Estudo de Caso - Desempenho de Algoritmos de Aprendizado de Máquina

Igor Ryan Bacelar Frota
Instituto Federal do Maranhão
Coelho Neto-MA, Brasil
igorb@acad.ifma.edu.br

Samuel Silva dos Santos
Instituto Federal do Maranhão
Coelho Neto-MA, Brasil
santos.samuel1@acad.ifma.edu.br

***Resumo*—O presente trabalho visa avaliar o desempenho de modelos de aprendizado de máquina sobre uma base de dados de classificação de cogumelos. Os algoritmos implementados foram o KNN, o SVM, o Random Forest e o XGBoost, tal validação ocorre via avaliação da métricas de desempenho resultantes da acurácia de cada um dos algoritmos.**

***Palavras-chave*—aprendizado de máquina, knn, svm,random forest, xgboost**

I. Introdução

Nos últimos anos, algoritmos de aprendizado de máquina têm se destacado na solução de problemas de classificação em diversas áreas, como saúde, segurança, agricultura e biotecnologia [2]. Um aspecto fundamental nesse contexto é a avaliação do desempenho de diferentes algoritmos em tarefas específicas, considerando fatores como precisão, tempo de processamento e capacidade de generalização.

Este estudo tem como objetivo comparar o desempenho de diferentes algoritmos de aprendizado de máquina utilizando uma base de dados amplamente conhecida na literatura: a base Classificação de Cogumelos, disponível no repositório UCI Machine Learning Repository e na plataforma Kaggle. Essa base contém informações sobre características físicas de cogumelos, com cada instância rotulada como comestível ou venenosa “Fig. 1”.

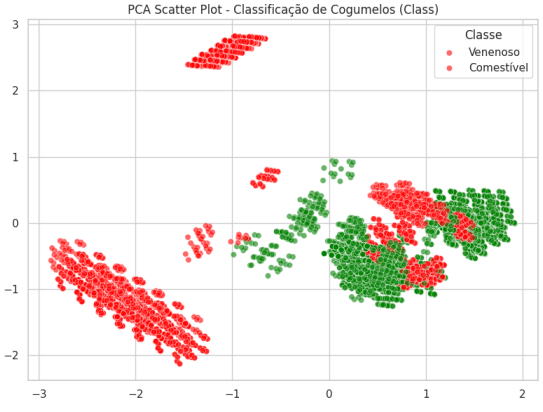

Fig. 1. Distribuição das classes

A escolha dessa base se justifica por sua estrutura rica em atributos categóricos, que não favorece nenhum algoritmo específico. Isso a torna adequada para avaliar técnicas de pré-processamento, como Label Encoding, One-Hot Encoding e redução de dimensionalidade com PCA. Além disso, sua estrutura equilibrada permite uma análise imparcial entre os algoritmos.

Foram utilizados algoritmos como K-Nearest Neighbors (KNN), Random Forest, Máquina de Vetores de Suporte (SVM) e eXtreme Gradiente Boosting (XGBoost). Este trabalho, portanto, busca analisar como diferentes abordagens de pré-processamento e a implementação desses algoritmos influenciam no desempenho final dos modelos.

II. Algoritmos de Aprendizado de Máquinas

Neste estudo, foram escolhidos quatro algoritmos supervisionados de classificação comumente utilizados em tarefas de aprendizado de máquina. Cada um representa uma abordagem distinta para a resolução de problemas de classificação.

A. KNN

O KNN é um algoritmo de classificação que se baseia em instâncias, classificando novos dados de acordo com a maioria das classes dos k vizinhos mais próximos no espaço de características. A distância mais comum que se utiliza é a Euclidiana. O KNN não passa por uma fase de treinamento explícita, sendo considerado um algoritmo ”preguiçoso” (*lazy learner*). Isso pode ser um problema em situações com grandes volumes de dados, já que o custo computacional na fase de predição pode ser alto. [5]

B. SVM

A SVM é um algoritmo supervisionado que tem como objetivo encontrar o hiperplano ideal que separa os dados em diferentes classes da melhor forma possível. Quando os dados não são linearmente separáveis, ela utiliza funções de kernel para transformar os dados em um espaço de maior dimensão, onde a separação se torna viável. A SVM se destaca em espaços de alta dimensão e é resistente ao *overfitting*, especialmente quando as margens entre as classes são bem definidas. [3]

C. Random Forest

A Random Forest é um algoritmo que combina múltiplos classificadores, utilizando várias árvores de decisão para

chegar a decisões mais robustas. Cada árvore é treinada com um subconjunto aleatório dos dados e dos atributos, uma técnica conhecida como bagging. A decisão final é feita por meio de uma votação da maioria. Esse algoritmo é conhecido por sua boa capacidade de generalização, resistência ao overfitting e por lidar bem com dados ruidosos. [6]

D. XGBoost

O XGBoost é uma implementação eficiente da técnica de Gradient Boosting, que constrói modelos sequenciais, onde cada novo modelo busca corrigir os erros dos anteriores. Ele é bastante popular em competições de ciência de dados, graças à sua alta performance e capacidade de regularização, que ajuda a reduzir o *overfitting*. O XGBoost combina rapidez, precisão e flexibilidade, sendo eficaz tanto para tarefas de classificação quanto de regressão, muito indicado para aplicações em bases de dados com número elevado de variáveis categóricas [4].

