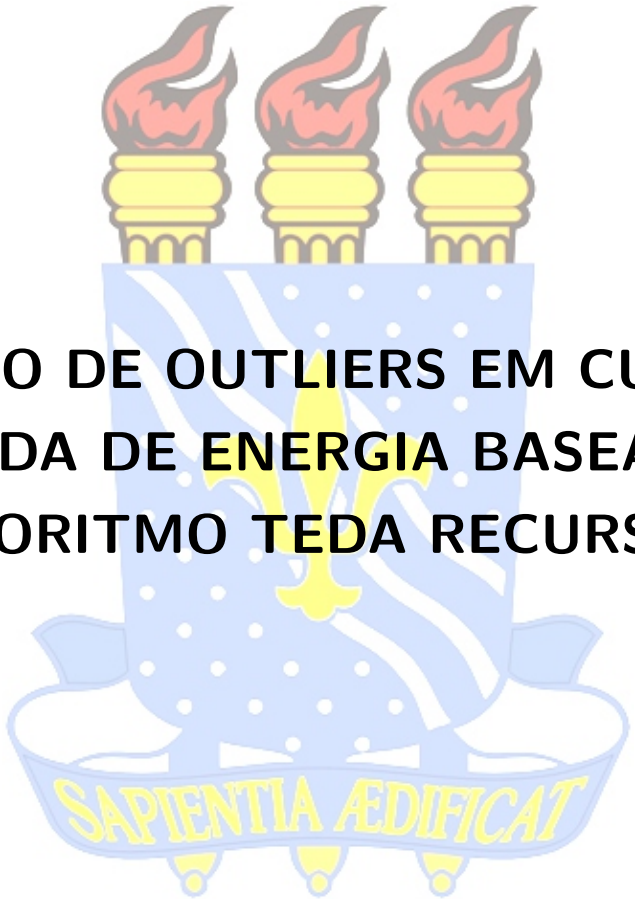
UNIVERSIDADE FEDERAL DA PARAÍBA
CENTRO DE ENERGIAS ALTERNATIVAS E RENOVÁVEIS
PROGRAMA DE PÓS GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Ricardo Wagner Correia Guerra Filho



DETECÇÃO DE OUTLIERS EM CURVAS DE DEMANDA DE ENERGIA BASEADO NO ALGORITMO TEDA RECURSIVO

João Pessoa
2023

Ricardo Wagner Correia Guerra Filho

DETECÇÃO DE OUTLIERS EM CURVAS DE DEMANDA DE ENERGIA BASEADO NO ALGORITMO TEDA RECURSIVO

Dissertação apresentada ao Programa de Pós Graduação em Engenharia Elétrica da Universidade Federal da Paraíba como exigência para a obtenção do título de Mestre em Engenharia Elétrica.

Universidade Federal da Paraíba

Centro de Energias Alternativas e Renováveis

Programa de Pós Graduação em Engenharia Elétrica

Orientador: Prof. Dr. Juan Moises Maurício Villanueva

João Pessoa

2023

Catalogação na publicação
Seção de Catalogação e Classificação

G934d Guerra Filho, Ricardo Wagner Correia.
Detecção de outliers em curvas de demanda de energia baseado no algoritmo TEDA recursivo / Ricardo Wagner Correia Guerra Filho. - João Pessoa, 2023.
80 f. : il.

Orientação: Juan Moises Maurício Villanueva.
Coorientação: Yuri Percy Molina Rodriguez.
Dissertação (Mestrado) - UFPB/CEAR.

1. Engenharia elétrica. 2. Outliers. 3. Curva de demanda. 4. Autoencoders. 5. Algoritmo TEDA. I. Maurício Villanueva, Juan Moises. II. Molina Rodriguez, Yuri Percy. III. Título.

UFPB/BC CDU 621.3(043)

Elaborado por MAGNOLIA FELIX DE ARAUJO - CRB-15/883

UNIVERSIDADE FEDERAL DA PARAÍBA – UFPB
CENTRO DE ENERGIAS ALTERNATIVAS E RENOVÁVEIS – CEAR
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA – PPGEE

A Comissão Examinadora, abaixo assinada, aprova a Dissertação

DETECÇÃO DE OUTLIERS EM CURVAS DE DEMANDA DE ENERGIA BASEADO NO ALGORITMO TEDA RECURSIVO

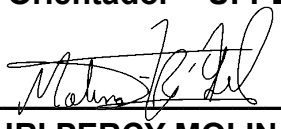
Elaborada por

RICARDO WAGNER CORREIA GUERRA FILHO

como requisito parcial para obtenção do grau de
Mestre em Engenharia Elétrica.

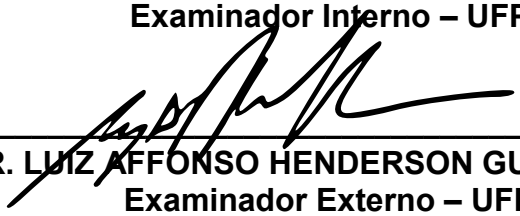
COMISSÃO EXAMINADORA

PROF. DR. JUAN MOISES MAURICIO VILLANUEVA
Orientador – UFPB



PROF. DR. YURI PERCY MOLINA RODRIGUEZ
Coorientador – UFPB

PROF. DR. EULER CÁSSIO TAVARES DE MACEDO
Examinador Interno – UFPB



PROF. DR. LUIZ AFFONSO HENDERSON GUEDES DE OLIVEIRA
Examinador Externo – UFRN

João Pessoa/PB, 28 de agosto de 2023

AGRADECIMENTOS

A Deus, por ter me dado a vida e todas as oportunidades que a acompanharam.

À minha família pelo apoio e incentivo durante toda essa jornada em especial à minha mãe Léa Trindade Crispim.

A minha esposa, Sheylla Monteiro e minha filha Natalie, pelo apoio incondicional, por me motivar sempre a buscar o melhor de mim e por toda paciência.

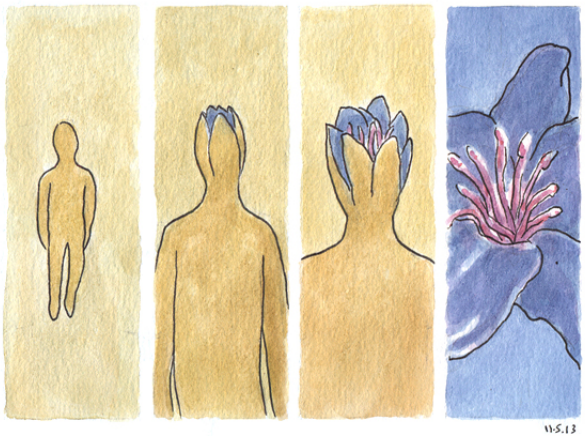
Ao Grupo de Inteligência Computacional Aplicada (GICA), em especial os professores Juan, Yuri e Euler, pela oportunidade de participar deste e pela confiança que depositaram em mim ao longo dos anos.

Ao meu orientador, Prof. Juan, pessoa de uma humanidade imensa, por todos ensinamentos ao longo destes anos de convívio.

Aos meus amigos, os espelhos da minha alma, refletem minhas virtudes e imperfeições onde através desse diálogo silencioso, descubro a essência da minha própria existência. Dentre estes, especialmente a meus amigos Diego, Joel, Kaique e Jean.

À Maria Iná Correia Guerra, minha avó, a pessoa que mais me apoiou em toda minha jornada, sempre com um novo ensinamento sobre a vida. Hoje, muito de mim é fruto dela.

E a todos que de alguma forma contribuíram para a realização deste trabalho.



- Vidi Descaves

RESUMO

A detecção de *outliers* é um problema importante que foi pesquisado em diversas áreas e domínios de aplicação. Muitas técnicas de detecção de *outliers* foram desenvolvidas especificamente para determinados domínios de aplicação, enquanto outras são mais genéricas. Os *outliers* podem ser de diferentes tipos, neste trabalho serão abordados os *outliers* do tipo pontuais (globais) e contextuais. O presente trabalho visa detectar essas falhas ocorridas durante a medição ou comunicação, e corrigir estas falhas afim de dar mais confiabilidade ao conjunto. O estudo foi realizado tendo como ferramentas alguns algoritmos de clusterização para etiquetar os *outliers*(clusterização), algoritmo de classificação KNN para a detecção dos *outliers*(classificação), algoritmos de aprendizado profundo do tipo Autoencoders e algoritmos evolutivos baseado em excentricidade, TEDA. Para a correção de *outliers* é utilizada a interpolação linear. São apresentadas ainda comparações de resultados entre técnicas como Z-Score, Z-Score Modificado, K-Means, C-Means, Autoencoders, Sparsed Autoencoders, TEDA e TEDA Diff, sendo estas comparações avaliadas de acordo com métricas utilizadas pela literatura. Com a finalidade de verificar o desempenho da técnica proposta foram usados dados de potência ativa de uma subestação de energia do estado da Paraíba. Este trabalho forneceu uma melhor compreensão das diferentes direções em que a pesquisa foi realizada sobre o tópico de detecção de *outliers* e como as técnicas desenvolvidas em uma área (a exemplo do TEDA) podem ser aplicadas nos domínios para os quais não se destinavam originalmente, onde para diferentes cenários de *outliers* em uma curva de demanda elétrica se conseguiu um coeficiente de correlação de matthew's sempre superior a 0.70, acima da média quando comparado com as outras técnicas testadas.

Palavras-chave: Outliers, Curva de demanda, Aprendizado profundo, Autoencoders, TEDA.

ABSTRACT

The detection of*outliers* is an important problem that has been researched in several areas and application domains. Many outlier detection techniques have been developed specifically for certain application domains, while others are more generic. *Outliers* can be of different types, in this work we will address *outliers* of the type punctual (global) and contextual. The present work aims to detect these failures that occurred during the measurement or communication, and to correct these failures in order to give more reliability to the set. The study was carried out using some clustering algorithms to label *outliers*(clustering), KNN classification algorithm for detecting *outliers*(classification), Autoencoders deep learning algorithms and TEDA auto evolutive algorithms. For the correction of outliers, linear interpolation is used. Comparisons of results between techniques such as Z-Score, Modified Z-Score, K-Means, C-Means, Autoencoders, Sparsed Autoencoders, TEDA and TEDA Diff are also provided, and these comparisons are evaluated according to the metrics used in the literature. In order to verify the performance of the proposed techniques, active power data from an energy substation in the state of Paraíba were used. This work provided a better understanding of the different directions in which research was carried out on the topic of *outliers* detection and how the techniques developed in one area (such as TEDA) can be applied in domains for which they were not originally intended, where for different scenarios of *outliers* in an electrical demand curve, a matthew's correlation coefficient was always higher than 0.70, above average when compared to the other tested techniques.

Keywords: outliers, Demand curve, Deep learning, Autoencoders, TEDA.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

MDA	Mediana de Desvio Absoluto
KM	*K-Means*
FCM	*Fuzzy C-Means*
KNN	*K-Nearest Neighbors*
AE	*Autoencoder*
AEE	*Autoencoder Esparso*
MLP	*Multilayer Perceptron*
ReLU	*Rectified Linear Unit*
TanH	*Hiperbolic Tangent Function*
SE JPS	Substação João Pessoa
TP	Verdadeiros Positivos
TN	Verdadeiros Negativos
FP	Falsos Positivos
FN	Falsos Negativos
MCC	Coeficiente de Correlação de Matthews
MAPE	Erro Percentual Absoluto Médio
MSE	Erro Médio Quadrático
SSE	Soma dos Erros ao Quadrado
TEDA	Typicality and Eccentricity based Data Analytics

SUMÁRIO

1 INTRODUÇÃO

1.1 PERTINÊNCIA E MOTIVAÇÃO DO TRABALHO

Big Data é uma realidade sem volta, e está claro que a análise de dados já mudou várias indústrias e o setor elétrico é o próximo. A análise de dados ajudou a Amazon a se tornar a central de varejo que é hoje (SMITH; LINDEN, 2017), enquanto empresas como Airbnb (GALLAGHER, 2018) e Uber (CRAMER; KRUEGER, 2016) aplicaram com sucesso grande volume de dados para agitar os setores hoteleiro e de transporte, respectivamente.

No setor elétrico, a crescente descarbonização e descentralização das redes de energia requer soluções mais inteligentes e flexíveis - e o *Data Analytics* é uma forte aliada. Os dados permitem o gerenciamento em tempo real da infra-estrutura da rede, previsões mais confiáveis para investimento e manutenção, e possibilita o desenvolvimento de novos modelos de negócios e serviços de energia.

Com a crescente penetração das tecnologias de informação tradicionais e emergentes na energia elétrica juntamente com o rápido desenvolvimento da reforma do mercado de eletricidade, a indústria de energia elétrica acumulou uma grande quantidade de dados (YU et al., 2015). As questões de qualidade dos dados têm se tornado cada vez mais proeminentes, o que afeta a precisão e a eficácia da mineração de dados de eletricidade e da análise de grandes volumes de dados de energia. Também está intimamente relacionada à segurança e confiabilidade da operação do sistema de energia elétrica e gestão baseada em apoio à decisão com base em dados.

No momento, grandes quantidades de dados de consumo de eletricidade têm sido acumulado. É difícil para as pessoas descobrirem diretamente o conhecimento que está escondido por trás desses dados, que são heterogêneos e inconsistentes (MOSLEHI; KUMAR, 2010). Para melhorar a qualidade dos dados de consumo de eletricidade, suas características devem ser plenamente consideradas. Com o desenvolvimento da área de tecnologia da informação e de redes inteligentes, os dados de consumo de eletricidade apresentam as seguintes características (CHEN et al., 2017):

- O volume de dados tem aumentado rapidamente. Com o desenvolvimento de redes inteligentes, muitas empresas de energia elétrica estabeleceram centros de computação em nuvem para processar os dados. Assim, ele faz os dados disponíveis aumentarem.
- A transmissão de dados e a velocidade de processamento aceleraram muito devido ao estabelecimento de equipamentos e sistemas inteligentes (MADNICK; ZHU, 2006).

- A gestão é mais precisa. A tecnologia de mineração de dados faz análise e gerenciamento dos dados dos medidores de energia elétrica mais precisa (*smart meters*).

Os volumes de dados de consumo de eletricidade são enormes. Além disso, os tipos de dados de consumo de eletricidade são vários . Através da análise desses dados é possível extrair conhecimento de alto valor de empresas de energia elétrica para, consequentemente, aumentar seu nível de rentabilidade e controle. Zhang, Wang e Li (2010) relatam que em alguns casos, quando a taxa de utilização dos dados aumenta em 10%, as empresas de energia elétrica podem aumentar o lucro de 20-49%. Portanto, devemos melhorar a taxa de utilização de dados de consumo de eletricidade. Entretanto, se os dados forem redundantes e confusos, aumentará a dificuldade de se obter um conhecimento efetivo. Se os dados estiverem errados e desatualizados, isso resultará em análise incorreta dos dados resultados.

As questões de qualidade dos dados de consumo de eletricidade podem ser divididas em três categorias, a saber, dados de ruído, dados incompletos e *outliers*. Os dados de ruído referem-se a dados que são contra a lógica empresarial (HAO; ZHOU; SONG, 2015) ou seja, não condizem com a realidade do negócio, portanto, podem afetar os resultados da análise dos dados. Dados incompletos referem-se aos dados de valores de atributos ausentes na fonte de dados, ou aos dados que se desviaram das características estatísticas (HEYONG; MINGJIAN; BINGCHUAN, 2010). Os *outliers* referem-se a dados que se desviam da maioria dos dados do conjunto de dados.

Assim, é muito importante e pertinente a pesquisa na área de detecção de *outliers* em curvas de demanda (ANDRADE et al., 2018) e é por este motivo que este trabalho tem por objetivo abordar a detecção e correção de anomalias, intituladas neste trabalho como *outliers*, em curvas de potência elétrica. Essas curvas são na verdade séries temporais e podem ser resumidads como um conjunto de observações, cada uma sendo registrada em um tempo específico (TEICHGRAEBER; BRANDT, 2022).

Ainda, ao analisar o problema dos *outliers* em séries temporais, percebem-se diversas linhas de pensamentos que trazem diversas problemáticas em relação à influência destes "erros" na análise de dados para diversas finalidades. Assim, verifica-se a importância de uma correção que se comporte mais próxima possível da realidade física no qual vieram os dados (potência elétrica), para que aplicações posteriores tenham um resultado com o menor erro possível.

Diversas são as aplicações em que são utilizadas séries temporais, e no caso das curvas de potência, podemos citar a aplicação para previsão de demanda futura.

É importante que, por exemplo, uma empresa de distribuição de energia elétrica tenha em mãos dados precisos e confiáveis que permitam uma previsão de demanda futura com certa exatidão, para não ocorrer em compra de potência energética acima ou abaixo

dos limites estabelecidos pela RESOLUÇÃO NORMATIVA Nº109, DE 26 DE OUTUBRO DE 2004 da Agência Nacional de Energia Elétrica - ANEEL (BRASIL, 2004). Assim sendo, é importante saber que os dados obtivos através de medição, por vezes, apresentam alguns problemas. De acordo com Aggarwal (2015), na maioria das aplicações, os dados são criados por um ou mais processos geradores, que podem refletir a atividade no sistema ou observações coletadas sobre entidades. Quando o processo de geração se comporta de maneira incomum, resulta na criação de *outliers*. E segundo Hawkins (1980) um *outlier* é uma observação que se desvia das outras observações de uma forma que suscita suspeitas de que foi gerada por um mecanismo diferente.

Ademais, detectar os *outliers* em séries temporais para que sua presença não venha a gerar conclusões errôneas, quando da análise dos dados. Neste trabalho explica-se alguns dos diversos métodos existentes, sendo eles o Z-Score, K-Means, C-Means, Autoencoder e TEDA, fazendo comparações, trazendo o estado da arte na resolução do problema, porém, para algumas aplicações à exemplo da previsão de dados futuros, apenas detectar os *outliers* nao é suficiente, sendo necessária a correção destes dados para que a aplicação não tenha resultado enviesado, com erros e termine por não refletir a realidade e por este motivo uma técnica de correção de dados é aplicada como complemento.

Algumas técnicas de inteligência computacional são adaptadas e utilizadas com o objetivo de correção de outliers, a exemplo das técnicas *Z-Score*, Z-Score Modificado, *K-Means*, *C-Means*, *Autoencoder*, *Autoencoder Esparso*, *TEDA* e o *TEDA Diff*. Ainda, são propostos oito casos de estudo e aplicação, onde as técnicas mencionadas tem seus resultados avaliados e comparados entre elas.

Ao estudar o tema, pretende-se trazer de uma visão científica a viabilidade do método proposto para correção de *outliers*, mostrando suas vantagens e desvantagens em relação a outros métodos, assim como o resultado prático decorrente da aplicação em dados reais fornecidos por empresa de distribuição de energia.

1.2 OBJETIVO GERAL

O objetivo deste trabalho consiste em utilizar de técnica de inteligência computacional para realizar a detecção e correção de *outliers* em curvas de demanda elétrica com a finalidade de melhorar a qualidade dos dados e possibilitar uma maior precisão nas previsões de demanda elétrica.

Ainda, abordar o tema da detecção e correção de *outliers* em séries temporais, especificamente curvas de demanda elétrica e suas particularidades, comparando diversas técnicas, suas aplicações e resultados através de levantamento bibliográfico com o objetivo de atingir o estado da arte.

1.3 OBJETIVO ESPECÍFICO

O objetivo específico desta pesquisa é avaliar o desempenho do algoritmo TEDA (Typicality and Eccentricity Data Analytics) como uma técnica para a detecção de outliers em curvas de demanda elétrica. Para alcançar esse propósito, serão conduzidas análises comparativas e avaliações quantitativas para investigar o seguinte:

- A criação de um conjunto de dados de potência elétrica, abrangendo cenários realistas, que incluam a introdução controlada de *outliers*, tanto pontuais quanto contextuais, nas curvas de demanda;
- A implementação e adaptação do algoritmo TEDA, juntamente com outros métodos de detecção de *outliers* amplamente utilizados, como Z-Score, K-Means, C-Means e Autoencoder;
- A realização de uma avaliação rigorosa do desempenho do algoritmo TEDA em comparação com as demais técnicas empregadas, utilizando métricas apropriadas de avaliação de detecção de *outliers*, como MSE (*Mean Squared Error*), MAPE (*Mean Absolute Percentage Error*) e MCC (*Matthews Correlation Coefficient*);
- A construção de uma tabela comparativa com os resultados obtidos por cada método em diferentes cenários com *outliers*, permitindo uma compreensão detalhada das capacidades do TEDA em identificar de forma precisa e eficiente as anomalias presentes nas curvas de demanda elétrica;
- A discussão aprofundada dos resultados, enfatizando as vantagens e limitações do algoritmo TEDA em relação às outras técnicas avaliadas, bem como as possíveis aplicações práticas desse algoritmo na detecção de *outliers* em contextos reais de distribuição de energia elétrica.

Ao alcançar esses objetivos específicos, espera-se obter *insights* relevantes sobre a eficácia e o potencial do algoritmo TEDA como uma solução viável para detectar *outliers* em curvas de demanda elétrica. Os resultados e discussões obtidos poderão contribuir para aprimorar o conhecimento em gerenciamento de dados elétricos e apoiar a tomada de decisões mais informadas e eficazes no setor de energia elétrica, visando garantir maior confiabilidade e eficiência operacional do sistema elétrico.

1.3.1 CONTRIBUIÇÕES

Como contribuição, este trabalho oferece excelentes algoritmos para detecção de *outliers* pontuais e contextuais, como o Autoenconder e o TEDA. Sendo o algoritmo TEDA de baixo custo computacional, *online*, auto evolutivo e aparamétrico.

Ademais, a variedade TEDA Diff se mostrou bastante eficiente, sendo possível comparar e até se sair melhor quando comparado em alguns casos com o algoritmo Autoencoder.

1.4 ORGANIZAÇÃO DO TRABALHO

Este trabalho foi estruturado em cinco capítulos. No primeiro capítulo, foi realizada uma introdução, abordando a importância e motivação do tema. Além disso, foram citados os objetivos gerais e específicos.

No segundo capítulo, será feita uma fundamentação teórica sobre o tema detecção e correção de anomalias. Conceitos relacionados ao tema como séries temporais, tipos de anomalias, algumas técnicas existentes serão discutidas e exemplificadas.

O terceiro capítulo abordará a metodologia utilizada, partindo da descrição da técnica utilizada partindo de uma visão geral e terminando por uma descrição de todas as técnicas utilizadas e o seu funcionamento, através do que foi implementado.

No quarto capítulo, os resultados serão apresentados em forma de tabela com a performance de cada técnica proposta em relação aos casos trazidos e de acordo com as métricas utilizadas e será feita uma sucinta explicação deles.

O quinto capítulo constará da conclusão do trabalho, comparando os resultados obtidos com os esperados ao traçar os objetivos. Também será discutido as futuras aplicações e utilidades deste projeto, assim como, proposta de continuação e expansão.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo serão abordados conceitos teóricos necessários para o entendimento deste trabalho. Inicialmente será explicado o que é uma série temporal e suas particularidades, seguindo pela definição de *concept drift*.

Nas séries temporais podem existir anomalias ou *outliers* e assim apresentam-se suas definições e tipos. Em uma sequência lógica, algumas técnicas de detecção de anomalias são apresentadas, finalizando com um quadro comparativo delas.

2.1 SÉRIES TEMPORAIS

As séries temporais contêm um conjunto de valores que normalmente são gerados pelo tempo de medição contínuo. Por isso, podemos dizer que uma série temporal é um processo estocástico pois evolui com o tempo de acordo com as leis probabilísticas (DAS, 1994).

Geralmente os valores colhidos consecutivamente da série não mudam muito significativamente ou são alterados de maneira suave. Nesses casos, alterações repentinas nos registros de dados subjacentes podem ser consideradas eventos anômalos e a descoberta de *outliers* em séries temporais está intimamente relacionada ao problema de detecção destes eventos anômalos.

