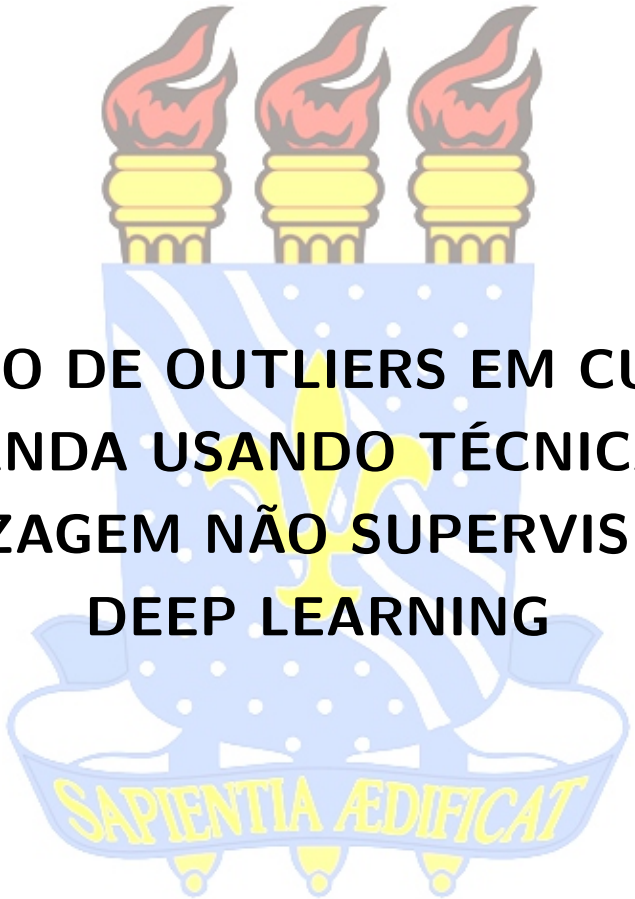
UNIVERSIDADE FEDERAL DA PARAÍBA
CENTRO DE ENERGIAS ALTERNATIVAS E RENOVÁVEIS
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

Ricardo Wagner Correia Guerra Filho



DETECÇÃO DE OUTLIERS EM CURVAS DE DEMANDA USANDO TÉCNICAS DE APRENDIZAGEM NÃO SUPERVISIONADA E DEEP LEARNING

João Pessoa
2020

Ricardo Wagner Correia Guerra Filho

DETECÇÃO DE OUTLIERS EM CURVAS DE DEMANDA USANDO TÉCNICAS DE APRENDIZAGEM NÃO SUPERVISIONADA E DEEP LEARNING

Trabalho de Conclusão de Curso apresentado à Coordenação do Curso de Engenharia Elétrica da Universidade Federal da Paraíba como exigência para a obtenção do título de Bacharel em Engenharia Elétrica.

Universidade Federal da Paraíba

Centro de Energias Alternativas e Renováveis

Curso de Graduação em Engenharia Elétrica

Orientador: Prof. Dr. Juan Moises Maurício Villanueva

João Pessoa

2020

Catalogação na publicação
Seção de Catalogação e Classificação

F481d Filho, Ricardo Wagner Correia Guerra.
DETECÇÃO DE OUTLIERS EM CURVAS DE DEMANDA USANDO TÉCNICAS DE APRENDIZAGEM NÃO SUPERVISIONADA E DEEP LEARNING / Ricardo Wagner Correia Guerra Filho. - João Pessoa, 2020.
62 f. : il.

Monografia (Graduação) - UFPB/CEAR.

1. Outliers. 2. Curva de demanda. 3. Aprendizado profundo. 4. Autoencoders. I. Título

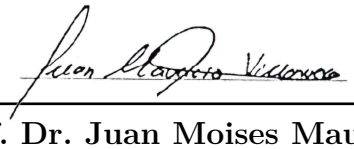
UFPB/BC

Ricardo Wagner Correia Guerra Filho

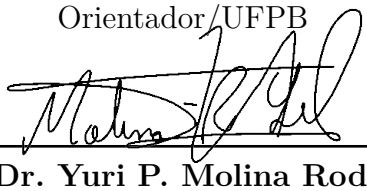
DETECÇÃO DE OUTLIERS EM CURVAS DE DEMANDA USANDO TÉCNICAS DE APRENDIZAGEM NÃO SUPERVISIONADA E DEEP LEARNING

Trabalho de Conclusão de Curso apresentado à Coordenação do Curso de Engenharia Elétrica da Universidade Federal da Paraíba como exigência para a obtenção do título de Bacharel em Engenharia Elétrica.

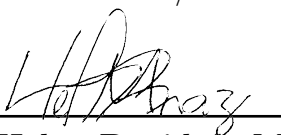
Trabalho aprovado. João Pessoa, 03 de agosto de 2020:



Prof. Dr. Juan Moises Maurício Villanueva
Orientador/UFPB



Prof. Dr. Yuri P. Molina Rodriguez
Avaliador Interno/UFPB



Prof. Dr. Helon David de Macêdo Braz
Avaliador Interno/UFPB

João Pessoa
2020

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 namorada, Sheylla Monteiro, pelo apoio incondicional, por ter me acalmado nos momentos de dificuldade, 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, Euler e Helon, 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

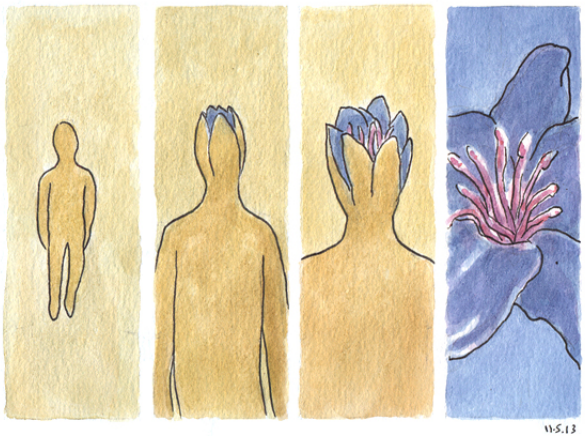
Aos meus amigos, Diego Cavalcanti, Joel Adelaide e Iuri Araujo, vocês foram indispensáveis durante toda a minha vida acadêmica.

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.

Aos amigos do GICA, agradeço por todo o conhecimento compartilhado ao longo desse período e pela amizade construída, em especial, Kaique, Jeane, Jean, Filipe, Frank, Rafael, Caio, Bruno, Jonathan, Celso e Cristiane.

Aos amigos da graduação, Luan Florencio, João Neto, Henrique Raldi. Juntos nos apoiamos e vencemos desafios.

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 de pesquisa 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) e algoritmos de aprendizado profundo do tipo Autoencoders. Para a correção de outliers é utilizada a interpolação linear. É apresentada ainda comparações de resultados entre técnicas como Z-Score, Z-Score Modificado, K-Means, C-Means, Autoencoders e Sparsed Autoencoders, 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. É esperado que esta pesquisa forneça uma melhor compreensão das diferentes direções em que a pesquisa foi realizada sobre esse tópico e como as técnicas desenvolvidas em uma área podem ser aplicadas nos domínios para os quais não se destinavam.

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

ABSTRACT

The detection of outliers is an important problem that has been researched in several research 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) and Autoencoders deep learning 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 and Sparsed Autoencoders 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. It is expected that this research will provide a better understanding of the different directions in which the research was carried out on this topic and how the techniques developed in an area can be applied in the domains for which they were not intended.

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

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

SUMÁRIO

1 INTRODUÇÃO

1.1 PERTINÊNCIA E MOTIVAÇÃO DO TRABALHO

Este trabalho monográfico tem por objetivo abordar a detecção e correção de anomalias, entituladas 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 (DAS, 1994).

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 potênca, 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 fiáveis 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 (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 tanto das outras observações 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 divérsos métodos existentes, sendo eles o Z-Score, K-Means, C-Means e Autoencoder, 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* e *Autoencoder Esparso*. 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 forma justa e 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 OBJETIVOS

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.

1.2.1 OBJETIVO GERAL

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.

1.2.2 OBJETIVO ESPECÍFICO

Delimitar o tema em relação a aplicação de técnica de inteligência computacional para deteção e correção de outliers em curva de demanda elétrica, especificamente utilizando métodos não supervisionados Z-Score, K-Means, C-Means e métodos supervisionados *deep learning* autoencoder para atingir o objetivo do trabalho. Abordar sua especificade, sua finalidade, cabimento, avaliar resultados, vantagens e desvantagens e outros fatores próprios deste.

1.3 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 peformance 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

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 a detecção dos outliers em uma série temporal é necessário que compreender que existem outros aspectos que entram em jogo ao lidar com essas, como nível, tendencia 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.

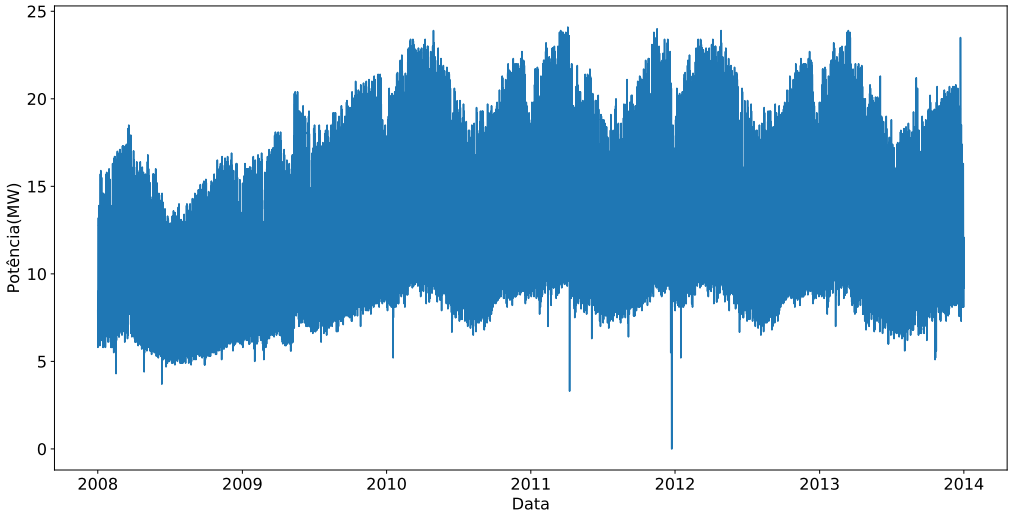
Para realizar algumas análises iniciais, usaremos dados medidos de potência ativa numa frequência de 1 amostra a cada 15 minutos, um exemplo de uma curva de demanda é ilustrado na Figura 1.

