MÉTODOS DE APRENDIZADO DE MÁQUINA PARA CLASSIFICAÇÃO DE SOLOS UTILIZANDO PARÂMETROS *IN SITU*[1]

Alberthus Koops Lordello[2]

RESUMO

Analisar e subsequentemente classificar o solo é parte importante de muitos processos de engenharia. Porém os testes de laboratório são custosos e demorados e seus resultados se referem apenas à localização da qual a amostra foi obtida, precisando extrapolar os resultados para a aplicação em grandes áreas. Ensaios de campo, como o CPTu, têm sido usados devido a velocidade de testagem e consistência dos resultados. Esse estudo busca classificar os solos usando algoritmos de aprendizado de máquina como: árvore de decisão, floresta aleatória, KNN, regressão logística e naive Bayes Gaussiano. Para o treinamento e teste é usado um *dataset* desbalanceado de 1.862 amostras, obtidas combinando diversos *datasets* que contém testes feitos em vários países com características de solo diferentes. Os testes feitos em cada solo é o CPTu e ensaio de laboratório para mensurar a densidade real dos grãos e o peso específico do solo. Obtiveram-se resultados promissores, com acurácia superior a 93%.

Palavras-chave: Aprendizado de Máquina. Classificação. Solo. CPTu. Ensaio de campo.

Abstract

Analyzing the soil and classifying it is an important part of many engineering processes. However laboratory tests are expensive and time-consuming and its results apply only to the sampling location and it needs to be extrapolated before being applied to bigger areas. Field tests, such as CPTu, have been used due to its testing speed, and result consistency. This study aims to classify soils using machine learning algorithms, using decision tree, random forest, KNN, logistic regression and Gaussian Naive Bayes. To test and train the model, an unbalanced dataset of 1862 samples concatenates multiple datasets that are composed from tests made in multiple countries with different soil characteristics. The tests made on each soil sample are CPTu and laboratory tests to measure the soil specific gravity and soil specific weight. We succeeded at obtaining promising results, with accuracy superior to 93%.

Key-words: Machine learning. Classification. Soil. CPTu. Field Test.

---

[1] Trabalho de Conclusão de Curso apresentado como requisito parcial para obtenção do grau de bacharel no Curso de Ciência e Tecnologia no Centro Tecnológico de Joinville (CTJ), Universidade Federal de Santa Catarina (UFSC), sob orientação do Prof. Dr. Ricardo José Pfitscher.

[2] Graduando como Bacharel em Ciência e Tecnologia E-mail: alberthus.lordello@gmail.com.

1. INTRODUÇÃO:

Segundo Aydin (2023), o solo é classificado por meio da identificação de suas características e propriedades físicas. A partir dessas características e propriedades são definidos os parâmetros que possibilitam precisão para a qualificação dos solos e sua identificação através de ensaios de laboratório das amostras coletadas. Adicionalmente, alguns desses parâmetros são passíveis de serem estimados a partir dos resultados de ensaios de campo. Aydin (2023) ressalta que uma adequada prospecção e qualificação do solo é fundamental para projetos geotécnicos apropriados e planejamento das fundações de edificações.

Puri (2018) ressalta que os ensaios de laboratório são custosos, necessitando de equipamentos especializados e uma equipe técnica qualificada para analisar os resultados. Nierwinski *et al.* (2023) acrescenta o desafio adicional de, a partir de uma amostra do solo utilizada nos ensaios de laboratório, extrapolar as informações significativas para toda a área do projeto.

Um complemento aos ensaios de laboratório é o ensaio de penetração do cone com poropressão (CPTu). Conforme relatado por Bhattacharya (2006), nesse ensaio de campo um cone metálico realiza a perfuração do solo à uma velocidade constante. Assim, a partir da força aplicada é obtida a mensuração da resistência do solo. Ainda, pode-se medir a poropressão e o atrito lateral por sensores presentes na ponteira cônica. O CPTu revela as características do solo de forma contínua, permitindo a identificação de diferentes camadas de solo. Desse modo, esse ensaio se diferencia dos testes de laboratórios por serem seus resultados obtidos imediatamente por meio de medição de campo, prescindindo do envio de amostras a serem analisadas em laboratório. As variáveis identificadas por meio do CPTu são: resistência de ponta do cone (qt); resistência do atrito lateral (fs); e poropressão (u).

Nos ensaios de laboratório são mensurados os parâmetros: densidade real dos grãos (*G*); e o peso específico do solo (γ). Conforme descreve Nierwinski *et al.* (2023), a densidade real dos grãos é uma propriedade intrínseca do solo, que permite distinguir um solo de outro. Já o peso específico do solo (γ) é um parâmetro essencial para engenharia geotécnica e projetos de construção civil, atrelado ao cálculo de tensões no solo.

Chandan (2018) relata que há diversos exemplos de uso de algoritmos de aprendizado de máquina para classificação de solo na literatura. Segundo ele, os objetivos de cada algoritmo são diversos, variando de identificação de locais com alta chance de deslizamentos de terra até classificação de solo para uso em projetos de engenharia ou agricultura.

No presente estudo serão usados diversos algoritmos de aprendizado de máquina para classificar solos utilizando os coeficientes obtidos pelo CPTu e os parâmetros G e γ, obtidos em laboratório. Outra característica desse estudo é a utilização de amostras variadas provenientes de diversas localizações, para os diversos tipos de solo.