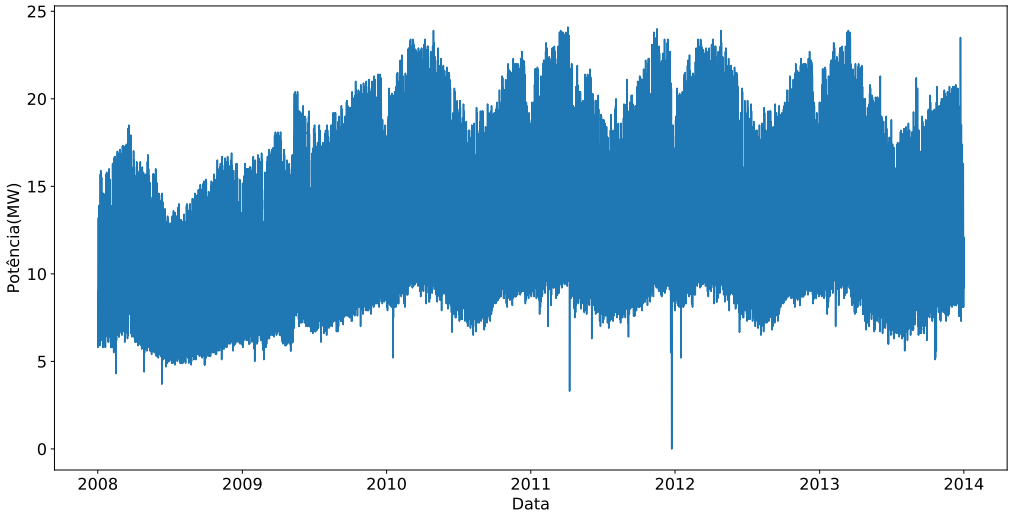
Para que seja possível realizar a detecção dos *outliers* em uma série temporal, é necessário compreender que existem outros aspectos que devem ser considerados, tais como: nível, tendência e sazonalidade, assim como estacionariedade e autocorrelação.

2.1.1 SÉRIES TEMPORAIS DO TIPO CURVA DE DEMANDA ELÉTRICA

No presente trabalho iremos abordar a detecção e correção de outliers em séries temporais do tipo curva de demanda elétrica, que são séries univariadas de potência elétrica medidas com o tempo. Essas séries tem algumas particularidades que devem ser levantadas antes de tratarmos do objetivo específico do trabalho.

Para realizar algumas análises iniciais, usaremos dados medidos de potência ativa numa frequência de 1 amostra a cada 15 minutos. Um exemplo de uma curva de demanda é ilustrado na Figura 1.

Figura 1 – Curva de demanda da subestação João Pessoa.



Fonte: Autoria própria.

2.1.2 TENDÊNCIA E SAZONALIDADE

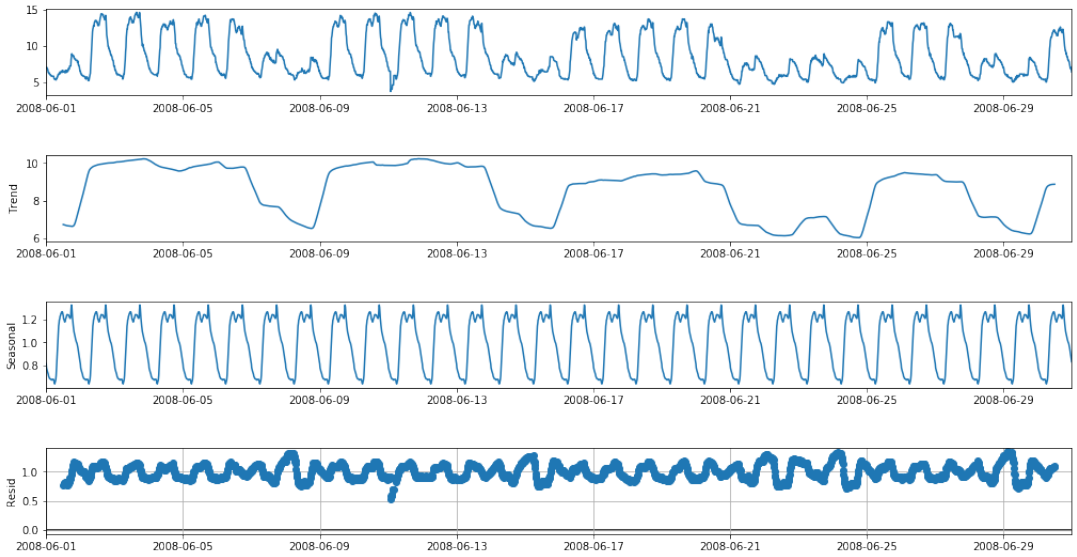
De acordo com Hyndman e Athanasopoulos (2018) existe uma tendência quando há um aumento ou diminuição nos dados, ou seja, estes dados não precisam ser lineares. Por outro lado, um padrão sazonal ocorre quando uma série temporal é afetada por fatores sazonais, como a época do ano ou o dia da semana. O mesmo autor explica que sazonalidade ocorre sempre com frequência fixa e conhecida. A curva de potência elétrica apresentada na Figura 1 reflete a sazonalidade, que é induzida em padrão diário e semanal.

Para extrair a tendência e sazonalidade de uma série temporal, pode-se utilizar o método clássico de decomposição que se originou na década de vinte por Persons (1923). É um procedimento relativamente simples e constitui o ponto de partida para a maioria dos outros métodos de decomposição de séries temporais. Em muitas séries temporais, a amplitude das variações sazonais e irregulares aumenta à medida que o nível da tendência aumenta. Nesta situação, um modelo multiplicativo é geralmente apropriado. No modelo multiplicativo do método clássico, a série temporal original é expressa como o produto de tendência, componentes sazonais e irregulares.

$$Série = Tendência \times Sazonalidade \times Resíduo \tag{2.1}$$

Na Figura 2 é ilustrada a curva de demanda e as componentes de tendência, sazonalidade e o resíduo respectivamente. É importante verificar nesta Figura que em uma curva de demanda a tendência e a sazonalidade são componentes de grande influência nesta série temporal.

Figura 2 – Série temporal decomposta.



Fonte: Autoria própria.

As curvas da Figura 2 foram obtidas através do método multiplicativo utilizando ferramenta computacional *STATS MODEL* (STATS MODELS, 2020).

2.1.3 ESTACIONARIEDADE

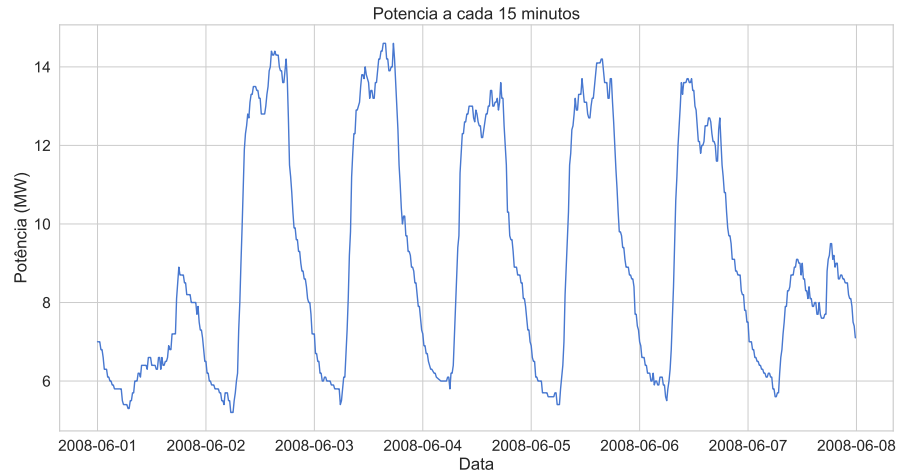
A estacionariedade é uma característica importante das séries temporais. Uma série temporal é considerada estacionária se suas propriedades estatísticas não mudarem ao longo do tempo. Em outras palavras, ele tem média e variação constantes, e a covariância é independente do tempo.

Idealmente, é desejado ter uma série temporal estacionária para modelagem. Obviamente, nem todas as séries são estacionárias, mas é possível fazer diferentes transformações para torná-las estacionárias.

No nosso caso, a série é notadamente afetada por tendência e sazonalidade, o que faz com que as estatísticas sumárias calculadas sobre a série não sejam consistentes com o tempo, como a média e a variância das observações.

Os métodos clássicos de análise e previsão de séries temporais estão preocupados em tornar estacionários os dados de séries temporais não estacionárias, identificando e removendo tendências e removendo efeitos sazonais, porém não é o objetivo deste trabalho a discussão minuciosa sobre o tema. Ilustra-se na Figura 3 a sazonalidade da curva de demanda elétrica, na qual é possível observar que existe um comportamento cíclico diário com horários de picos de consumo bastante definidos para cada dia da semana.

Figura 3 – Sazonalidade da curva de demanda.



Fonte: Autoria própria.

2.1.4 *CONCEPT DRIFT*

Esse problema das mudanças nas relações subjacentes nos dados é chamado de *concept drift* no campo de aprendizado de máquina (WANG; SCHLOBACH; KLEIN, 2010).

A modelagem preditiva é o problema de aprender um modelo a partir de dados históricos e usar o modelo para fazer previsões sobre novos dados no qual não sabemos a resposta.

Muitas vezes, esse mapeamento é considerado estático, o que significa que o mapeamento aprendido com dados históricos é igualmente válido no futuro no qual novos dados e que as relações entre os dados de entrada e saída não mudam.

Em alguns casos, as relações entre os dados de entrada e saída podem mudar ao longo do tempo, o que significa que, por sua vez, há alterações na função de mapeamento subjacente desconhecida.

As alterações podem ser consequentes, como as previsões feitas por um modelo treinado em dados históricos mais antigos não mais corretas ou tão corretas quanto poderiam ser se o modelo fosse treinado em dados históricos mais recentes.

Essas alterações, por sua vez, podem ser detectadas e, se detectadas, pode ser possível atualizar o modelo aprendido para refletir essas alterações.

De acordo com Maimon e Rokach (2005), isso traz dois desafios importantes. O primeiro desafio é detectar quando ocorre o *concept drift*. O segundo desafio é manter os padrões atualizados sem induzi-los considerando o conjunto de histórico completo.

Na maioria das aplicações de análise de dados, padrões e relações evoluem ao longo do tempo e devem ser analisados quase em tempo real, portanto, os modelos construídos para fazer essas análises rapidamente se tornam obsoletos (ŽLIOBAITĖ; PECHENIZKIY; GAMA, 2016).

Para lidar com um possível problema de *concept drift*, a maneira mais comum é não lidar com isso e assumir que os dados não mudam com o tempo (estacionário no caso das séries temporais). Porém, sabe-se que isso nem sempre ocorre e Hashmani et al. (2020) traz algumas formas que seriam re-treinar o modelo, atualizar o modelo com novos dados (aproveitando o modelo antigo), colocar pesos de importância em variáveis de entrada, etc.

Alguns modelos são capazes de automaticamente se adequar a mudanças de conceito, como é o exemplo do *Typicality and Eccentricity based Data Analytics* (TEDA) (ANGELOV, 2013).

2.1.5 ALGORITMOS ONLINE

Os algoritmos *online* são uma classe importante de algoritmos que processam dados em um fluxo contínuo, tomando decisões sobre cada dado que chega. Estudados extensivamente em ciência da computação e engenharia, são amplamente utilizados em aplicações como roteamento de rede e sistemas de recomendação. Neste tópico, será explorado o conceito de algoritmos *online*, suas vantagens e limitações, trazendo exemplos de seu uso.

Uma das principais vantagens dos algoritmos *online* é a capacidade de processar dados em tempo real com informações limitadas, tornando-os eficientes e escaláveis. De acordo com Lin et al. (2022), os algoritmos *online* podem lidar com grandes volumes de dados de forma eficiente e eficaz em tempo real, tornando-os uma ferramenta importante para muitas aplicações modernas. Essa capacidade é particularmente útil em aplicações onde os dados são contínuos e o processo de tomada de decisão deve ser feito rapidamente, como no caso de fluxos de dados em uma rede de computadores.

No entanto, os algoritmos *online* têm limitações que devem ser levadas em consideração. Uma das principais limitações conforme é enunciado por Karp (1992), é que algoritmos *online* podem tomar decisões abaixo do ideal devido à informação limitada disponível em cada etapa, levando a uma perda de desempenho em comparação com algoritmos *offline* que têm acesso a toda a entrada antecipadamente.

Apesar de suas limitações, algoritmos *online* têm sido aplicados com sucesso em diversas aplicações, como sistemas de recomendação, onde algoritmos *online* são usados para personalizar recomendações para cada usuário com base em seu histórico de navegação e comportamento em tempo real (CABRERA-SÁNCHEZ et al., 2020). No roteamento de rede, algoritmos *online* são usados para otimizar o fluxo de dados pela rede, reduzindo o

congestionamento e melhorando o desempenho (POUPART et al., 2016). Na publicidade online, algoritmos *online* são usados para determinar qual anúncio exibir para um usuário com base em seu histórico de navegação e comportamento em tempo real (BARBOSA et al., 2023).

Por fim, os algoritmos *online* são uma ferramenta importante para o processamento de dados em tempo real com informações limitadas. Eles têm vantagens como eficiência e escalabilidade, mas também limitações como tomada de decisão abaixo do ideal. Algoritmos *online* têm sido aplicados com sucesso em diversas aplicações, sendo necessárias mais pesquisas para melhorar seu desempenho e superar suas limitações.

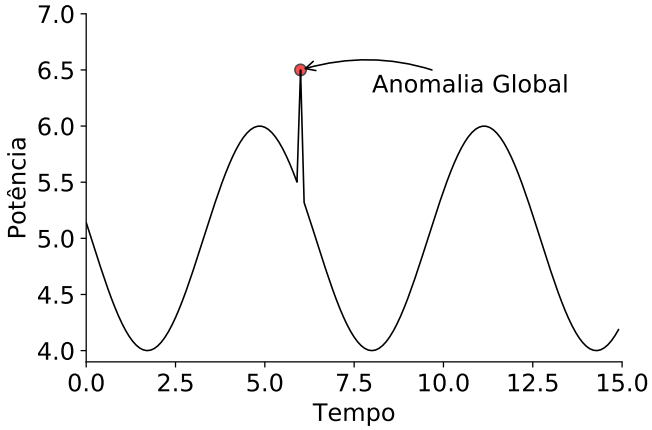
2.2 TIPOS DE OUTLIERS

Um aspecto importante antes de se aplicar uma técnica de detecção de *outliers* é saber a natureza do *outlier* que buscamos. Os *outliers* podem ser classificadas em três categorias.

2.2.1 OUTLIER GLOBAL

Se um ponto, representando um dado individual pode ser considerado anômalo em relação ao restante dos dados, o ponto é denominado um *outlier* global. Este é o tipo mais simples de *outlier* e é o foco da maioria das pesquisas sobre detecção de anomalias.

Figura 4 – Exemplo *outlier* global.



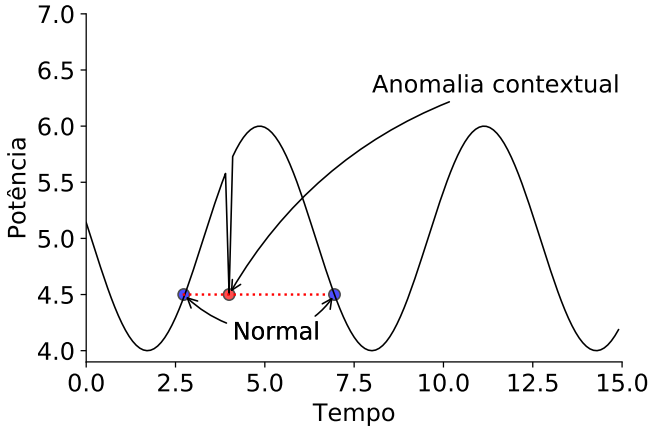
Fonte: Autoria própria.

Por exemplo, na ilustração presente na Figura 4, o ponto marcado em vermelho fica fora do limite das regiões normais e, portanto, é considerado uma anomalia, pois é diferente dos pontos de dados considerados normais.

2.2.2 OUTLIER CONTEXTUAL

Se uma instância de dados é anômala em um contexto específico, mas não de outra forma, é denominado *outlier* contextual, também conhecido como *outlier* condicional (SONG et al., 2007). A noção de um contexto é induzida pela estrutura no conjunto de dados e deve ser especificada como parte da formulação do problema e em séries temporais o tempo é um atributo contextual que determina a posição de uma instância em toda a sequência.

Figura 5 – Exemplo *outlier* contextual.



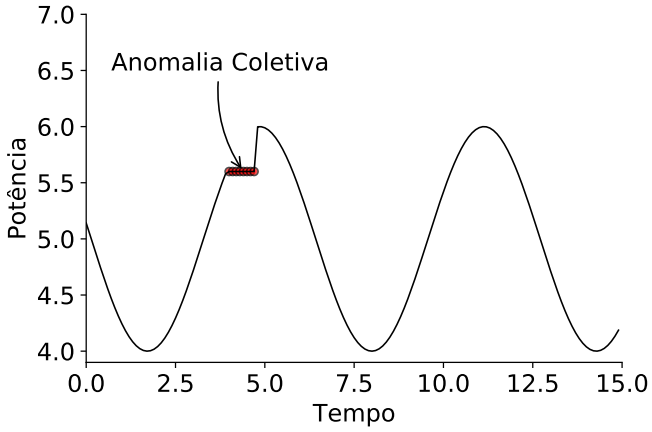
Fonte: Autoria própria.

Por exemplo, o baixo consumo de energia elétrica é algo considerado normal nas primeiras horas do dia, porém, não é normal no meio do dia, momento em que os consumidores residenciais e industriais estão em plena execução de suas atividades, ou seja, no contexto do tempo, o baixo consumo de energia naquele momento poder ser considerado um *outlier* contextual, como se ilustra na Figura 5.

2.2.3 OUTLIER COLETIVO

Se uma coleção de pontos de dados relacionados for anômalo em relação a todo o conjunto de dados, será denominada uma anomalia coletiva (Figura 6). Os pontos de dados individuais em uma anomalia coletiva podem não ser considerado um *outlier*, mas as ocorrências juntas como uma coleção é considerado *outlier*.

Figura 6 – Exemplo *outlier* coletivo.



Fonte: Autoria própria.

Neste trabalho, o foco será na detecção e correção de *outliers* apenas do tipo global e contextual.

2.3 TÉCNICAS DE DETECÇÃO DE OUTLIERS

Uma introdução sobre séries temporais e a natureza dos dados em uma curva de demanda elétrica foi apresentada e neste momento serão explanadas algumas técnicas de detecção de *outliers* e suas particularidades.

Primeiramente, é importante saber que métodos de detecção de *outlier* podem ser divididos entre métodos univariados e métodos multivariados. Métodos univariados envolvem a análise de uma única variável, enquanto a análise multivariada examina duas ou mais variáveis. A maioria das análises multivariadas envolve uma variável dependente e múltiplas variáveis independentes.

Outra taxonomia fundamental dos métodos de detecção externa é entre métodos paramétricos (estatísticos) e métodos não paramétricos isentos de modelo (WILLIAMS et al., 2002). Os métodos paramétricos estatísticos assumem uma distribuição subjacente conhecida das observações (HAWKINS, 1980)ou, pelo menos, baseiam-se em estimativas estatísticas de parâmetros de distribuição desconhecidos (HADI, 1992).

Os rótulos associados a um dado indicam se este dado é normal ou anômalo. Deve-se notar que a obtenção de dados rotulados que sejam precisos e representativos de todos os tipos de comportamento é frequentemente caro. A rotulagem é geralmente feita manualmente por um especialista humano e, portanto, é necessário um esforço substancial para obter o conjunto de dados de treinamento rotulado.

Normalmente, obter um conjunto rotulado de instâncias de dados anômalos que cubra todo o tipo possível de comportamento anômalo é mais difícil do que obter rótulos para o comportamento normal. Além disso, o comportamento anômalo é frequentemente dinâmico por natureza, por exemplo, novos tipos de anomalias podem surgir, para os quais não há dados rotulados. Por esse motivo a literatura divide as técnicas de aprendizagem de máquina, onde podemos incluir a detecção de *outliers*, em três tipos de abordagens.

- Abordagem supervisionada;
- Abordagem semi-supervisionada;
- Abordagem não-supervisionada.

As técnicas que utilizam a abordagem **supervisionada** assumem que a disponibilidade de um conjunto de dados de treinamento se encontra rotulado em dados normais e anômalos. Existem dois problemas principais que surgem na detecção supervisionada de *outliers*. Primeiro, os dados anômalos se encontram em menor quantidade comparados aos dados normais do conjunto de treinamento e em segundo lugar, a obtenção de rótulos precisos e representativos, especialmente para o grupo de dados anômalos, geralmente é um desafio.

As técnicas que utilizam a abordagem **semi-supervisionada** assumem que os dados de treinamento são rotulados apenas para os dados "normais". Essa abordagem não necessáriamente requer rótulos nos dados anômalos e por isso eles são mais amplamente aplicáveis do que as técnicas supervisionadas. A abordagem típica usada em tais técnicas é criar um modelo para a classe correspondente ao comportamento normal e usar o modelo para identificar anomalias nos dados de teste.

Por fim, as técnicas que operam na abordagem **não-supervisionado** não requerem dados de treinamento e, portanto, são mais amplamente aplicáveis. As técnicas nesta categoria assumem implícitamente que instâncias normais são muito mais frequentes que anomalias nos dados de teste. Se essa suposição não for verdadeira, essas técnicas sofrerão uma alta taxa de alarmes falsos (falsos positivos e falsos negativos). Muitas técnicas semi-supervisionadas podem ser adaptadas para operar em um modo não supervisionado, usando uma amostra do conjunto de dados não rotulados como dados de treinamento. Essa adaptação assume que os dados do teste contêm muito poucas anomalias e o modelo aprendido durante o treinamento é robusto para essas poucas anomalias.

Neste trabalho estaremos utilizando técnicas nao supervisionadas por natureza para detecção de *outliers* e não entraremos em detalhes sobre as abordagens, que são bem explicadas em Freitas et al. (2019).

2.3.1 ALGORITMOS DE DETECÇÃO DE OUTLIERS

Neste tópico serão apresentadas as principais técnicas de detecção de *outliers*, tais como: z-score, z-score modificado, k-means, Fuzzy c-means, KNN, Autoencoders e TEDA.

2.3.1.1 Z-SCORE

O *z-score* é uma medida numérica estatística que fornece uma idéia de quão longe um dado está da média. Mas tecnicamente, porém, é uma medida de desvio padrão abaixo ou acima da população onde aquele dado se encontra, se convertendo em uma pontuação bruta.

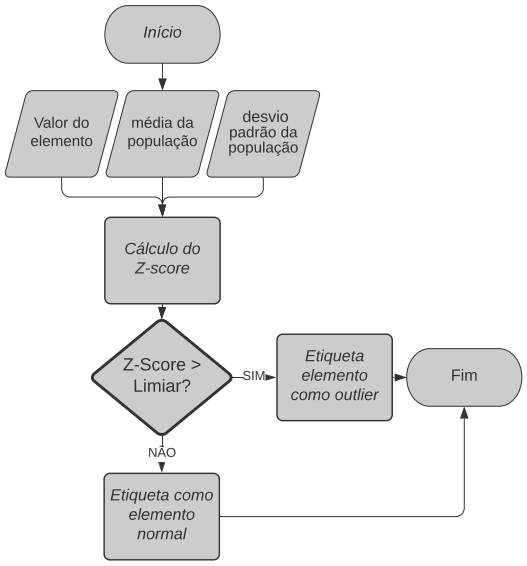
Shiffler (1988) trouxe o z-score como uma solução simples para detecção de *outliers*. Basicamente precisamos saber a média e o desvio padrão do conjunto e assim calculamos o *z-score* de cada ponto deste conjunto, atribuindo uma pontuação, conforme equação (2.2).

$$z = \frac{(X - \mu)}{\sigma} \tag{2.2}$$

em que: z é o *z-score*, X é o valor do elemento, μ é média da população e σ é o desvio padrão da população.

Para usar o *z-score* como método de detecção de anomalias devem-se separar os dados que tiverem um *z-score* que se desvia além de um limiar, podendo então serem candidatos a um *outlier* (Figura 7). Este limiar é geralmente 3, de acordo com Kannan, Manoj e Arumugam (2015) e Garcia (2012),

Figura 7 – Fluxograma Z-Score.



