Análise Comparativa de Modelos de Machine Learning para Predição da Dificuldade de Questões de Matemática do ENEM

Ana Laura Tavares Costa
Universidade Federal de Viçosa

Resumo

Este trabalho apresenta uma análise comparativa de diferentes modelos de aprendizado de máquina aplicados à predição da dificuldade de questões de matemática do Exame Nacional do Ensino Médio (ENEM). O objetivo é avaliar a eficácia de diferentes algoritmos e estratégias de modelagem na estimativa do parâmetro de dificuldade de itens.

1 Introdução

O Exame Nacional do Ensino Médio (ENEM) é a principal porta de entrada para o ensino superior no Brasil. Todos os anos, milhões de participantes realizam a prova em todo o país. Segundo dados levantados pelo MEC (Ministério da Educação), o exame registrou mais de 3,9 milhões de inscrições em 2023. Tais informações evidenciam a relevância social e educacional desta avaliação.

Para os participantes, entender a dificuldade de cada questão é uma etapa fundamental na organização dos seus estudos e na preparação para o dia da prova. Contudo, a dificuldade dos itens é formalizada pelo Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira (INEP) através de parâmetros da Teoria de Resposta ao Item (TRI). Este método estatístico, embora robusto, apresenta uma notória complexidade, tornando a interpretação da dificuldade pouco acessível ao público geral.

Neste contexto, este projeto investiga a viabilidade de utilizar modelos de Inteligência Artificial para classificar a dificuldade de questões do ENEM, com foco na prova de Matemática. A pergunta central que norteia este trabalho é: "É possível construir um modelo preditivo que classifique com acurácia a dificuldade oficial de uma questão, e qual abordagem de Machine Learning se mostra mais eficaz?". Para responder a essa questão, este artigo apresenta a metodologia utilizada, uma análise comparativa de três algoritmos de classificação e, por fim, discute os resultados obtidos e as conclusões do estudo.

2 Trabalhos Relacionados

Andrade, Tavares e Valle (2000) detalham a aplicação da Teoria de Resposta ao Item (TRI) no contexto brasileiro, explicando a relevância dos parâmetros de dificuldade (B), discriminação (A) e acerto casual (C) para uma aferição precisa da proficiência dos alunos. Este referencial fundamenta a decisão de utilizar o Parâmetro B como a medida ideal para a criação da variável alvo `dificuldade_oficial` no projeto.

A tarefa de prever a dificuldade de questões utilizando IA já foi explorada na literatura. Costa, Campelo e Campos (2018) propõem um classificador automático de questões de Matemática em língua portuguesa, com o objetivo de diferenciá-las entre Questões Problema (QPM) e Questões Não Problema (QNPM), a fim de fomentar práticas pedagógicas associadas ao Pensamento Computacional (PC). Utilizando técnicas de Processamento de Linguagem Natural (PLN) e Aprendizado de Máquina, especialmente o classificador *Naive Bayes* com vetorização TF-IDF, os autores demonstram que é possível automatizar essa classificação com alta capacidade preditiva.

Jardim (2022) propôs um modelo classificador automático para determinar o nível de dificuldade de questões do ENEM, utilizando técnicas de Processamento de Linguagem Natural e uma Rede Neural Convolucional. As questões foram vetorizadas com *embeddings* semânticos e rotuladas conforme a taxa de acerto dos participantes, explorando a chamada sabedoria das multidões. O modelo demonstrou bom desempenho preditivo e foi integrado a um Sistema de Apoio à Decisão (SAD), com potencial aplicação em sistemas educacionais adaptativos.

3 Metodologia

O desenvolvimento deste projeto seguiu um fluxo de trabalho estruturado em quatro etapas principais: preparação dos dados, definição da variável alvo, engenharia de atributos e, finalmente, a modelagem e avaliaçao comparativa.