1.1. *Dataset*

O *dataset* utilizado no presente estudo foi cedido pela pesquisa desenvolvida pela Profa. Helena Nierwinski *et al.* (2023) da UFSC, que por sua vez utilizou dados disponibilizados pelo Dr. Paul W. Mayne, amostras da literatura (OLIVEIRA, 1997; BRUGGER, 1996; BARONI, 2010; BARBOSA, 2018) e dados fornecidos por uma companhia privada. O *dataset* recebido possui informações relativas a dez tipos

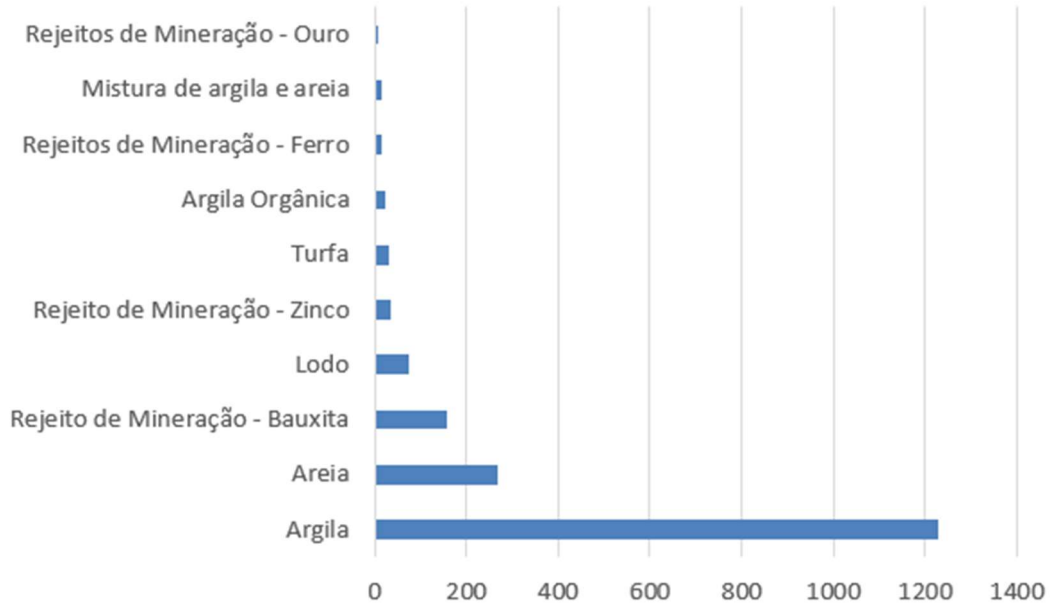
diferentes de solo, totalizando 1.862 amostras que são distribuídas de acordo com a Figura 1. Por fim, a Tabela 1 descreve as variáveis de cada coluna do *dataset*.

Tabela 1 – Descrição do *dataset* e suas variáveis

variável	descrição	obtida por
Solo	Tipo de solo	-
G	Densidade real dos grãos	ensaio em laboratório
qt	Resistência de ponta do cone	CPTu
fs	Resistência de atrito lateral	CPTu
u	Poropressão	CPTu
γ	Peso específico do solo	ensaio em laboratório
mq	coef. de variação da resistência pela profundidade	Derivado do CPTu
Rf	Razão de atrito lateral	Derivado do CPTu

Fone: Autor.

Figura 1 – Distribuição dos tipos de solo no *dataset*



Fonte: Autor.

2. FUNDAMENTAÇÃO TEÓRICA

2.1. Transformada Yeo-Johnson

A transformada de Yeo-Johnso é feita de acordo com a Equação 1 e é considerada uma poderosa ferramenta para determinação automática de transformação exponencial de dados contaminados por *outliers* (RIANI, 2023). Ela é uma expansão da transformação Box-Cox, que aceita amostras com valores nulos e negativos. Ela apresenta bom desempenho para distribuições log-normal e a interpretação do resultado é que a transformada de Yeo-Johnson obtém o λ que mais aproxime o resultado da distribuição gaussiana.

$$y_i^{(\lambda)} = \begin{cases} ((y_i + 1)^{\lambda} - 1)/\lambda & \text{if } \lambda \neq 0, y \geq 0 \\ \ln(y_i + 1) & \text{if } \lambda = 0, y \geq 0 \\ -((-y_i + 1)^{(2-\lambda)} - 1)/(2 - \lambda) & \text{if } \lambda \neq 2, y < 0 \\ -\ln(-y_i + 1) & \text{if } \lambda = 2, y < 0 \end{cases} \quad (1)$$