Fonte: Autoria própria.

2.3.1.2 Z-SCORE MODIFICADO

Existe um problema no *Z-Scores* ao utilizar a média da amostra e o desvio padrão da amostra, pois a média das amostras pode ser bastante afetada por alguns valores extremos (*outliers*) ou mesmo por um único valor extremo. Para resolver esse problema, a mediana e a mediana do desvio absoluto (MDA) são empregadas nas *Z-Scores* modificadas, em vez da média e desvio padrão da amostra, respectivamente (IGLEWICZ; HOAGLIN, 1993).

$$MDA = mediana\,|x_i - \widetilde{x}| \tag{2.3}$$

em que: $\widetilde{x}$ é a mediana da amostra.

$$M_i = \frac{0,6745(x_i - \widetilde{x})}{MDA} \tag{2.4}$$

Iglewicz e Hoaglin (1993) sugeriram que as observações são rotuladas como *outliers* quando $|M_i|$>3,5. A pontuação do M_i é eficaz para dados normais da mesma maneira que o *Z-score*.

2.3.1.3 K-MEANS

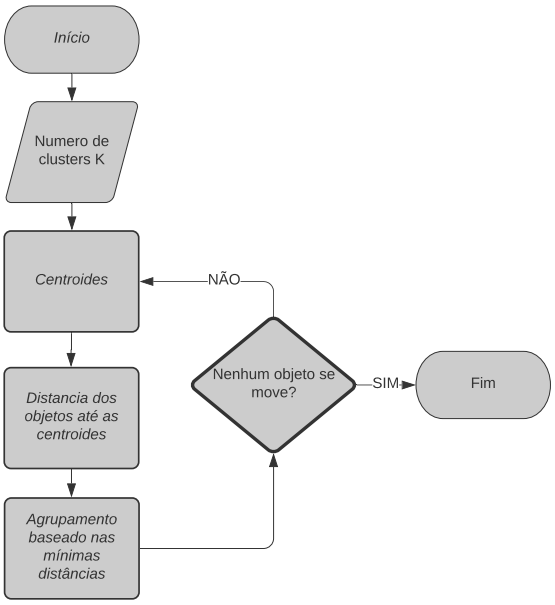
K-means (KM) é uma técnica de agrupamento (clusterização) criada por MacQueen et al. (1967) e é um dos mais simples algoritmos de aprendizado não supervisionado que resolvem este conhecido problema. O procedimento segue uma maneira simples e fácil de classificar um determinado conjunto de dados por meio de um certo número de *clusters* (suponha k *clusters*), como ilustrado no fluxograma da Figura 8.

A idéia principal é definir k centroides, um para cada *cluster*. O próximo passo é pegar cada ponto pertencente a um determinado conjunto de dados e associá-lo ao centroide mais próximo. Quando nenhum ponto está pendente, o primeiro passo é concluído e um agrupamento inicial é feito. Nesse ponto, precisamos recalcular k novos centroides como baricentros dos aglomerados resultantes da etapa anterior.

Depois de termos esses k novos centroides, uma nova ligação deve ser feita entre os mesmos pontos de conjunto de dados e o novo centroide mais próximo. Um *loop* foi gerado. Como resultado desse *loop*, podemos notar que os k centroides mudam de localização passo a passo até que não sejam feitas mais alterações. Em outras palavras, os centroides não se movem mais.

Finalmente, este algoritmo visa minimizar uma função objetivo, neste caso uma função do erro quadrático. A função objetiva da Equação (2.5)

Figura 8 – Fluxograma K-Means.



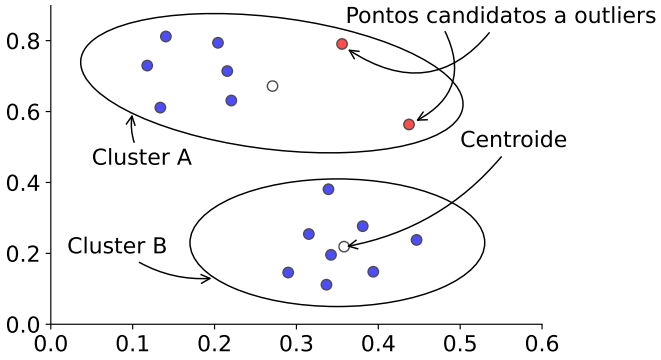
Fonte: Autoria própria.

$$J = \sum_{i=1}^{k} \sum_{x \in S_i} \|x - \mu_i\|^2 \tag{2.5}$$

em que: $\|x - \mu_i\|^2$ é uma medida de distância escolhida entre um ponto de dados x e o centro de *cluster* μ_i, sendo um indicador da distância dos n pontos de dados de seus respectivos centroides.

Existem algumas formas de utilizar K-means como técnica de detecção de *outlier*. Cada *cluster* tem um centroide, e os pontos pertencentes a este *cluster* estão definidos pela distância deles para com este centroide. Então, pode-se categorizar *outliers* de acordo com a distância absoluta deste para com o centroide, bem como de acordo com a distância relativa, que seria a distância absoluta comparada com a média das distâncias dos pontos do *cluster* para o centroide, à exemplo do observado na Figura 9.

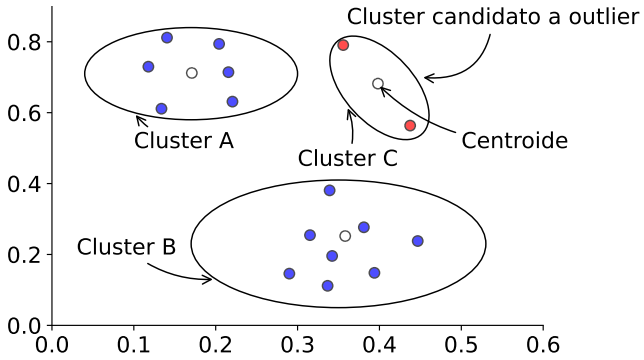
Figura 9 – Exemplo 1 K-Means.



Fonte: Autoria própria.

No caso onde nos deparamos com um conjunto de dados univariado, podemos ainda categorizar os *outliers* como pontos pertencentes aos *clusters* extremos do conjunto de dados (Figura 10). Ambos os casos são exemplificados em Yoon, Kwon e Bae (2007).

Figura 10 – Exemplo 2 K-Means



Fonte: Autoria própria.

O k-means como técnica de detecção de *outlier* é um método simples e eficiente e sua execução tem um baixo custo computacional, porém o k-means não permite o desenvolvimento de um conjunto ideal de *clusters* e, para obter resultados efetivos, deve-se decidir sobre os *clusters* antes, sendo que a escolha da quantidade de *clusters* é essencial para otimização do resultado. Existem várias propostas na literatura para escolher o melhor valor para K, como o método do cotovelo, silhueta validação cruzada e outros (KODINARIYA; MAKWANA, 2013).

2.3.1.4 FUZZY C-MEANS

O Fuzzy C-Means (FCM) foi desenvolvido por Dunn (1973), e melhorado por Bezdek (1981) e é uma técnica de clustering em que cada ponto de dados pode pertencer a mais de um *cluster*. Como o K-means, o número desejado de *clusters* tem que ser pré-definido e posicionados para executar o Fuzzy c-means. Uma métrica também é usada para comparar todos os objetos ao *cluster*, mas a comparação é feita usando uma média ponderada que leva em consideração o grau de pertinência do objeto a cada *cluster*. No final do algoritmo, uma lista do grau estimado de pertinência do objeto a cada um dos *clusters* c é impressa e o objeto pode ser atribuído ao *cluster* para o qual o grau de associação é maior.

Ao contrário do método KM, o FCM é mais flexível porque atribuem cada objeto a diferentes *clusters* com diferentes graus de pertinência, conforme mencionado acima. Em outras palavras, enquanto a associação a um *cluster* é exatamente 0 ou 1 no KM, ela varia entre 0 e 1 no FCM.

Portanto, nos casos em que não podemos decidir facilmente que os objetos pertencem a apenas um cluster, especialmente com os conjuntos de dados com ruídos ou *outliers*, o FCM pode ser melhor que o KM. Por esse motivo. Matematicamente falando, FCM minimiza a função objetivo definida como:

$$J = \sum_{k=1}^{c} \sum_{i=1}^{N} (u_{ki})^2 \left| x_i - c_k \right|^2 ; \tag{2.6}$$

em que:

- u_{ki}é o grau de pertinência do objeto i em relação ao *cluster* k;
- $|x_i - c_k|^2$ é a distância ao quadrado entre o vetor de observações do objeto i e o vetor que representa o centróide (protótipo) do *cluster* μ; e,
- N é o número de observações da amostra.

Para o cálculo do grau de pertinência (u_{ki}) deve-se utilizar valores para o fuzzificador maior que 1. Segundo Babuška (2012), Höppner et al. (1999) e Pal e Bezdek (1995) o paramêtro ideal é geralmente 2 e é o expoente fuzzy que determina o grau de nebulosidade da partição final ou, em outras palavras, o grau de sobreposição entre os grupos.

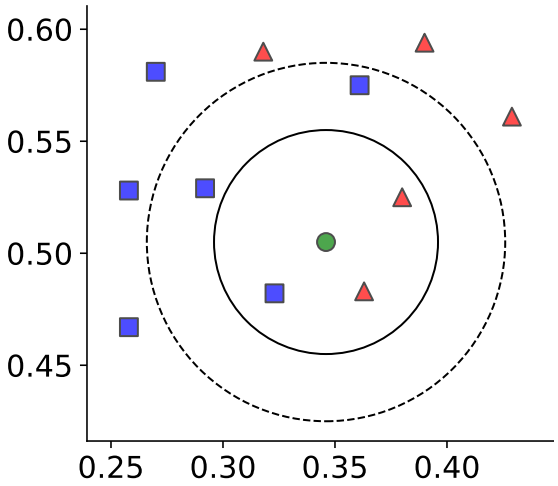
Quando utilizado como técnica de detecção de *outlier*, o FCM funciona da mesma forma que o KM, acrescentando o parâmetro fuzzificador. O parâmetro fuzzificador m, quando se aproxima de 1 tendem a tornar os *clusters* mais densos, pois a centroide tende a se posicionar mais próxima da concentração dos objetos, porém quando m tende para o infinito, ocorre o contrário, os centroides tendem a se posicionar de maneira mais difusa.

Desta forma, com um melhor controle do posicionamento dos centroides é possível isolar os *outliers*, opção esta que não existe controle no k-means.

2.3.1.5 K-NEAREST NEIGHBORS

O K-Nearest Neighbors (kNN) é um método de classificação não paramétrico, de abordagem supervisionada, proposto por Cover e Hart (1967), em que o vizinho mais próximo é calculado com base no valor de k, que especifica quantos vizinhos mais próximos devem ser considerados para definir a classe de um ponto de dados de amostra.

Figura 11 – Exemplo KNN.



Fonte: Autoria própria.

Por exemplo, da ilustração na Figura 11 pode-se observar que a amostra de teste (ponto verde) deve ser classificada em quadrados azuis ou em triângulos vermelhos. Se $k = 3$, ele é atribuído aos triângulos vermelhos porque existem 2 triângulos e apenas 1 quadrado dentro do círculo interno. Se $k = 5$, ele é atribuído aos quadrados azuis (3 quadrados vs. 2 triângulos dentro do círculo externo). Como esse algoritmo depende da distância para classificação, a normalização dos dados de treinamento pode melhorar drasticamente sua precisão (PIRYONESI; EL-DIRABY, 2020).

A vantagem do KNN como método de classificação é que o seu treinamento é rápido, de simples implementação e robusto, porém pode ficar tendencioso em decorrência da escolha do K e ser facilmente enganado por atributos irrelevantes (BHATIA et al., 2010).

2.3.1.6 AUTOENCODERS

Inicialmente, pode-se dizer que um autoencoder é um tipo de rede neural artificial usada para aprender codificações de dados eficientes de maneira supervisionada (KRAMER,

1991) e suas primeiras aplicações remotam a década de 1980 (SCHMIDHUBER, 2015).

Um autoencoder consiste em um encoder que aprende a representação latente z dos dados e um decoder que reconstrói os dados $d(z)$ com base na idéia de como os dados estão estruturados, ou seja, o objetivo da saída do autoencoder é "copiar"as entradas.

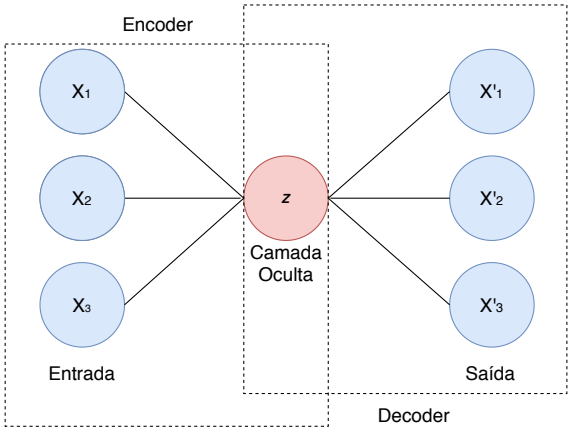
De acordo com Jia e Zhang (2018), a topologia de um autoencoder ilustrado na Figura 12, pode ser modelado pela Equação (2.7). Em que $f(\cdot)$ e $g(\cdot)$ são as funções de encoder e decoder, w e b são os parâmetros destes referente a pesos e *bias*.

$$\begin{cases} z = f(w_e, b_e; X) \\ X' = g(w_d, b_d; z) \end{cases} \tag{2.7}$$

Assim como os outros modelos, treinar um autoencoder significa minimizar uma função objetivo, neste caso representada pela Equação (2.8), onde $\theta = (w_e, b_e; w_d, b_d)$.

$$J(\theta) = \frac{1}{n} \sum_{i=1}^{n} \left\| x_i - x_i^{'} \right\|_2^2 \tag{2.8}$$

Figura 12 – Exemplo autoencoder.



Fonte: Autoria própria.

Quando usado para detecção de *outliers*, tendo em vista que no geral as camadas ocultas dos autoencoders consistem em muito menos neurônios, para reconstruir a entrada o mais verissímil possível, os pesos ocultos capturam os parâmetros mais representativos dos dados de entrada originais e ignoram as especificidades detalhadas dos dados de entrada, como *outliers*. Em outras palavras, dados normais são muito mais fáceis de reconstruir do que *outliers*.

Com base no que foi dito, quanto maior a diferença entre a saída (ou seja, a entrada reconstruída) e a entrada original, maior a probabilidade que os dados de entrada correspondentes sejam uma anomalia (XIA et al., 2015).

Para construção de um modelo autoencoder é necessária a determinação de alguns parâmetros. Inicialmente é necessário criar uma **arquitetura**, e geralmente os autoencoders são construídos através da conhecida *Multilayer Perceptron* (MLP), onde os neurônios são dispostos em camadas e cada camada tem todos os seus membros interligados com os membros da camada seguinte. Nos autencoders, temos a simetria do encoder e decoder, então a MLP tem que ser construída respeitando essa limitação, sendo normalmente disposto como ilustrado na Figura 12, uma camada de entrada, uma cada oculta e uma camada de saída.

Outro parâmetro de necessária definição é a função de ativação que restringe a saída dos neurônios. Existem diversas funções de ativação na literatura (SHARMA, 2017) e iremos aqui explanar apenas sobre duas delas: H*yperbolic Tangent Function* (Tanh) e *Rectified Linear Unit Function* (ReLU).

A função de ativação TanH é contínua e diferenciável, os valores situam-se no intervalo de -1 a 1. A TanH é bastante usada como função de ativação, pois possui gradientes que não estão restritos a variar em uma determinada direção e também, é centrado em zero (KARLIK; OLGAC, 2011).

Por outro lado, a função de ativação ReLU possuiu uma função derivativa, tornando-a não linear e por por isso, permitindo a sua utilização em *deep learning* através do *backpropagation*. Uma vantagem da ReLU é que nem todos os neurônios são ativados ao mesmo tempo. Isso implica que um neurônio será desativado apenas quando a saída da transformação linear for zero, tornando esta função mais eficiente que as outras, com um menor custo computacional e rápida convergência (SHARMA, 2017).

Tendo sido mencionado o *backpropagation*, vale explicar que este é um algoritmo de treinamento baseado em *gradient descent*, que visa minimizar uma função. A aplicação deste algorítmo à função de erro quadrática permite encontrar pesos que atingem valores cada vez menores, tornando o modelo gradualmente mais preciso. Mais informações podem ser encontradas em Hecht-Nielsen (1992).

Ainda, os hiperparâmetros da rede devem ser definidos e estes são: A quantidade de épocas, função custo, otimizador do *gradient descent* e a taxa de aprendizado. A explanação sobre estas variáveis será deixada a cargo de Junior (2019).

Por fim, existem algumas variantes dos autoencoders (JIA; ZHANG, 2018) e neste trabalho iremos abordar, além do autoencoder padrão a variante autoencoder esparso, que modifica o autoencoder padrão, incluindo-se restrições esparsas. Em *deep learning*, estas restrições são introduzidas através de regularizadores se utilizando de uma técnica que faz pequenas modificações no algoritmo de aprendizado, de modo que o modelo se generalize melhor, ou seja, evitando o *over-fitting*.

2.3.1.7 TYPICALITY AND ECCENTRICITY DATA ANALYTICS - TEDA

Uma estrutura de análise de dados baseada na tipicidade e excentricidade (TEDA) foi introduzida por Angelov (2013) com o objetivo de generalizar e evitar as suposições bem conhecidas, mas muito restritivas, nas quais a abordagem estatística tradicional e a teoria da probabilidade se baseiam, a saber:

- a) independência das amostras de dados individuais (observações) umas das outras;
- b) grande (teoricamente, infinito) número de amostras de dados;
- c) pressuposto prévio da distribuição ou kernel (na maioria das vezes, normal/Gaussiano).

De fato, a teoria da probabilidade foi desenvolvida com processos puramente aleatórios em mente, que são usados como exemplos em qualquer livro, como jogos e jogos de azar. É perfeitamente adequado para descrever esses processos e variáveis puramente aleatórias. No entanto, quando temos em mãos processos reais (por exemplo, climáticos, econômicos, físicos, biológicos, sociais, psicológicos, etc.) que não são puramente aleatórios, temos dependência entre amostras, não distribuições normais/Gaussianas e não um número infinito de observações.

Ou seja, as premissas acima mencionadas estão sendo violadas ou ignoradas. Este é um problema bem conhecido que é resolvido geralmente por mistura de distribuições normais, ortogonalizações como PCA, etc.

Para esses casos supõe-se que sistemas geralmente estão *offline* e exigem que todos os dados sejam conhecidos com antecedência e levam a novos recursos/variáveis de entrada não reais, mas derivados. Outras alternativas incluem métodos como por exemplo filtragem de partículas (GORDON; RISTIC; ARULAMPALAM, 2004) e aprendizado teórico da informação (PRINCIPE et al., 2000) e (PRINCIPE, 2010) que não são paramétricos. No entanto, eles ainda assumem um kernel gaussiano ou outro para representar a vizinhança ao redor da amostra de dados (o campo). O TEDA, pelo contrário, não requer nenhuma suposição de kernel. O problema com a falta de quantidade infinita de dados e o fato de que teoricamente é um requisito geralmente está sendo completamente ignorado.

Para evitar todos esses problemas, foi recentemente introduzida a nova abordagem TEDA (ANGELOV, 2013) que pode ser caracterizada da seguinte forma:

- É inteiramente baseado nos dados e sua distribuição mútua no espaço de dados;
- Não são feitas suposições prévias;
- Não são necessários kernels;

- Nenhum limite e parâmetros específicos do usuário ou do problema precisam ser pré-especificados;
- Não requer independência das amostras de dados individuais (observações) – pelo contrário, TEDA se baseia (faz uso) de sua dependência mútua;
- Também não requer um número infinito de observações e pode trabalhar com apenas 3 amostras de dados.

O TEDA pode ser visto como uma estrutura estatística alternativa que pode trabalhar eficientemente com qualquer dado, exceto processos puramente aleatórios nos quais amostras de dados individuais (observações) são completamente independentes umas das outras. Para tais casos (por exemplo, jogos de azar, jogos, ruído branco, etc.), a teoria tradicional da probabilidade é a melhor ferramenta a ser usada, pois foi desenvolvida com esses processos em mente e amadureceu ao longo de três ou mais séculos (COSTA et al., 2015).

No entanto, para processos de dados reais - que são a maioria dos casos na natureza, o TEDA é mais adequado, pois não se baseia em suposições que não são satisfeitas por tais processos. Um exemplo de problema real muito simples de uma dimensão (1D) de análise climática será dado para o qual a análise estatística tradicional falha em identificar a anomalia, enquanto o TEDA é muito eficiente.

O TEDA é baseado em várias novas quantidades que são todas baseadas na análise de proximidade/semelhança no espaço de dados. Estes não são exatamente os mesmos que a densidade usada em estatísticas e outras áreas, nem o mesmo que entropia. O termo tipicidade usado no TEDA é um pouco semelhante ao termo recentemente introduzido com o mesmo nome em (OSHERSON; SMITH, 1997) para descrever "até que ponto os objetos são 'bons exemplos' de um conceito".

Vamos supor um espaço de dados $\mathfrak{X} \in \mathrm{R}^2$, que consiste em n amostras de dados dimensionais. Para este espaço, podemos definir uma distância $d(x, y)$. Então, vamos considerar as amostras de dados como uma sequência ordenada $\{x_1, x_2, ..., x_k, ...\}$ $x_i \in \mathrm{R}^2$ $i \in \mathrm{N}$, em que o índice k pode ter o significado físico de instante de tempo quando a amostra de dados chegou.

O TEDA introduz as seguintes características válidas para cada amostra de dado:

a) *proximidade acumulada*, π de um ponto particular $\mathfrak{X} \in \mathrm{R}^2$, para todos as amostras de dados restantes até $k^{ésimo}$ para um dado $j^{ésimo}$ $(j > 1)$ amostra de dado calculado quando k $(k > 1)$ amostras de dados estão disponíveis (ANGELOV, 2013):

$$\pi_k(x_j) = \pi_{jk} = \sum_{i=1}^{k} d_{ij} k > 1 \tag{2.9}$$

em que: d_{ij} denota a medida de distância entre as amostras de dados, x_i e x_j. A distância pode ser Euclidiana, Mahabolis, Cosseno, etc.

b) *excentricidade* da $j^{ésima}$ amostra de dados calculada quando k ($k > 2$) amostras de dados estão disponíveis e não tem o mesmo valor (ANGELOV, 2013):

$$\xi_{jk} = \frac{2\pi_{jk}^{k}}{\sum_{i=1}^{k} \pi_{ik}} \sum_{i=1}^{k} \pi_{ik} > 0 \quad k > 2 \tag{2.10}$$

c) *tipicidade* da $j^{ésimo}$ ($j > 1$) amostra calculada quando k ($k > 2$) amostras não idênticas estão disponíveis:

$$\tau_{jk} = 1 - \xi_{jk} \quad k > 2 \quad \sum_{i=1}^{k} \pi_{ik} > 0 \tag{2.11}$$

Assim, tanto a *excentricidade* ξ quanto a *tipicidade* τ estão entre 0 e 1 e suas contrapartes normalizadas $\zeta = \xi/2$ e $t = \tau/(k-2)$ somam 1 (ANGELOV, 2013). Eles também podem ser definidos tanto localmente quanto globalmente da mesma forma que a proximidade acumulada, π. A *tipicidade* se assemelha as *membership functions* de um sistema $Fuzzy$, no entanto, eles diferem pois a tipicidade não requer suposições anteriores. Ou seja, a tipicidade representa tanto o padrão de distribuição espacial quanto a frequência de ocorrência de uma amostra de dados simultaneamente e por amostra de dados, não "em média".

Além disso, ainda de acordo com Angelov (2014) em casos que seja preciso uma aplicação online, as variáveis podem ser calculadas recursivamente para certos tipos de distâncias. Por exemplo, se usar Distância euclidiana tem-se:

$$\mu_k = \frac{k-1}{k}\mu_{k-1} + \frac{1}{k}x_k \;\; \mu_1 = x_1 \tag{2.12}$$

A forma com que o TEDA lida com a análise de dados em relação a teoria da probabilidade tradicional é especial, pois esta é baseada em diferentes conjuntos de suposições, enquanto a TEDA não requer nenhuma suposição a princípio, além do fato de que o processo observado não é um processo aleatório puro. No entanto, a chamada análise "$n\sigma$" para anomalias pode ser comprovadamente aplicável em ambas as análises, TEDA e estatística.