3.1 Fonte e Preparação dos Dados

Foram utilizados os dados públicos do INEP (Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira), especificamente os arquivos `ITENS_PROVA` e `MICRODADOS_ENEM` das edições de 2022 e 2023. Não foram utilizados dados de outros anos (2021, 2020, etc.) pois eles não possuem informações consis-

tentes sobre os parâmetros da Teoria de Resposta ao Item, parte relevante da Engenharia de Features deste trabalho. Optou-se por dar enfoque nas questões da aplicação regular, pois é a prova com maior número de participantes. Então, foi feita uma análise de frequência no conjunto de microdados dos participantes presentes, selecionando os 4 códigos de prova mais frequentes. Em seguida, o conjunto de itens foi filtrado para selecionar apenas questões de Matemática (`'SG_AREA' == 'MT'`), que não foram anuladas (`'IN_ITEM_ABAN' == 0`) e que pertenciam ao conjunto de provas da aplicação regular. Este processo resultou em um dataset final de 86 itens únicos para análise.

3.2 Definição da Variável Alvo

A variável alvo a ser predita, denominada `dificuldade_oficial`, foi criada a partir do parâmetro B da Teoria de Resposta ao Item (`NU_PARAM_B`), que de acordo com o INEP, "está associado à dificuldade do item, sendo que quanto maior seu valor, mais difícil o item é". Utilizou-se a função `qcut` da biblioteca `Pandas` para segmentar as 86 questões em três quantis (tercis) de mesmo tamanho, atribuindo os rótulos 'Fácil', 'Média' e 'Difícil'. Tal abordagem garante classes balanceadas para a tarefa de classificação, tornando a avaliação dos modelos mais justa.

3.3 Engenharia de Atributos (Features)

O trabalho possui dois conjuntos de *features* distintos, cada um criado em um momento específico do desenvolvimento.

- **Conjunto Básico:** É o conjunto de *features* inicial do projeto. Estão inclusos apenas dados intrínsecos ao item, como os parâmetros A (`NU_PARAM_A`) e C (`NU_PARAM_C`). O parâmetro A, segundo o INEP, "é o poder de discriminação do item para diferenciar os participantes que dominam dos participantes que não dominam a habilidade avaliada". Já o parâmetro C é "a probabilidade de um participante acertar o item não dominando a habilidade exigida". Além dos parâmetros mencionados, também foi inclusa a variável (`CO_HABILIDADE`), relacionada à habilidade avaliada pela questão. Por ser uma variável categórica, foi transformada em formato numérico pela técnica de *One-Hot-Encoding*.

- **Conjunto Avançado:** Além das *features* básicas, incorpora duas novas *features* calculadas a partir de uma amostra aleatória de 100.000 alunos dos microdados: a `taxa_acerto`, que informa o percentual de acerto da questão, e a `taxa_em_branco`, que é o percentual de participantes que deixaram a questão em branco.

O primeiro conjunto foi criado na etapa inicial do projeto, realizando-se com ele os primeiros testes. Já o segundo conjunto foi criado em uma tentativa de melhorar os resultados obtidos na fase inicial do projeto. Além disso, também pensou-se na hipótese de que capturar a forma como os participantes interagem com as questões seria capaz de fornecer informações relevantes acerca da dificuldade das questões. Assim, surgiu o segundo conjunto de *features*.

3.4 Modelagem e Avaliação Comparativa

Para a tarefa de classificação, foram utilizadas três técnicas de aprendizado supervisionado: **Regressão Logística** (linear), ***Support Vector Machine* (SVM)** (baseado em margem) e ***Random Forest*** (baseado em ensemble de árvores). Esses modelos foram escolhidos visando explorar a complexidade do problema de predição a partir de diferentes abordagens: uma linear, uma baseada em maximização de margem e uma baseada em comitês de decisão (*ensemble*).

A **Regressão Logística** foi escolhida como o ponto de partida e *baseline* linear do projeto. Por ser um modelo mais simples, rápido e de fácil interpretação, seu objetivo principal era testar a hipótese mais básica: "Existe uma relação linear simples entre as features de uma questão e sua dificuldade?". A performance desse modelo serviu como uma régua inicial, resultados mais baixos indicariam que o problema é não-linear, justificando a necessidade de algoritmos mais complexos.

O **SVM** foi selecionado por sua abordagem conceitualmente distinta e sua eficácia em encontrar fronteiras de decisão não-lineares complexas através do ajuste de kernel. Diferente da Regressão Logística, que é probabilística, o SVM busca o hiperplano ideal que maximiza a margem entre os pontos de dados mais próximos de classes opostas. Sua inclusão foi importante para testar a hipótese de que, mesmo sendo um problema complexo, os dados poderiam ser separados por uma fronteira de decisão clara.