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 é sempre de uma 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 sazonalidae de uma série temporal, pode-se utilizar o

Figura 1 – Curva de demanda subestação João Pessoa



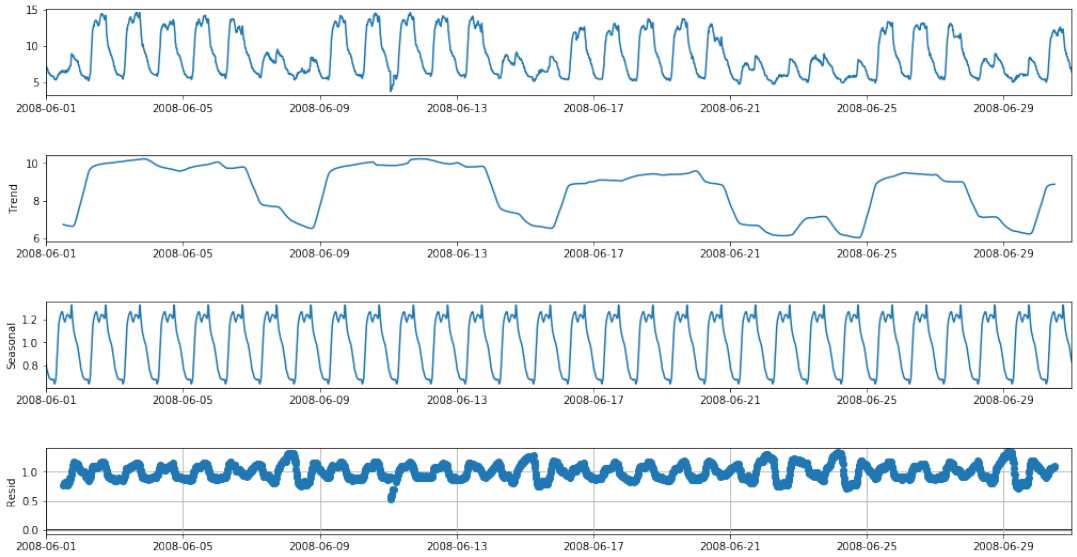
Fonte: Autoria própria.

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.

$$Serie = Tendencia \times Sazonalidade \times Residuo \tag{2.1}$$

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

Figura 2 – Série temporal decomposta



Fonte: Autoria própria.

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

2.1.3 ESTACIONARIEDADE

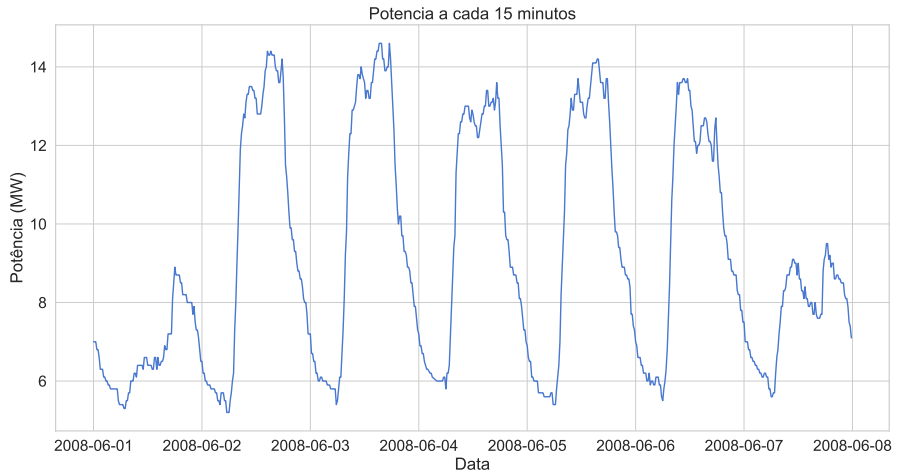
A estacionariedade é uma característica importante das séries temporais. Uma série temporal é considerada estacionária se suas propriedades estatísticas não mudarem ao longo do tempo. Em outras palavras, ele tem média e variação constantes, e a covariância é independente do tempo.

Idealmente, queremos ter uma série temporal estacionária para modelagem. Obviamente, nem todas as séries são estacionárias, mas podemos 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 seja consistente 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 minunciosa sobre o tema. A Figura 3 ilustra a sazonalidade da curva de demanda elétrica, onde 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.

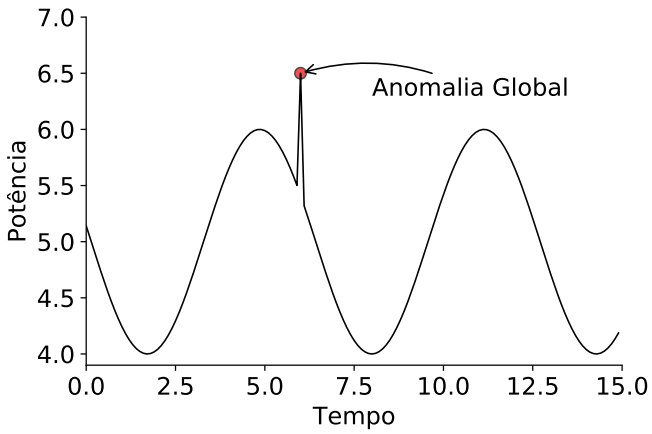
2.2 TIPOS DE OUTLIERS

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



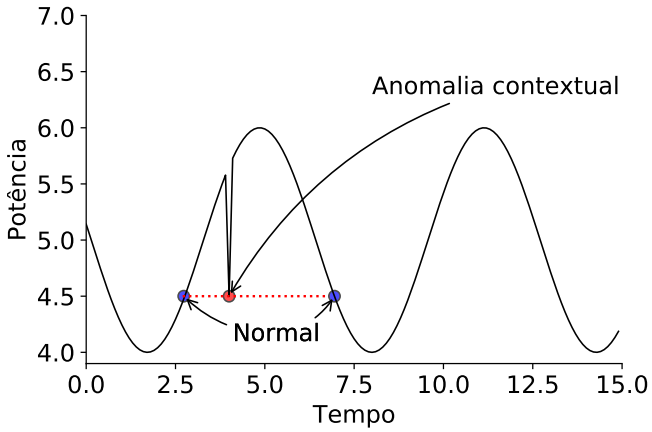
Fonte: Autoria própria.

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 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



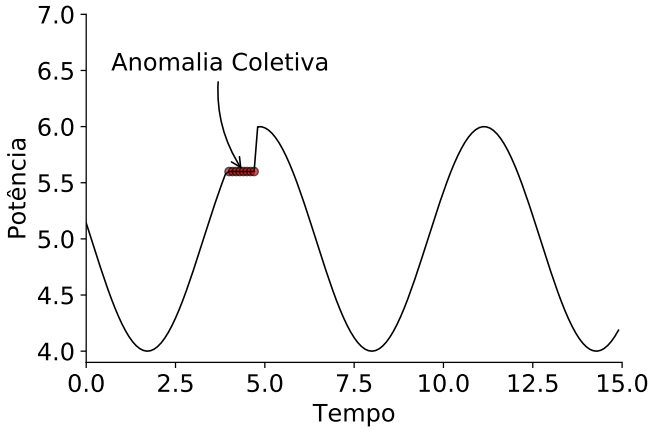
Fonte: Autoria própria.

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 iremos focar em deteção 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 será explanado algumas técnicas de detecção de *outliers* e suas particularidades.

Primeiramente, é importante saber que métodos de detecção 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 semisupervisionada;
- Abordagem não supervisionada;

As técnicas que utilizam a abordagem **supervisionada** assumem que a disponibilidade de um conjunto de dados de treinamento rotulado em dados normais e anomalos. 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 anomalos, geralmente é um desafio.

As técnicas que utilizam a abordagem **semissupervisionada** 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 implícita 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 iremos trazer alguns exemplos e aplicações de algumas técnicas de detecção de outliers.

2.3.1.1 Z-SCORE

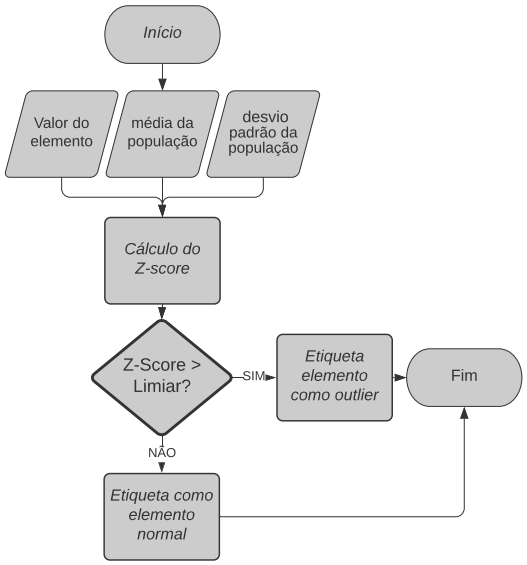
O *z-score* é uma medida numérica estatística que fornece uma idéia de quão longe um ponto de dado está da média. Mais tecnicamente, porém, é uma medida de desvio padrão abaixo ou acima da população onde aquele dado se encontra, se convertendo em uma pontuação bruta.

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).

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

Onde 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 basta separar os dados que tiverem um *z-score* que se desvia além de um liminar podem então serem candidatos a um outlier (Figura 7. Este liminar 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 podem ser bastante afetadas 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).

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

Onde $\widetilde{x}$ 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 Mi é eficaz para dados normais da mesma maneira que o *Z-score*.

2.3.1.3 K-MEANS

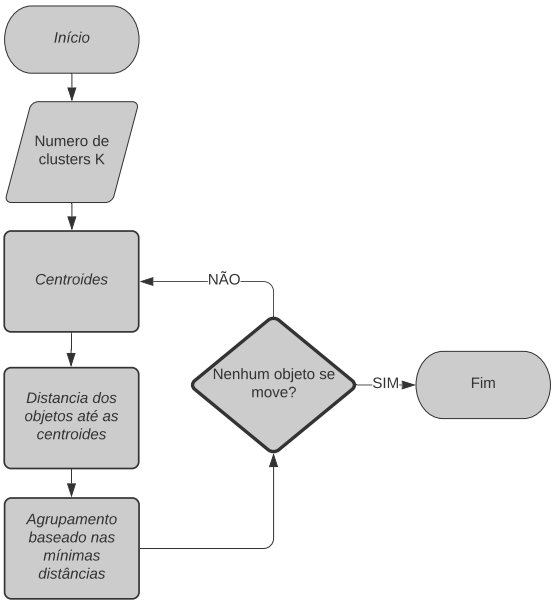
K-means (KM) é uma técnica de agrupamento (clusterização) criada por MacQueen et al. (1967) e é um dos mais simples algoritmos de aprendizado não supervisionado que resolvem este conhecido problema. O procedimento segue uma maneira simples e fácil de classificar um determinado conjunto de dados por meio de um certo número de clusters (suponha k clusters), bem ilustrado através do fluxograma da Figura 8.