Isso pode ser facilmente comprovado a partir da definição da excentricidade em TEDA (usando distância euclidiana (2.12) e a definição de variância (2.13) descrita na 2.14.

$$\sigma_k^2 = \sum_{i=1}^{k} \frac{(x_i - \mu_k)^T (x_i - \mu_k)}{k} \tag{2.13}$$

$$\xi_k = \frac{1}{k} + \frac{(\mu_k - x_k)^T(\mu_k - x_k)}{k\sigma_k^2} \tag{2.14}$$

Para problemas de *Big Data* quando o valor de k pode se tornar enorme e usar a proximidade acumulada π_k pode levar a problemas de limitação de hardware, pode-se usar Equação 2.14 em relação à Equação 2.12 e atualização do desvio padrão que é semelhante ao 2.12 (ANGELOV, 2012):

$$\sigma_k^2 = \frac{k-1}{k}\sigma_{k-1}^2 + \frac{1}{k-1}\left\|x_k - \mu_k\right\|^2 \quad \sigma_1^2 = 0 \tag{2.15}$$

Então, finalmente Angelov (2014) explica que a condição que fornece exatamente o mesmo resultado como a inequação de Chebyshev pode ser dada como:

$$\zeta > \frac{n^2+1}{2k} \tag{2.16}$$

A importância deste resultado não pode ser subestimada. Ela demonstra que para uma distribuição arbitrária e com apenas três amostras de dados é possível analisar a excentricidade normalizada por amostra de dado, e se essa exceder a condição, então temos exatamente o mesmo resultado que a inequação de Chebyshev forneceria.

2.3.2 QUADRO COMPARATIVO ENTRE OS ALGORITMOS DESCRITOS

Alguns pontos são importantes de trazer quando da funcionalidade dos algoritmos abordados neste trabalho. Classificamos eles em suas capacidades de ser não supervisionado, necessidade de treinamento prévio, parametrização, se funciona *online* (*stream* de dados) e se consegue se adaptar ao *concept drift*.

Tabela 1 – Classificação dos algoritmos.

Modelo	Não supervisionado	Não necessita treinamento	Não Parametrizável	Algoritmo online	Lida com concept drift
Z-Score	SIM	MÉDIA DO CONJUNTO	SIM	NÃO	NÃO
KM+KNN	SIM	NÃO	NÃO	NÃO	NÃO
FCM+KNN	SIM	NÃO	NÃO	NÃO	NÃO
Autoencoder	SIM	NÃO	NÃO	NÃO	NÃO
TEDA	SIM	SIM	SIM	SIM	SIM

A classificação da Tabela 1 é importante pois o resultado de performance pode variar de acordo com algumas capacidades, sendo em algumas situações, ser necessário fazer *tradeoff* na hora da escolha do modelo por motivo de restrições relacionada a classificação.

3 METODOLOGIA

Neste capítulo será apresentada a metodologia do trabalho, onde realizar-se-a explicação minuciosa de todos os recursos e métodos aplicados em busca de atingir o objetivo deste trabalho.

3.1 MATERIAIS

3.1.1 LINGUAGEM DE PROGRAMAÇÃO PYTHON

Na literatura existem várias linguagens de programação, entretanto, foi escolhida a linguagem de programação Python para realizar a criação dos algoritmos necessários para o andamento da pesquisa. Python é uma linguagem de programação interpretada, de alto nível e de uso geral. Criado por Guido Van Rossum e lançado pela primeira vez em 1991, a filosofia de design do Python enfatiza a legibilidade do código com seu uso notável de espaço em branco significativo. Suas construções de linguagem e abordagem orientada a objetos têm como objetivo ajudar os programadores a escrever código claro e lógico para projetos de pequena e grande escala (KUHLMAN, 2012).

Python é digitado dinamicamente e suporta vários paradigmas de programação, incluindo programação estruturada (principalmente procedural), orientada a objetos e funcional. O Python é frequentemente descrito como uma linguagem "baterias incluídas"devido à sua abrangente biblioteca padrão (ROSSUM et al., 2007).

3.1.1.1 MÓDULOS PYTHON

Para facilitar a pesquisa e evolução do trabalho, foram utilizados alguns módulos (bibliotecas) Python. É importante citar alguns destes módulos e suas especificações.

O **Scikit-learn**, (GÉRON, 2022), é uma biblioteca de aprendizado de máquina de *software* livre para a linguagem de programação Python. Em que possui vários algoritmos de classificação, regressão e *clustering*, incluindo máquinas de vetores de suporte, *random-forest*, k-means e DBSCAN, foi projetado para interoperar com as bibliotecas numéricas e científicas Python NumPy e SciPy. Esta biblioteca é utilizada na pesquisa como auxílio para construção do modelo K-Means de clusterização e o K-Nearest Neighbors para classificação.

O **fuzzy-c-means** é um módulo Python que implementa o algoritmo de clustering Fuzzy C-means e foi utilizado na pesquisa com o propósito de construir os modelos C-means.

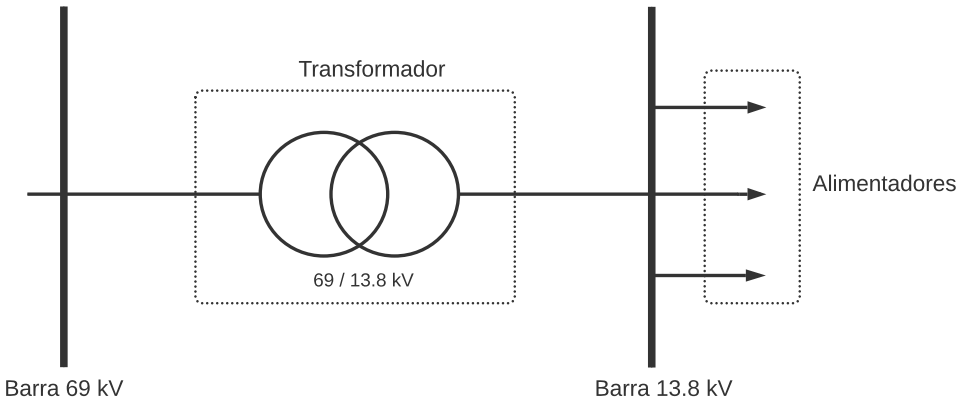
O **Keras** é uma biblioteca de rede neural de código aberto escrita em Python. De acordo com (GÉRON, 2022) o *keras* foi projetado para permitir experimentação rápida com redes neurais profundas, ele se concentra em ser fácil de usar, modular e extensível, funcionando em cima da plataforma ***tensorflow*** de *deep-learning* do google. Utilizamos o Keras para implementação dos modelos de deteção de *outliers* baseados em *deep-learning* que foram apresentados neste trabalho.

Outras bibliotecas python foram amplamente usadas como **NumPy** (matrizes), **Pandas** (dados).

3.1.2 CONJUNTO DE DADOS DE SUPORTE

Para realizar algumas análises iniciais, foram utilizados dados medidos de potência ativa numa frequência de 1 amostra a cada 15 minutos. As subestações são compostas geralmente por um padrão conforme o diagrama unifilar presente na Figura 13.

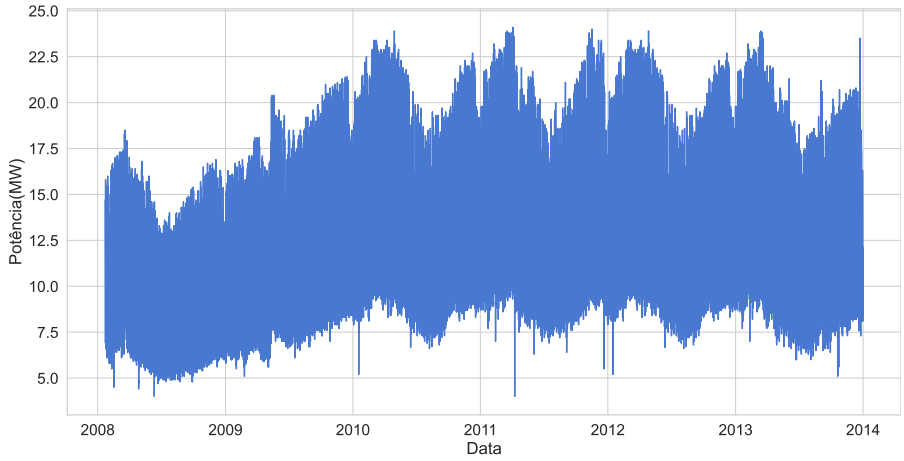
Figura 13 – Diagrama unifilar resumido de uma subestação.



Fonte: Autoria própria.

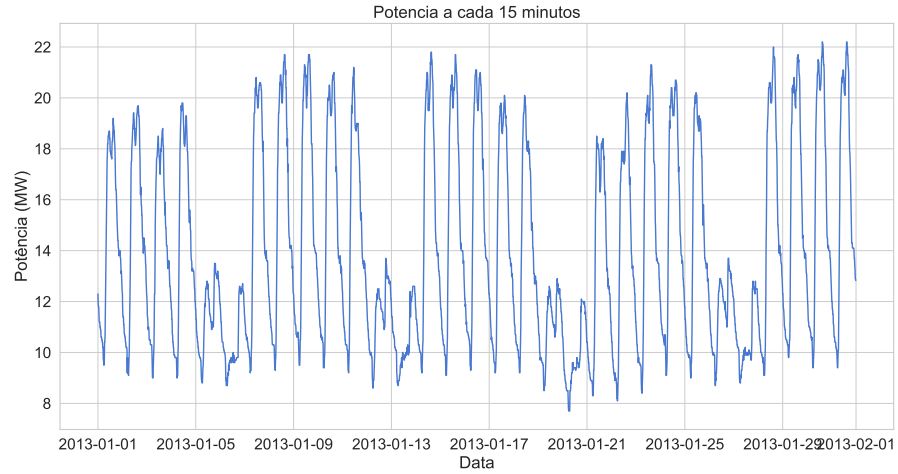
Neste trabalho, utiliza-se especificamente dados da subestação João Pessoa (SEJPS) localizada na cidade de mesmo nome, esses dados são referentes as medições realizadas na barra de 69 kV, do período de janeiro de 2008 a dezembro de 2013 (Figura 14). Foram selecionados 31 dias de medição do conjunto de dados da SEJPS compreendendo o dia 01 de janeiro de 2013 até o dia 31 de janeiro de 2013 inclusive. Estes dados são ilustrados na Figura 15.

Figura 14 – Série temporal completa.



Fonte: Autoria própria.

Figura 15 – Dados para treino e validação.

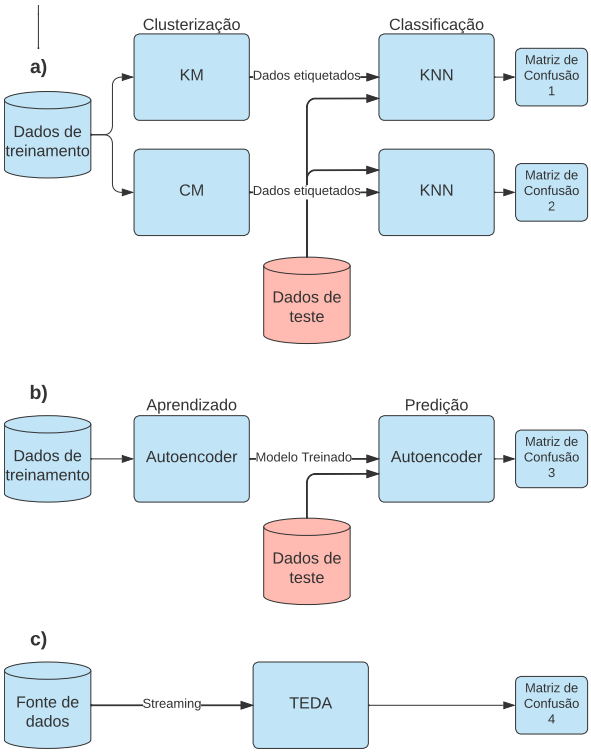


Fonte: Autoria própria.

3.2 MÉTODOS

Neste trabalho serão abordados métodos não supervisionados, a exemplo do K-Means, Fuzzy C-Means e TEDA, além de métodos supervisionados, como o Autoencoder, que serão utilizados como base para processo de detecção de *outliers*, conforme apresentado no diagrama de fluxo da Figura 16.

Figura 16 – Processo de detecção de outlier no KM, CM e Autoencoder e TEDA.



Fonte: Autoria própria.

Da Figura 16 a) é possível verificar que os métodos do K-Means e Fuzzy C-Means passam naturalmente pelo processo de clusterização, onde recebem as etiquetas de classes, no qual, para avaliação posterior, se utiliza o KNN com o conjunto de dados de teste, finalizando com a matriz de confusão. Tudo se resume a um processo de clusterização e classificação.

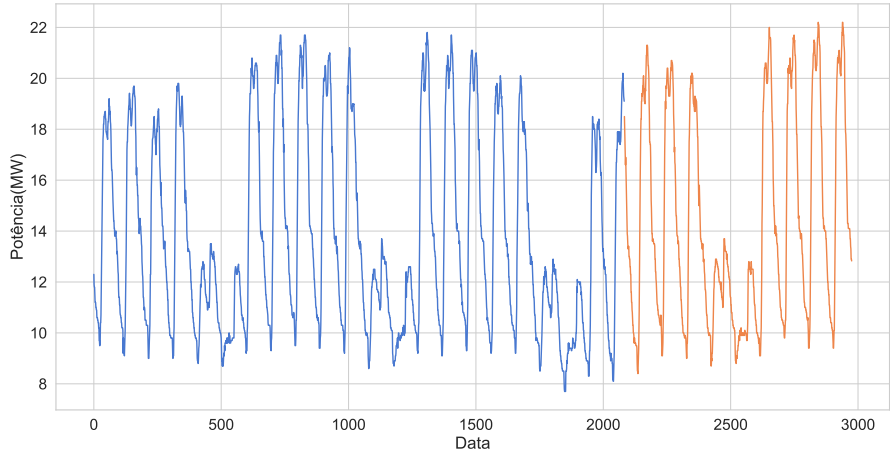
Para o Autoencoder (Figura 16 b), por se tratar de um método supervisionado, dados etiquetados de treinamento passam por um processo de aprendizado e em seguida, é realizada a classificação dos itens de teste utilizando-se do modelo treinado.

Por fim (Figura 16 c)), o fluxo do TEDA onde os dados podem vir em *stream*, sem necessidade de treinamento, clusterização ou parametrização, realizando uma detecção online dos *outliers*.

3.2.1 MODIFICAÇÃO DOS DADOS PARA TREINO E AVALIAÇÃO

Este conjunto representa 2976 pontos de medição, coletados em um intervalo de 15 minutos entre eles. Separou-se os dados para treino e validação, onde 70% dos dados foram separados para treino e 30% para validação (Figura 17).

Figura 17 – Dados separados treino/validação.

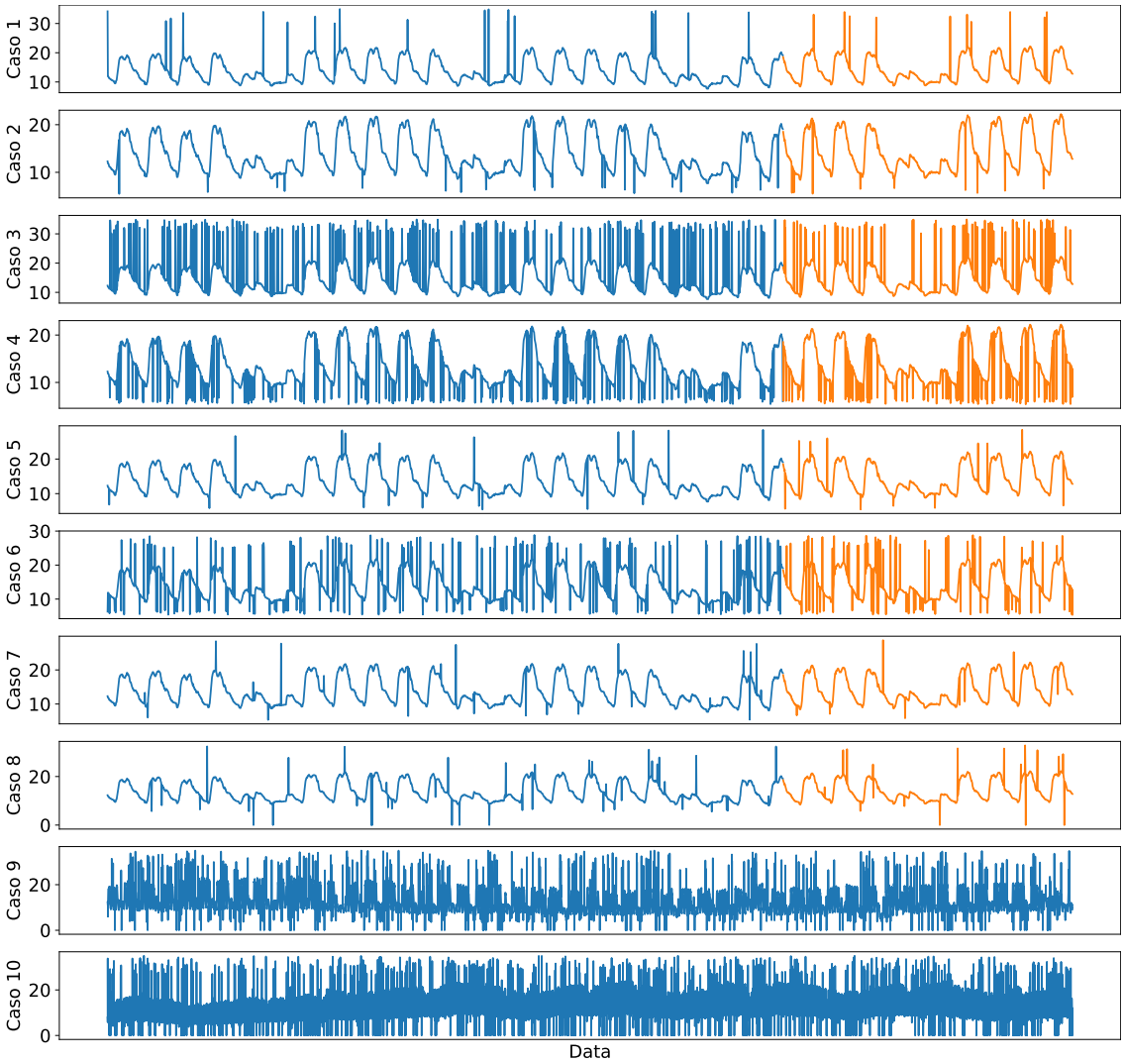


Fonte: Autoria própria.

Em cima dos dados de treino e validação foram inseridos *outliers* aleatoriamente para realização dos testes, formando oito casos de estudo, conforme ilustrado na Figura 18 e definidos da seguinte forma:

- **Caso 1:** 30 *outliers* do tipo picos inseridos (1% do conjunto);
- **Caso 2:** 30 *outliers* do tipo vales inseridos (1% do conjunto);
- **Caso 3:** 298 *outliers* do tipo picos inseridos (10% do conjunto);
- **Caso 4:** 298 *outliers* do tipo vales inseridos (10% do conjunto);
- **Caso 5:** 30 *outliers* do tipo vales e picos inseridos (1% do conjunto);
- **Caso 6:** 298 *outliers* do tipo vales e picos inseridos (10% do conjunto);
- **Caso 7:** 30 *outliers* do tipo vales e picos com *outliers* contextuais inseridos (1% do conjunto);
- **Caso 8:** 60 *outliers* do tipo vales e picos com *outliers* contextuais inseridos (2% do conjunto);
- **Caso 9:** 716 *outliers* do tipo vales e picos com *outliers* contextuais inseridos (2% do conjunto);
- **Caso 10:** 1200 *outliers* do tipo vales e picos com *outliers* contextuais inseridos (0.57% do conjunto).

Figura 18 – Dados com *outliers* inseridos.



Fonte: Autoria própria.

3.2.2 MÉTRICAS DE AVALIAÇÃO

Para possibilitar uma análise comparativa entre as diferentes técnicas levantadas neste trabalho, algumas métricas de avaliação foram levadas em consideração.

3.2.2.1 ERRO MÉDIO QUADRÁTICO

O erro médio quadrático (MSE) de um estimador é a média dos quadrados dos erros, ou seja, a diferença quadrática média entre os valores estimados e o valor atual, representada pela Equação (3.1):

$$MSE = \frac{1}{n}\sum_{t-1}^{n}(A_t - P_t)^2 \tag{3.1}$$

onde A_t é o valor atual e P_t é o valor previsto.

3.2.2.2 ERRO PERCENTUAL ABSOLUTO MÉDIO

Já o erro percentual absoluto médio (MAPE), é uma medida da precisão de um método de previsão em estatísticas. Geralmente expressa a precisão como uma proporção definida pela Equação (3.2):

$$MAPE = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{A_t - P_t}{A_t}\right| * 100\% \tag{3.2}$$

em que: A_t é o valor atual e P_t é o valor previsto. O valor absoluto neste cálculo é somado para cada ponto previsto no tempo e dividido pelo número de pontos ajustados n. Multiplicar por 100% faz com que seja um erro em porcentagem.

3.2.2.3 MATRIZ DE CONFUSÃO

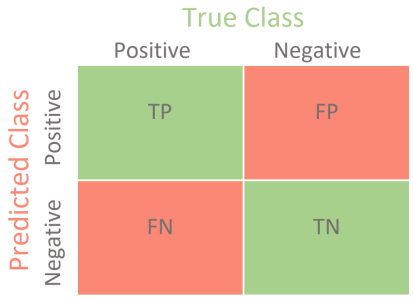
No problema de classificação estatística, a matriz de confusão, também conhecida como matriz de erros (STEHMAN, 1997), é um layout de tabela específico que permite a visualização do desempenho de um algoritmo, tipicamente um aprendizado supervisionado (em aprendizado não supervisionado, geralmente é chamado de matriz correspondente).

Dentro da matriz de confusão existem quadrantes que representam quatro parâmetros, e com esses parâmetros é possível realizar cálculos de alguns tipos de métricas de erro. Especificando os parâmetros, temos:

- Verdadeiros Positivos (TP) - Esses são os valores positivos previstos corretamente, o que significa que o valor da classe real é sim e o valor da classe prevista também é sim. Por exemplo, se o valor real da classe indicar que o dado não é um *outlier* e a classe prevista informa a mesma coisa.
- Verdadeiros Negativos (TN) - Esses são os valores negativos previstos corretamente, o que significa que o valor da classe real é não e o valor da classe prevista também é não. Por exemplo, se a classe real indicar que é um *outlier* e a classe prevista diz a mesma coisa.
- Falsos Positivos (FP) - Quando a classe real é não e a classe prevista é sim. Por exemplo, se a classe real diz que é um *outlier*, mas a classe prevista informa que não.
- Falsos Negativos (FN) - Quando a classe real é sim, mas a classe prevista não. Por exemplo, se o valor real da classe indicar que o ponto não é um *outlier* e a classe prevista informa é um *outlier*.

A Figura 19 retrata uma matriz de confusão, onde o quadrante superior esquerdo representa os TP, o quadrante superior direito representa os FN, o quadrante inferior esquerdo representa os FP e o último quadrante representa os TN.

Figura 19 – Matriz de confusão.



De posse da matriz de confusão é possível calcular uma métrica de erro intitulada coeficiente de correlação de Matthews (MCC), que é usado no aprendizado de máquina como uma medida da qualidade das classificações binárias (de duas classes), introduzidas pelo bioquímico Brian W. Matthews (MATTHEWS, 1975). O MCC pode ser calculado diretamente da matriz de confusão usando a equação 3.3.

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TP + FN)}} \tag{3.3}$$

O coeficiente leva em consideração positivos e negativos verdadeiros e falsos e geralmente é considerado uma medida equilibrada que pode ser usada mesmo que as classes sejam de tamanhos muito diferentes (BOUGHORBEL; JARRAY; EL-ANBARI, 2017).