2.2. Modelos

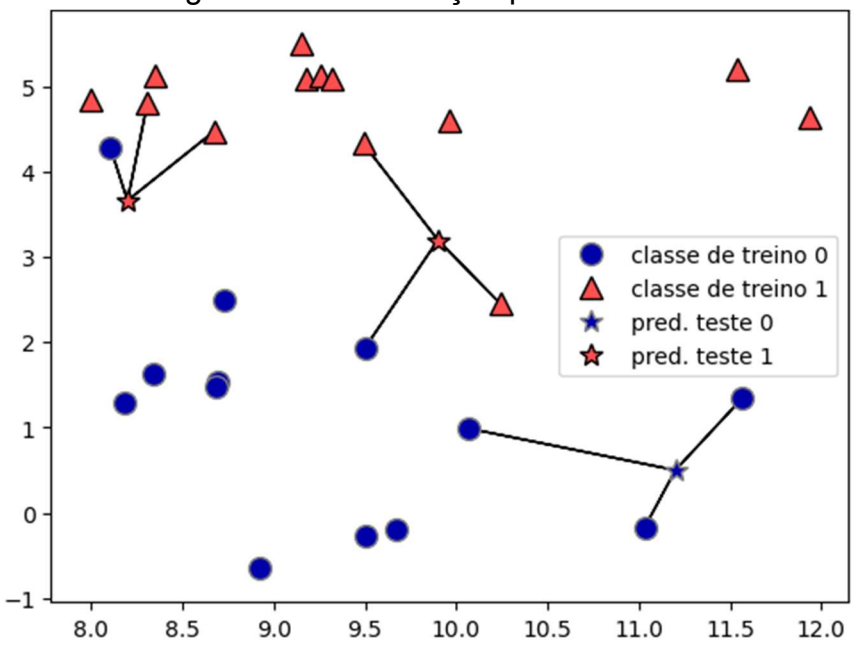
Diversos algoritmos de aprendizado de máquina serão usados a fim de comparar sua efetividade para classificação do solo. Segue uma breve explicação de cada algoritmo avaliado.

2.2.1. KNN (*K-Nearest Neighbor*)

O modelo K-vizinho próximo assume que uma amostra está próxima de outras amostras de mesma classe. Logo, para a classificação binária de uma nova amostra, o algoritmo encontra os k-vizinhos mais próximos nas observações dos dados do treino e, então, faz uma votação para definir a classe predita. A Figura 2 ilustra o processo descrito para k=3 (MÜLLER; GUIDO, 2016).

Para classificação multi-classes, quando há mais de duas classes, o modelo usa a abordagem “um contra o resto” (do inglês, *one-vs-rest*). Nela, para cada classe é feito um modelo de classificação binária, que classifica sua classe contra todas as outras, o resto. Para fazer a predição de um ponto, todos os modelos gerados são avaliados, sendo o resultado aquele para qual a sua classe obteve maior pontuação (MÜLLER; GUIDO, 2016).

Figura 2 – Classificação por KNN usando k =3.



Fonte: MÜLLER; GUIDO, 2016.

2.2.2. Regressão Logística

Na regressão logística o modelo linear dado por $\beta_0 + \beta_1 x$ substitui o *z* na função logística $\frac{1}{1+e^{-z}}$, resultando na Equação 2 (ALBON, 2018):

$$P(y_i = 1 \mid X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}} \quad (2)$$

Em que $P(y_i = 1 \mid X)$ é a probabilidade da i-ésima observação, y_i, ser da classe 1 dado os dados de treino X. Os parâmetros a serem aprendidos são dados por β_0, β_1.

2.2.3. Gaussian Naive Bayes.

O modelo parte do teorema de Bayes descrito pela Equação 3:

$$P(A \mid B) = \frac{P(B \mid A)\, P(A)}{P(B)} \quad (3)$$

A partir do teorema de Bayes obtém-se o teorema classificador de naive Bayes na Equação 4, ao assumir que cada atributo apresenta distribuição gaussiana e é independente dos demais atributos (*naive*). Por consequência dos atributos serem independentes, as probabilidades resultantes também o são (ALBON, 2018).

$$P(y \mid x_1, \cdots, x_j) = \frac{P(x_1, \cdots x_j \mid y) P(y)}{P(x_1, \cdots, x_j)} \quad (4)$$

Em que, de acordo com Albon (2018):

- $P(y \mid x_1, \cdots, x_j)$ é a probabilidade da classe *y* dado os atributos da amostra. É a probabilidade que deseja-se saber para a classificação;
- $P(x_1, \ldots x_j \mid y)$ é a probabilidade dos atributos amostrados sabendo que ele pertence a classe *y*;
- $P(y)$ é a probabilidade da classe *y* nos dados de treino; e
- $P(x_1, \ldots, x_j)$ é a probabilidade dos atributos sem considerar sua classe.

2.2.4. Árvore de decisão

Para classificar se uma pessoa é idosa, por exemplo, perguntamos se ela tem 60 anos ou mais. Esses testes são feitos intuitivamente com base em conhecimentos prévios. Para o algoritmo de árvore de decisão os testes são feitos a partir dos dados de treinamento, em que no seu processo de aprendizado se formula os testes sequenciais para realizar a classificação corretamente.

Para formular os testes, o algoritmo faz uso de critérios matemáticos calculados durante o processo de aprendizado. O critério usado para a formulação do

teste em cada nó é o de obter a menor impureza; uma métrica para mensurar a probabilidade de classificação incorreta dos nós resultantes, também chamado de ramos. Formalmente, a tomada de decisão em cada nó é dada pelo par de atributo e teste usado para a divisão do *dataset*. A impureza Gini dada pela equação 5 é a métrica mais usada para calcular a impureza residual, a qual se deseja minimizar e é dada por (BONACCORSO, 2017):

$$G(t) = 1 - \sum_{i=1}^{c} p_i^2 \quad (5)$$

"Em que *G(t)* é a impureza Gini no *t* e p_i é a proporção de observações da classe *c* no nó *t"* (MÜLLER; GUIDO, 2016).