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

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

Finalmente, este algoritmo visa minimizar uma função objetiva, neste caso uma

Figura 8 – Fluxograma K-Means



Fonte: Autoria própria.

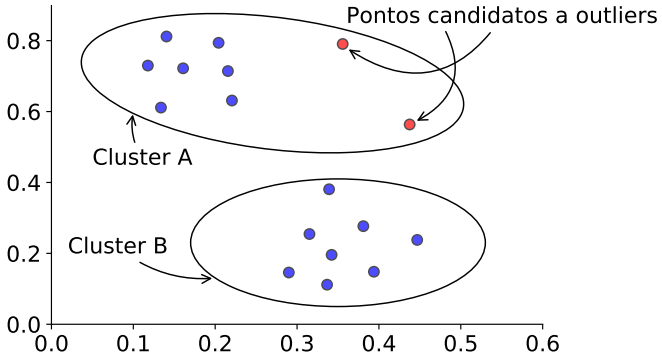
função do erro quadrático. A função objetiva da equação (2.5)

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

Onde $\|x - \mu_i\|^2$ 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 seu centroide, e os pontos pertencentes a este cluster estão definidos pela distancia deles para com este centroide. Então, pode-se categorizar outliers de acordo com a distancia absoluta deste para com o centroide, bem como de acordo com a distancia relativa, que seria a distancia absoluta comparada com a média das distâncias dos pontos do cluster para o centroide, à exemplo do observado na Figura 9.

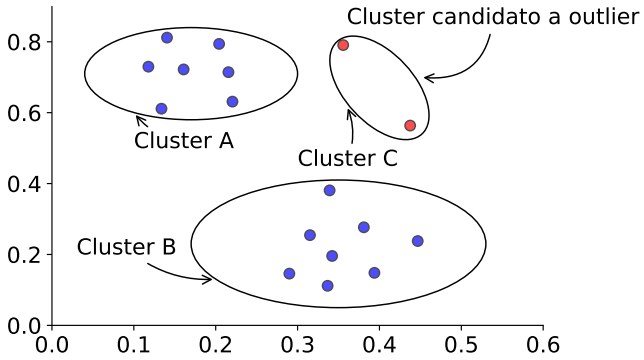
Figura 9 – Exemplo 1 k-means



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:

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

onde,

- u_{ki} é o grau de pertinência do objeto i em relação ao cluster k;
- $\left| x_i - c_k \right|^2$ 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.

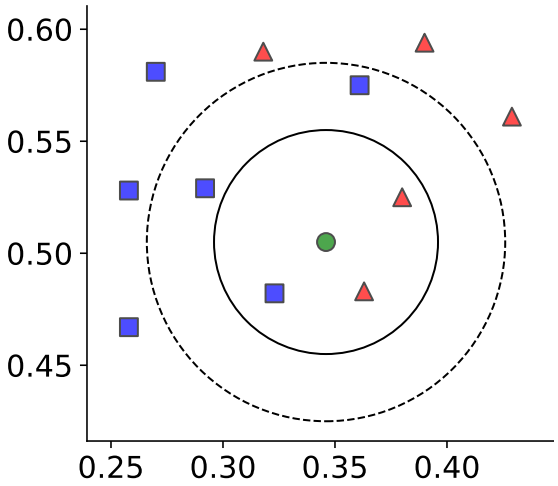
Quando utilizado como técnica de detecção de outlier, o FCM funciona da mesma forma que o KM, acrescentando o parâmetro fuzzificador. O parâmetro fuzzificador m, quando se aproxima de 1 tendem a tornar os clusters mais densos, pois a centroide tende a se posicionar mais próxima da concentração dos objetos,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

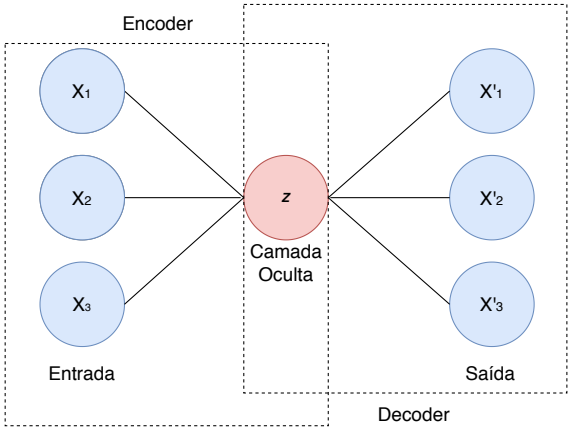
A vantagem do KNN como método de classificação é que o seu treinamento é rapido, de simples implementação e robusto, porém pode ficar tendencioso em decorrência da escolha do K e ser facilmente enganado por atributos irrelevantes (BHATIA et al., 2010).

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).

Figura 12 – Exemplo autoencoder



Fonte: Autoria própria.

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) autoencoder da Figura 12 pode ser modelado pela equação (2.7). Onde $f(\cdot)$ e $g(\cdot)$ são as funções de encoder e decoder, w e b são os parametros 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 $\theta = (w_e, b_e; w_d, b_d)$.

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

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 parametros 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 construidos 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 construida respeitando essa limitação, sendo normalmente disposto como ilustrado na Figura 12, uma camada de entrada, uma cada oculta e uma camada de saída.

Outro parâmentro 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 dois deles: Hyperbolic 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. 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 é uma função que parece linear porém 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 ajuda a 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 parâmetros externos 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 do autoencoder padrão apenas com a inclusão de restrições esparsas, onde, 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 INTERPOLAÇÃO LINEAR

Se tratando de um trabalho que tem como objetivo a deteção e correção de outliers, é importante que seja fundamentada a técnica utilizada para correção dos dados.

Na matemática, a interpolação linear é um método de ajuste de curvas usando polinômios lineares para construir novos pontos de dados dentro do intervalo de um conjunto discreto de pontos de dados conhecidos (WIKIPEDIA, 2020b). Se os dois pontos conhecidos são dados pelas coordenadas (x_0, y_0) e (x_1, y_1), o interpolante linear é a linha reta entre esses pontos. Para um valor x no intervalo (x_0, x_1), o valor y ao longo da linha reta é dado a partir da equação (2.9)

$$\frac{y - y_0}{x - x_0} = \frac{y_1 - y_0}{x_1 - x_0}, \tag{2.9}$$

Assim, esta é a técnica utilizada neste trabalho para realizar a correção dos outliers. Por ser bastante conhecida na literatura, justifica-se a breve fundamentação. Ao leitor, caso deseje aprofundamento sobre o tema, o livro de Davis (1975) é uma referência clássica e completa.

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 (bibliotécas) Python. É importante citar alguns destes módulos e suas especificações.

O **Scikit-learn**, de acordo com o Wikipédia (WIKIPEDIA, 2020c), 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 neigbors 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.

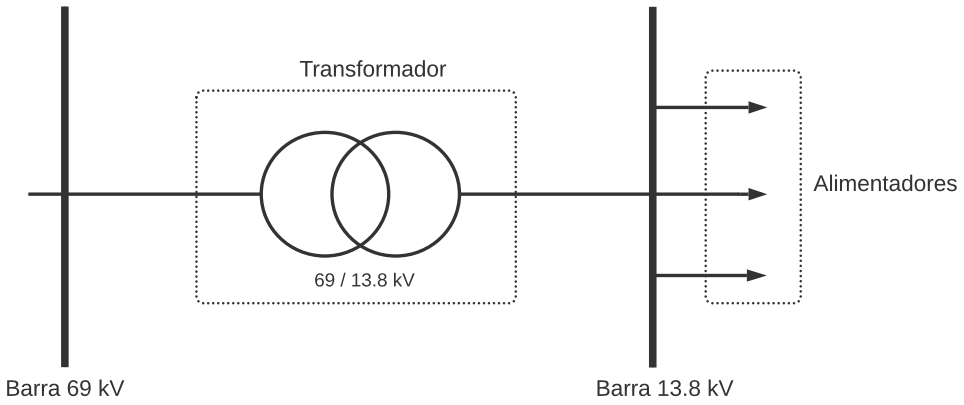
O **Keras** é uma biblioteca de rede neural de código aberto escrita em Python. De acordo com (WIKIPEDIA, 2020a) o keras foi projetado para permitir experimentação rápida com redes neurais profundas, ele se concentra em ser fácil de usar, modular e extensível, funcionando em cima da plataforma ***tensorflow*** 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, usa-se 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.

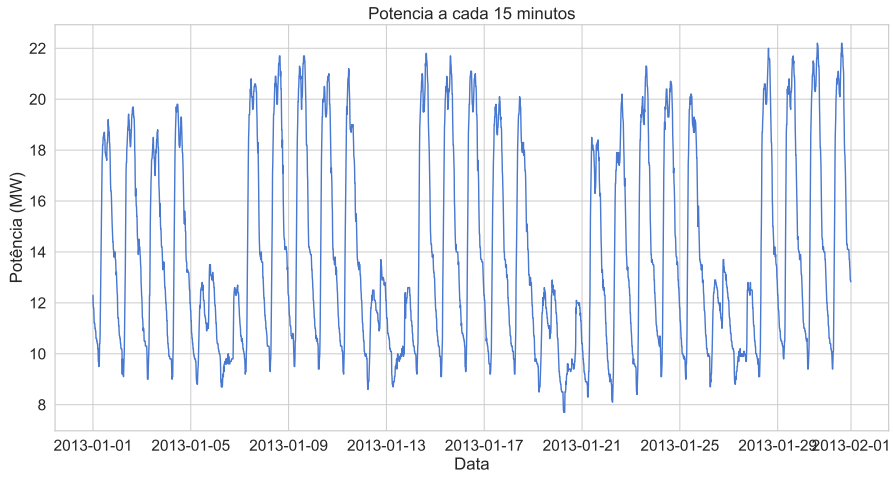
Figura 13 – Diagrama unifilar resumido de uma subestação.



Fonte: Autoria própria.

Neste trabalho, utiliza-se especificamente dados da subestação João Pessoa (JPS) 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. 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 14.

Figura 14 – Dados para treino e validação.

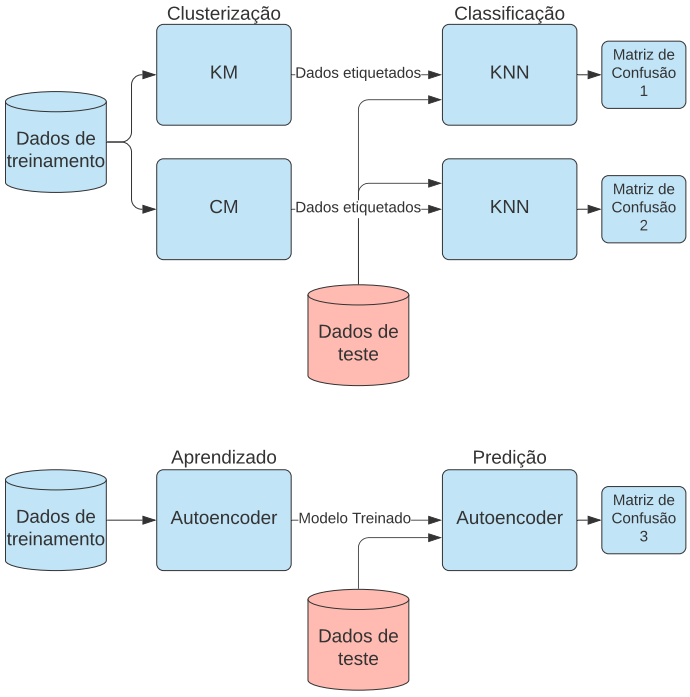


Fonte: Autoria própria.

