Redução de Dimensionalidade para Árvores Aleatórias

Walter Bueno[1], Olavo Silva[1], José A. Nacif[1], Ricardo Ferreira[1]

[1]email: walter.bueno@ufv.br, Universidade Federal de Viçosa, Brazil

Resumo. *A redução de dimensionalidade pode simplificar os modelos de aprendizado de máquina, melhorando o desempenho computacional sem perda de informações relevantes. Este artigo investiga a aplicação de métodos de redução de dimensionalidade em conjuntos de dados utilizados para a construção de árvores aleatórias, uma técnica amplamente empregada em aprendizado de máquina. Foram analisadas várias abordagens, incluindo Análise de Componentes Principais (PCA), t-Distributed Stochastic Neighbor Embedding (t-SNE), técnicas de compressão com K-means e coeficiente Gini, avaliando os impactos nos modelos de árvores aleatórias. Os resultados indicam que é possível realizar a redução de dimensionalidade sem perdas na acurácia das árvores aleatórias. A implementação fez uso do pacote scikit-learn para as técnicas e a base dados sendo load digit. Os experimentos estão disponíveis em um repositório público. Utilizando K-means, obtivemos uma redução de 7,6 vezes no número de nodos das árvores preservando a acurácia.*

1. Introdução

A crescente disponibilidade de dados tem impulsionado o uso de técnicas de aprendizado de máquina para extração de padrões e previsões. Dentre essas técnicas, as árvores aleatórias destacam-se por sua robustez e eficácia. A presença de muitos atributos ou alta dimensionalidade dos dados, aumenta a complexidade, gerando desafios significativos. Uma alternativa é aplicar técnicas de redução de dimensionalidade.

Neste trabalho investigamos técnicas de redução de dimensionalidade como Análise de Componentes Principais (PCA) e técnicas de redução com visualização de dados como t-Distributed Stochastic Neighbor Embedding (t-SNE). Além das duas técnicas, também investigamos métodos de redução de dimensionalidade com técnicas de agrupamento. Pode-se também, após a utilização dessas técnicas, aplicar técnicas de quantização para obter uma redução maior sem perda de acurácia. Desse modo, este estudo tem como objetivo investigar o impacto da aplicação dessas técnicas na construção de modelos de árvores aleatórias. Especificamente, busca-se avaliar a acurácia e a eficiência. Os experimentos e códigos desenvolvidos utilizaram a biblioteca padrão com scikit-learn [Pedregosa et al. 2011].

Este artigo está organizado da seguinte forma. Na Seção 2 apresentamos as técnicas de redução de dimensionalidade avaliadas. A Seção 3 descreve a metodologia de como as técnicas foram avaliadas. A Seção 4 avalia o método K-means na redução de dimensionalidade. A Seção 5 ilustra os principais resultados de experimentos. Finalmente, a Seção 6 apresenta as principais conclusões.

2. Técnicas Avaliadas

Neste trabalho, avaliamos diversas técnicas de redução de dimensionalidade e quantização com o objetivo de otimizar a acurácia e reduzir o tamanho das árvores de decisão geradas.

2.1. PCA

O PCA (Análise de Componentes Principais) [Pearson 1901] é uma técnica de redução de dimensionalidade linear que transforma os dados em um novo sistema de coordenadas, onde as componentes principais capturam a maior variação nos dados. Neste trabalho utilizamos o PCA da biblioteca scikit-learn [Pedregosa et al. 2011] com dois e três componentes principais para avaliar a acurácia das árvores aleatórias. Além disso, avaliamos o desempenho do PCA com e sem quantização. Avaliamos também o impacto na acurácia com a utilização de mais de 3 componentes no PCA e o tamanho das árvores.