2.2.5. Floresta Aleatória (*Random Forest*)

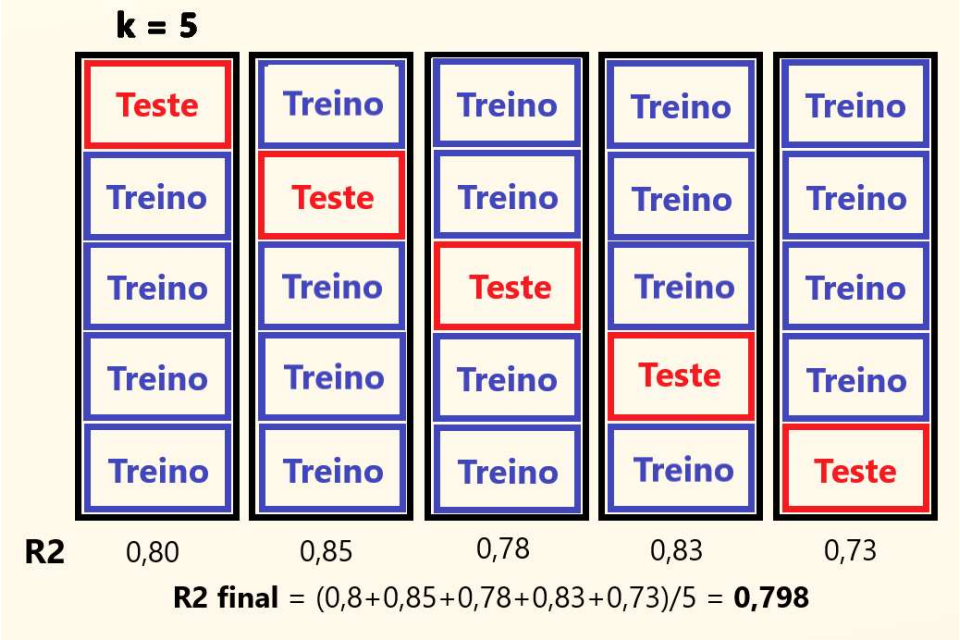
A floresta aleatória é um algoritmo que utiliza um conjunto de árvores de decisão em que o resultado é escolhido com base no voto da maioria. O que diferencia cada árvore é que cada uma é treinada utilizando um subconjunto dos dados. Consequentemente, cada árvore é treinada de forma mais fraca, produzindo predição menos precisa. Por fim, o resultado é a combinação dos resultados individuais por média ou voto da maioria (BONACCORSO, 2017).

As árvores de decisão tendem a se ajustar excessivamente aos dados de treinamento (*overfitting*, do inglês). Para evitar esse problema, a floresta aleatória treina múltiplas árvores de forma incompleta, sendo assim cada árvore *overfit* de forma diferente, e ao tirar a média dos resultados reduz-se o efeito do *overfit*. A política limitar o treinamento da árvore é utilizar uma quantia reduzida de atributos por árvore e a realização de *bootstrap* nas amostras, que consiste em retirar e adicionar cópia de amostra aleatória múltiplas vezes (MÜLLER, 2016).

2.3. Validação cruzada

Na validação cruzada os dados são divididos em *k* partições, e então o algoritmo é executado *k*-vezes, sendo que em cada execução se usa uma partição diferente para teste, e as demais para treino assim como ilustrado na Figura 3. Consequentemente, se tem *k* resultados de performance é possível calcular estimativas mais confiáveis da performance do algoritmo, como a média e o desvio padrão.

Figura 3 – Ilustração de validação cruzada.



Fonte: Didática Tech.

3. METODOLOGIA

Para realizar a classificação dos solos foram seguidos os seguintes passos: (i) análise exploratória de dados; (ii) pré-processamento; (iii) modelagem, usando diferentes técnicas de aprendizado de máquina; e (iv) análise dos resultados obtidos.

3.1. Análise Exploratória de Dados

Análise exploratória de dados é um passo essencial na pesquisa e análise. O objetivo primário da análise exploratória é examinar o dado para obter a sua distribuição, identificar *outliers* e outras anomalias, e assim direcionar testes dedicados à avaliação de suas hipóteses (KOMOROWSKI, 2016). Para analisar a relação entre os atributos e tipos de solos foram feitos gráficos de dispersão usando diferentes cores para indicar os tipos de solos. Ao analisar a distribuição de amostras para cada atributo é utilizado a Estimativa de Densidade Kernel (*Kernel Density Estimation* - KDE).

3.2. Pré-processamento

O pré-processamento é o tratamento dos dados com o objetivo de melhorar o desempenho dos algoritmos de aprendizado de máquina. O pré-processamento foi dividido em três etapas: (i) tratamento de *outliers*; (ii) Mudança de escala; e (iii) Balanceamento dos dados usando SMOTE.

3.2.1. Tratamento de *outliers*

O impacto que os *outliers* têm em algoritmos lineares pode ser reduzido pelo uso de algoritmos de mudança de escala. Adicionalmente, há o fato de as classes

serem desbalanceadas e, consequentemente, considera-se prudente evitar a perda de informação. Logo, optou-se em usar uma abordagem extremamente conservadora para o tratamento de *outliers*.

3.2.2. Mudança de escala

Muitos algoritmos de aprendizado de máquina são sensíveis à escala dos dados (MÜLLER; GUIDO, 2016). Portanto, para otimizar a precisão e velocidade de convergência dos algoritmos foi usado o método de mudança de escala Yeo-Johnson.

3.2.3. Balanceamento usando SMOTE

A intenção por trás desse algoritmo é a ideia de que quando há um número limitado de amostras de uma classe, o algoritmo não tem um conjunto de elementos rico o suficiente para construir as regras de classificação. Ao serem criadas novas amostras similares, mas não idênticas às já existentes, o algoritmo tem a oportunidade de aprender um conjunto mais robusto de regras de classificação (BRUCE, 2020).