Não será usado o conhecido F1-score neste trabalho como métrica de avaliação pois, segundo Chicco e Jurman (2020), o *score* F1 é menos verdadeiro e informativo que MCC na classificação da avaliação binária.

Importante frisar que para o MSE e MAPE, quanto menor o valor, melhor (são métricas de erro) e para o MCC, que vem da matriz de confusão, quanto maior melhor, sendo representado a precisão na identificação dos *outliers*.

3.2.3 PARÂMETROS INTERNOS DOS MODELOS

Nesta subseção serão discutidos e determinados alguns parâmetros necessários para a construção dos modelos de detecção de *outliers* utilizando as técnicas anteriormente mencionadas.

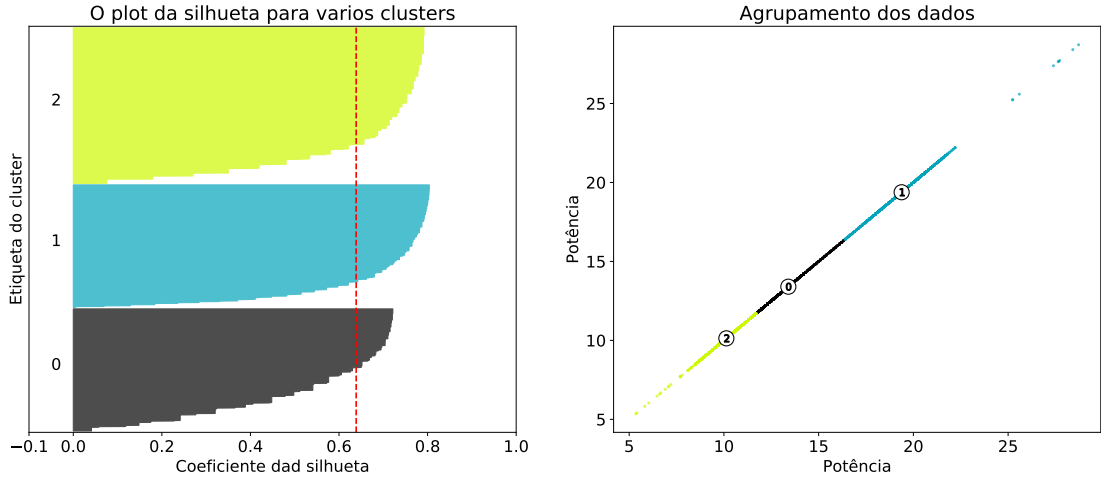
3.2.4 NÚMERO DE CLUSTERS PARA K-MEANS E FUZZY C-MEANS

Para construção de um modelo baseado em KM e FCM, é necessária a determinação de um parâmetro que determina quantidade de *clusters* no modelo. Conforme já mencionado

anteriormente, existem várias propostas na literatura para escolher a melhor quantidade de grupos (*clusters*), como o método do cotovelo, silhueta validação cruzada e outros (KODINARIYA; MAKWANA, 2013).

Para este trabalho, optou-se pelo método da silhueta para decidir o número de *clusters* que será utilizado. O coeficiente da silhueta é calculado usando a distância intra-cluster média (a) e a distância média do *cluster* mais próximo (b) para cada amostra. O coeficiente de silhueta para uma amostra é (b - a) / máximo (a, b). Para esclarecer, b é a distância entre uma amostra e o *cluster* mais próximo do qual a amostra não faz parte.

Figura 20 – Análise pelo método da silhueta com 3 *clusters*.



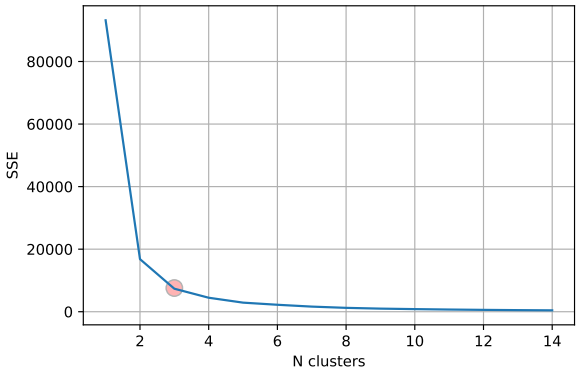
Fonte: Autoria própria.

Para uma boa escolha neste método, é importante que os *clusters* não tenham muita diferença no tamanho e que todos estejam ao menos na média do coeficiente da silhueta. Na Figura 20 apresenta-se bem essa situação, na qual o número ideal de *clusters* foi indicado como sendo k=3.

Para uma melhor validação, foi realizada uma comparação com o método do cotovelo, em que a ideia por trás deste é implementar o agrupamento em um determinado conjunto de dados para um intervalo de valores de k (n *clusters*, por exemplo, $k = 1$ a 10) e, para cada valor de k, calcular a soma dos erros ao quadrado (SSE). Assim, é traçado um gráfico K versus SSE, em que se o gráfico de linhas parecer um braço - um círculo vermelho na Figura 21, o "cotovelo"no braço é o valor ideal de k (número do *cluster*).

Verificado que o valor 'ideal' de número de *clusters* foi revelado como sendo 3 por meio de dois métodos, este será o valor utilizado como referência no trabalho.

Figura 21 – Análise pelo método do cotovelo.



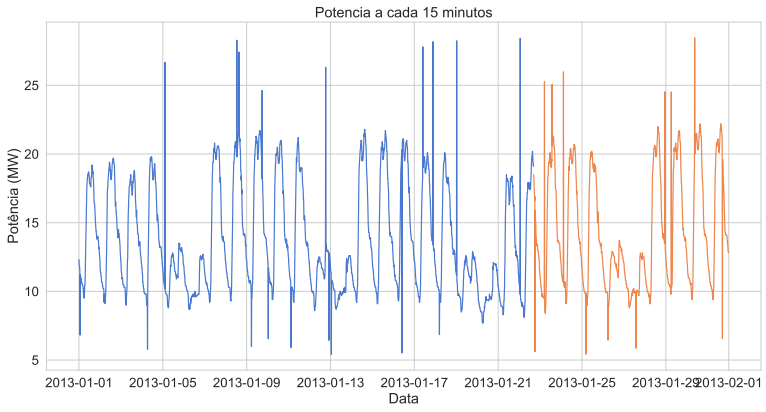
Fonte: Autoria própria.

3.2.5 MODELOS PARA DETECÇÃO E CORREÇÃO DE OUTLIERS

Em todos os modelos aqui exemplificados foi utilizada a regressão linear como técnica para correção dos dados identificados como *outliers* e esta subseção tem como objetivo apresentar e exemplificar os modelos de detecção destes.

Para exemplificar, será utilizado o conjunto do caso 5 (30 *outliers* do tipo vales e picos inseridos (1% do conjunto), conforme apresentado na Figura 22. As técnicas estão detalhadas de forma individual no Capítulo 2.

Figura 22 – Dados caso 5.



Fonte: Autoria própria.

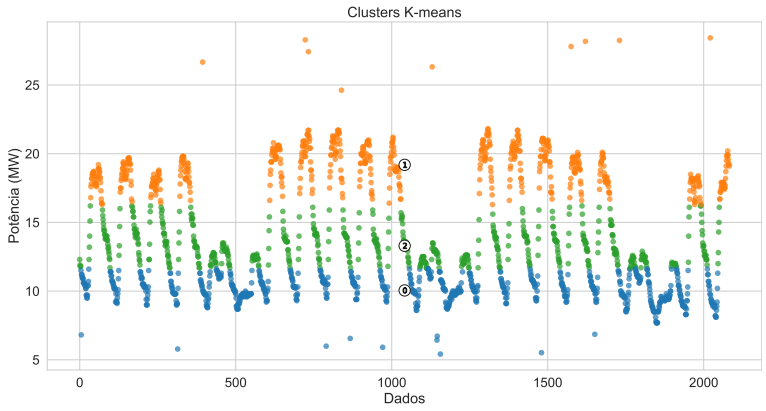
3.2.5.1 KM + KNN

Utilizando o algoritmo KM, realizou-se a etiquetagem dos dados de treinamento por meio de clusterização (Figura 23), sinalizando os *outliers* que possuíssem uma distância

Euclidiana superior a $\mu \times 4 \times \sigma$, conforme equação (3.4). O resultado é apresentado na Figura 24.

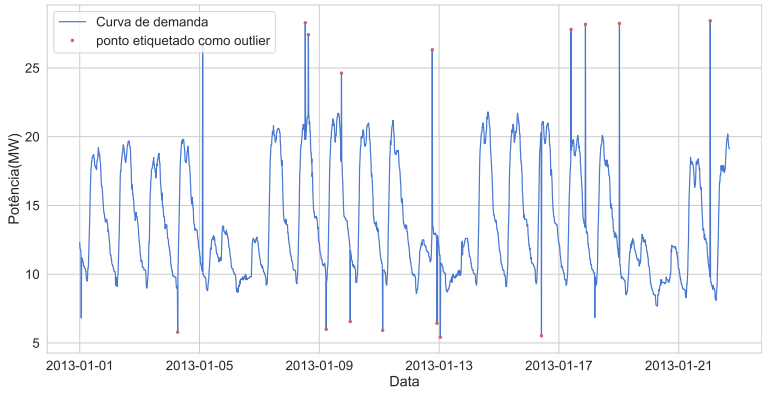
$$outlier = \|x - \mu_i\|^2 > (\mu \times 4 \times \sigma) \tag{3.4}$$

Figura 23 – K-Means: Clusterização.



Fonte: Autoria própria.

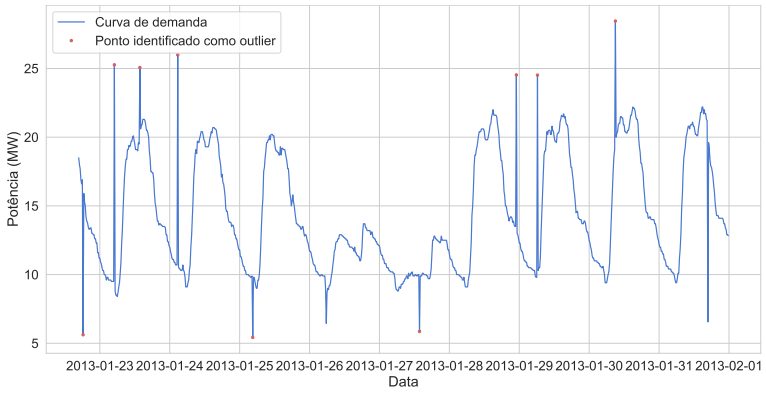
Figura 24 – K-Means: Etiquetagem.



Fonte: Autoria própria.

Em seguida, o algoritmo KNN foi alimentado com os dados etiquetados pelo KM e assim foi realizada a classificação dos dados de validação em *outliers* ou não (Figura 25). Para este trabalho foi escolhido o valor de $k = 5$ e a distância euclidiana como parâmetros do KNN pois estes resultaram Erro Percentual Absoluto Médio inferior a 10% nos casos testados. Importante mencionar que K precisa ser impar para não existir empates durante a classificação.

Figura 25 – K-Means: Classificação dos *outliers* com KNN.

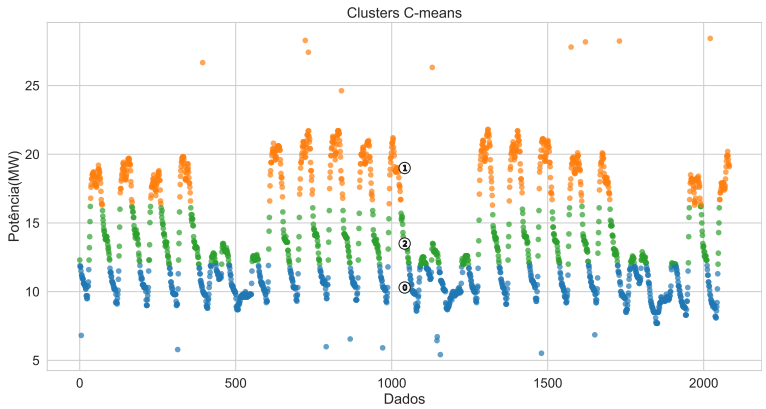


Fonte: Autoria própria.

3.2.5.2 FCM + KNN

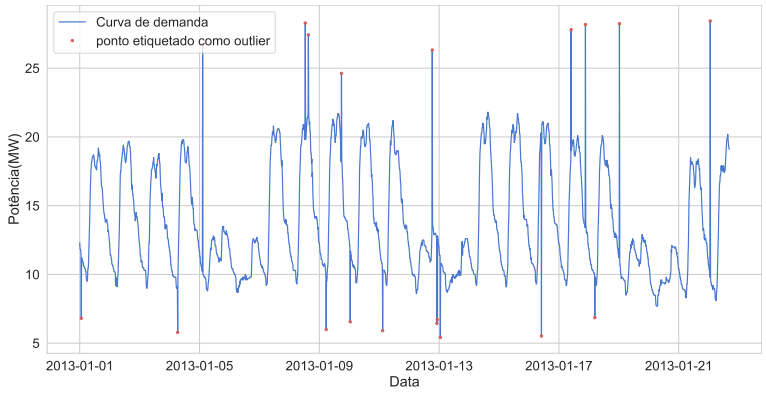
Utilizando o algoritmo FCM, realizou-se a etiquetagem dos dados de treinamento por meio de clusterização (Figura 23), sinalizando os *outliers* aqueles com uma distância euclidiana superior a $\mu \times 4 \times \sigma$ (mesmo caso do KM, equação (3.4)) e coeficiente de fuzzificação igual a 50 (Figura 27).

Figura 26 – Fuzzy C-Means: Clusterização.



Fonte: Autoria própria.

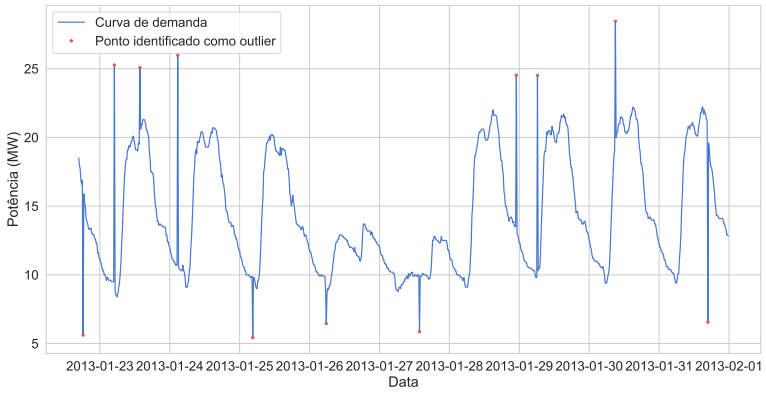
Figura 27 – Fuzzy C-Means: Etiquetagem.



Fonte: Autoria própria.

Em seguida, é utilizado o KNN para classificação dos *outliers* dos dados de validação de acordo com os dados etiquetados pelo C-Means (Figura 28). Para este trabalho foi escolhido o valor de $k = 5$ e a distância euclidiana como parâmetros do KNN pois estes resultaram Erro Percentual Absoluto Médio inferior a 10% nos casos testados.

Figura 28 – C-Means: Classificação dos *outliers* com KNN.

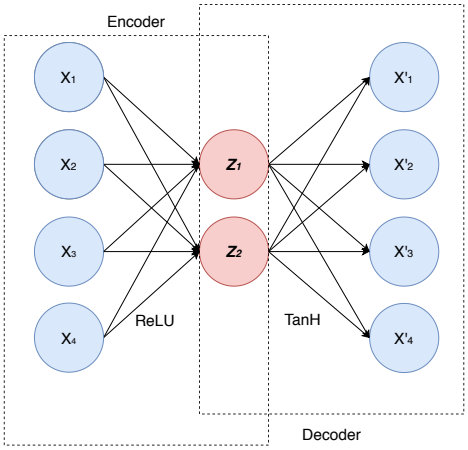


Fonte: Autoria própria.

3.2.5.3 AUTOENCODER

O modelo do autencoder apresentado possui quatro entradas. Essas entradas representam dados de potência em ordem cronológica $(X_{n-3}, X_{n-2}, X_{n-1}, X_n,)$ e estes são inseridos na rede por meio do codificador com função de ativação ReLu, uma camada oculta com dois neurônios e função e ativação TanH (escolhidos empiricamente), e quatro saídas por meio do decodificador, conforme Figura 29. Os parâmetros externos do modelo foram escolhidos conforme Tabela 2.

Figura 29 – Modelo Autoencoder.



Fonte: Autoria própria.

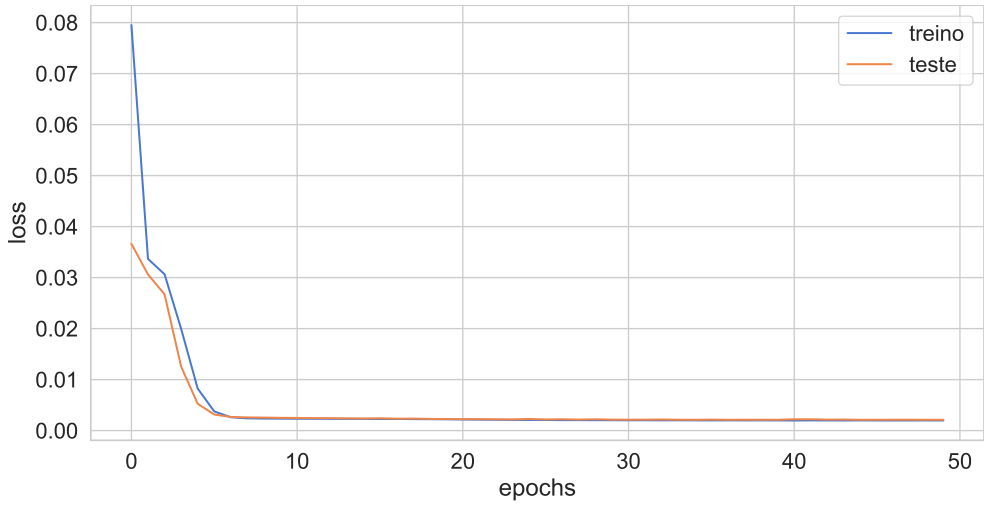
Tabela 2 – Autoencoder: Parâmetros externos de treinamento.

Hiperparâmetros	Valores
Épocas	50
Cost Function	MSE
Otimizador do *gradient descent*	Adam
Taxa de aprendizagem	0.01

O método consiste em realizar o treinamento do modelo utilizando os dados separados para este fim. Em seguida, utiliza-se os dados de validação como entrada do modelo e calcula-se o erro entre este e o conjunto de saída, utilizando do MSE, conforme Figura 30.

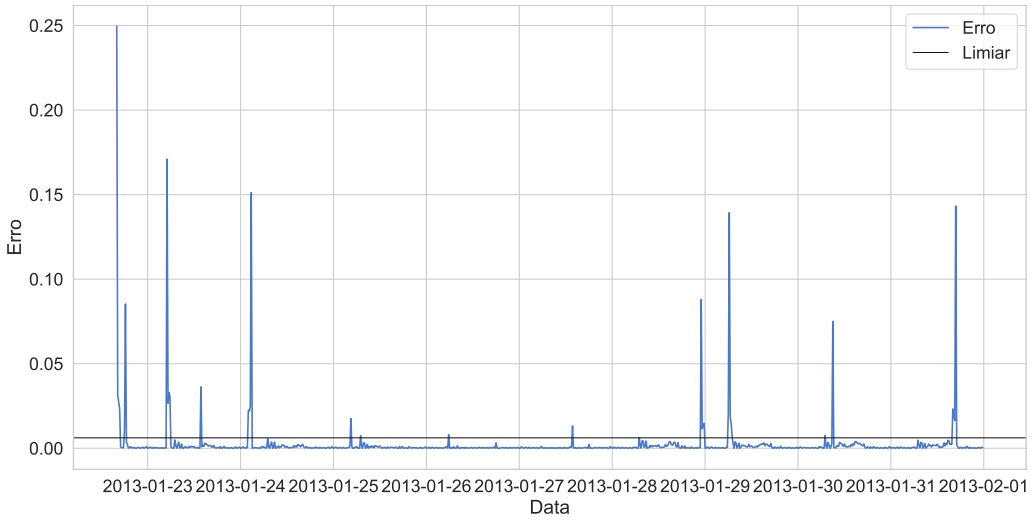
Para detecção dos *outliers*, é colocado um limiar na curva de erro (Figura 31), no qual os valores que excederem este limiar serão considerados *outliers*. O limiar foi configurado como sendo 5 vezes a média truncada em 1%.

Figura 30 – Autoencoder: Erro de treinamento.



Fonte: Autoria própria.

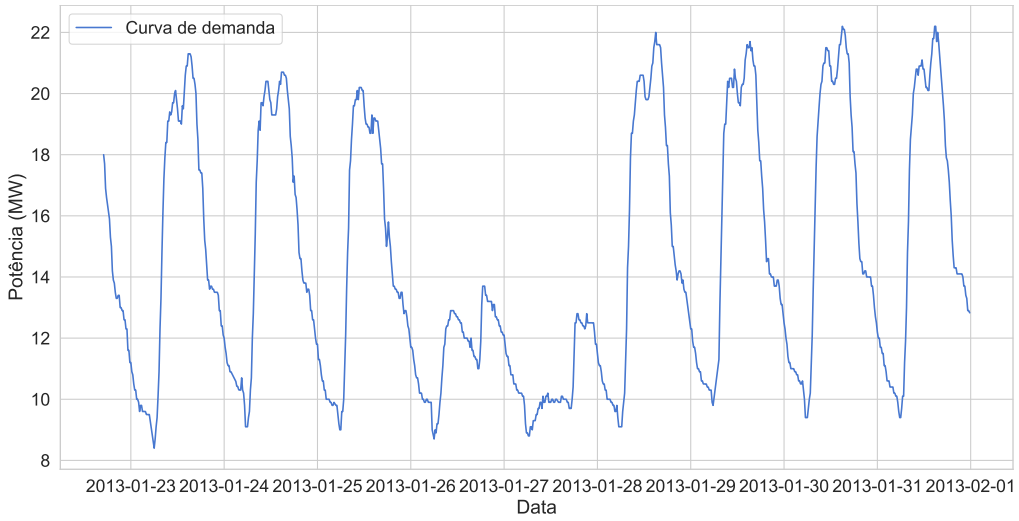
Figura 31 – Autoencoder: Curva de erro e limiar.



Fonte: Autoria própria.

Por fim, os dados são corrigidos por meio da interpolação linear (Figura 32) e as métricas de avaliação são utilizadas para avaliação dos resultados.

Figura 32 – Autoencoder: Correção dos dados.



Fonte: Autoria própria.

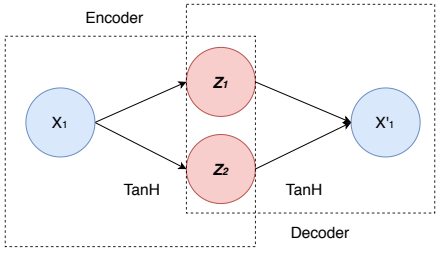
3.2.5.4 AUTOENCODER ESPARSO

Para a construção deste modelo optou-se por apenas uma entrada de dado X_i(valor de potência da série) no codificador com função de ativação TanH, uma camada oculta com dois neurônios e função e ativação TanH (escolhidos empiricamente) e uma saída por meio do decodificador, conforme Figura 33. Os parâmetros externos do modelo foram escolhidos conforme Tabela 3.

Tabela 3 – Autoencoder: Parâmetros externos de treinamento.

Parâmetros	Valores
Épocas	100
Cost Function	MSE
Otimizador do *gradient descent*	Adam
Taxa de aprendizagem	0.01

Figura 33 – Modelo Autencoder Esparso.