O **Random Forest** foi escolhido como o modelo de alta performance para o projeto. Sendo um método de *ensemble* que combina centenas de árvores de decisão, ele possui características ideais para este problema:

- **Alta flexibilidade:** É capaz de aprender relações não-lineares e complexas sem assumir distribuições entre os dados.

- **Modelação de interações:** Sua principal vantagem é a capacidade de capturar interações complexas entre as *features* de forma natural, algo que outros modelos não fazem trivialmente.

- **Robustez:** Não exige padronização de dados e é menos propenso a *overfitting* do que uma única árvore de decisão.

- **Interpretabilidade:** Fornece a importância de cada *feature* como um subproduto do seu treinamento, o que foi crucial para a análise final dos dados.

A avaliação dos modelos foi realizada por meio de **Validação Cruzada com 5 *folds* (k=5)**. Para garantir uma comparação justa e tratar as diferentes escalas das *features*, foi utilizado um `Pipeline` do `Scikit-learn` que aplica a padronização dos dados (`StandardScaler`) antes do treinamento de cada modelo em cada rodada de avaliação. As métricas de performance utilizadas foram a **Acurácia** e o **F1-*Score* Ponderado**. Finalmente, foi realizado um ajuste de hiperparâmetros com `GridSearchCV` para encontrar a configuração ótima de cada algoritmo para este problema.

4 Resultados

Como mencionado anteriormente, os modelos foram avaliados utilizando acurácia e F1-*Score* ponderado, calculando-se seus valores médios por meio da Validação Cruzada (k=5). O modelo *Random Forest* obteve o melhor desempenho, seguido pelo modelo de Regressão Logística. Com a menor performance dentre os três, temos o SVM. A avaliação ocorreu em três fases: na fase inicial do projeto, com o conjunto base de *features*; em seguida, após o acréscimo das *features* `taxa_acerto` e `taxa_em_branco`; e, por fim, após o ajuste de hiperparâmetros para cada um dos algoritmos.

4.1 Desempenho Inicial dos Modelos

A primeira etapa avaliou os modelos utilizando apenas as *features* intrínsecas dos itens: os parâmetros A e C da Teoria de Resposta ao Item (TRI) e a habilidade avaliada pela questão. O objetivo foi estabelecer uma linha de base de performance. Os resultados são apresentados na Tabela 1.

Tabela 1: Resultados comparativos dos modelos

Modelo	Acurácia Média	F1-Score Médio
Regressão Logística	34,77%	32,44%
SVM	37,19%	35,02%
RF	41,76%	41,12%

Nesta fase inicial, o *Random Forest* já se mostrou superior aos outros modelos, atingindo uma acurácia de 41,76%, enquanto os outros modelos tiveram resultados próximos a de um palpite aleatório (33,3%). Isso forneceu a primeira evidência de que a relação entre os atributos e a dificuldade da questão é complexa e não-linear, sendo melhor capturada por modelos de árvore de decisão.

4.2 Adição de Features Extras

A segunda etapa incluiu o cálculo de *features* extras, com objetivo de avaliar o impacto na performance de cada um dos modelos. Nesta segunda parte, houve uma melhora relativa da performance dos três algoritmos. O *Random Forest* seguiu sendo o modelo que alcançou os melhores resultados, atingindo 47,71% de acurácia média. A Tabela 2 detalha os resultados para cada um dos modelos.

Tabela 2: Resultados comparativos dos modelos com a adição de *features*

Modelo	Acurácia Média	F1-Score Médio
Regressão Logística	39,41%	37,46%
SVM	34,84%	32,90%
RF	47,71%	45,38%

O ganho de performance do *Random Forest* pode ser analisado da seguinte perspectiva: O conjunto de *features* básico era composto por parâmetros psicométricos teóricos do item (e.g., discriminação e acerto casual). Em contraste, as *features* avançadas, `taxa_acerto` e `taxa_em_branco`, representam uma evidência empírica, refletindo a interação real de uma amostra de 100.000 estudantes com cada questão. A arquitetura deste modelo, por ser um método de *ensemble* não-linear, foi particularmente eficaz em capitalizar estas novas informações. Este resultado confirma a hipótese de que captar informações sobre a interação dos participantes com as questões poderia fornecer dados significativos aos modelos.