3.2 MÉTODOS

Neste trabalho serão abordados métodos não supervisionados, a exemplo do K-Means e Fuzzy C-Means e métodos supervisionados, como o Autoencoder, que serão utilizados como base para processo de detecção de outliers, conforme apresentado no diagrama de fluxo da Figura 15.

Figura 15 – Processo de deteção de outlier no KM, CM e Autoencoder.

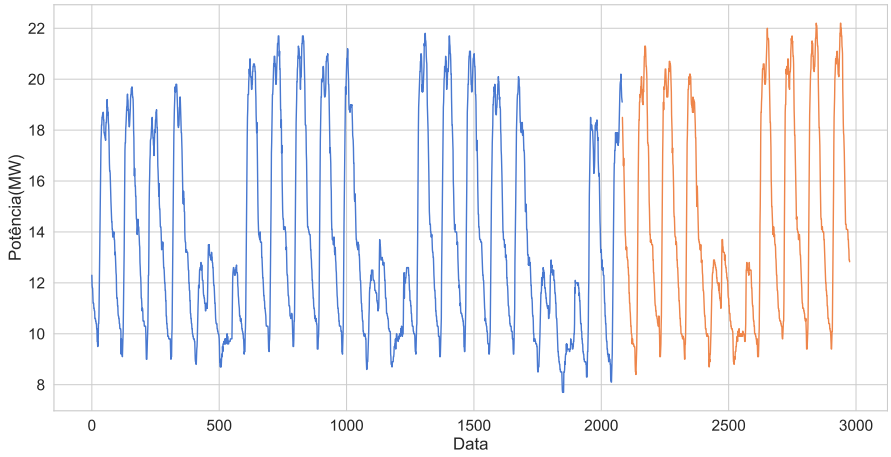


Fonte: Autoria própria.

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 16).

Figura 16 – Dados separados treino/validação.

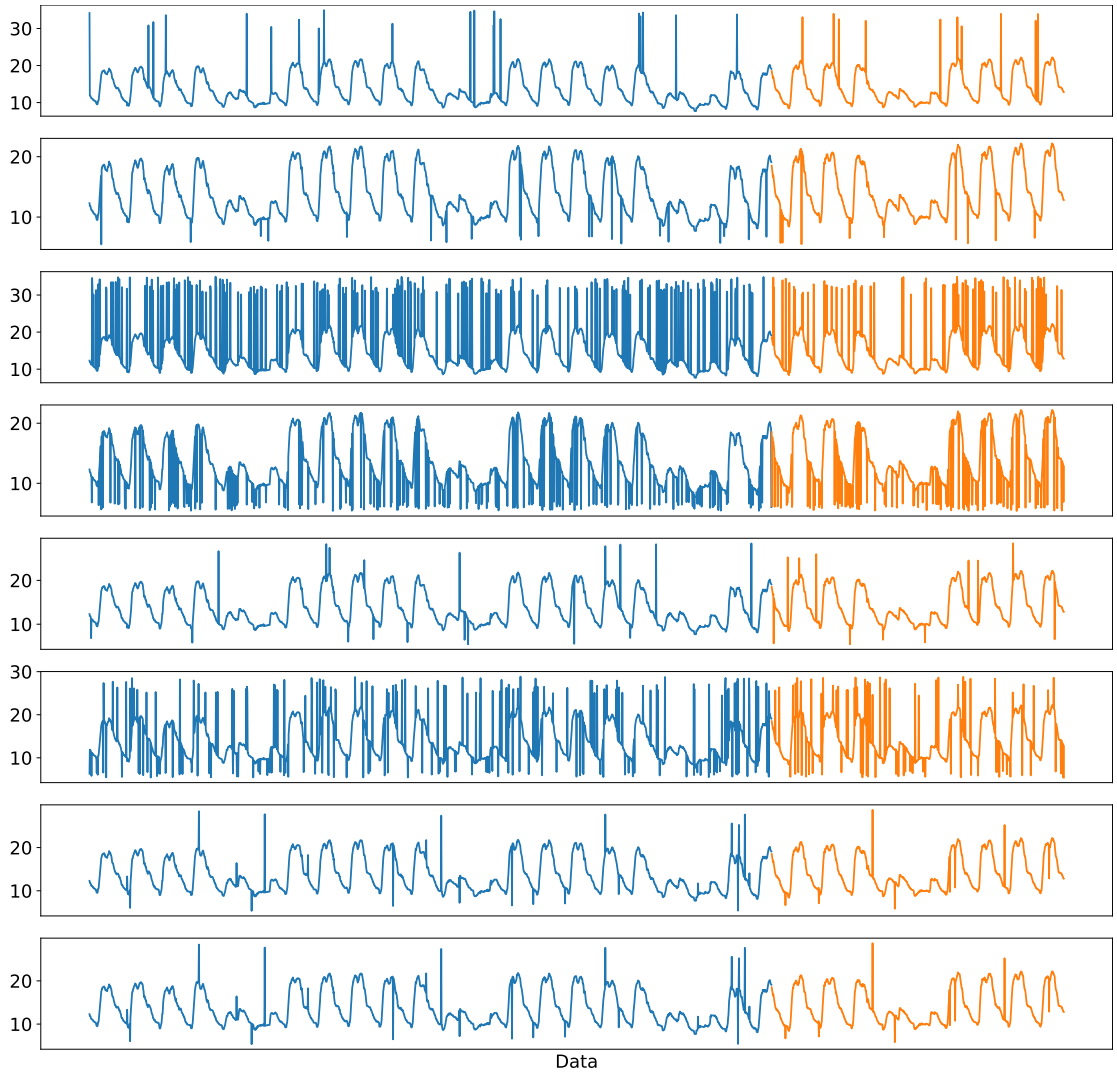


Fonte: Autoria própria.

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 17 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).

Figura 17 – 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

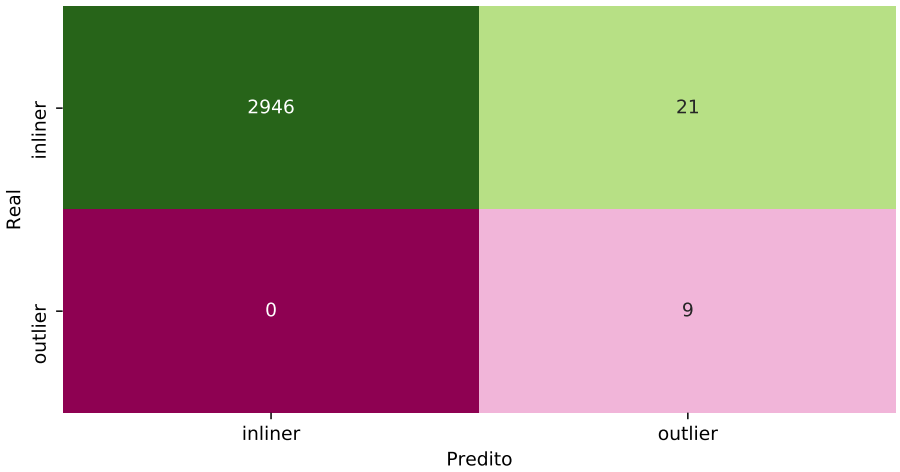
No problema de classificação estatística, uma matriz de confusão, também conhecida como matriz de erros (STEHMAN, 1997), é um layout de tabela específico que permite a visualização do desempenho de um algoritmo, tipicamente um aprendizado supervisionado (em aprendizado não supervisionado, geralmente 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 18 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 ultimo quadrante representa os TN.

Figura 18 – Matriz de confusão



Fonte: Autoria própria.

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 escore 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 será discutido e determinado alguns parâmetros necessários para a construção dos modelos de detecção de outliers utilizando as técnicas anteriormente

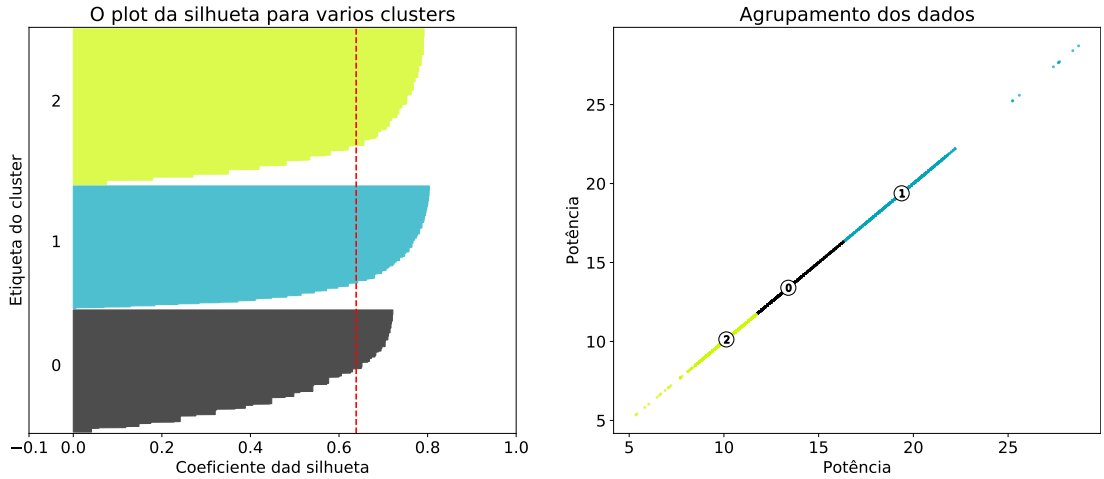
mencionadas.

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 19 – Análise pelo método da silhueta com 3 clusters

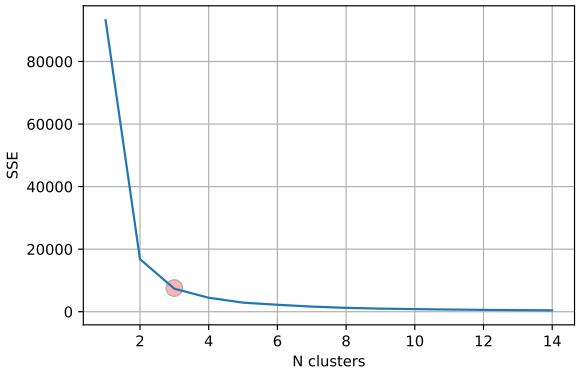


Fonte: Autoria própria.