O algoritmo SMOTE (do inglês, *Synthetic Minority Over-sampling Technique*) cria novas amostras da classe minoritária, tendo como base um elemento dela e seus K-vizinhos próximos. Assim, ele faz uma média ponderada aleatória dos vizinhos para gerar a nova amostra. A estratégia utilizada para a aplicação do SMOTE é apenas realizar a sobre amostragem de classes que tenham menos de 60 amostras nos dados de treino, a fim de que todas tenham ao menos 60 amostras sem que haja uma sobre amostragem.

3.3. Modelagem

Com o objetivo de comparar a eficácia de diferentes algoritmos de aprendizado de máquina foram escolhidos os seguintes algoritmos de classificação: Árvore de decisão, Floresta Aleatória, KNN, Regressão Logística multinomial e Gaussian Naive Bayes. Primeiramente foi comparado o efeito da mudança de escala usando o método de Yeo-Johnson e também foi comparado o efeito da sobre amostragem SMOTE.

Em seguida é feito a optimização dos hyperparâmetros, parâmetros que afetam o aprendizado e consequentemente o resultado dos algoritmos. A optimização é feita em três etapas. Primeiro se define um conjunto de valores que será testado para cada hyperparâmetro, em seguida é feito o treinamento e predição usando todas as combinações de hyperparâmetros obtidas. Por fim, os hyperparâmetros usados que apresentaram melhor resultado são escolhidos. Vale ressaltar que nem todos os hyperparâmetros testados afetam a efetividade do treinamento.

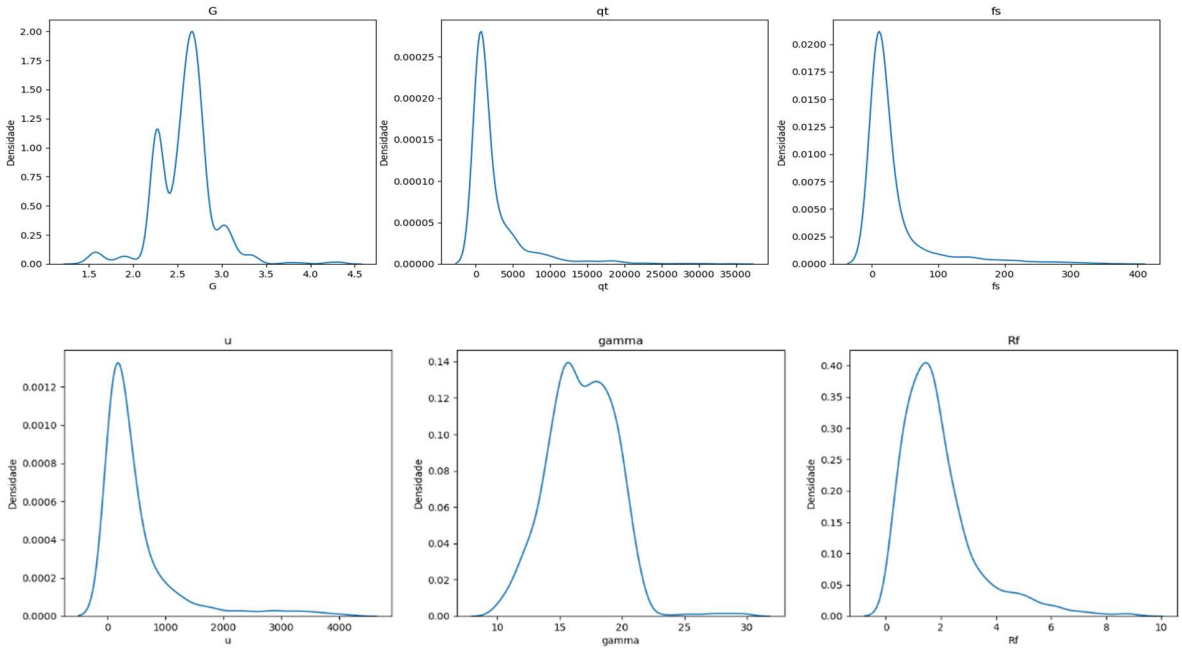
4. RESULTADOS E DISCUSSÃO

4.1. Análise Exploratória de Dados

Na Figura 4 é possível observar um gráfico que ilustra a distribuição de amostras em função dos valores que cada variável. Nas imagens que se referem aos atributos *qt*, *fs*, *u* e *Rf* observa-se que a característica da distribuição é semelhante à log-normal, na qual ela apresenta uma cauda longa com grande variação de amplitude para as amostras. Esse padrão não é desejável para treino de algoritmos lineares. Consequentemente é escolhido o uso do algoritmo Yeo-Johnson para fazer a mudança de escala, pois ele tem bons resultados para esse tipo de distribuição.