2.2. t-SNE

t-SNE (t-distributed Stochastic Neighbor Embedding) [Van der Maaten and Hinton 2008] é um método de visualização de dados, atribuindo a cada ponto de dados uma localização em um mapa bidimensional ou tridimensional, posicionando pontos próximos para objetos semelhantes e pontos distantes para objetos dissimilares. Assim como o PCA, neste trabalho usamos o t-SNE com duas e três dimensões para avaliar a acurácia das árvores aleatórias. Avaliamos a redução com e sem quantização. O t-SNE pode reduzir melhor que o PCA, porém não pode ser usado na inferência pois precisa ser re-calculado para todos os pontos caso um ponto novo seja adicionado.

2.3. Quantização

A quantização é uma técnica simples de redução de dimensionalidade. Neste trabalho usamos uma abordagem de redução na representação binária dos números. Em geral, os números são de ponto flutuante padrão IEEE 754 com 32 bits. Neste formato temos 1 bit de sinal, 8 bits de expoente e 23 bits de mantissa. Seja e_{min} e e_{max} os valores com o maior e menor expoente, respectivamente. Fizemos uma recodificação dos números usando somente a faixa de expoentes. Por exemplo se e_{min} é 2^{-8} e o e_{max} é 2^{+7}, recodificamos com 4 bits para os 16 valores de expoentes, onde $2^{-8} = 0000, 2^{-7} = 0001, \ldots, 2^{7} = 1111$ e removemos a mantissa. Avaliamos a acurácia dos modelos com e sem quantificação.

2.4. K-means

O K-means [Penha et al. 2018] é um algoritmo de agrupamento amplamente utilizado que também pode ser usado para a redução de dimensionalidade [Silva et al. 2023]. Ele particiona os dados em k agrupamentos, minimizando a soma das distâncias entre eles. A métrica de distância mais usada é a distância euclidiana, com a soma dos quadrados das distâncias entre os pontos de dados e os pontos centrais ou centroide de cada agrupamento. A implementação mais usada do algoritmo funciona de forma iterativa, inicializando os k centroides aleatoriamente e, classificando todos os pontos nos agrupamentos mais próximos, e em seguida, atualizando-os os centros de cada agrupamento com base na média dos pontos atribuídos classificado em cada grupo. O processo continua até que os centroides dos grupos não mudem significativamente de posição entre as iterações.

2.5. Coeficiente Gini

O índice de Gini, também conhecido como coeficiente de Gini ou Gini impurity, é uma métrica que mede a impureza ou desordem em um conjunto de dados. Ele é geralmente utilizado na construção de árvores de decisão e random forest [Penha et al. 2023]. O coeficiente Gini pode ser usado no K-means e k-medoids para avaliação da qualidade dos agrupamentos [Laber and Murtinho 2019].

3. Redução de Dimensionalidade

Neste trabalho usamos dois conjuntos de dados padrões de aprendizado de máquina: digits e mnist [Pedregosa et al. 2011]. Ambos são bi-dimensionais, com 8x8 e 28x28 atributos, respectivamente. Na primeira etapa sem quantização, cada amostra é linearizada em uma dimensão, como ilustrado na Figura 1, onde a amostra 8x8 do *digits* é transformada em um vetor de 64 atributos. Em seguida, o vetor é aplicado aos métodos de redução de dimensionalidade (PCA ou t-SNE), reduzindo cada amostra para uma representação com 2 ou 3 atributos. A acurácia do modelo após a redução (com 2 ou 3 dimensões) é comparada com o modelo sobre o conjunto de dados original (com 64 ou 784 atributos).