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. A Figura 19 apresenta-se bem essa situação, no 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 20, o "cotovelo"no braço é o valor ideal de k (número do cluster).

Figura 20 – Análise pelo método do cotovelo



Fonte: Autoria própria.

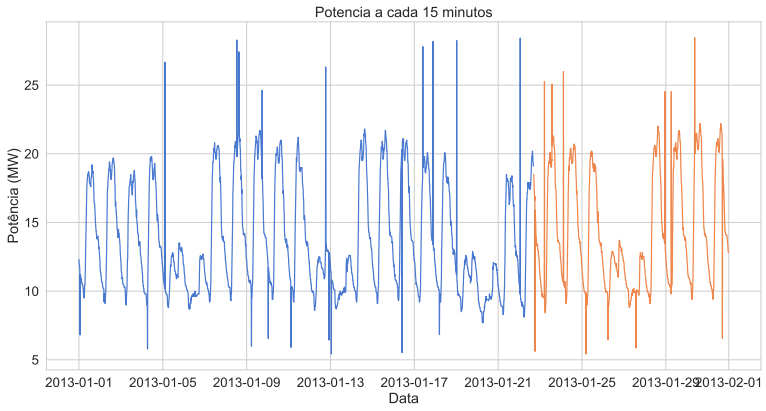
Verificado que o valor 'ideal' de número de clusters foi revelado como sendo 3 por meio de dois métodos, este será o valor utilizado como referência no trabalho.

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)),(Figura 21). As técnicas individualmente estão detalhadas no capítulo 2.

Figura 21 – Dados caso 5



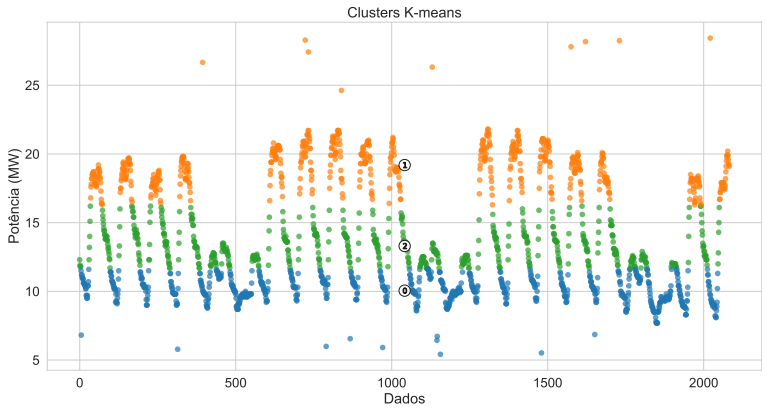
Fonte: Autoria própria.

3.2.5.1 KM + KNN

Utilizando o algoritmo KM, realizou-se a etiquetagem dos dados de treinamento por meio de clusterização (Figura 22), sinalizando os outliers aqueles com uma distância euclidiana superior a $\mu \times 4 \times \sigma$, conforme equação (3.4). O resultado é apresentado na Figura 23.

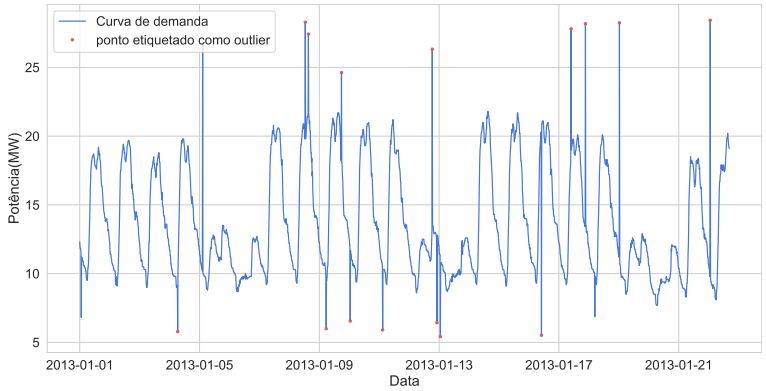
$$outlier = \|x - \mu_i\|^2 > (\mu \times 4 \times \sigma) \tag{3.4}$$

Figura 22 – K-Means: Clusterização



Fonte: Autoria própria.

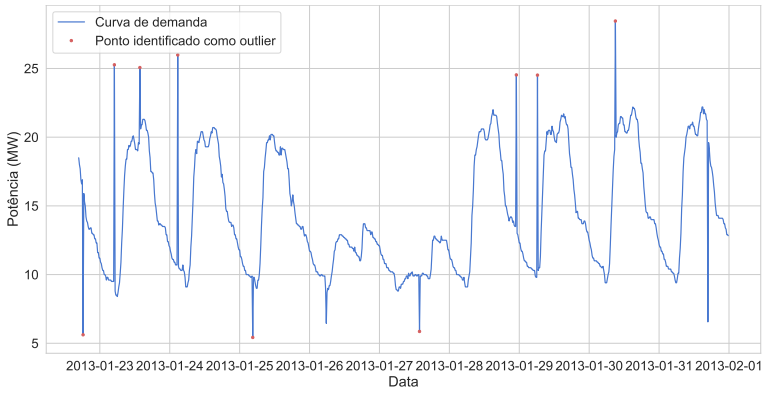
Figura 23 – K-Means: Etiquetagem



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 24). Para este trabalho foi escolhido o valor de $k = 5$ e a distância euclidiana como parâmetros do KNN pois estes resultaram em uma boa performance (tentativa e erro).

Figura 24 – K-Means: Classificação dos outliers com KNN

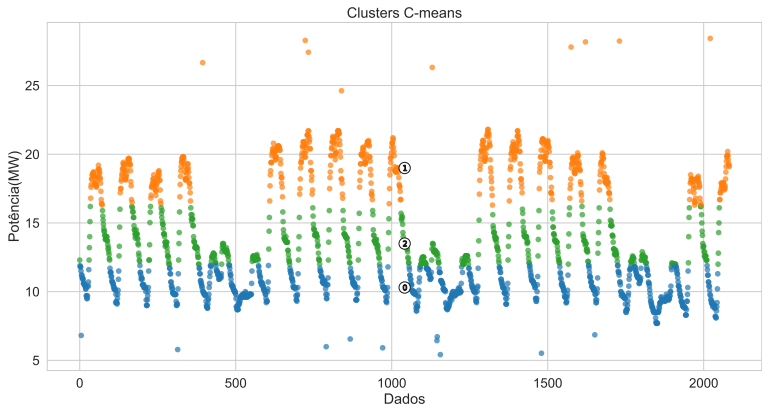


Fonte: Autoria própria.

3.2.5.2 FCM + KNN

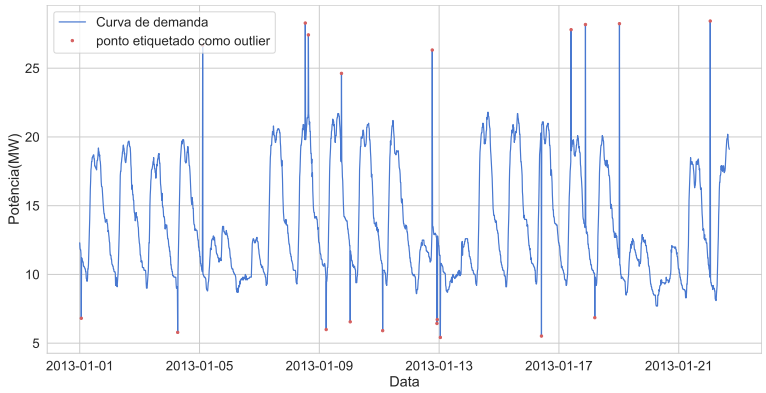
Utilizando o algoritmo FCM, realizou-se a etiquetagem dos dados de treinamento por meio de clusterização (Figura 22), sinalizando os outliers aqueles com uma distância euclidiana superior a $\mu \times 4 \times \sigma$ (mesmo caso do KM, equação (3.4)) e coeficiente de fuzzificação igual a 50 (Figura 26).

Figura 25 – Fuzzy C-Means: Clusterização



Fonte: Autoria própria.

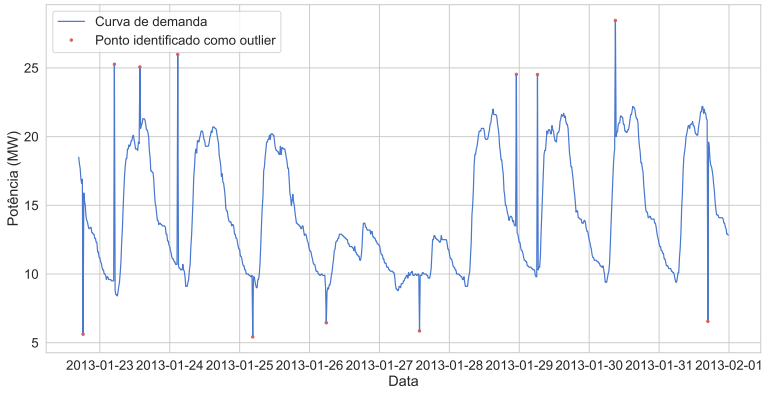
Figura 26 – Fuzzy C-Means: Etiquetagem



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 27). Para este trabalho foi escolhido o valor de $k = 5$ e a distância euclidiana como parâmetros do KNN pois estes resultaram em uma boa performance (tentativa e erro).

Figura 27 – C-Means: Classificação dos outliers com KNN

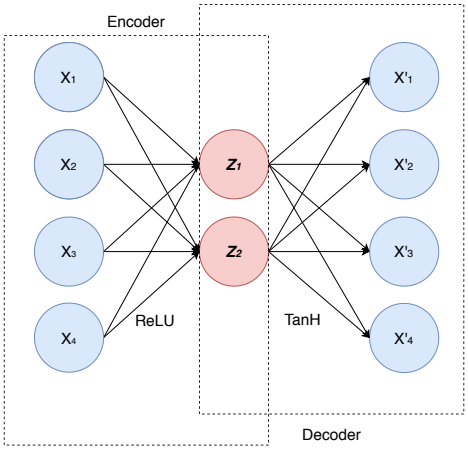


Fonte: Autoria própria.

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 28. Os parâmetros externos do modelo foram escolhidos conforme Tabela 1.

Figura 28 – Modelo Autoencoder



Fonte: Autoria própria.