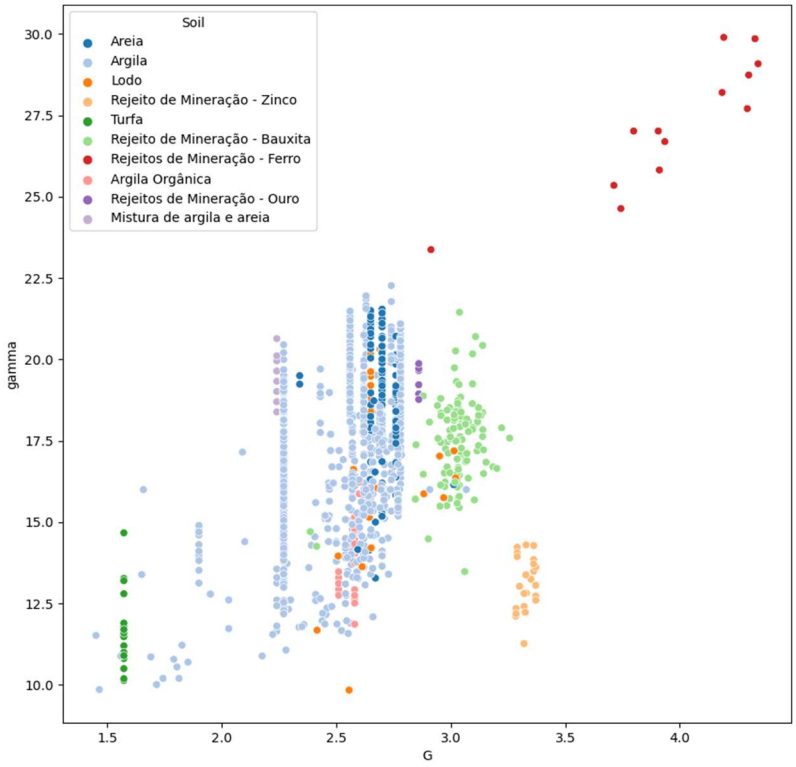
Outro gráfico decisivo para o pré-processamento dos atributos é a Figura 5 pois ela ilustra a relação entre os tipos de solo e as variáveis G e γ. Nele observa-se que os pontos extremos das variáveis G e γ estão associados a diferentes tipos de solo, um padrão que seria prejudicial perder em um algoritmo de remoção de *outliers.* Como nos demais atributos não se observa outros pontos de interesse optou-se por excluir G e γ do tratamento de *outliers,* tratando apenas as demais variáveis.

Figura 4 – Distribuição por atributo



Fonte: Autor.

Figura 5 – gráfico de dispersão G vs γ



Fonte: Autor.

4.2. Pré-processamento

Devido às observações obtidas pela análise exploratória de dados optou-se por não remover *outliers* das variáveis G e γ e adotar uma abordagem conservadora para a remoção de *outliers* das outras variáveis. Sendo assim, foi retirado apenas 2% das extremidades de cada um dos outros atributos. A Tabela 3 e 4 mostram as diferenças de valores para cada um dos atributos. Pode-se notar que foram eliminadas 119 amostras e as mudanças dos valores mínimos, máximos, médias e desvio padrão de todas as colunas menos G e γ. Com ênfase principal nos valores sublinhados na tabela 4.

Tabela 3 – Distribuição estatística dos parâmetros antes do tratamento de *outlier*

Parâmetros estatísticos	**Variáveis**					
	G	qt (kPa)	fs (kPa)	u (kPa	γ (kN/m³)	Rf
amostras	1.488	1.488	1.488	1.488	1.488	1.488
média	2,58	3.039,97	39,80	525,56	16,76	2,00
σ	0,33	7.393,01	87,66	812,53	2,75	2,06
min	1,45	30,37	0,00	0,00	9,84	0,00
25%	2,29	394,98	6,54	102,62	14,90	0,95
50%	2,62	862,62	13,11	251,58	16,70	1,58
75%	2,70	2.650,71	31,23	535,25	18,77	2,41
max	4,34	102.000,00	1.125,10	6.078,00	29,89	47,14

Fonte: Autor.

Tabela 4 – Distribuição estatística dos parâmetros depois do tratamento de *outlier*

Parâmetros estatísticos	Variáveis					
	G	qt (kPa)	fs (kPa)	u (kPa	γ (kN/m³)	Rf
amostras	1369	1369	1369	1369	1369	1369
média	2,58	2.392,91	32,82	503,34	16,75	1,91
σ	0,32	3.917,12	52,90	694,06	2,68	1,38
min	1,45	106,61	1,49	0,00	9,84	0,21
25%	2,28	416,16	6,98	117,00	15,02	0,99
50%	2,62	869,86	13,25	264,15	16,70	1,61
75%	2,71	2.524,05	30,20	546,23	18,68	2,38
max	4,34	34.507,38	373,49	4.155,00	29,89	9,07

Fonte: Autor.

4.3. Modelos

Para treino dos modelos foi reservado 80% dos dados e 20% foi reservado para teste e validação. Uma vez separado os dados foi feito uma avaliação das eficácia dos métodos de pré-processamento propostos, cada modelo foi avaliado usando os dados com e sem a mudança de escala pelo método Yeo-Johnson e com ou sem o uso do algoritmo de sobre amostragem SMOTE. A Tabela 5 mostra os resultados obtidos para medida F (*f1-score*), que utiliza os valores globais de positivos verdadeiros, positivos falsos e falsos negativos para ser calculado. Ao observar os resultados, conclui-se que a mudança de escala traz melhora significativa para a performance dos algoritmos KNN, regressão logística e Gaussiana Naive Bayers. Já o uso do algoritmo SMOTE não levou a um melhor desempenho, e em alguns casos até a desempenho pior. Como os tipos de solo com menos amostras não são de importância fundamental para a classificação não se justifica utilizar o SMOTE no algoritmo de classificação, sendo assim os testes seguintes não o utilizarão.