A variedade desses algoritmos possibilita uma comparação detalhada de desempenho, especialmente em conjuntos de dados que possuem atributos categóricos, como o que foi utilizado neste estudo.

III. METODOLOGIA

Para este estudo comparativo, foi utilizado o conjunto de dados conhecido como "Mushroom Dataset", que possui 8124 instâncias e 22 atributos categóricos, além da variável-alvo chamada *"class"*, que nos diz se o cogumelo é comestível ou venenoso. Todos os atributos são do tipo nominal, existindo apenas alguns valores indefinidos.

O primeiro passo consistiu na exploração e no pré-processamento da base de dados [1]. Foi necessário tratar variáveis com tipos indefinidos, o que ocorria com os valores da variável *stall-root* (raíz do talo) em que para 2480 amostras este valor era 'desconhecido'. No entanto, considerando a quantidade expressiva de informações, optou-se por manter esses dados, assumindo-os como válidos para a análise.

Por se tratar de uma base composta exclusivamente por atributos categóricos, foram aplicadas diferentes estratégias de codificação, conforme o algoritmo e o cenário avaliado. Para o algoritmo K-Nearest Neighbors (KNN), no cenário ideal, utilizou-se a técnica de One-Hot Encoding, dado que esse método depende de cálculos de distância e não interpreta adequadamente valores inteiros como categorias. Nos demais cenários para o KNN, aplicou-se o Label Encoding.

Para os algoritmos Random Forest, XGBoost e Máquina de Vetores de Suporte (SVM), foram utilizados tanto One-Hot Encoding quanto Label Encoding, uma vez que esses algoritmos, por serem baseados em árvores ou em funções kernel – o que é o caso do SVM –, não são sensíveis à natureza ordinal dos dados codificados numericamente.

A escolha do tipo de codificação variou de acordo com o cenário experimental considerado, respeitando as características de cada técnica e seu impacto no desempenho dos modelos.

Além da codificação, foram aplicadas técnicas de normalização, como o StandardScaler e o Min-Max Scaler, considerando que algoritmos como KNN e SVM são sensíveis à escala dos atributos. A base de dados foi dividida em conjuntos de treino e teste na proporção de 80/20, assegurando a preservação da proporção entre as classes (cogumelos comestíveis e venenosos) em ambos os subconjuntos.

Em determinados experimentos, especialmente nos cenários mais elaborados para os algoritmos KNN e SVM, também foram aplicadas técnicas de balanceamento de classes, como o Synthetic Minority Over-sampling Technique (SMOTE), com o objetivo de mitigar possíveis enviesamentos provocados por desbalanceamento entre as categorias.

A redução de dimensionalidade com PCA foi utilizada em alguns cenários, principalmente nas melhores configurações, para verificar se a simplificação das variáveis poderia trazer ganhos de generalização. A ideia era manter mais de 80 d da variância explicada com o menor número possível de componentes.

Cada algoritmo foi avaliado em três cenários distintos, definidos a partir de variações reais no nível de preparo dos dados, complexidade e condições de treino:

a) Cenário desfavorável: Neste cenário os dados apresentam alta sobreposição entre classes e ausência de técnicas de balanceamento, normalização ou redução de dimensionalidade. Foi aplicado apenas um pré-processamento básico com codificação de rótulos (Label Encoding), e os hiperparâmetros foram mantidos em configurações simples. Assim como a introdução de ruídos nos atributos alvos.

b) Cenário intermediário: Neste cenário os dados passaram por um tratamento moderado. Embora técnicas como PCA ou balanceamento não tenham sido utilizadas, os atributos estão mais organizados e há menor sobreposição entre as classes. Esse cenário simula uma rotina prática comum, em que há um preparo razoável, porém sem ajustes otimizados.

c) Cenário ideal: Este é o cenário mais favorável, com dados devidamente tratados, menor presença de ruído, separação clara entre classes e aplicação de técnicas como normalização, balanceamento de classes (como SMOTE) e redução de dimensionalidade via PCA. Os modelos também foram ajustados utilizando Grid Search para a definição dos melhores hiperparâmetros, como o valor ótimo de k para o KNN e o número ideal de árvores para a Random Forest. Esse cenário representa a condição mais propícia para extrair o máximo desempenho dos algoritmos.