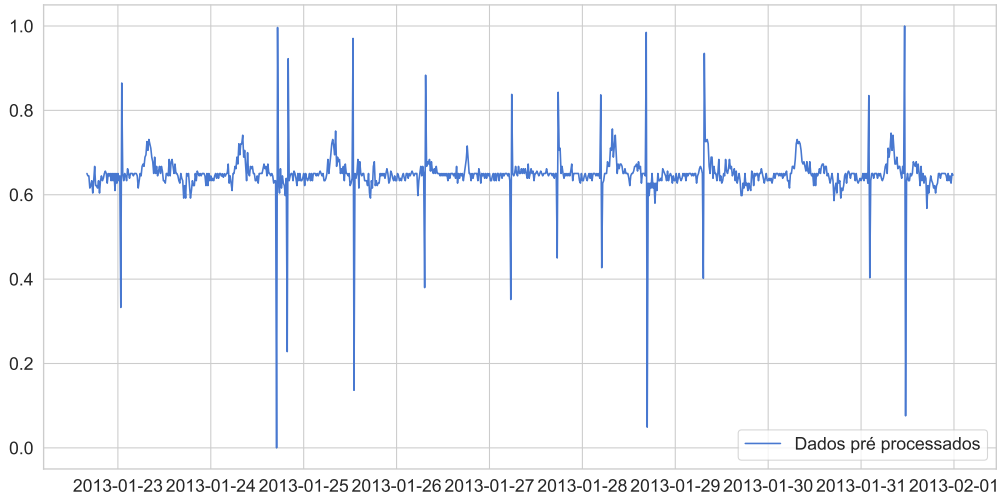


Fonte: Autoria própria.

O diferencial no modelo é que neste caso foi inserido um regularizador, que impõe uma restrição de esparsidade às unidades ocultas, o autoencoder ainda descobrirá uma estrutura interessante nos dados, mesmo se o número de unidades ocultas for maior que as de entrada. Sem entrar muito a fundo no assunto, um regularizador ajuda o modelo a evitar o *overfitting* (ZAREMBA; SUTSKEVER; VINYALS, 2014) e assim possibilita que o treinamento seja executado em 100 épocas sem se preocupar com este problema.

Ainda para este modelo, os dados foram pré processados, passando por uma diferenciação de primeiro grau e um escalonamento entre zero e um. O resultado do pré processamento é apresentado na Figura 34.

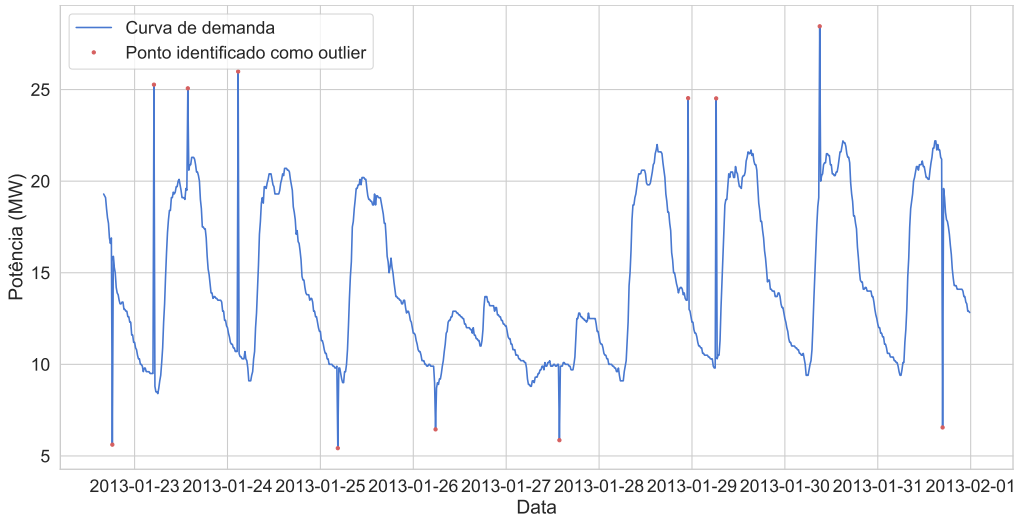
Figura 34 – Autoencoder Esparso: Dados pré processados.



Fonte: Autoria própria.

A diferenciação ajuda a estabilizar a média de uma série temporal, removendo as alterações no nível da mesma e, portanto, eliminando (ou reduzindo) a tendência e a sazonalidade (HYNDMAN; ATHANASOPOULOS, 2018). Já o escalonamento basicamente ajuda a normalizar os dados dentro de um intervalo específico, ajudando na velocidade de processamento do modelo.

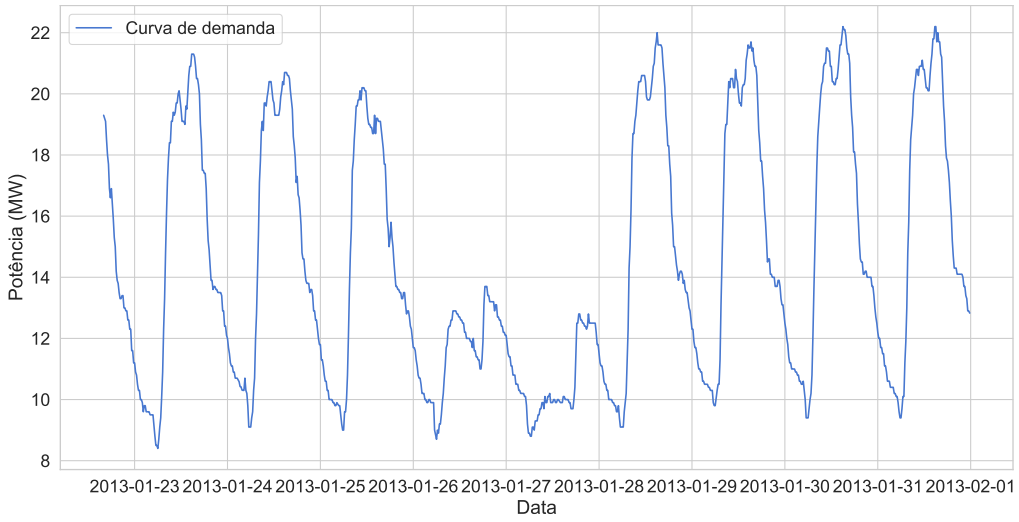
Figura 35 – Autoencoder Esparso: Detecção de outliers.



Fonte: Autoria própria.

Para detecção dos *outliers* o processo é o mesmo do autoencoder padrão, onde é colocado um limiar na curva de erro, no qual os valores que excederem este limiar serão considerados *outliers*. O limiar foi configurado também como sendo 3 vezes a média truncada em 1%. Por fim, os dados são corrigidos através da interpolação linear (Figura 36).

Figura 36 – Autoencoder Esparso: Correção dos dados.

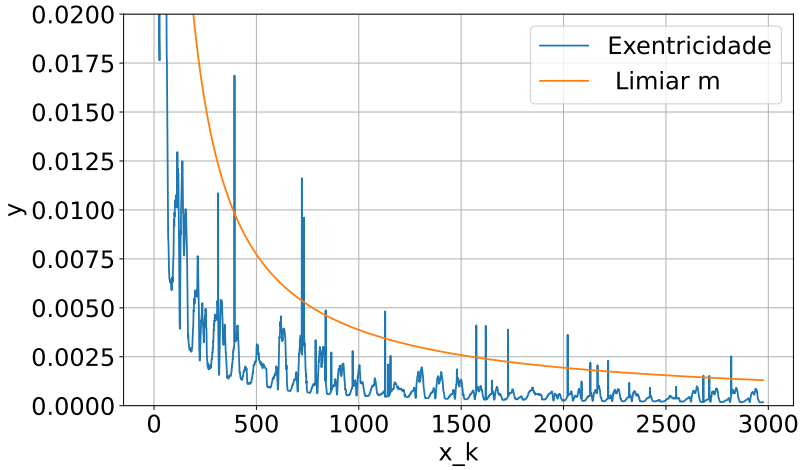


Fonte: Autoria própria.

3.2.5.5 TEDA

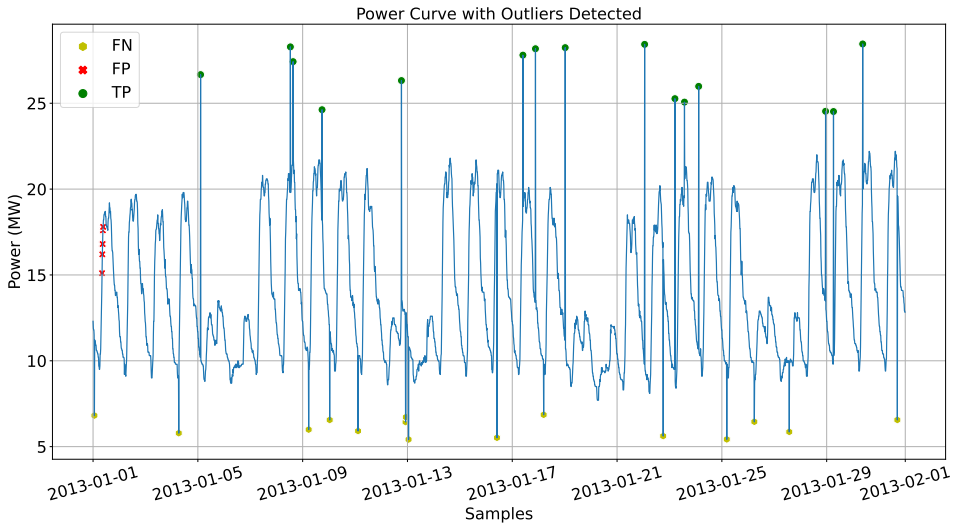
Angelov (2014) considera o algorítmo do TEDA como de aprendizado autônomo, e Bezerra et al. (2016) afirma que esta consideração se deve ao fato de o ciclo completo do algoritmo ser baseado apenas na análise dos dados, não sendo necessária informações prévias sobre as amostras. Conclui ainda que, por conta da dinâmica das iterações, o algorítmo é capaz de se adaptar a mudança nos dados (*concept drift* e *concept evolution*). Desta forma não é necessário o treinamento prévio ou utilização de modelos matemáticos, onde, com apenas a entrada de um dado x_i se calcula a média, variância e excentricidade, onde finalmente 2.16 é aplicado para saber se determinado x_i pode ser considerado um *outlier*.

Figura 37 – Excentricidade de cada dado e o limiar de detecção.



Fonte: Autoria própria.

Figura 38 – Indicadores de detecção de acordo com a matriz de correlação.



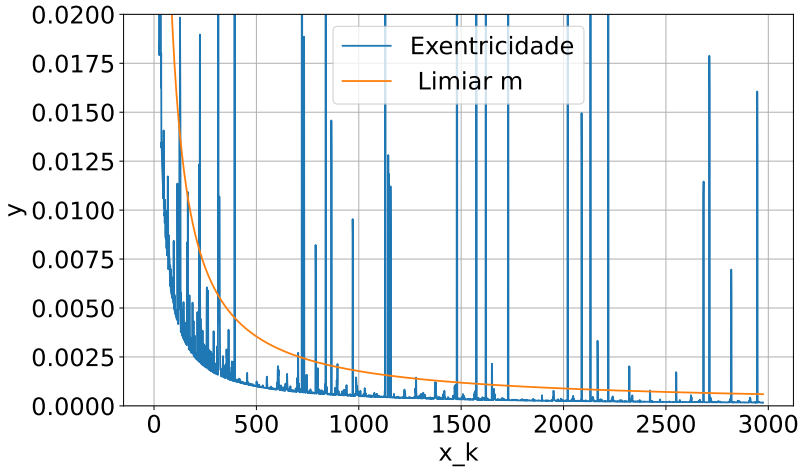
Fonte: Autoria própria.

A Figura 37 ilustra a excentricidade de cada dado bem como o limiar de detecção, indicando que alguns pontos foram detectados, que é ilustrado na Figura 38.

3.2.5.6 TEDA DIFF

Além de todas as vantagens existentes no algoritmo TEDA para detecção de anomalias, esta versão insere o fator da diferenciação para melhorar a detecção. Isto acontece pois a diferenciação ajuda a estabilizar a média de uma série temporal, removendo as alterações no nível da mesma e, portanto, eliminando (ou reduzindo) a tendência e a sazonalidade (HYNDMAN; ATHANASOPOULOS, 2018).

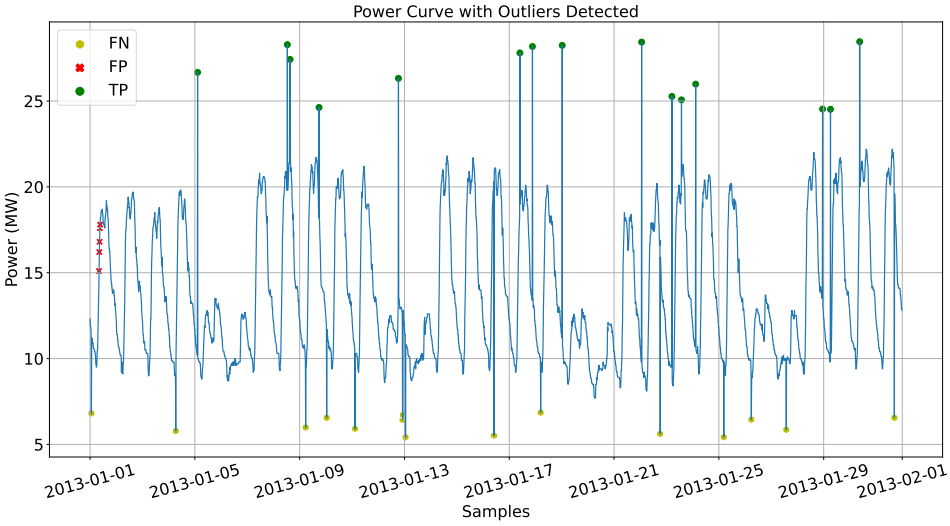
Figura 39 – Excentricidade de cada dado e o limiar de detecção.



Fonte: Autoria própria.

Uma nova variável é adicionada ao algoritmo, que guardará informação do dado anterior ao analisado, assim, a diferença é calculada ($x_{diff} = x_{k+1} - x_k$) e o resultado é que segue para os cálculos posteriores e recursivos de média, variância, excentricidade, etc.

Figura 40 – Indicadores de detecção de acordo com a matriz de correlação.



Fonte: Autoria própria.

A Figura 39 ilustra a excentricidade de cada dado bem como o limiar de detecção, indicando que alguns pontos foram detectados, que é ilustrado na Figura 40.

4 RESULTADOS

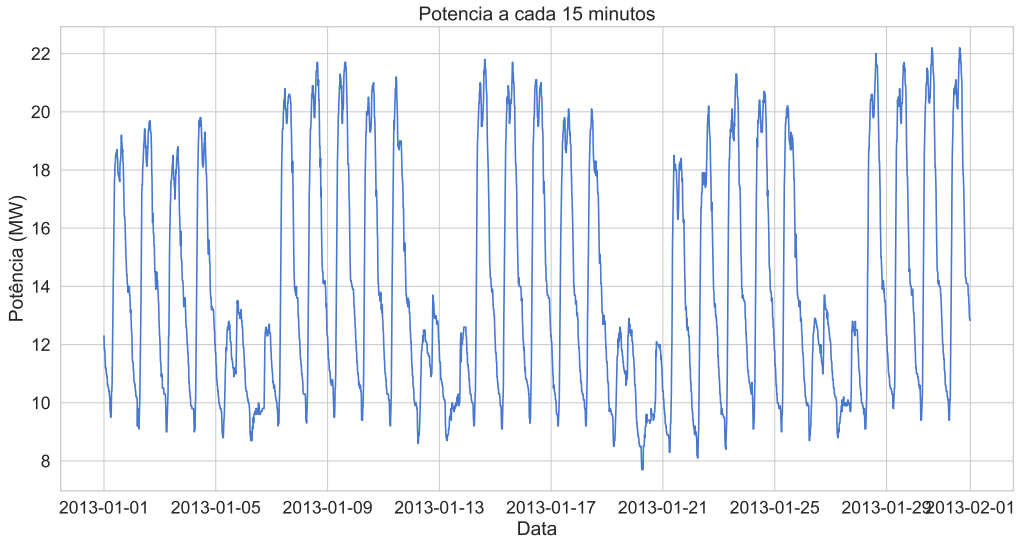
Neste capítulo serão demonstrados e discutidos os resultados obtidos neste trabalho, trazendo os exemplos e aplicações do trabalho desenvolvido. Inicialmente será realizada uma comparação entre os métodos KM+KNN e FCM+KNN como ferramenta para detecção de *outliers*, com levantamento de algumas avaliações e considerações.

Em seguida, o método com melhor desempenho será comparado aos modelos de *deep learning* autoencoders, novamente com algumas avaliações e considerações e mais uma vez, o mais performático irá ser comparado com o algorítmo auto evolutivo TEDA. Por fim, será apresentado um estudo comparativo que permitirá ilustrar algumas métricas de avaliação que foram consideradas.

4.1 DESCRIÇÃO DO CONJUNTO DE DADOS

Para realizar as primeiras comparações foram utilizados dados de 1 mês de medição da SEJPS compreendendo o período de 01 de janeiro de 2013 até 31 de janeiro de 2013 (Figura 41).

Figura 41 – Dados janeiro SEJPS.

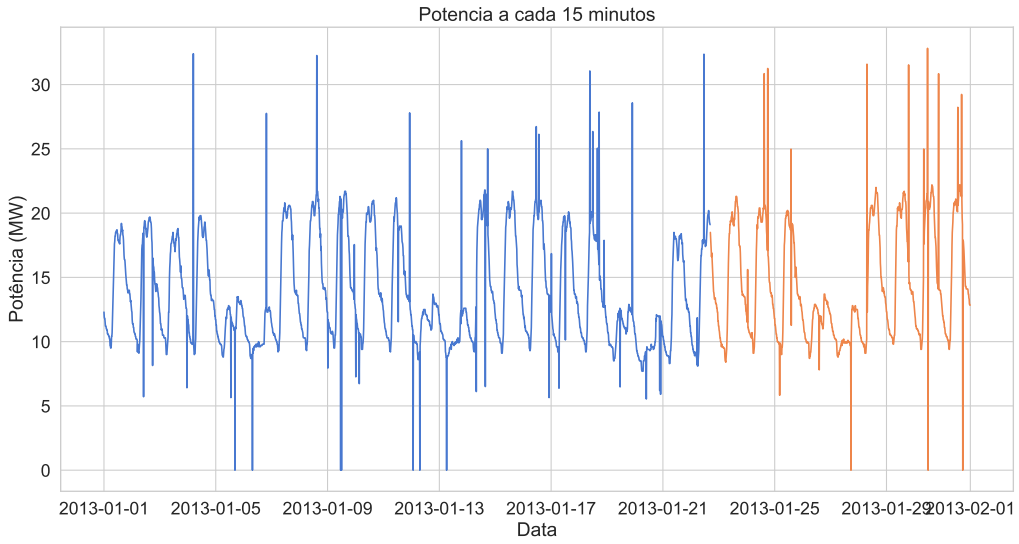


Fonte: Autoria própria.

Este conjunto de dados representa 2976 pontos de medição, coletados em um intervalo de 15 minutos entre eles. Separou-se os dados para treino e validação, onde 70% dos dados foram reservados para o treino e 30% para validação.

Outliers do tipo global e contextual foram inseridos aleatoriamente no conjunto em uma quantidade representativa de 1% dos dados, conforme pode-se verificar na Figura 42. Adicionalmente, em um dos casos estudados foi utilizado todo parte ou todo o conjunto de dados (Figura 14) para validação de alguns modelos (casos 9 e 10).

Figura 42 – Dados separados treino/validação e com outliers inseridos.



Fonte: Autoria própria.

Este conjunto (Figura 42) representa o *Caso 8* especificado no capítulo de metodologia, trazido apenas como exemplo visual.

4.2 CASO A: ALGORÍTIMOS KM+KNN E FCM+KNN

Nesta seção será realizada a comparação dos algorítimos KM+KNN e FCM+KNN em relação ao conjunto de dados da Seção 4.1 utilizando as métricas de avaliação anteriormente descritas. Na Tabela 4 apresentam-se os erros obtidos na avaliação dos modelos. As Figuras 43 e 44 apresentam as matrizes de confusão resultante de ambos os modelos e por fim, apresentam-se nas Figuras 45 e 46 os dados corrigidos após a detecção.

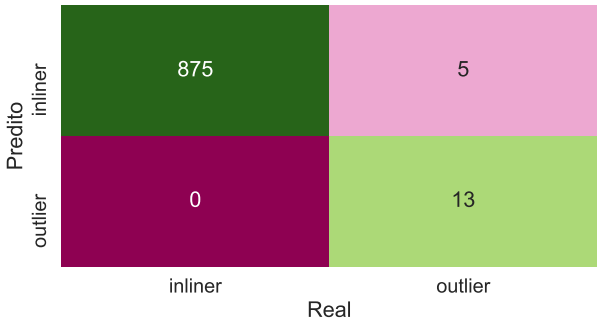
Tabela 4 – Caso A: Resultado.

	MSE	**MAPE**	**MCC**
KM	2.59	3.05%	0.43
FCM	2.54	2.96%	0.45

É possível verificar dos resultados que o algoritmo FCM+KNN teve um melhor desempenho no critério do MCC e MAPE em relação ao KM+KNN. Tal resultado se deve principalmente ao parâmetro fuzzificador existente no método FCM, que ameniza

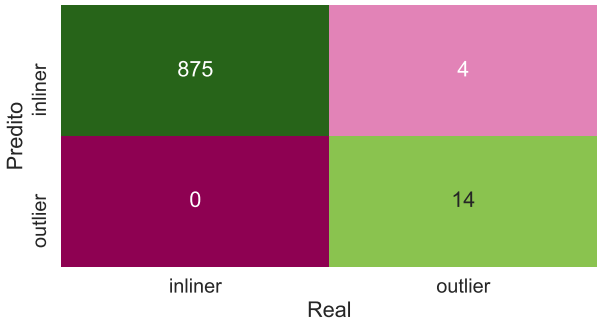
a influência dos *outliers* na posição dos centroides dos clusters, permitindo que estes se mantenham mais afastados e assim, possam detectar os *outliers* que se encontram mais distantes através da distância euclidiana.

Figura 43 – Caso A: Matriz de confusão KM+KNN.



Fonte: Autoria própria.

Figura 44 – Caso A: Matriz de confusão FCM+KNN.



Fonte: Autoria própria.

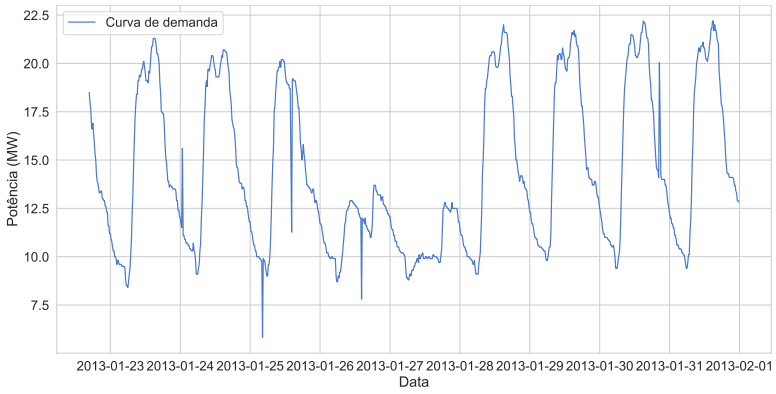
Assim, pode-se observar que ambos os métodos não fizeram detecção de dados falso positivos (FP) porém o KM falhou em detectar 5 *outliers* e o FCM 4, de um total de 8. O interessante é que 3 dos *outliers* não detectados são do tipo contextual, muito difícil de serem detectados pelos métodos aqui avaliados.

Verificado que o FCM é relativamente superior ao KM, utilizaremos esse para a comparação na próxima seção.

4.3 CASO B: ALGORÍTIMOS FCM+KNN E AUTOENCODER

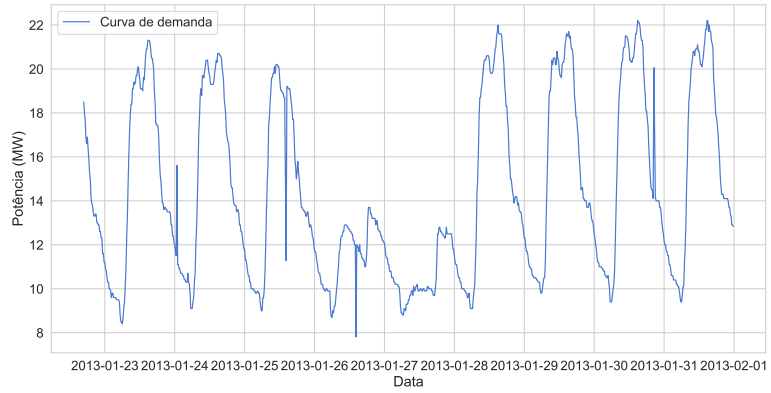
Desta vez, foram avaliados o FCM+KNN em relação ao Autoencoder padrão. Apresenta-se na Tabela 5 os erros obtidos na avaliação dos modelos. As Figuras 47 e 48 ilustram respectivamente a matriz de confusão e os dados corrigidos após a detecção do algoritmo autoencoder.