Tabela 5 – Avaliação do efeito da mudança de escala Yeo-Johnson e uso de SMOTE

Modelo	Com escala com smote	Sem escala com smote	Com escala sem smote	Sem escala sem smote
Árvore de decisão	89,25%	89,25%	90,32%	90,86%
Floresta aleatória	94,09%	93,28%	94,62%	93,82%
KNN	90,32%	67,20%	91,67%	74,19%
Regressão logística	86,29%	71,51%	88,44%	74,46%
Gaussian Naive Bayes	85,48%	64,52%	86,83%	64,25%

Fonte: Autor.

Para avaliação dos modelos usados será utilizado as seguintes métricas: precisão que é o número de acertos dividido pelo número de ocorrências preditas; *recall* que é o número de acertos pelo número real de ocorrências; e a Medida F(do inglês, *f1-score*) é uma pontuação que combina precisão e *recall*.

Para obter melhores resultados dos algoritmos foi feita a otimização dos hyperparâmetros como descrito, utilizando apenas os dados de treino dividido em cinco partições para utilização do método de validação cruzada. Encontrando assim os valores ótimos de hyperparâmetro para cada algoritmo testado. Por fim, usando os dados de treino e os hyperparâmetros obtidos foi calculada a precisão, *recall*, e a medida F (*f1-score*) utilizando validação cruzada.

Os resultados obtidos encontram-se na Tabela 6 para cada algoritmo usado. Destaca-se que todos os algoritmos obtiveram resultados próximos, sendo que floresta Aleatória obteve a melhor precisão enquanto árvore de decisão obteve melhor *recall* e Medida F. Os algoritmos lineares, usando mudança de escala Yeo-Johnson, obtiveram resultados próximos aos algoritmos baseados em árvores, embora inferiores. Por ter o melhor resultado para *recall* e Medida F e segundo melhor para precisão o modelo da árvore de decisão será usado no passo seguinte para analisar a matriz de confusão dos resultados obtidos.

Tabela 6 – Performance de cada modelo após optimização

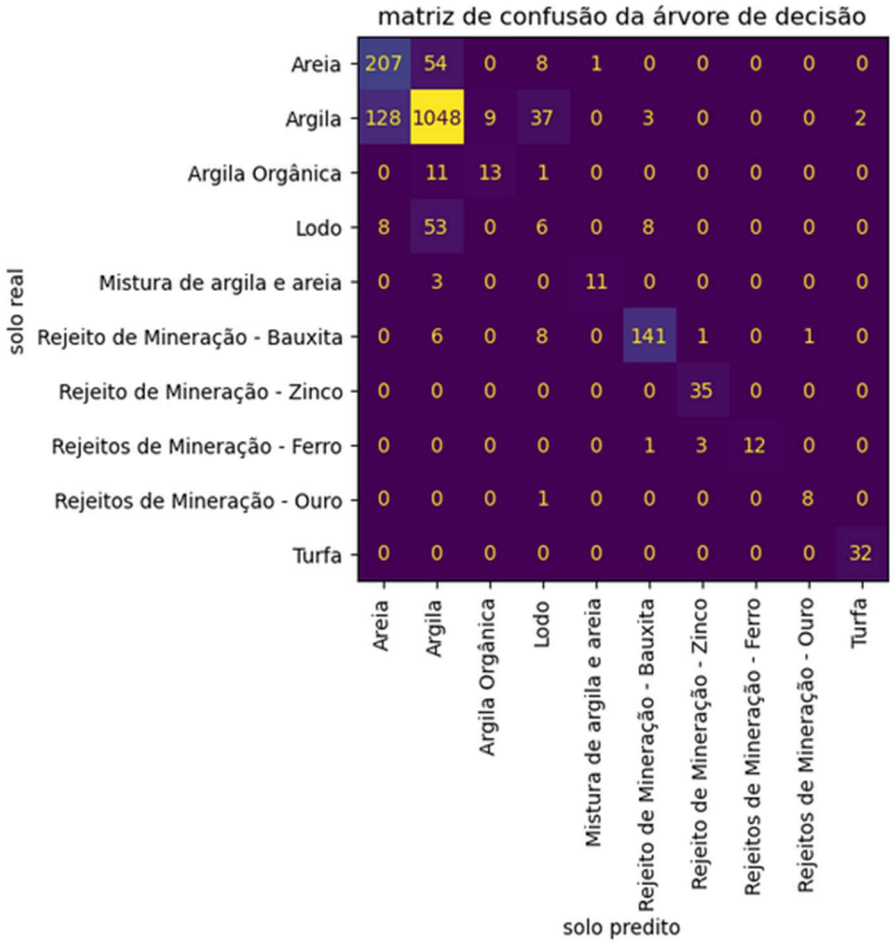
Modelos	Precisão	Recall	Medida F
Árvore de decisão	91,01%	82,93%	81,46%
Floresta aleatória	93,57%	77,34%	80,60%
KNN	89,26%	64,93%	65,87%
Regressão logística	88,68%	62,09%	61,79%
Gaussian Naive Bayes	87,51%	82,02%	80,10%

Fonte: Autor.

Por fim, utilizando a árvore de decisão com os hyperparâmetros otimizados é feita a matriz de confusão da Figura 6. Para calcular a matriz de confusão foi dividido todos os dados utilizando validação cruzada com k=5. Foi refeito todos os passos citados, como tratamento de *outliers*, mudança de escala e o resultado obtido em cada etapa foi somado para obter a matriz de confusão final. Optou-se por essa abordagem por concluir que a contaminação dos dados de teste pelos dados de teste é baixa e os ganhos de informação obtidos por ter mais dados utilizados para a formação da matriz de confusão justificam as perdas.