A métrica utilizada para avaliação dos modelos foi a acurácia, precisão, recall e F1-score como métricas de avaliação. Essas métricas complementares oferecem uma análise mais completa do desempenho do modelo, especialmente em problemas de classificação com dados desbalanceados ou onde o custo de falsos positivos e falsos negativos difere significativamente.

O objetivo da análise foi identificar o impacto direto que o nível de preparação dos dados exerce sobre o desempenho dos algoritmos, observando o comportamento do KNN, Random Forest, SVM e XGBoost frente a diferentes graus de qualidade nos dados de entrada.

IV. Resultados e Discussão

Após o treinamento a avaliação do desempenho dos quatro algoritmos de aprendizado de máquina — SVM, KNN, Random Forest e XGBoost — em três cenários distintos de preparação de dados.

a) KNN: Com uma acurácia de 0.79 no pior cenário e 1 no intermediário e no cenário favorável, o KNN demonstrou uma capacidade razoável de classificação, mesmo em condições adversas. Sua sensibilidade a ruídos e a escolha do número de vizinhos (k) foram fatores que influenciaram positivamente seu desempenho, permitindo que o modelo identificasse vizinhos relevantes, apesar da alta sobreposição entre classes.

b) SVM: No melhor dos cenários "Fig. 2" o SVM apresentou um ótimo desempenho com uma acurácia de 1 (100%), no ambiente intermediário "Fig. 3", embora com algumas modificações como inserção de ruídos na variável alvo, remoção de variáveis com alta correlação com o alvo a acurácia ficou em 0.88 demonstrando assim um bom desempenho do modelo mesmo que nestas condições. No pior dos cenários "Fig. 4", aumentou-se o índice de ruídos, selecionou-se as variáveis com baixa correção com o alvo para compor as variáveis preditoras, modificou-se o kernel do modelo para um menos indicado para esse tipo de base o que resultou em uma acurácia de 0.67.

🎯 Acurácia - SVM (One-Hot Encoding): 1.0000

📋 Relatório de Classificação - SVM:

	precision	recall	f1-score	support
Venenoso	1.00	1.00	1.00	1257
Comestível	1.00	1.00	1.00	1181
accuracy			1.00	2438
macro avg	1.00	1.00	1.00	2438
weighted avg	1.00	1.00	1.00	2438

Fig. 2. SVM: melhor cenário

🎯 Acurácia - SVM (One-Hot Encoding): 0.8860

📋 Relatório de Classificação - SVM:

	precision	recall	f1-score	support
Venenoso	0.83	0.98	0.90	1257
Comestível	0.98	0.78	0.87	1181
accuracy			0.89	2438
macro avg	0.90	0.88	0.88	2438
weighted avg	0.90	0.89	0.88	2438

Fig. 3. SVM: cenário intermediário

🎯 Acurácia - SVM (One-Hot Encoding): 0.6790

📋 Relatório de Classificação - SVM:

	precision	recall	f1-score	support
Venenoso	0.63	0.97	0.76	433
Comestível	0.90	0.35	0.50	380
accuracy			0.68	813
macro avg	0.77	0.66	0.63	813
weighted avg	0.76	0.68	0.64	813

Fig. 4. SVM: pior cenário

c) Random Forest: O desempenho do Random Forest de 1 em seu caso favorável mas foi inferior, com uma acurácia de 0.64 no cenário desfavorável e 0.92 no intermediário. A complexidade do modelo e a dificuldade em capturar padrões significativos em um cenário com alta variabilidade e ruídos podem ter contribuído para esse resultado. A falta de otimização dos hiperparâmetros, como o número de árvores e a profundidade máxima, também pode ter impactado negativamente sua capacidade de generalização.

d) XGBoost: Sendo este algoritmo um dos que melhor se aplica a bases de dados com elevado número de variáveis categóricas – o que é o caso da base em estudo – conseguiu uma acurácia de 1 no melhor dos cenários "Fig. 5"; no cenário intermediário "Fig. 6", com inserção de ruídos e remoção de variáveis com alta correlação e alteração dos hiperparâmetros reduzindo a profundidade máxima (`max_depth`) para 1 e limitando o número de árvores e combinadas em 5 (`n_stimators`) a acurária teve uma leve redução para 0.94 o que demonstra a alta performance do modelo mesmo que tal situação; já no pior dos cenários "Fig. 7" onde dobrou-se o número de ruídos e reduzindo drasticamente o número de variáveis preditoras limitando-as apenas às que possuíam baixa correlação com a variável alvo, observou-se um redução drástica da acurácia para apenas 0.64.