De forma contraintuitiva, a performance do SVM foi inferior em relação ao *baseline*, apresentando uma acurácia média 3% menor que a inicial, com seus hiperparâmetros padrões. Isso sugere que a simplificação do espaço de decisão tornou o kernel não-linear padrão do SVM menos eficaz para este novo conjunto de dados, indicando que uma otimização de hiperparâmetros ou a escolha de um kernel mais simples seria necessária para que o modelo usufrua do novo conjunto de informações.

Assim como o *Random Forest*, a Regressão Logística teve um leve salto de performance, indo de 34,77% para 39,41% de acurácia média. Tal fato indica que, com a adição das novas *features*, a relação entre as variáveis atributo e a variável alvo tornou-se mais linear e direta.

4.3 Ajuste de Hiperparâmetros e Desempenho Final

Após a criação e adição de novas *features*, foi realizado o ajuste de hiperparâmetros para otimização dos modelos. O objetivo foi realizar a comparação final com as melhores configurações possíveis de cada algoritmo, extraindo o potencial máximo para o segundo conjunto de dados. Para isso, foi utilizada a técnica de ***Grid Search* com Validação Cruzada (`GridSearchCV`, com k=5)**. Os principais hiperparâmetros otimizados foram: `C` (regularização) para Regressão Logística e SVM, `gamma` para o SVM, e `n_estimators` (número de ávores) e `max_depth` (profundidade máxima) para o *Random Forest*. A Tabela 3 resume os melhores hiperparâmetros encontrados para cada modelo e a acurácia média final correspondente.

Após a otimização, é possível confirmar a superioridade do *Random Forest* para este problema, que atingiu uma acurácia média final de 51,11%. Mesmo com os ajustes, os modelos de Regressão Logística e SVM alcançaram pouco mais de 40% de acurácia média. Esta etapa, embora tenha proporcionado um ganho de per-

Tabela 3: Resultados finais de performance após ajuste de hiperparâmetros.

Modelo	Parâmetros	Acurácia Média
Regressão Logística	`C`: 0.1	40,58%
SVM	`C`: 0.1, `gamma`: 0.01	41,76%
RF	`max_depth`: 20, `n_estimators`: 50	51,11%

formance singelo, foi importante para validar que os resultados obtidos representam o potencial máximo de cada algoritmo para o conjunto de *features* disponível. Assim, o desempenho do *Random Forest* de 51,11% é o resultado que melhor caracteriza a previsibilidade da dificuldade das questões de matemática do ENEM com a abordagem desenvolvida neste projeto.

Além destes resultados, pode-se analisar também a relevância de cada *feature* do problema para o *Random Forest*. Esta técnica revela a contribuição relativa de cada uma delas para a tomada de decisão do modelo. O gráfico abaixo ilustra essas informações.

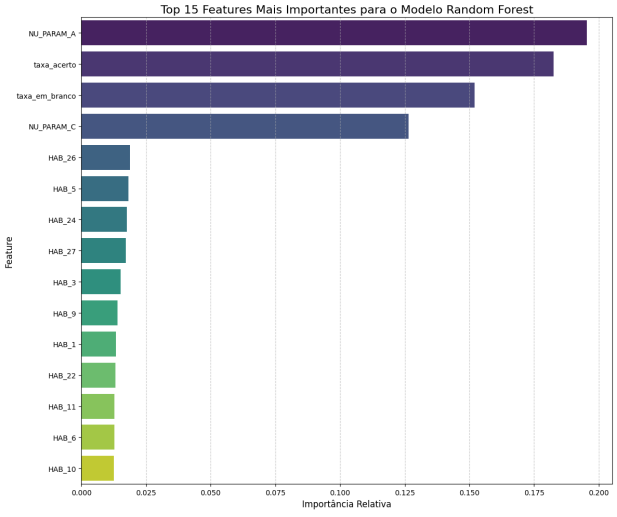

Figura 1: Importância das *features* para o *Random Forest*

Observa-se que o Parâmetro de Discriminação da TRI (`NU_PARAM_A`) emergiu como o atributo de maior poder preditivo. Este resultado sugere que a capacidade de uma questão em diferenciar alunos de alta e baixa proficiência, uma propriedade psicométrica fundamental, está intrinsecamente correlacionada com seu nível de dificuldade final (`NU_PARAM_B`), do qual a variável alvo `dificuldade_oficial` é derivada.