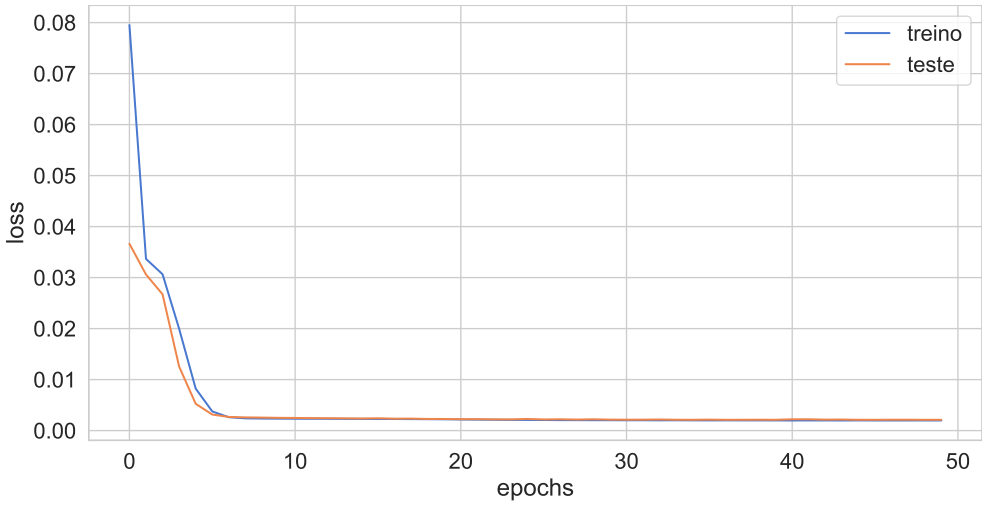
Tabela 1 – Autoencoder: Parâmetros externos de treinamento

Parâmetros	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 29.

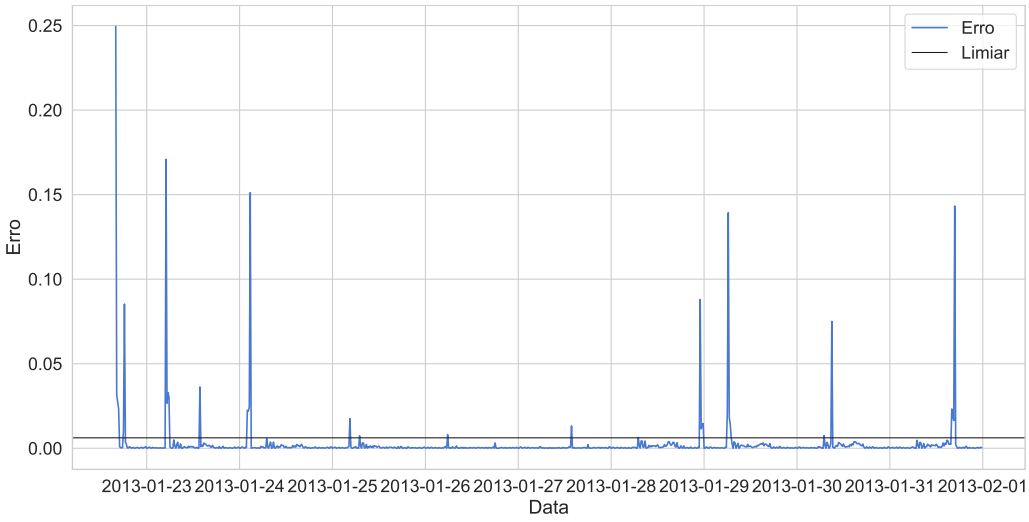
Para detecção dos outliers, é colocado um limiar na curva de erro (Figura 30), no qual os valores que excederem este limiar serão considerados outliers. O limiar foi setado como sendo 5 vezes a média truncada em 1%.

Figura 29 – Autoencoder: Erro de treinamento



Fonte: Autoria própria.

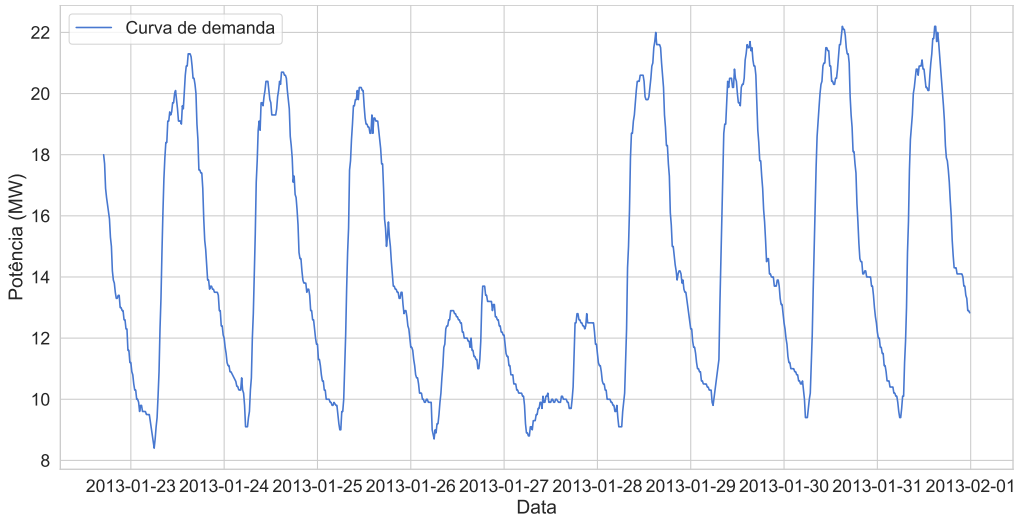
Figura 30 – Autoencoder: Curva de erro e limiar



Fonte: Autoria própria.

Por fim, os dados são corrigidos por meio da interpolação linear (Figura 31) e as métricas de avaliação são utilizadas para avaliação dos resultados.

Figura 31 – Autoencoder: Correção dos dados



Fonte: Autoria própria.

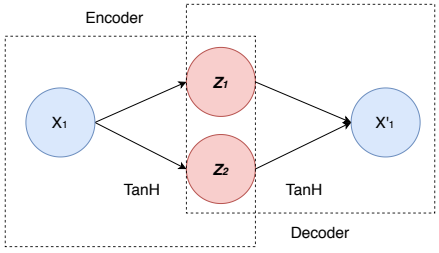
3.2.5.4 AUTOENCODER ESPARSO

Para a construção deste modelo optao-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 32. Os parâmetros externos do modelo foram escolhidos conforme Tabela 2.

Tabela 2 – Autoencoder: Parâmetros externos de treinamento

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

Figura 32 – Modelo Autencoder Esparso

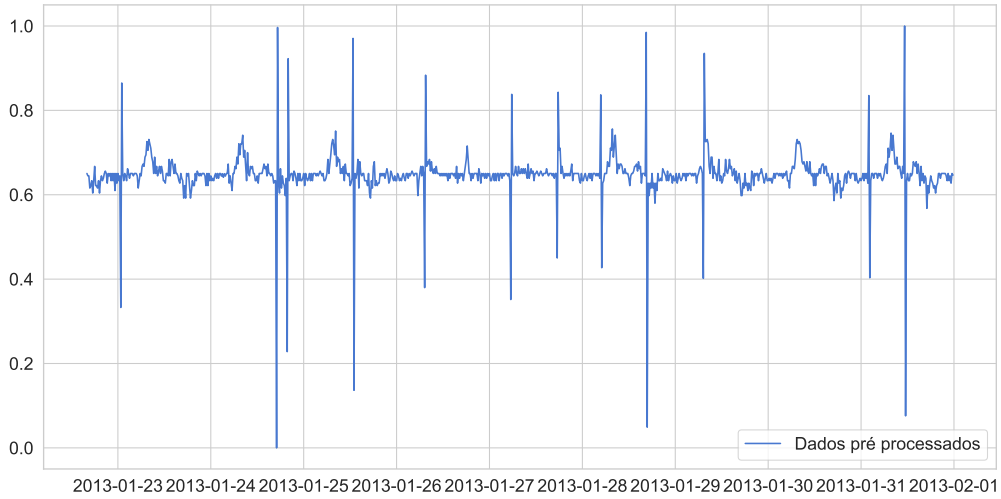


Fonte: Autoria própria.

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 famoso 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 33.

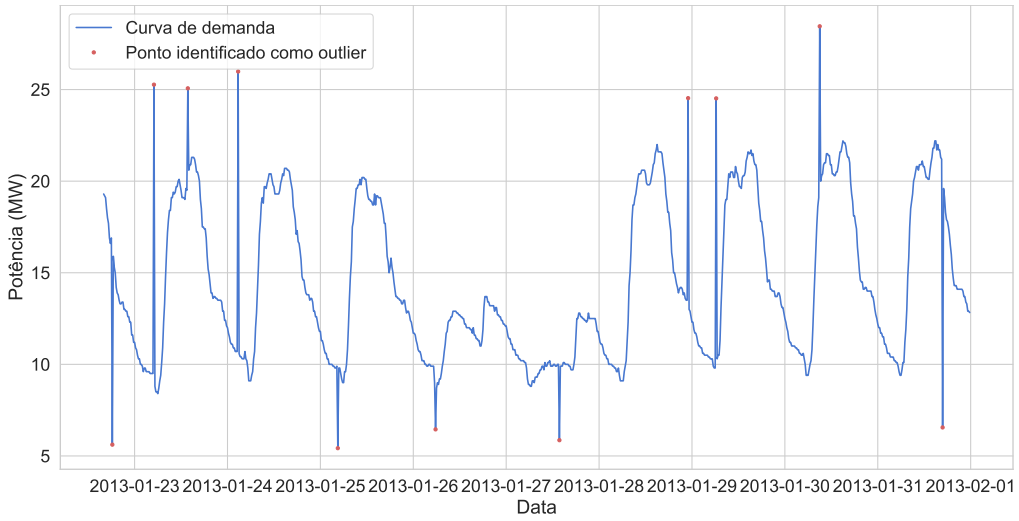
Figura 33 – Autoencoder Esparso: Dados pré processados



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.

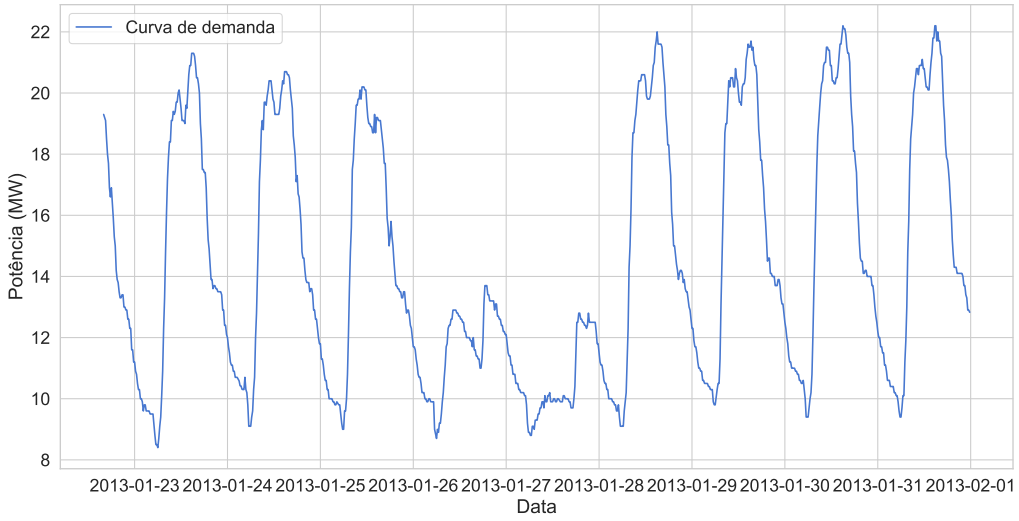
Figura 34 – Autoencoder Esparso: Detecção de outliers



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 setado também como sendo 3 vezes a média truncada em 1%. Por fim, os dados são corrigidos através da interpolação linear (Figura 35).