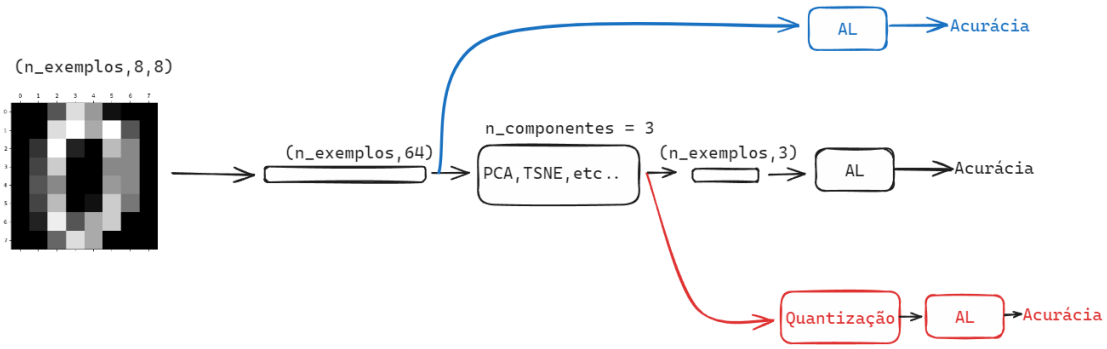

Figura 1. Três abordagens para construção das árvores aleatórias (al): dados originais, redução com PCA ou t-SNE, redução seguida de quantização.

Além da redução, avaliamos também a aplicação da etapa de quantização. Os 2 ou 3 atributos gerados pela redução são números com ponto flutuante. Como explicado na seção 2.3, transformamos cada um destes números de 32 bits para uma representação de 4 a 8 bits usando apenas os expoentes.

4. Quantização com K-means

O K-means pode ser usado para reduzir o número de atributos. Por exemplo, se temos 5 atributos com 8 bits cada, podemos agrupá-los em 4 grupos (k_0 a k_3) como ilustra a Tabela 1. Assim iremos reduzir de $5 \cdot 8 = 40$ bits para 2 bits, pois precisamos de apenas 2 bits para codificar os 4 centroides k_0 a k_3.

Tabela 1. Redução de Dimensionalidade com K-means.

Atributos						
A	B	C	D	E	centroides	Classe
43	23	38	10	1	k_1	X
35	19	45	22	3	k_2	Y
52	30	28	17	5	k_0	X
40	25	33	15	2	k_3	Y
47	20	40	12	4	k_1	X

O K-means é uma técnica de aprendizado não supervisionado. Mas pode ser usado junto com as técnicas de aprendizado supervisionado, como etapa de redução. Suponha o exemplo da Tabela 1, podemos observar que o centroide k_1 tem 2 elementos e ambos são da classe X. Portanto, este grupo em torno de k_1 separa bem os elementos da classe X. Se for um problema supervisionado, podemos usar o Gini para avaliar a qualidade de separação dos dados através do cálculo dos coeficientes Gini.

5. Resultados Experimentais

Todos os experimentos foram conduzidos usando o Google Colab [Canesche et al. 2021] com a biblioteca Scikit-learn [Pedregosa et al. 2011]. Como referência usamos a acurácia do modelo de árvores aleatórias padrão sobre dois conjuntos de dados: Mnist e Digits. Como ilustrado na Figura 1, avaliamos a acurácia em função da redução usando PCA, T-sne e quantização, considerando 4 árvores com profundidade ilimitada.

5.1. PCA e t-SNE com e sem quantização

Tabela 2. Redução e Quantização com Árvores Aleatórias.

Digits 8x8 ou 64 atributos, 1.797 amostras							
	Dados Entrada			Quant 1		Quant 2	
Abordagem	Atributos (bits)	Bits	Acur.	bits	Acur.	bits	Acur.
Original	64 Int 4	256	0,891	-	-	-	-
PCA 2	2 Float 32	64	0,591	10	0,597	8	0,583
PCA 3	3 Float 32	96	0,688	15	0,688	12	0,644
t-SNE 2	2 Float 32	64	0,983	10	0,969	8	0,913
t-SNE 3	3 Float 32	96	0,972	15	0,977	12	0,916
Mnist 28x28 ou 784 atributos, 60.000 amostras							
Abordagem	Entrada (bits)	Bits	Acur.				
Original	784 Int 8	6.272	0,907	-	-	-	-
PCA 2	Float 32	64	0,463	-	-	-	-
PCA 3	Float 32	96	0,460	-	-	-	-
t-SNE 2	Float 32	64	0,96864	-	-	-	-
t-SNE 3	Float 32	96	0,96885	-	-	-	-