🎯 Acurácia - XGBoost (Label Encoding): 1.0000

📋 Relatório de Classificação - XGBoost:

	precision	recall	f1-score	support
Venenoso	1.00	1.00	1.00	1257
Comestível	1.00	1.00	1.00	1181
accuracy			1.00	2438
macro avg	1.00	1.00	1.00	2438
weighted avg	1.00	1.00	1.00	2438

Fig. 5. XGB: melhor cenário

🎯 Acurácia - XGBoost (Label Encoding): 0.9430

📋 Relatório de Classificação - XGBoost:

	precision	recall	f1-score	support
Venenoso	0.97	0.91	0.94	1257
Comestível	0.91	0.97	0.94	1181
accuracy			0.94	2438
macro avg	0.94	0.94	0.94	2438
weighted avg	0.94	0.94	0.94	2438

Fig. 6. XGB: cenário intermediário

🎯 Acurácia - XGBoost (Label Encoding): 0.6470

📋 Relatório de Classificação - XGBoost:

	precision	recall	f1-score	support
Venenoso	0.81	0.44	0.57	433
Comestível	0.58	0.88	0.70	380
accuracy			0.65	813
macro avg	0.69	0.66	0.64	813
weighted avg	0.70	0.65	0.63	813

Fig. 7. XGB: pior cenário

V. Conclusões

Após implementação e avaliação das métricas de desempenho de cada modelo aplicado à base de dados de classificação de cogumelos venenosos e comestíveis, foi possível averiguar que com o tratamento adequado do *dataset*

no que diz respeito à análise exploratória dos dados, pré-processamento e neste inclui-se o tratamento de valores ausentes, codificação de variáveis categóricas de acordo com tipo de modelo a ser aplicado, normalização ou padronização dos dados, uso de técnicas adequadas de balanceamento, todos os modelos apresentaram ótimo desempenho.

Logo ao se analisar os desempenhos dos algoritmos nos três cenários propostos tem-se que:

- Todos os modelos alcançaram elevadas acurácias. O que, em boa parte, deve-se ao fato da base possuir variáveis com alto poder discriminativo como a variável odor, por exemplo.
- Dada a alta previsibilidade da base, modelos simples como o KNN ou mesmo os modelos baseados em árvores de decisão – como o Random Forest e XGBoost – mesmo com um configuração simples de hiperparâmetos (`max_depth=1`, `n_stimators=5`) apresentaram um bom desempenho.
- O KNN tem bom desempenho alcançado elevada acurácia porém mais sensível dada sua dependência de um valor ideal a ser atribuído ao número de vizinhos (k).
- A SVM com o kernel RBF também se destaca, demonstrando um excelente desempenho com a adequada codificação; demonstrando-se eficiente em problemas com fronteiras de decisão não lineares.
- Modelos simples, mesmo com a inserção de ruídos, remoção de algumas variáveis de alta correlação os algoritmos ainda performaram bem.
- Por se tratar de uma base relativamente bem balanceada, sem valores ausentes e dados ”limpos”, há pouca necessidade de se aplicar técnicas avançadas como SMOTE, normalização ou limpeza textual (mesmo assim tais técnicas foram aplicadas visando extrair o máximo desempenho dos modelos).

Por fim, a base de cogumelos apresenta características que favorecem a classificação: dados limpos, balanceados e alguns atributos altamente informativos. Como resultado, todos os modelos testados – KNN, SVM, Random Forest e XGBoost – atingiram desempenhos excepcionais, com acurácia próxima ou igual a 100%. Isso indica que o problema tem baixa complexidade, e a separação entre as classe é clara. Portanto, esta base é um excelente estudo de caso para validar algoritmos de classificação, avaliar técnicas de codificação e interpretar a importância de variáveis.

REFERÊNCIAS

[1] Ferreira, R. G. C., de Miranda, L. B. A., Pinto, R. A., Pessutto, L. R. C., Pereira, M. A., de Andrade, A. L. C., and Marques, L. T. (2021). Preparação e Análise Exploratória de Dados. SAGAH, Porto Alegre.

[2] Lenz, M. L., Neumann, F. B., Santarelli, R., and Salvador, D.. Fundamentos de Aprendizagem de Máquina. SAGAH, Porto Alegre. 2020.

[3] Scikit-learn - Support Vector Machines (SVM). Disponível em: https://scikit-learn.org/stable/modules/svm.html#svm. Acesso em 31-maio-2025.

[4] Scikit-learn - Ensembles: Gradient boosting, random forests, bagging, voting, stacking. Disponível em: https://scikit-learn.org/stable/modules/ensemble.html. Acesso em 31-maio-2025.

[5] Scikit-learn – K-Nearest Neighbors and Random Forest classifiers. Disponível em: https://scikit-learn.org/stable/supervised_learning.html. Acesso em: 31-maio-2025.

[6] Scikit-learn – Random Forest Classifier. Disponível em: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html. Acesso em: 31-maio-2025.