Figura 35 – Autoencoder Esparso: Correção dos dados



Fonte: Autoria própria.

4 RESULTADOS

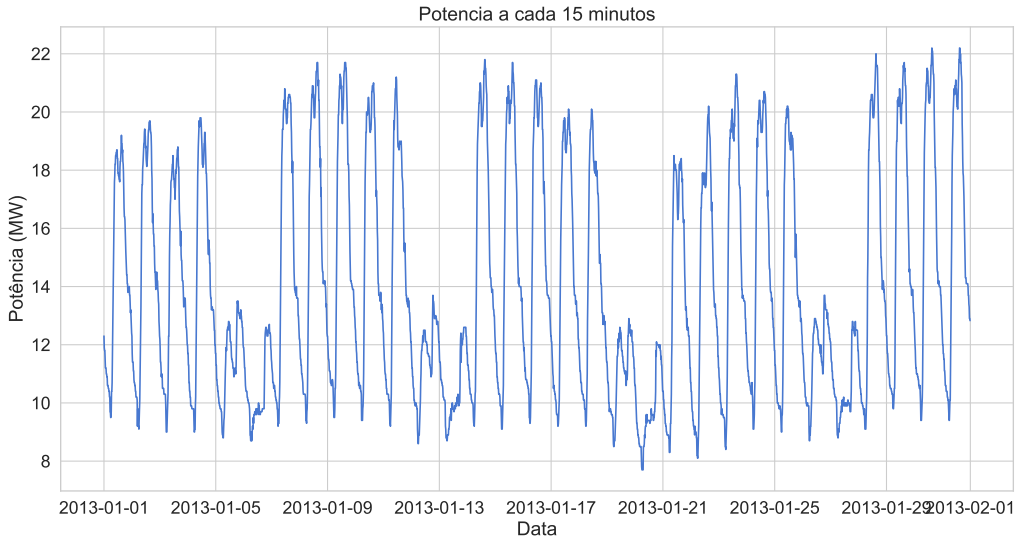
Neste capítulo será demonstrado e discutido os resultados obtidos neste trabalho, trazendo os exemplos e aplicações do trabalho desenvolvido. Inicialmente será realizada uma comparação entre os métodos KM+KNN e FCM+KNN como ferramenta para detecção de outliers, com levantamento de algumas avaliações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

Em seguida, o método com melhor performance será comparado aos modelos de *deep learning* autoencoders, novamente com algumas avaliações e considerações. Por fim, será apresentada uma tabela que irá ilustrar o todos os métodos estudados em virtude das métricas de avaliação que foram proposta.

4.1 DESCRIÇÃO DO CONJUNTO DE DADOS

Para realizar as primeiras comparações utilizao-se dados de 1 mês de medição da SE JPS compreendendo o período de 01 de janeiro de 2013 até 31 de janeiro de 2013 (Figura 36).

Figura 36 – Dados janeiro SE JPS

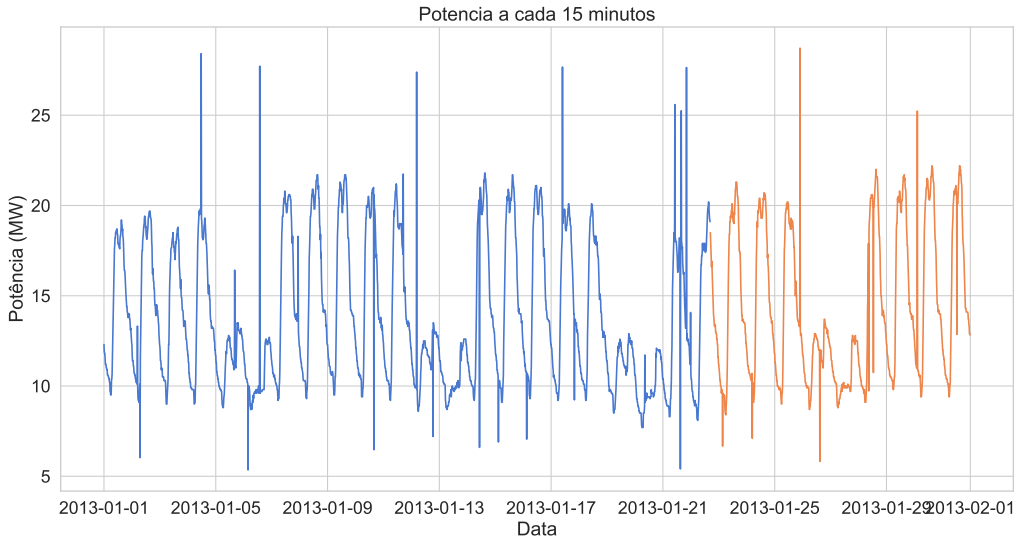


Fonte: Autoria própria.

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 separados para 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 37.

Figura 37 – Dados separados treino/validação e com outliers inseridos



Fonte: Autoria própria.

Este conjunto representa o *Caso 7* especificado no capítulo de metodologia.

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

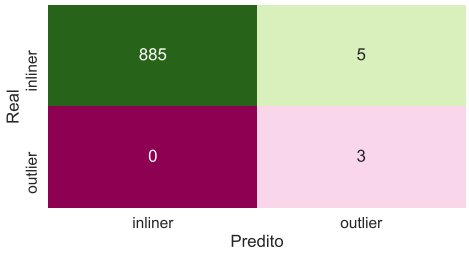
Nesta seção será realizada a comparação dos algorítimos KM+KNN e FCM+KNN em relação ao conjunto de dados da seção 4.1 utilizando as métricas de avaliação anteriormente descritas. A tabela 3 trás os erros obtidos na avaliação dos modelos. As Figuras 38 e 39 apresentam as matrizes de confusão resultante de ambos os modelos e por fim as Figuras 40 e 41 mostraram os dados corrigidos após a detecção.

Tabela 3 – Caso A: Resultado

	MSE	**MAPE**	**MCC**
KM	0.263	0.22%	0.61
FCM	0.253	0.18%	0.70

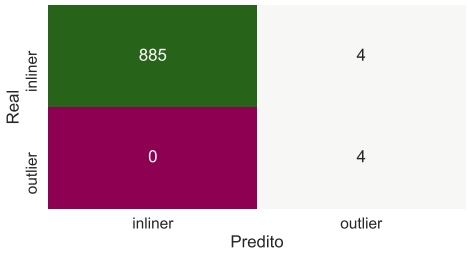
É possível verificar dos resultados que o algoritmo FCM+KNN teve um melhor desempenho no critério do MCC e MAPE em relação ao KM+KNN. Tal resultado se deve principalmente ao parâmetro fuzzificador existente no método CM, que ameniza a influência dos outliers na posição dos centroides dos clusters, permitindo que estes se mantenham mais afastados e assim possam detectar os outliers que se encontram mais distantes através da distância euclidiana.

Figura 38 – Caso A: Matriz de confusão KM+KNN



Fonte: Autoria própria.

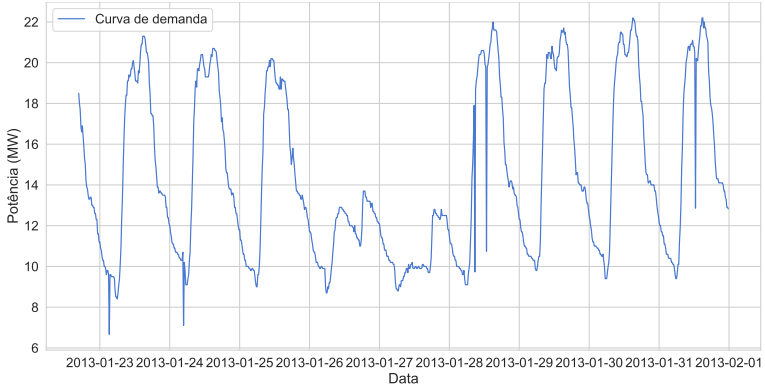
Figura 39 – Caso A: Matriz de confusão FCM+KNN



Fonte: Autoria própria.

Assim, pode-se observar que ambos os métodos não fizeram detecção de dados *inliers* como *outliers* (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.

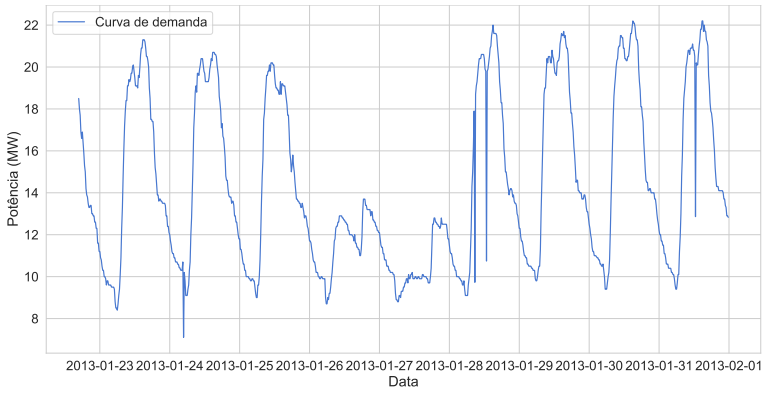
Figura 40 – Caso A: Dados corrigidos KM+KNN



Fonte: Autoria própria.

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

Figura 41 – Caso A: Dados corrigidos FCM+KNN



Fonte: Autoria própria.

4.3 CASO B: ALGORÍTIMO FCM+KNN E AUTOENCODER

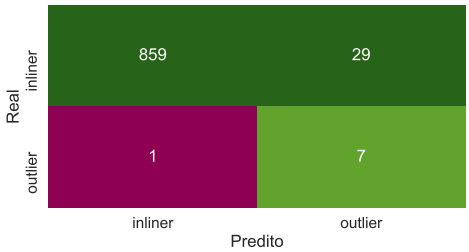
Desta vez, foram avaliados o FCM+KNN em relação ao Autoencoder padrão. A Tabela 4 trás os erros obtidos na avaliação dos modelos. As Figuras 42 e 43 ilustram respectivamente a matriz de confusão e os dados corrigidos após a detecção do algoritmo autoencoder.