A Tabela 2 apresenta um resumo dos resultados da redução e da acurácia para os métodos de PCA e t-SNE com e sem quantização. A primeira linha mostra o conjunto de dados original. A coluna **bits** mostra a quantidade de bits para o modelo original e para a redução gerada pelas abordagens PCA e t-sne com 2 e 3 variáveis. Podemos observar que o PCA com 2 ou 3 componentes tem uma perda significativa de acurácia para 59% e 69%, respectivamente.

Já o t-SNE apresenta ótimos resultados de acurácia. Porém é uma técnica que pode ser usada apenas na compactação dos dados. Pois para ser usada na inferência, se adicionando uma nova amostra, todos os valores do t-SNE devem ser recalculados, o que tem um alto custo computacional. O t-SNE é útil para dizer que o conjunto de dados tem uma boa distribuição e pode ser bem classificado.

Para comparar o efeito da quantização, usamos a metodologia descrita na Seção 2.3. As saídas do PCA e t-SNE geram 2 ou 3 números float 32. Reduzimos para 5 e 4 bits cada saída usando apenas o expoente ajustado entre o menor e maior valor. Assim, dois valores de 32 bits serão mapeados em 10 bits, 5 bits cada para quantização 1 na coluna **quant. 1** e 8 bits na coluna **quant. 2**. Para o conjunto de dados digits com PCA, a quantização teve um bom impacto na redução sem perda de acurácia. Para o t-SNE não teve perda para 5 bits e uma pequena perda com 4 bits.

O conjunto de dados Mnist é 10 vezes maior, sendo 30 vezes mais lento para ser calculado usando a redução com t-SNE. Apesar do t-SNE não ser usado na inferência, pode gerar uma boa compactação sem perda de acurácia, ao contrário, separa bem as

classes que a acurácia final é melhor que os dados originais. Em compensação, o PCA com 2 e 3 componentes gera um resultado com perda de acurácia para a faixa de 40%. Na Tabela 2 não avaliamos a quantização, uma vez que os resultados para PCA não foram satisfatórios para o PCA com 2 e 3 componentes e para t-SNE, o custo computacional seria alto.

5.2. Acurácia versus Componentes PCA

Para melhor avaliar o Mnist e PCA, fizemos a avaliação da acurácia em função dos componentes como ilustra a Figura 2. No lado esquerdo temos a avaliação do *Digits*. Podemos observar que precisamos de 5 ou mais componentes para ter uma acurácia próxima de 90%. Mas é interessante notar que com apenas 1 componente usando a quantização 1 (5 bits), podemos obter 70%. Do lado direito, temos o conjunto de dados Mnist, onde precisamos de mais de 5 componentes para ter 70% e mais de 10 componentes para termos próximo de 85%.

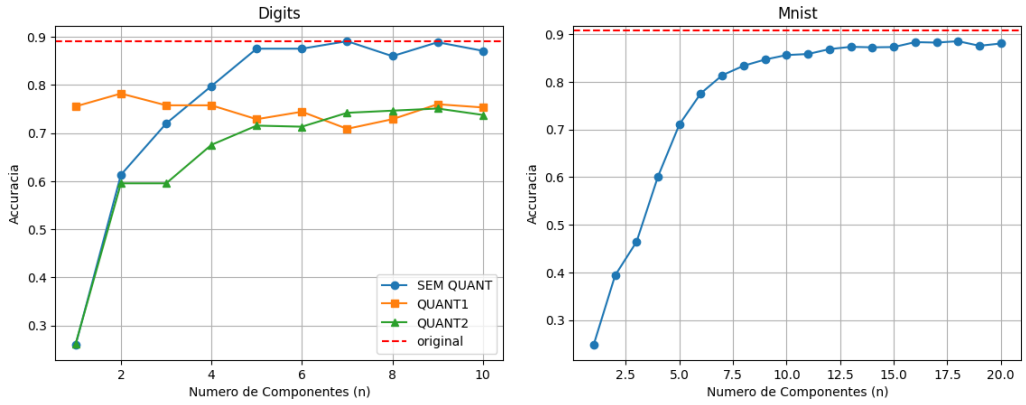

Figura 2. Acurácia versus número de componentes PCA para o Digits e Mnist.