Na matriz de confusão as linhas se referem à classe real e as colunas ao que foi predito. Por exemplo, o elemento que se encontra na intersecção da linha e coluna argila se refere ao número de elementos corretamente classificados como argila. Já na linha argila e coluna lodo está o número de elementos que são argila, mas foram erroneamente classificados como lodo. Ao analisar a matriz de confusão na Figura 6 pode-se observar que o algoritmo classificou erroneamente lodo em 90,15% dos casos tendo um *recall* de 8%. Outros solos com baixa acurácia foram argila orgânica com 59%, areia com 60,3%. Já os demais apresentam desempenho satisfatório como a argila com 89,1%, rejeito de mineração – bauxita com 92,7% entre outros.

Figura 6 – matriz de confusão floresta aleatória



Fonte: Autor.

5. CONCLUSÃO

O estudo do solo é uma área importante para diferentes processos de engenharia. Para a classificação de solos, foi possível observar que (i) o uso da transformada Yeo-Johnson implicou em melhorias significativas apenas para modelos lineares; (ii) o uso de sobre amostragem utilizando o algoritmo SMOTE não causou impacto significativo; (iii) o modelo não é restrito a uma localização geográfica, pois foi treinado com amostras de solos de diversas localizações geográficas; e (iv) ao observar a matriz de confusão os solos com o pior desempenho de classificação são o lodo, argila e areia. Todos os outros tipos de solo foram corretamente classificados.

Futuras melhorias para o processo seria: a realização de teste com outros algoritmos de aprendizado de máquina ou *deep learning*; utilizar o modelo disponibilizado por Nierwinski *et al.* (2023) para substituir testes de laboratório feitos para determinar o peso específico do solo (γ); e usar o processo proposto por Bhattacharya (2006), realizando o *clustering* de amostras consecutivas do CPTu e as utilizar em conjunto no algoritmo de classificação.

REFERÊNCIAS

ALBON, C. **Machine learning with python cookbook: Practical solutions from preprocessing to deep learning**. O'Reilly Media, Inc. 2018.

AYDIN, Y. *et al*. Use of machine learning techniques in soil classification. **Sustainability**, v. 15, n. 3, p. 2374. 2023.

BARBOSA, H. **Banco de dados geotécnico das argilas moles da Região Metropolitana do Recife (RMRecife).** 2018. Tese de Mestrado. Universidade Federal de Pernambuco.

BARONI, M. **Investigação geotécnica em solos moles da Barra da Tijuca com ênfase em ensaios in situ.** 2010. Ph.D. Tese. Dissertação de MSc., COPPE/UFRJ, Rio de Janeiro, Brasil, *in print*.

BHATTACHARYA, B.; SOLOMATINE, D. Machine learning in soil classification. **Neural networks**, v. 19, n. 2, p. 186-195. 2006.

BONACCORSO, G. **Machine learning algorithms**. Packt Publishing Ltd. p.155-158 e 167. 2017.

BRUCE, P.; BRUCE, A.; GEDECK, P. **Practical statistics for data scientists: 50+ essential concepts using R and Python**. O'Reilly Media, 2020.

BRUGGER, P. **Análise de Deformações em Aterros sobre Solos Moles.** Tese de Doutorado, COPPE/UFRJ, 1996. Xv, 255 p. 29, 7.

CHANDAN, T. Recent trends of machine learning in soil classification: A review. **Int. J. Comput. Eng. Res**, v. 8, p. 25-33. 2018.

DIDÁTICA TECH. **O pacote Caret – linguagem R.** Disponível em: <https://didatica.tech/o-pacote-caret-linguagem-r/>. Acesso de acesso: 24 jun. 2023.

KOMOROWSKI, M.; Marshall, D.; Salciccioli, J.; Crutain, Y. (2016). Exploratory Data Analysis. Em: **Secondary Analysis of Electronic Health Records.** Springer, Cham. 2016. Disponível em: <https://doi.org/10.1007/978-3-319-43742-2_15>. Acesso em: 2 mai. 2023.

MÜLLER, A.; GUIDO, S. **Introduction to machine learning with Python: a guide for data scientists**. O'Reilly Media, Inc. 2016.

NIERWINSKI, H.; PFITSCHER, R.; MENEGAZ, T.; ODEBRECHT, E. **A Machine Learning Method for Soil Unit Weight Estimation.** (no prelo)

OLIVEIRA, J., 1997. **Ensaios de Piezocone em um Depósito de Argila Mole da Cidade do Recife.** Tese de mestrado. Programa de Pós-Graduação em Engenharia Civil, COPPE/UFRJ.

PURI, N.; PRASAD, H.; JAIN, A. Prediction of geotechnical parameters using machine learning techniques. **Procedia Computer Science**, v. 125, p. 509-517, 2018.

RIANI, M.; ATKINSON, A.; CORBELLINI, A. Automatic robust Box–Cox and extended Yeo–Johnson transformations in regression. **Statistical Methods & Applications**, v. 32, n. 1, p. 75-102, 2023.

APÊNDICE

APÊNDICE A – código fonte de implementação dos modelos testados

O código fonte do projeto encontra-se no repositório do GitHub e está disponível para acesso público, porém a base de dados usada não está disponível. A URL do repositório é <https://github.com/alberthuslordello/TCC>.