Figura 45 – Caso A: Dados corrigidos KM+KNN.



Fonte: Autoria própria.

Figura 46 – Caso A: Dados corrigidos FCM+KNN.



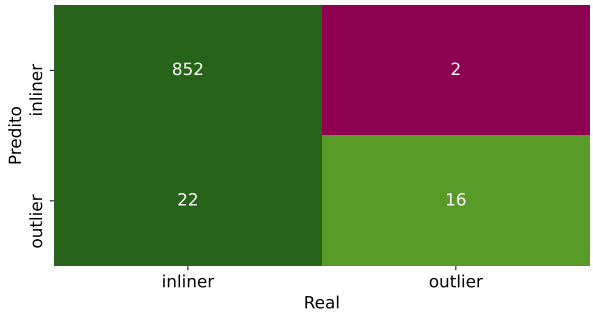
Fonte: Autoria própria.

Tabela 5 – Caso B: Resultado.

	MSE	**MAPE**	**MCC**
CM	2.54	2.96%	0.45
Autoencoder	1.18	1.86%	0.63

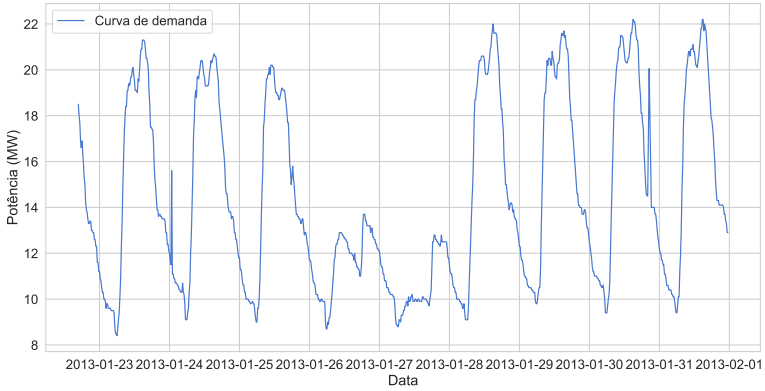
Da matriz de confusão do autoencoder pode-se verificar que alguns pontos foram detectados erroneamente como sendo *outliers*, porém eram *inliers*, caso este que não ocorreu com o KM e CM, porém em compensação os 3 outliers do tipo contextual, que não foram detectados pelo CM, foram aqui detectados.

Figura 47 – Caso B: Matriz de confusão Autoencoder.



Fonte: Autoria própria.

Figura 48 – Caso B: Dados corrigidos Autoencoder.



Fonte: Autoria própria.

É possível analisar que existiu um compromisso neste caso estudado, no qual, o autoencoder melhor atuou para detectar os *outliers*, inclusive os contextuais, porém alguns Falso Negativos foram etiquetados. Pode-se dizer que, neste caso, o autoencoder foi superior ao CM pois, mesmo detectando alguns Falso Negativos, após a correção, os erros MAPE e MSE foram menores quando comparados com os dados originais.

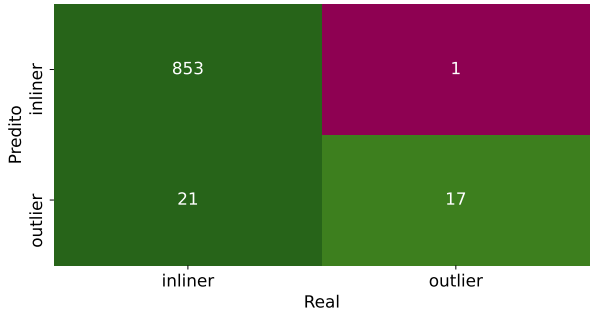
4.4 CASO C: ALGORÍTIMOS AUTOENCODER E AUTOENCODER ESPARSO

Aqui é trazido o algoritmo autoencoder esparso à prova. Na Tabela 6 apresentam-se os erros obtidos na avaliação dos modelos. Ilustra-se nas Figuras 49 e 50 respectivamente a matriz de confusão e os dados corrigidos após a detecção do algoritmo autoencoder esparsado.

Tabela 6 – Caso C: Resultado.

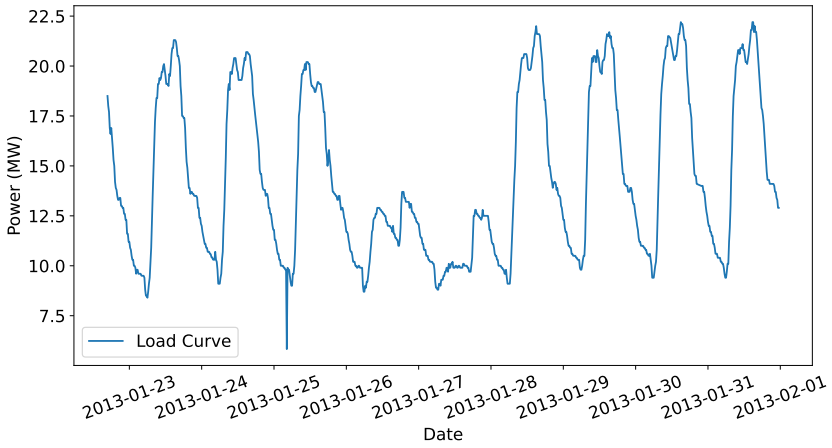
	MSE	MAPE	MCC
Autoencoder	1.18	1.84%	0.63
Autoencoder Esparsado	0.57	2.26%	0.71

Figura 49 – Caso C: Matriz de confusão Autoencoder Esparso.



Fonte: Autoria própria.

Figura 50 – Caso C: Dados corrigidos Autoencoder Esparso.



Fonte: Autoria própria.

O resultado foi superior, onde Autoencoder esparsado fez menos detecção de FN e FP no total, resultando em um MCC melhor.

4.5 CASO D: ALGORÍTIMOS AUTOENCODER ESPARSO E O TEDA

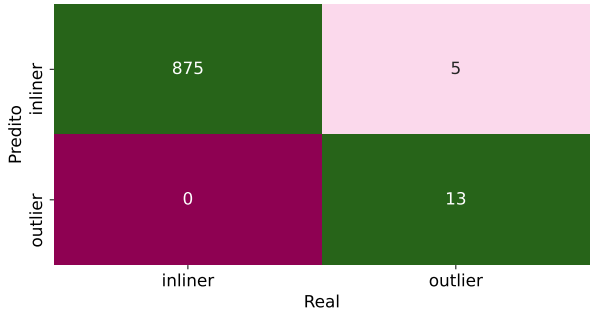
Um resultado superior foi obtido com o autoencoder esparso em relação ao autoencoder puro e agora comparamos aqui ele com o TEDA. Na Tabela 7 ilustram-se os erros

obtidos na avaliação dos modelos. E nas Figuras 51 e 53 ilustram-se respectivamente a matriz de confusão e os dados corrigidos após a detecção do algoritmo TEDA.

Tabela 7 – Caso D: Resultado.

	MSE	**MAPE**	**MCC**
Autoencoder Esparsado	0.57	2.26%	0.71
TEDA	0.19	0.28%	0.85

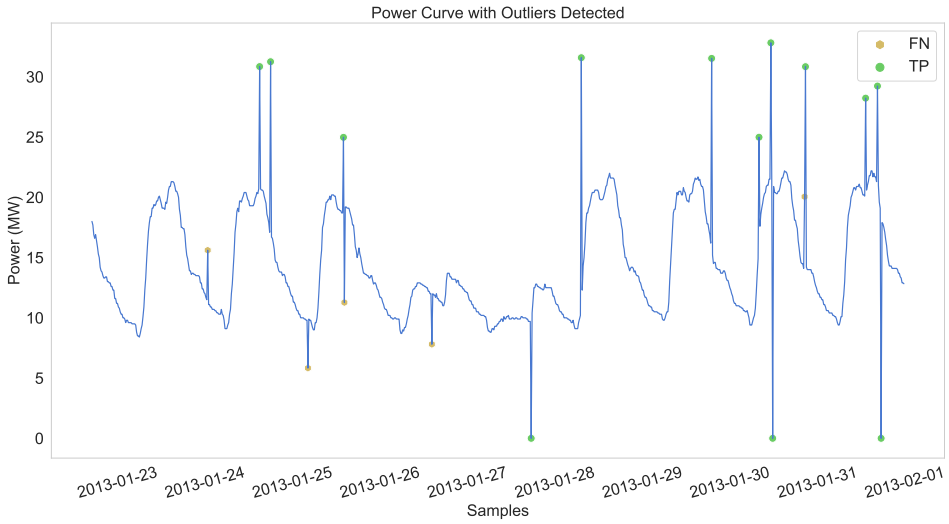
Figura 51 – Caso D: Matriz de confusão TEDA.



Fonte: Autoria própria.

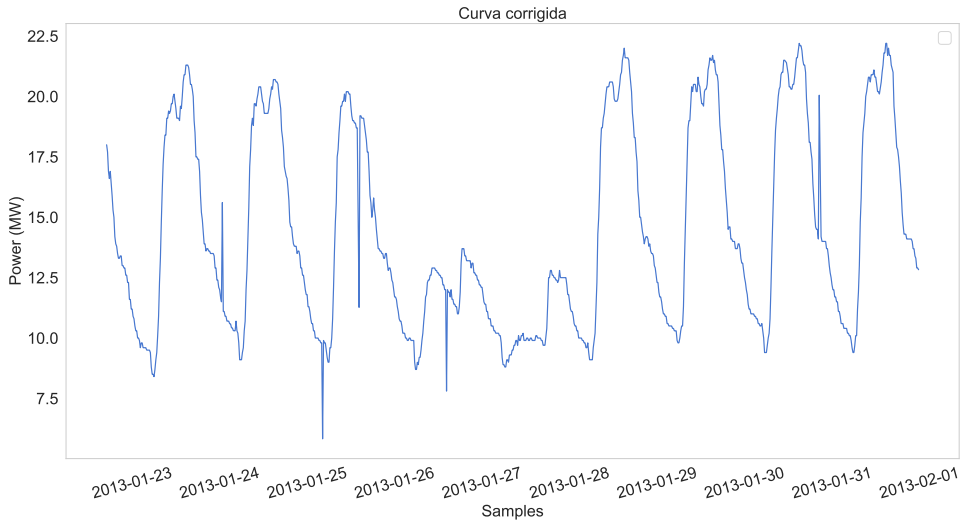
Conforme é possível observar, o TEDA se saiu melhor comparado com o autoencoder esparsado, declarando seis falsos positivos no início (momento que ainda está "aprendendo"o funcionamento da série) (Figura 52).

Figura 52 – Caso D: TEDA - Detecção de outliers.



Fonte: Autoria própria.

Figura 53 – Caso D: Dados corrigidos TEDA.



Fonte: Autoria própria.

É possível perceber mais facilmente que o TEDA falhou em detectar os vales e anomalias contextuais apresentados na Figura 53.

4.6 CASO E: ALGORÍTIMOS TEDA E O TEDA DIFF

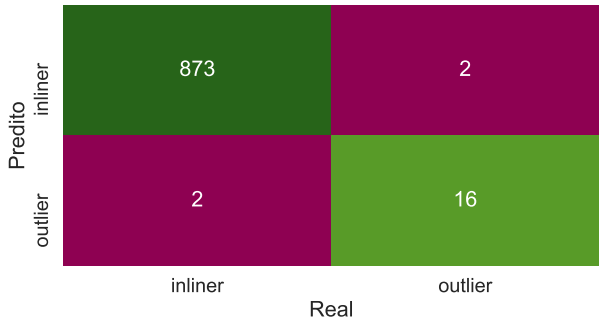
Mais uma vez é realizada a comparação, agora, do resultado do TEDA com o TEDA Diff. Na Tabela 8 são ilustrados os erros obtidos na avaliação dos modelos. As Figuras 54 e 56 ilustram respectivamente a matriz de confusão e os dados corrigidos após a detecção do algoritmo TEDA.

Tabela 8 – Caso E: Resultado.

	MSE	**MAPE**	**MCC**
TEDA Diff	0.56	0.37%	0.89
TEDA	0.19	0.28%	0.85

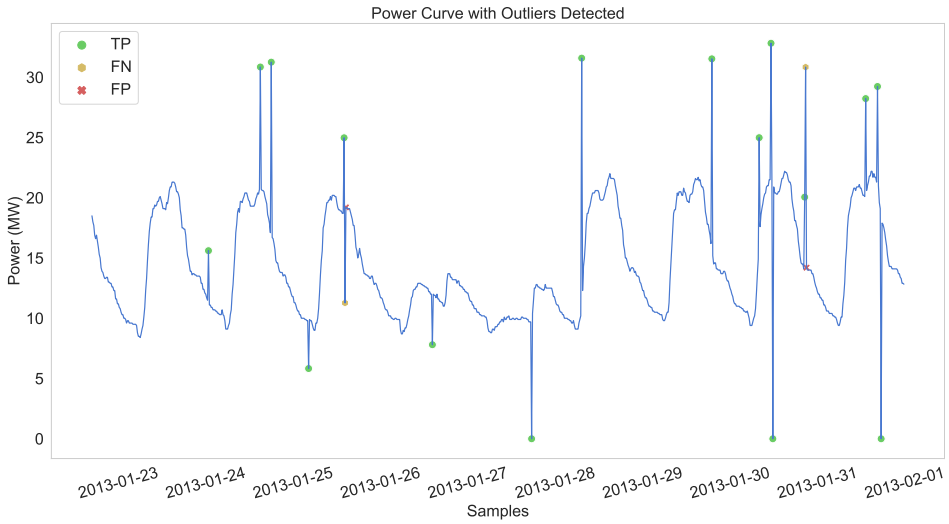
Neste caso, apesar de detectar corretamente um FN a menos, dois FP foram detectados pelo TEDA Diff em comparação ao TEDA, dando uma pequena diferença no resultado entre os dois (Figura 55). Para saber se em maiores conjuntos de dados essa diferença ainda existe, testaremos mais dois casos.

Figura 54 – Caso E: Matriz de confusão TEDA Diff.



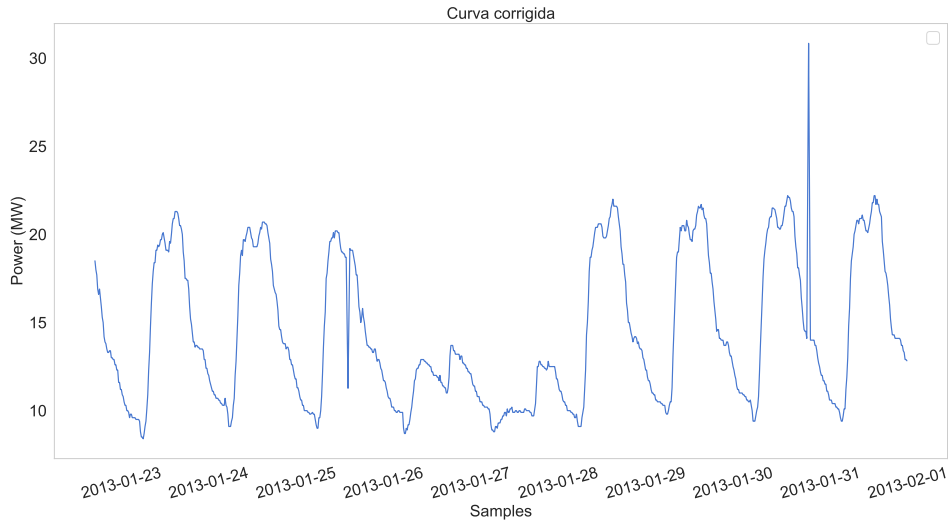
Fonte: Autoria própria.

Figura 55 – Caso E: TEDA Diff - Detecção de outliers.



Fonte: Autoria própria.

Figura 56 – Caso E: Dados corrigidos TEDA Diff.



Fonte: Autoria própria.

É possível perceber mais facilmente que o TEDA Diff obteve sucesso em detectar boa parte das anomalias conforme pode-se observar na Figura 56.

4.7 CASO F: ALGORÍTIMOS TEDA E O TEDA DIFF EM SÉRIE TEMPORAL DE 1 ANO

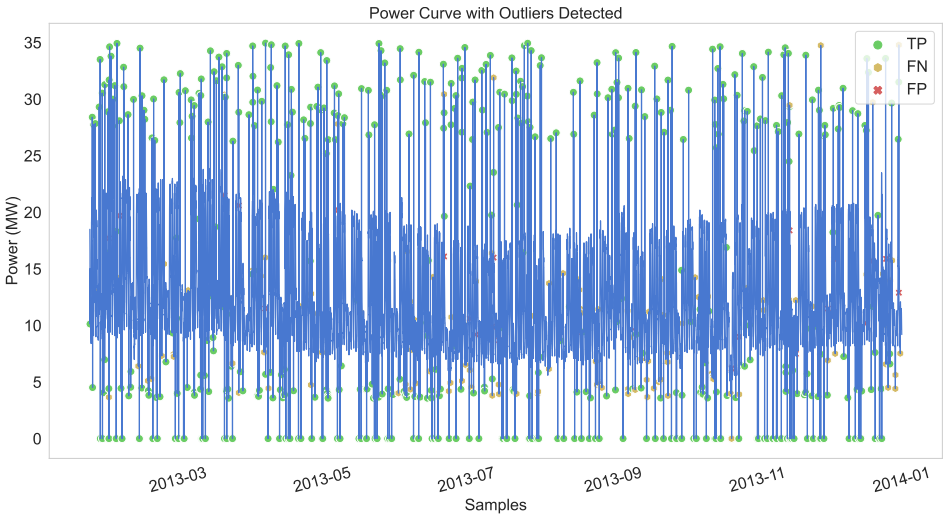
Nesta seção, é realizada uma comparação em situações onde é possivel encontrar uma série temporal mais longa, com um ano (caso 09), para verificar se os algoritmos TEDA e TEDA diff se beneficiam do fato de serem online. Para efeito de comparações, é trazido ainda o resultado do algoritmo Autoencoder esparso.

Tabela 9 – Caso F: Resultado com conjunto do caso 09.

	MSE	**MAPE**	**MCC**
Autoenconder Esparsado	1.1	2.61%	0.52
TEDA	0.46	0.47%	0.73
TEDA Diff	0.15	0.23%	0.87

Do resultado encontrado na Tabela 9 podemos concluir que para uma série temporal mais longa, tanto o TEDA quanto o TEDA diff se sairam melhor, como esperado, por serem auto adaptativos e funcionar de forma online.

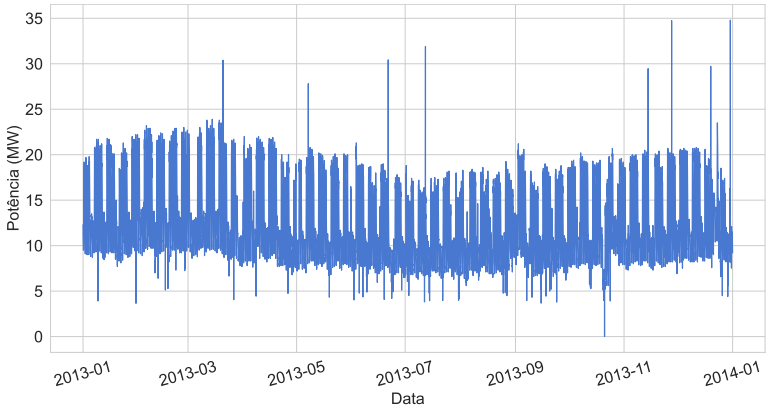
Figura 57 – Caso F: TEDA Diff - Detecção de outliers no caso 09.



Fonte: Autoria própria.

A maioria dos picos, vales e zeros foram detectados, passando como falso negativos apenas alguns poucos picos e zeros e majoritariamente as anomalias contextuais (Figura 57).

Figura 58 – Caso F: Dados corrigidos TEDA Diff no caso 09.



Fonte: Autoria própria.

Comparando o resultado da Figura 57 com a Figura 58 é possível perceber o quanto eficiente foi o algoritmo TEDA diff no caso em questão e em acordo com a Tabela 9.

4.8 CASO G: ALGORÍTIMOS TEDA E O TEDA DIFF EM SÉRIE TEMPORAL DE 6 ANOS

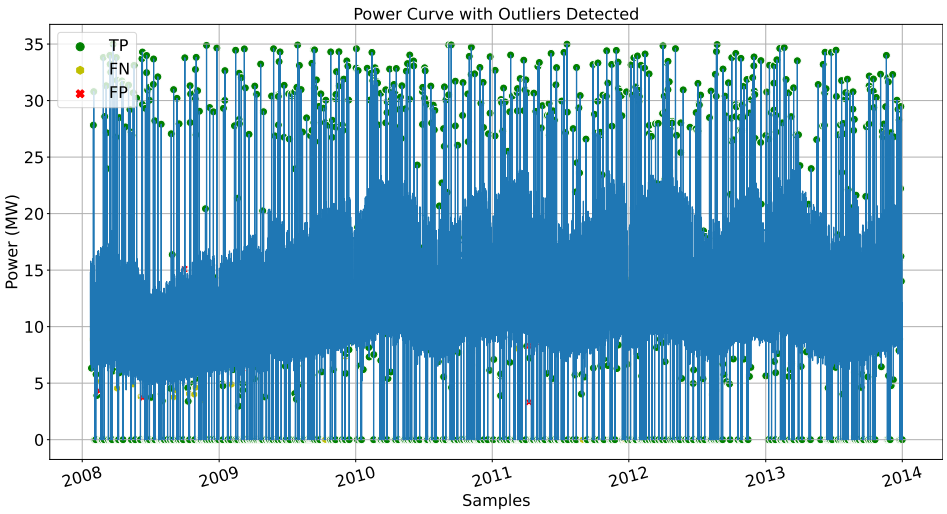
Aqui, finalizando os casos, é realizada uma última comparação em situações onde é possivel encontrar uma série temporal mais longa ainda, com seis anos de informações (caso 10), para forçar ainda mais a capacidade de detecção de anomalias dos algoritmos TEDA e TEDA diff, tendo em vista que já se conseguiu observar do caso F que esses algoritmos realmente se beneficiam do fato de serem online. Mais uma vez, para efeito de comparações, é trazido ainda o resultado do algoritmo Autoencoder esparso.

Tabela 10 – Caso G: Resultado com conjunto do caso 10.

	MSE	**MAPE**	**MCC**
Autoenconder Esparsado	0.61	2.12%	0.49
TEDA	0.07	0.27%	0.41
TEDA Diff	0.08	0.2%	0.97

Do resultado encontrado na Tabela 10 podemos concluir que para uma série temporal mais longa ainda, o Autoencoder esparso teve um resultado satisfatório quando comparado com o TEDA, porém o resultado do TEDA diff foi ainda melhor.

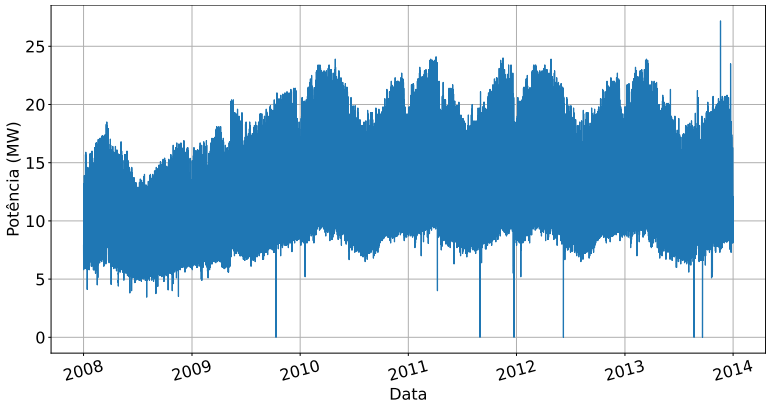
Figura 59 – Caso G: TEDA Diff - Detecção de outliers no caso 10.