Tabela 4 – Caso B: Resultado

	MSE	**MAPE**	**MCC**
CM	0.253	0.18%	0.70
Autoencoder	0.019	0.11%	0.40

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 7 dos 8 outliers foram detectados, inclusive os 3 outliers do tipo contextual, que não foram detectados pelo CM.

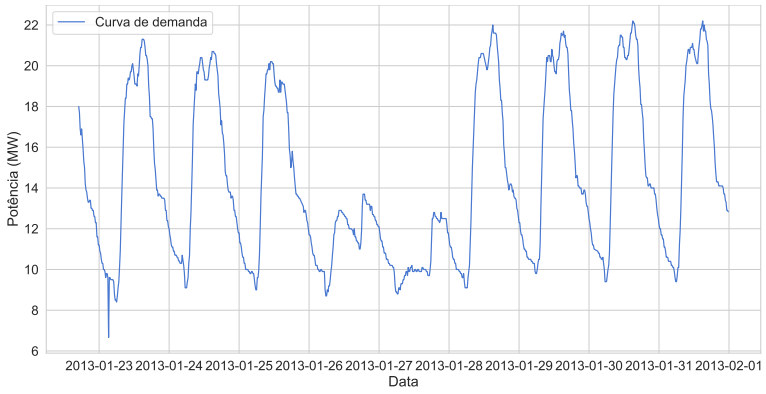
Figura 42 – Caso B: Matriz de confusão Autoencoder



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

Figura 43 – Caso B: Dados corrigidos Autoencoder



Fonte: Autoria própria.

inliers foram detectados erroneamente. 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 menor quando comparados com os dados originais limpos.

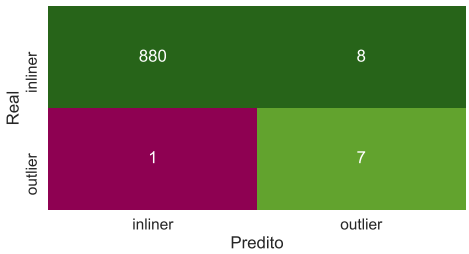
4.4 CASO C: ALGORÍTIMO AUTOENCODER E AUTOENCODER ESPARSO

Por fim, finalizando esta etapa de comparações, é trazido o algoritmo autoencoder esparso à prova. A Tabela 5 trás os erros obtidos na avaliação dos modelos. As Figuras 44 e 45 ilustram respectivamente a matriz de confusão e os dados corrigidos após a detecção do algoritmo autoencoder esparsado.

Tabela 5 – Caso C: Resultado

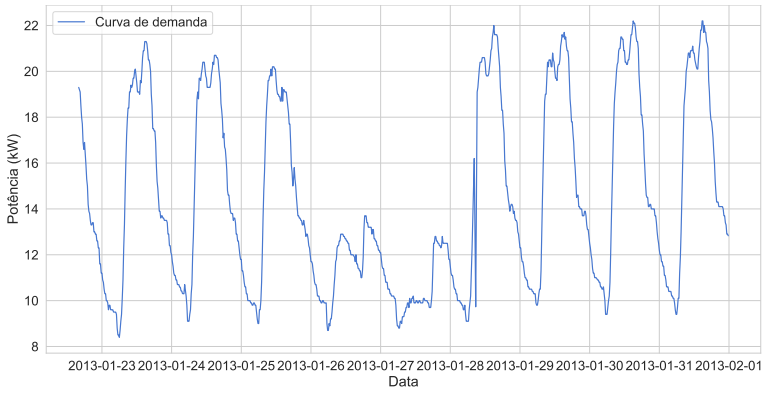
	MSE	**MAPE**	**MCC**
Autoencoder	0.019	0.11%	0.40
Autoencoder Esparsado	0.13	0.14%	0.63

Figura 44 – Caso C: Matriz de confusão Autoencoder Esparso



Fonte: Autoria própria.

Figura 45 – Caso C: Dados corrigidos Autoencoder Esparso



Fonte: Autoria própria.

O resultado foi praticamente igual ao do autoencoder, com diferença que o esparsado fez menos detecção de FN.

4.5 COMPARATIVO DE TODOS OS MÉTODOS

Neste estudo, optou-se por analisar a Tabela 6 contendo o resultado de todos os casos desenvolvidos na metodologia em relação a todos os algoritmos de detecção de outliers neste trabalho abordados.

Tabela 6 – 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:
Z-Score	**MSE**	0.0	0.74	27.89	7.8	0.78	11.34	0.53	2.85
	MAPE	0.01%	0.51%	11.45%	5.49%	0.54%	6.93%	0.57%	3.36%
	MCC	1.00	0.00	0.0	0.0	0.60	0.0	0.0	0.31
Z-Score	**MSE**	0.00	0.74	0.00	7.8	0.41	4.90	0.53	2.70
Modificada	**MAPE**	0.01%	0.51%	0.08%	5.49%	0.28%	3.32%	0.57%	3.25%
	MCC	1.00	0.00	1.00	0.0	0.736	0.64	0.0	0.359
K-Means	**MSE**	3.46	0.00	27.89	7.8	0.239	10.15	0.46	2.587
	MAPE	1.4%	0.01%	11.45%	5.49%	0.11%	6.17%	0.42%	3.05%
	MCC	0.0	1.0	0.0	0.0	0.90	0.37	0.497	0.429
C-Means	**MSE**	0.00	0.00	27.89	7.8	0.00	5.86	0.46	2.54
	MAPE	0.01%	0.01%	11.45%	5.49%	0.01%	3.93%	0.42%	2.96%
	MCC	1.0	1.0	0.0	0.0	1.0	0.58	0.497	0.45
Autoencoder	**MSE**	0.00	0.07	0.80	3.488	0.01	3.15	0.052	1.56
	MAPE	0.05%	0.18%	0.47%	3.9%	0.1%	2.83%	0.19%	2.36%
	MCC	0.58	0.52	0.919	0.439	0.458	0.58	0.656	0.526
Sparsed	**MSE**	0.00	0.00	4.97	1.31	0.00	1.55	0.00	0.339
Autoencoder	**MAPE**	0.01%	0.01%	2.86%	1.88%	0.03%	1.51%	0.05%	0.74%
	MCC	1.0	1.0	0.868	0.756	0.85	0.82	0.60	0.886

A tabela 6 ilustra 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 0,8. É importante mencionar que os casos 7 e 8 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 através da interpolação linar, um FP gera um erro muito pequeno, muitas vezes quase nulo, em relação a curva original sem contaminação, por outro lado, um FN trás um grande prejuízo agregado à deteção. Por exemplo, no caso B e C foi possível observaro que existiu a deteção 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 foi apresentando diversos métodos e técnicas com o objetivo de construção de modelos para realizar a deteção e correção de outliers em curvas de demanda.

Foram construídos modelos não supervisionados baseados no Z-SCORE+KNN, Z-SCORE MODIFICADO+KNN, KM+KNN, FCM+KNN, e modelos supervisionados deeplearning baseados em Autoencoder e Autencoder Esparso. Tais modelos foram postos a provas em conjunto de dados selecionados, tratados e com outliers inseridos virtualmente conforme explanado na seção 3.2.1.

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 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.

Para finalizar, o modelo deeplearning 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 6. 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*.

A performance dos modelos apresentados 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 deteção.

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 e o Autoencoder Esparsado quando se tratando de modelo supervisionado de *deeplearning.*

Para um trabalho futuro 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;
- Comparação do z-score com janelamento e sem janelamento;

Por fim, esse trabalho contribuiu de forma direta para a ampliação dos conhecimentos adquiridos ao longo a graduação em engenharia elétrica do autor, sobretudo na área de automação 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 14.

BABUŠKA, R. *Fuzzy modeling for control*. [S.l.]: Springer Science & Business Media, 2012. v. 12. Citado na página 28.

BEZDEK, J. C. Objective function clustering. In: *Pattern recognition with fuzzy objective function algorithms*. [S.l.]: Springer, 1981. p. 43–93. Citado na página 28.

BHATIA, N. et al. Survey of nearest neighbor techniques. *arXiv preprint arXiv:1007.0085*, 2010. Citado na página 29.

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 39.

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 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 39.

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 29.

DAS, S. *Time series analysis*. [S.l.]: Princeton university press, Princeton, NJ, 1994. v. 10. Citado 2 vezes nas páginas 14 e 17.

DAVIS, P. J. *Interpolation and approximation*. [S.l.]: Courier Corporation, 1975. Citado na página 32.

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 28.

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 23.

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 24.

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 22.

HAWKINS, D. M. *Identification of outliers*. [S.l.]: Springer, 1980. v. 11. Citado 2 vezes nas páginas 14 e 22.

HECHT-NIELSEN, R. Theory of the backpropagation neural network. In: *Neural networks for perception*. [S.l.]: Elsevier, 1992. p. 65–93. Citado na página 31.

HÖPPNER, F. et al. *Fuzzy cluster analysis: methods for classification, data analysis and image recognition*. [S.l.]: John Wiley & Sons, 1999. Citado na página 28.

HYNDMAN, R. J.; ATHANASOPOULOS, G. *Forecasting: principles and practice*. [S.l.]: OTexts, 2018. Citado 2 vezes nas páginas 17 e 48.

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 25.

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 30 e 31.

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 31.

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 24.

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 31.

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 27 e 40.

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 30.

KUHLMAN, D. A python book: Beginning python. *Advanced Python, and Python Exercises*, 2012. Citado na página 33.

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 25.

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 39.

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 28.

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 18.

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 29.

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 33.

SCHMIDHUBER, J. Deep learning in neural networks: An overview. *Neural networks*, Elsevier, v. 61, p. 85–117, 2015. Citado na página 30.

SHARMA, S. Activation functions in neural networks. *Towards Data Science*, v. 6, 2017. Citado na página 31.

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 24.

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 21.

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 19.

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 38.

WIKIPEDIA. *Keras — Wikipedia, The Free Encyclopedia*. 2020. Disponível em: <https://en.wikipedia.org/wiki/keras>. Acesso em: 17 de junho de 2020. Citado na página 34.

WIKIPEDIA. *Linear interpolation — Wikipedia, The Free Encyclopedia*. 2020. Disponível em: <https://en.wikipedia.org/wiki/Linear_interpolation>. Acesso em: 04 de agosto de 2020. Citado na página 32.

WIKIPEDIA. *scikit-learn — Wikipedia, The Free Encyclopedia*. 2020. Disponível em: <https://en.wikipedia.org/wiki/Scikit-learn>. Acesso em: 24 de junho de 2019. Citado na página 33.

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 22.

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 30.

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 27.

ZAREMBA, W.; SUTSKEVER, I.; VINYALS, O. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*, 2014. Citado na página 48.