5.3. K-means para Redução

Além do PCA e t-SNE, investigamos o uso do K-means. A Tabela 3 ilustra a distribuição das 10 classes em 15 agrupamento k_0 até o k_{14}. Para $k = 15$ temos a acurácia de 91,8%. Podemos observar que o grupo 1 capturou bem o dígito 0, onde 177 amostras no dígito 0 foram colocadas no grupo 1 que tem um gini próximo de 0, mostrando que separou bem aquela classe. Outro exemplo é o grupo 8 que capturou o dígito 6. Alguns dígitos ficaram mais distribuídos em 2 grupos, como é o caso do dígito 4. Com 15 grupos temos uma redução para 4 bits que é melhor que as reduções apresentadas na Tabela 2 para PCA e t-SNE, sem muita perda de acurácia.

5.4. PCA versus K-means

Podemos verificar qual seria a variação da acurácia em função do valor de k. A Figura 3(a) mostra os valores da acurácia com o valor de K entre 10 e 40. Para 10, que seria o mínimo, a acurácia é de 79,2% e precisamos de 4 bits para representar. Na faixa de 5 bits, o valor de $k = 29$ tem a acurácia de 94,5%. Apesar de ter boa acurácia para valores de K-means na faixa de 15 a 30 em comparação com as outras técnicas e uma ótima compactação dos dados com 4 ou 5 bits, o K-means tem um custo computacional

Tabela 3. Distribuição de classes e coeficientes de Gini por cluster.

Grupo	Gini	10 Classes de dígitos										Total
		0	1	2	3	4	5	6	7	8	9	
0	0,2766	0	1	0	3	0	71	0	0	7	2	84
1	0,0222	177	0	1	0	0	0	1	0	0	0	179
2	0,2487	0	0	11	8	0	0	1	0	138	2	160
3	0,0225	0	0	0	0	87	0	1	0	0	0	88
4	0,3383	0	0	2	4	2	0	0	87	2	11	108
5	0,1079	0	0	7	150	0	0	0	0	0	2	159
6	0,0437	1	0	0	0	88	1	0	0	0	0	90
7	0,4114	0	27	11	0	0	0	0	0	0	0	38
8	0,0223	0	0	0	0	0	1	176	0	1	0	178
9	0,3381	0	0	3	14	0	13	0	0	4	141	175
10	0,0957	0	0	0	2	0	96	0	0	1	2	101
11	0,2754	0	100	1	0	0	0	2	0	16	0	119
12	0,0416	0	0	139	1	0	0	0	0	2	0	142
13	0,4592	0	54	2	0	0	0	0	0	3	19	78
14	0,1168	0	0	0	1	4	0	0	92	0	1	98

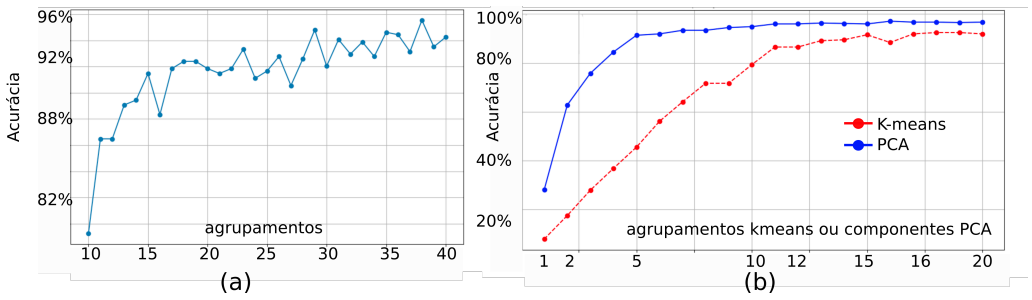

Figura 3. (a) Acurácia x Número de Agrupamentos para o Digits com Kmeans; (b) Comparação entre PCA e K-means.