Fonte: Autoria própria.

Praticamente todos os picos, vales e zeros foram detectados, inclusive as anomalias contextuais, dando a entender que mesmo a longo prazo, o algoritmo evolue de forma a manter as detecções mais constantes (Figura 59).

Figura 60 – Caso G: Dados corrigidos TEDA Diff no caso 10.



Fonte: Autoria própria.

Comparando o resultado da Figura 59 com a Figura 60 é possível perceber o quanto eficiente foi o algoritmo TEDA diff no caso em questão e em acordo com a Tabela 10.

4.9 COMPARATIVO DE TODOS OS MÉTODOS

Neste estudo,apresenta-se na Tabela 11 o resultado comparativo de todos os casos desenvolvidos na metodologia em relação a todos os algoritmos de detecção de *outliers* neste trabalho abordados.

Tabela 11 – Avaliação e todos os métodos em relação aos casos.

		Caso 1:	Caso 2:	Caso 3:	Caso 4:	Caso 5:	Caso 6:	Caso 7:	Caso 8:	Caso 9:	Caso 10:
Z-Score	**MSE**	0.0	0.74	27.89	7.8	0.78	11.34	0.53	2.85	0.47	0.05
	MAPE	0.01%	0.51%	11.45%	5.49%	0.54%	6.93%	0.57%	3.36%	0.47%	0.23%
	MCC	1.00	0.00	0.0	0.0	0.60	0.0	0.0	0.31	0.73	0.54
Z-Score	**MSE**	0.00	0.74	0.00	7.8	0.41	4.90	0.53	2.70	0.47	0.44
Modificada	**MAPE**	0.01%	0.51%	0.08%	5.49%	0.28%	3.32%	0.57%	3.25%	0.47%	0.46%
	MCC	1.00	0.00	1.00	0.0	0.736	0.64	0.0	0.36	0.73	0.33
K-Means	**MSE**	3.46	0.00	27.89	7.8	0.24	10.15	0.46	2.59	0.13	0.05
	MAPE	1.4%	0.01%	11.45%	5.49%	0.11%	6.17%	0.42%	3.05%	0.22%	0.23%
	MCC	0.0	1.0	0.0	0.0	0.90	0.37	0.49	0.43	0.82	0.54
C-Means	**MSE**	0.00	0.00	27.89	7.8	0.00	5.86	0.46	2.54	0.13	0.05
	MAPE	0.01%	0.01%	11.45%	5.49%	0.01%	3.93%	0.42%	2.96%	0.22%	0.23%
	MCC	1.0	1.0	0.0	0.0	1.0	0.58	0.49	0.45	0.82	0.54
Autoencoder	**MSE**	0.00	0.07	0.80	3.488	0.01	3.15	0.052	1.18	0.15	0.68
	MAPE	0.05%	0.18%	0.47%	3.9%	0.1%	2.83%	0.19%	1.84%	1.84%	2.23%
	MCC	0.58	0.52	0.919	0.439	0.46	0.58	0.66	0.63	0.60	0.17
Sparsed	**MSE**	0.00	0.00	4.97	1.31	0.00	1.55	0.00	0.57	1.10	0.61
Autoencoder	**MAPE**	0.01%	0.01%	2.86%	1.88%	0.03%	1.51%	0.05%	2.26%	2.61%	2.12%
	MCC	1.0	1.0	0.87	0.76	0.85	0.82	0.60	0.71	0.52	0.49
TEDA	**MSE**	0.00	0.74	15.5	7.80	0.41	7.36	0.30	0.19	0.46	0.07
	MAPE	0.01%	0.51%	6.84%	5.49%	0.28%	4.82%	0.27%	0.28%	0.47%	0.27%
	MCC	1.00	0.00	0.61	0.0	0.74	0.45	0.50	0.85	0.73	0.41
TEDA	**MSE**	0.00	0.00	6.77	1.52	0.00	2.41	0.00	0.56	0.15	0.08
Diff	**MAPE**	0.01%	0.02%	3.59%	2.23%	0.01%	2.27%	0.01%	0.37%	0.23%	0.2%
	MCC	1.00	0.95	0.76	0.70	1.00	0.71	1.00	0.89	0.87	0.97

Ilustra-se com auxílio da 11 a avaliação de todos os modelos do trabalho em relação aos casos propostos. Algumas conclusões podem ser extraídas desta tabela, onde em verde

temos os modelos que obtiveram melhor resultado naquele caso e em amarelo aqueles que obtiveram MCC superior a 0,7. É importante mencionar que os casos 7 a 10 são os únicos casos que trazem *outliers* contextuais em seus conjuntos.

Importante mencionar ainda que, quando se trabalha com classificação, existe um custo associado em se encontrar um FP e FN e para cada problema essas detecções tem um significado. Neste trabalho, o custo dos FN são bem maiores que os FP, pois após a correção por meio da interpolação linear, um FP gera um erro muito pequeno, muitas vezes quase nulo, em relação a curva original sem contaminação, por outro lado, um FN traz um grande prejuízo agregado à detecção. Por exemplo, no caso B e C foi possível observar que existiu a detecção de um número considerável de FP, porém, na avaliação do MSE e MAPE foi possível perceber que isso decorreu em quase nenhum resultado negativo em termos de erro.

5 CONCLUSÃO

Neste trabalho foram apresentados diversos métodos e técnicas com o objetivo de construção de modelos para realizar a detecção e correção de *outliers* em curvas de demanda.

Para validação dos modelos foram utilizadas as métricas de erro MAPE, MSE e MCC baseado na matriz de confusão. Dos métodos não supervisionados foi possível concluir que o Fuzzy C-Means com KNN teve uma melhor performance, tanto em relação ao *Caso A* quanto em uma visão geral. Isso se deu principalmente pelo diferencial fuzzy que impõe um parâmetro de lógica difusa no processo de posicionamento dos centroides no agrupamento dos dados, demonstrando ser um método robusto quando testado em detecção de *outliers* do tipo global, porém falhando quando da detecção dos outliers do tipo contextual em curvas de demanda elétrica.

O modelo *deep learning autoencoder* foi então comparado com o C-Means no *Caso B*, onde esse modelo se demonstrou superior na detecção de outliers contextuais, porém, obtendo uma baixa pontuação no MCC pelo motivo de ter detectado alguns FN, sendo possível concluir que este modelo pode vir a ser escolhido quando da presença de outliers contextuais, por outro lado, o C-Means consegue fazer o trabalho de detecção os *outliers* pontuais com um menor custo.

Em seguida, o modelo *deep learning* Autoencoder Espaçado foi trazido. Este modelo tem algumas especificações extras que permitem uma maior robustez na detecção de outliers, quando do tratamento correto dos dados (escalonamento e diferenciação). Em comparação com o autoencoder padrão para a detecção de outliers, este se demonstrou superior, tanto no *Caso B* quanto nos casos 1 a 8, conforme podemos observar da Tabela 11. Este resultado se deve não apenas ao pré processamento de dados como da inserção do parâmetro de restrição de esparsidade na camada oculta que previne, principalmente, do modelo treinar em 100 épocas sem *overfitting*.

Por fim, traz-se o algoritmo TEDA e TEDA Diff à prova, onde o segundo surpreende ao ter ótimos resultados inclusive quando posto para provar conjuntos com outliers contextuais (caso 7 a 10), se saindo inclusive melhor que o Autoencoder esparçado nos caso 7 a 10.

O desempenho da maioria dos modelos apresentados (exceção do TEDA) dependem sempre da escolha dos parâmetros internos destes, onde podemos apontar especialmente para a escolha da quantidade de clusters e da distância mínima de um ponto em relação ao seu centro para que este seja etiquetado como *outlier* nos modelos baseados em KM e

FCM.

Ainda, nos modelos supervisionados Autencoder e Autoencoder Esparso a complexidade ainda é maior, pois se trata de métodos *deep learning*, onde dezenas de parâmetros internos e externos necessitam ser configurados. Para os casos de detecção de outlier é importante mencionar que a escolha do limiar baseado no erro de treinamento/predição é uma tarefa que compreende um compromisso grande, no qual o ajuste muito próximo termina por detectar muitos FN e um ajuste menos ousado compromete a boa detecção.

Desta forma, a escolha de um modelo para detecção de outlier vem a depender dos tipos e da frequência dos outliers existentes e que se pretende corrigir em uma curva de demanda, sendo o FCM o que obteve os melhores resultados neste trabalho quando se fala em um modelo não supervisionado, o Autoencoder Esparsado quando se tratando de modelo de *deep learning* e o TEDA de modo geral, especialmente quando se necessita de um modelo não paramétrico e evolutivo.

Inclusive, importante mencionar que o TEDA diff se demonstrou o algoritmo mais estável, obtendo bons resultados (MCC acima de 0.7) em todos os casos, chamando atenção para sua robustez quando posto para tratar grande conjunto de dados em séries temporais de longo prazo (inclusive acima de 5 anos), onde o algoritmo Autoencoder esparso obteve resultado que foi decrescendo na medida que a série se distanciava dos do tempo em que a rede foi treinada. Assim, o resultado do TEDA diff se deve principalmente pela sua capacidade de se adaptar com o tempo.

Para continuidade do trabalho sugere-se:

- Otimização dos parâmetros;
- Proposição de novos modelos para detecção de outliers globais;
- Aplicação do z-score na curva do ruído, obtida através da decomposição multiplicativa;
- Substituição do Limiar através da inequação de Chebyshev no TEDA por um modelo Fuzzy com regras automáticas;

Por fim, esse trabalho contribui de forma direta para a ampliação dos conhecimentos adquiridos ao longo da pós-graduação em engenharia elétrica do autor, sobretudo na área de *Big Data* e sistemas inteligentes. Desta maneira o conhecimento utilizado nesse trabalho irá ser de grande utilidade na vida profissional do autor.

REFERÊNCIAS

AGGARWAL, C. C. Outlier analysis. In: SPRINGER. *Data mining*. [S.l.], 2015. p. 237–263. Citado na página 16.

ANDRADE, P. H. M. d. et al. Metodologia para a detecção e correção de outliers em curvas de potência de subestações utilizando técnicas de inteligência artificial. Universidade Federal da Paraíba, 2018. Citado na página 15.

ANGELOV, P. *Autonomous learning systems: from data streams to knowledge in real-time*. [S.l.]: John Wiley & Sons, 2012. Citado na página 39.

ANGELOV, P. Outside the box: an alternative data analytics framework. *Journal of Automation Mobile Robotics and Intelligent Systems*, v. 8, n. 2, p. 29–35, 2013. Citado 4 vezes nas páginas 23, 36, 37 e 38.

ANGELOV, P. Anomaly detection based on eccentricity analysis. In: IEEE. *2014 IEEE symposium on evolving and autonomous learning systems (EALS)*. [S.l.], 2014. p. 1–8. Citado 3 vezes nas páginas 38, 39 e 58.

BABUŠKA, R. *Fuzzy modeling for control*. [S.l.]: Springer Science & Business Media, 2012. v. 12. Citado na página 32.

BARBOSA, B. et al. Defining content marketing and its influence on online user behavior: a data-driven prescriptive analytics method. *Annals of Operations Research*, Springer, p. 1–26, 2023. Citado na página 24.

BEZDEK, J. C. Objective function clustering. In: *Pattern recognition with fuzzy objective function algorithms*. [S.l.]: Springer, 1981. p. 43–93. Citado na página 32.

BEZERRA, C. G. et al. A new evolving clustering algorithm for online data streams. In: IEEE. *2016 IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS)*. [S.l.], 2016. p. 162–168. Citado na página 58.

BHATIA, N. et al. Survey of nearest neighbor techniques. *arXiv preprint arXiv:1007.0085*, 2010. Citado na página 33.

BOUGHORBEL, S.; JARRAY, F.; EL-ANBARI, M. Optimal classifier for imbalanced data using matthews correlation coefficient metric. *PloS one*, Public Library of Science San Francisco, CA USA, v. 12, n. 6, p. e0177678, 2017. Citado na página 47.

BRASIL. Resolução normativa aneel n. 109, de 29 de outubro de 2004. institui a convenção de comercialização de energia elétrica. *Diário Oficial [da] República Federativa do Brasil*, Brasília, DF, 2004. ISSN 1677-7042. Disponível em: <http://pesquisa.in.gov.br/imprensa/jsp/visualiza/index.jsp?jornal=1&pagina=196&data=29/10/2004>. Citado na página 16.

CABRERA-SÁNCHEZ, J.-P. et al. Online recommendation systems: Factors influencing use in e-commerce. *Sustainability*, MDPI, v. 12, n. 21, p. 8888, 2020. Citado na página 23.

CHEN, W. et al. Data quality of electricity consumption data in a smart grid environment. *Renewable and Sustainable Energy Reviews*, Elsevier, v. 75, p. 98–105, 2017. Citado na página 14.

CHICCO, D.; JURMAN, G. The advantages of the matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation. *BMC genomics*, Springer, v. 21, n. 1, p. 6, 2020. Citado na página 47.

COSTA, B. S. J. et al. Online fault detection based on typicality and eccentricity data analytics. In: IEEE. *2015 International Joint Conference on Neural Networks (IJCNN)*. [S.l.], 2015. p. 1–6. Citado na página 37.

COVER, T.; HART, P. Nearest neighbor pattern classification. *IEEE transactions on information theory*, IEEE, v. 13, n. 1, p. 21–27, 1967. Citado na página 33.

CRAMER, J.; KRUEGER, A. B. Disruptive change in the taxi business: The case of uber. *American Economic Review*, v. 106, n. 5, p. 177–82, 2016. Citado na página 14.

DAS, S. *Time series analysis*. [S.l.]: Princeton university press, Princeton, NJ, 1994. v. 10. Citado na página 19.

DUNN, J. C. A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters. Taylor & Francis, 1973. Citado na página 32.

FREITAS, I. W. S. d. et al. Um estudo comparativo de técnicas de detecção de outliers no contexto de classificação de dados. Universidade Federal Rural do Semi-Árido, 2019. Citado na página 27.

GALLAGHER, L. *A história da Airbnb*. [S.l.]: Buzz Editora LTDA, 2018. Citado na página 14.

GARCIA, F. A. A. Tests to identify outliers in data series. *Pontifical Catholic University of Rio de Janeiro, Industrial Engineering Department, Rio de Janeiro, Brazil*, v. 50, 2012. Citado na página 28.

GÉRON, A. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*. [S.l.]: "O'Reilly Media, Inc.", 2022. Citado 2 vezes nas páginas 40 e 41.

GORDON, N.; RISTIC, B.; ARULAMPALAM, S. Beyond the kalman filter: Particle filters for tracking applications. *Artech House, London*, v. 830, n. 5, p. 1–4, 2004. Citado na página 36.

HADI, A. S. Identifying multiple outliers in multivariate data. *Journal of the Royal Statistical Society: Series B (Methodological)*, Wiley Online Library, v. 54, n. 3, p. 761–771, 1992. Citado na página 26.

HAO, S.; ZHOU, X.; SONG, H. A new method for noise data detection based on dbscan and svdd. In: IEEE. *2015 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*. [S.l.], 2015. p. 784–789. Citado na página 15.

HASHMANI, M. A. et al. Concept drift evolution in machine learning approaches: a systematic literature review. *International Journal on Smart Sensing and Intelligent Systems*, Exeley, Inc., v. 13, n. 1, p. 1, 2020. Citado na página 23.

HAWKINS, D. M. *Identification of outliers*. [S.l.]: Springer, 1980. v. 11. Citado 2 vezes nas páginas 16 e 26.

HECHT-NIELSEN, R. Theory of the backpropagation neural network. In: *Neural networks for perception*. [S.l.]: Elsevier, 1992. p. 65–93. Citado na página 35.

HEYONG, W.; MINGJIAN, L.; BINGCHUAN, C. The research of outlier data cleaning based on accelerating method. In: IEEE. *2010 2nd IEEE International Conference on Information Management and Engineering*. [S.l.], 2010. p. 153–155. Citado na página 15.

HÖPPNER, F. et al. *Fuzzy cluster analysis: methods for classification, data analysis and image recognition*. [S.l.]: John Wiley & Sons, 1999. Citado na página 32.

HYNDMAN, R. J.; ATHANASOPOULOS, G. *Forecasting: principles and practice*. [S.l.]: OTexts, 2018. Citado 3 vezes nas páginas 20, 56 e 59.

IGLEWICZ, B.; HOAGLIN, D. Volume 16: how to detect and handle outliers. *The ASQC basic references in quality control: statistical techniques*, EF Mykytka, Ed, v. 16, 1993. Citado na página 29.

JIA, W.-J.; ZHANG, Y.-D. Survey on theories and methods of autoencoder. *Computer Systems & Applications*, n. 5, p. 1, 2018. Citado 2 vezes nas páginas 34 e 35.

JUNIOR, A. A. da S. Aplicação de deep learning no preenchimento de falhas em dados micrometeorológicos. Universidade Federal de Mato Grosso, 2019. Citado na página 35.

KANNAN, K. S.; MANOJ, K.; ARUMUGAM, S. Labeling methods for identifying outliers. *International Journal of Statistics and Systems*, v. 10, n. 2, p. 231–238, 2015. Citado na página 28.

KARLIK, B.; OLGAC, A. V. Performance analysis of various activation functions in generalized mlp architectures of neural networks. *International Journal of Artificial Intelligence and Expert Systems*, v. 1, n. 4, p. 111–122, 2011. Citado na página 35.

KARP, R. M. On-line algorithms versus off-line algorithms: How much. In: *Algorithms, Software, Architecture: Information Processing 92: Proceedings of the IFIP 12th World Computer Congress*. [S.l.: s.n.], 1992. v. 1, p. 416. Citado na página 23.

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. Citado 2 vezes nas páginas 31 e 48.

KRAMER, M. A. Nonlinear principal component analysis using autoassociative neural networks. *AIChE journal*, Wiley Online Library, v. 37, n. 2, p. 233–243, 1991. Citado na página 34.

KUHLMAN, D. A python book: Beginning python. *Advanced Python, and Python Exercises*, 2012. Citado na página 40.

LIN, D. et al. Platogl: Effective and scalable deep graph learning system for graph-enhanced real-time recommendation. In: *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. [S.l.: s.n.], 2022. p. 3302–3311. Citado na página 23.

MACQUEEN, J. et al. Some methods for classification and analysis of multivariate observations. In: OAKLAND, CA, USA. *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*. [S.l.], 1967. v. 1, n. 14, p. 281–297. Citado na página 29.

MADNICK, S.; ZHU, H. Improving data quality through effective use of data semantics. *Data & Knowledge Engineering*, Elsevier, v. 59, n. 2, p. 460–475, 2006. Citado na página 14.

MAIMON, O.; ROKACH, L. Data mining and knowledge discovery handbook. Springer, 2005. Citado na página 22.

MATTHEWS, B. W. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, Elsevier, v. 405, n. 2, p. 442–451, 1975. Citado na página 47.

MOSLEHI, K.; KUMAR, R. A reliability perspective of the smart grid. *IEEE transactions on smart grid*, IEEE, v. 1, n. 1, p. 57–64, 2010. Citado na página 14.

OSHERSON, D.; SMITH, E. E. On typicality and vagueness. *Cognition*, Elsevier, v. 64, n. 2, p. 189–206, 1997. Citado na página 37.

PAL, N. R.; BEZDEK, J. C. On cluster validity for the fuzzy c-means model. *IEEE Transactions on Fuzzy systems*, IEEE, v. 3, n. 3, p. 370–379, 1995. Citado na página 32.

PERSONS, W. M. Correlation of time series. *Journal of the American Statistical Association*, Taylor & Francis, v. 18, n. 142, p. 713–726, 1923. Citado na página 20.

PIRYONESI, S. M.; EL-DIRABY, T. E. Role of data analytics in infrastructure asset management: Overcoming data size and quality problems. *Journal of Transportation Engineering, Part B: Pavements*, American Society of Civil Engineers, v. 146, n. 2, p. 04020022, 2020. Citado na página 33.

POUPART, P. et al. Online flow size prediction for improved network routing. In: IEEE. *2016 IEEE 24th International Conference on Network Protocols (ICNP)*. [S.l.], 2016. p. 1–6. Citado na página 24.

PRINCIPE, J. C. *Information theoretic learning: Renyi's entropy and kernel perspectives*. [S.l.]: Springer Science & Business Media, 2010. Citado na página 36.

PRINCIPE, J. C. et al. Learning from examples with information theoretic criteria. *Journal of VLSI signal processing systems for signal, image and video technology*, Springer, v. 26, n. 1, p. 61–77, 2000. Citado na página 36.

ROSSUM, G. V. et al. Python programming language. In: *USENIX annual technical conference*. [S.l.: s.n.], 2007. v. 41, p. 36. Citado na página 40.

SCHMIDHUBER, J. Deep learning in neural networks: An overview. *Neural networks*, Elsevier, v. 61, p. 85–117, 2015. Citado na página 34.

SHARMA, S. Activation functions in neural networks. *Towards Data Science*, v. 6, 2017. Citado na página 35.

SHIFFLER, R. E. Maximum z scores and outliers. *The American Statistician*, Taylor & Francis Group, v. 42, n. 1, p. 79–80, 1988. Citado na página 28.

SMITH, B.; LINDEN, G. Two decades of recommender systems at amazon. com. *Ieee internet computing*, IEEE, v. 21, n. 3, p. 12–18, 2017. Citado na página 14.

SONG, X. et al. Conditional anomaly detection. *IEEE Transactions on knowledge and Data Engineering*, IEEE, v. 19, n. 5, p. 631–645, 2007. Citado na página 25.

STATS MODELS. *Seasonal Decompose - Stats Models*. 2020. Disponível em: <https://www.statsmodels.org/stable/generated/statsmodels.tsa.seasonal.seasonal_decompose.html>. Acesso em: 04 de agosto de 2020. Citado na página 21.

STEHMAN, S. V. Selecting and interpreting measures of thematic classification accuracy. *Remote sensing of Environment*, Elsevier, v. 62, n. 1, p. 77–89, 1997. Citado na página 46.

TEICHGRAEBER, H.; BRANDT, A. R. Time-series aggregation for the optimization of energy systems: Goals, challenges, approaches, and opportunities. *Renewable and Sustainable Energy Reviews*, Elsevier, v. 157, p. 111984, 2022. Citado na página 15.

WANG, S.; SCHLOBACH, S.; KLEIN, M. What is concept drift and how to measure it? In: SPRINGER. *International Conference on Knowledge Engineering and Knowledge Management*. [S.l.], 2010. p. 241–256. Citado na página 22.

WILLIAMS, G. et al. A comparative study of rnn for outlier detection in data mining. In: IEEE. *2002 IEEE International Conference on Data Mining, 2002. Proceedings.* [S.l.], 2002. p. 709–712. Citado na página 26.

XIA, Y. et al. Learning discriminative reconstructions for unsupervised outlier removal. In: *Proceedings of the IEEE International Conference on Computer Vision*. [S.l.: s.n.], 2015. p. 1511–1519. Citado na página 34.

YOON, K.-A.; KWON, O.-S.; BAE, D.-H. An approach to outlier detection of software measurement data using the k-means clustering method. In: IEEE. *First International Symposium on Empirical Software Engineering and Measurement (ESEM 2007)*. [S.l.], 2007. p. 443–445. Citado na página 31.

YU, N. et al. Big data analytics in power distribution systems. In: IEEE. *2015 IEEE power & energy society innovative smart grid technologies conference (ISGT)*. [S.l.], 2015. p. 1–5. Citado na página 14.

ZAREMBA, W.; SUTSKEVER, I.; VINYALS, O. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*, 2014. Citado na página 56.

ZHANG, L.; WANG, X.; LI, X. *Power system operation information acquisition and maintenance technology*. Beijing: China Electric Power Press, 2010. Citado na página 15.

ŽLIOBAITĖ, I.; PECHENIZKIY, M.; GAMA, J. An overview of concept drift applications. *Big data analysis: new algorithms for a new society*, Springer, p. 91–114, 2016. Citado na página 23.