maior. Enquanto o PCA tem um custo linear de $2 \cdot K \cdot 64$, onde K é o número de componentes, 64 é o número de atributos do *digits* e o fator 2 existe pois é necessário multiplicar pelo peso e somar para cada componente. O K-means tem um custo $3 \cdot k \cdot 64$, pois além da subtração da distância e a multiplicação para elevar ao quadrado, precisa encontrar o centroide mais próximo, que gera um fator 3. Para uma implementação em hardware [da Silva Alves et al. 2023], podemos usar pipeline e o tempo de execução será equivalente para o PCA e o K-means.

5.5. Acurácia e Tamanho das Árvores

Outro ponto é observar o tamanho das árvores que irão fazer a classificação. Se usamos a redução de dimensionalidade, estamos buscando modelos mais simples. A Figura 4 mostra três configurações de árvores para diversos valores de K. Ao usarmos 100 árvores com profundidade ilimitada (Figura 4a), temos acurácia entre 80-95% mas o custo do tamanho das árvores varia de 2.000 a 8.000 nodos. Se reduzimos para 20 árvores com profundidade de 7, a acurácia fica entre 75-95% e reduzimos para 600 a 1.000 nodos como ilustrado na Figura 4b. Finalmente, para 10 árvores temos uma acurácia entre 90-

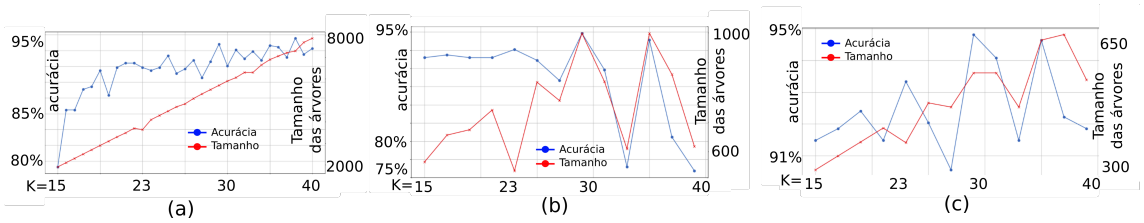

Figura 4. Vários K-means para Digits avaliando acurácia e tamanho das árvores: (a) 100 árv., Prof=Ilimitada; (b) 20 árv.,Prof=7; (c) 10 árv., Prof=10.

95% e apenas 300 a 600 nodos (Figura 4a). Além disso, a configuração k=23 tem uma redução na árvore e uma boa acurácia nas três situações.

5.6. Árvores e Métodos

Tabela 4. Acurácia e Tamanho das 10 árvores com diferentes k's (prof=10).

Número de Clusters (k)	Acurácia	Total de Nodos
19	0,92	370
21	0,91	410
23	0,9333	368

Se fixamos em 10 árvores com profundidade 10, fazendo uma busca refinada com os valores de K iguais a 19, 21 e 23, podemos observar uma boa acurácia e um valor reduzido de nodos nas árvores como ilustrado na Tabela 4. Com relação ao PCA, podemos observar que apesar da boa acurácia, não há redução das árvores como ilustrado na Tabela 5 onde para 21 componentes temos acurácia de 92,2% e um custo de 2.171 nodos.

Tabela 5. Acurácia e Tamanho das Árvores para diferentes valores de PCA

PCA componentes	3	6	9	12	15	18	21
Acurácia	0,741	0,889	0,919	0,926	0,926	0,925	0,922
Tamanho das Árvores	2.505	2.020	1.879	2.048	2.203	2.083	2.171

Finalmente, comparando com o conjunto de dados original do digits sem usar métodos de redução, podemos observar que temos um acurácia melhor mais com um custo bem superior no tamanho das árvores da ordem de 1.741 s 31.678 nodos.

Tabela 6. Random Forest no Conjunto Original digits sem reduções.

Configuração	Acurácia	Tamanho das Árvores
100 árvores, ilimitada	0,98	31.678
20 árvores, profundidade 7	0,95	3.060
10 árvores, profundidade 10	0,94	2.821
30 árvores, profundidade 5	0,92	1.741

6. Conclusões

Esse trabalho apresenta diversas técnicas de redução de dimensionalidade e quantização aplicadas a dois conjuntos de dados padrões da literatura: Mnist e Digits. O desafio é

avaliar quais técnicas são efetivas preservando a acurácia. Avaliamos o PCA, o t-SNE, o K-means e a quantização. Mostramos que existe um amplo espaço para exploração e que devemos observar os custos dos modelos de redução, além da acurácia. Para trabalhos futuros, serão investigados o uso de VQ-VAE para quantização, outras formas de compactação como o ZGIP e reduções mais agressivas explorando o desempenho de GPUs [Bueno et al. 2024] e uso de BDDs [Silva et al. 2024].

Agradecimentos

Apoio financeiro da Bolsa de Iniciação Científica da FAPEMIG programa Institucional, do projeto FAPEMIG APQ-01577-22, do CNPq e da UFV.

Referências

Bueno, W., Barros, O., Nacif, J., and Ferreira, R. (2024). Implementação paralela de múltiplos k-means em gpu. In *Simpósio em Sistemas Computacionais de Alto Desempenho*.

Canesche, M., Bragança, L., Neto, O. P. V., Nacif, J. A., and Ferreira, R. (2021). Google colab cad4u: Hands-on cloud laboratories for digital design. In *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1–5. IEEE.

da Silva Alves, M., Silva, L. B., Penha, J., Ferreira, R., and Nacif, J. A. M. (2023). Kcgra–uma arquitetura reconfigurável de domínio específico para k-means. In *Simpósio em Sistemas Computacionais de Alto Desempenho (SSCAD)*, pages 25–36. SBC.

Laber, E. and Murtinho, L. (2019). Minimization of gini impurity: Np-completeness and approximation algorithm via connections with the k-means problem. *Electronic Notes in Theoretical Computer Science*, 346:567–576.

Pearson, K. (1901). Liii. on lines and planes of closest fit to systems of points in space. *London, Edinburgh, and Dublin philosophical magazine and journal of science*, 2(11).

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830.

Penha, J., da Silva, A. K., Barros, O., Moreira, I., Nacif, J. A. M., and Ferreira, R. (2023). Avaliação de estilos de código para árvores de decisão em gpu com microbenchmarks. In *Anais do XXIV Simpósio em Sistemas Computacionais de Alto Desempenho*.

Penha, J. C., Bragança, L., Coelho, K., Canesche, M., Silva, J., Comarela, G., Nacif, J. A. M., and Ferreira, R. (2018). A gpu/fpga-based k-means clustering using a parameterized code generator. In *Symp on High Performance Computing Systems (WSCAD)*.

Silva, A., Barros, O., Moreira, I. Nacif, J., and Ferreira, R. (2024). Implementações eficientes de random forest em fpga de baixo custo para internet das coisas e computação de borda. In *Simpósio em Sistemas Computacionais de Alto Desempenho*.

Silva, O. A., Silva, A. K., Moreira, Í. G., Nacif, J. A., and Ferreira, R. S. (2023). Rdsf: Everything at same place all at once-a random decision single forest. In *2023 XIII Brazilian Symposium on Computing Systems Engineering (SBESC)*, pages 1–6. IEEE.

Van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, 9(11).