Em seguida, as duas *features* criadas através da engenharia de atributos, `taxa_acerto` e `taxa_em_branco`, figuram com alta relevância, somando juntas mais de 31% da importância total. O destaque da `taxa_acerto` confirma que o desempenho prático dos estudantes é uma representação bastante fiel da dificuldade teórica. Adicionalmente, a relevância da `taxa_em_branco` indica que o comportamento de evasão da questão (quando um participante opta por não responder) também é um forte sinal de sua complexidade.

5 Modelos Generativos

Ferramentas de Inteligência Artificial Generativa (como Gemini e Chat GPT) foram utilizadas para geração de código em Python para tarefas como a implementação da função de processamento de dados com a biblioteca `pandas`, a configuração de pipelines de avaliação com `Scikit-learn` e a visualização de dados com `Matplotlib`. Também foram utilizadas nas etapas de depuração, auxiliando a identificar erros de sintaxe e lógica no código. Essas ferramentas também forneceram assistência para estruturar o documento `LaTeX` deste relatório e sugestões de melhorias na redação do texto.

6 Código e Implementação

A implementação foi realizada em Python, utilizando bibliotecas como `Scikit-learn`, `pandas`, `Matplotlib` e `numpy`, dentre as principais. O código completo está disponível no repositório: <https://github.com/analauratvrs/Projeto-Final-IA.git>

7 Conclusão e Trabalhos Futuros

Este trabalho investigou a viabilidade de se prever a dificuldade oficial de itens de matemática do ENEM, definida pela Teoria de Resposta ao Item (TRI), a partir de atributos intrínsecos e empíricos das questões. Os resultados confirmam que é possível treinar modelos de *Machine Learning* para esta tarefa com uma performance superior à de um palpite aleatório. A análise comparativa, um dos objetivos centrais, demonstrou que o modelo *Random Forest* otimizado alcançou o melhor desempenho, com uma acurácia média final de 51,11%, validando-o como a arquitetura mais adequada para a estrutura do problema.

Um dos principais desafios foi a base de dados reduzida, já que foi possível obter apenas 86 questões após a filtragem de dados, algo que provavelmente tem forte influência neste resultado final obtido. Além disso, o estudo das *features* para treinamento dos modelos exigiu uma análise minuciosa para encontrar o conjunto mais adequado e de maior relevância para este trabalho, o que demandou tempo considerável.

A análise da interação dos participantes com as questões, feita pelo cálculo das *features* `taxa_acerto` e `taxa_em_branco`, demonstrou ser uma fonte promissora de informações, dado seu impacto positivo nos resultados. Trabalhos futuros podem explorar com maior profundidade este panorama, com a construção de novas *features* que enriqueçam ainda mais as análises.

Para uma abordagem ainda mais robusta, sugere-se também a utilização de recursos mais avançados, como técnicas de Processamento de Linguagem Natural para fazer análise do texto dos enunciados, a fim de capturar a complexidade semântica.

Referências

1 DE ANDRADE, Dalton Francisco; TAVARES, Heliton Ribeiro; DA CUNHA VALLE, Raquel. Teoria da Resposta ao Item: conceitos e aplicações. **ABE, São Paulo**, 2000. Disponível em: (link para o documento)

2 COSTA, Erick John Fidelis; CAMPELO, Claudio Elizio Calazans; CAMPOS, Lívia Maria Rodrigues Sampaio. Classificação automática de questões problema de matemática para aplicações do pensamento computacional na educação. 2018. In: **Anais dos Workshops do VII Congresso Brasileiro de Informática na Educação (WCBIE 2018)**. Fortaleza, CE, Brasil: SBC, 2018. Disponível em: <http://www.repositorio.ufc.br/handle/riufc/44048>.

3 JARDIM, Rafael Ris-Ala José. *Desenvolvimento de um modelo classificador de questões para o cenário educacional brasileiro fundamentado em ciência de dados.* **Universidade Federal do Rio de Janeiro (UFRJ), Rio de Janeiro, 2022.** Disponível em: <http://objdig.ufrj.br/15/teses/38051.pdf>.

4 INSTITUTO NACIONAL DE ESTUDOS E PESQUISAS EDUCACIONAIS ANÍSIO TEIXEIRA (INEP). 3,9 milhões estão inscritos no Enem 2023. Gov.br, 29 jun. 2023. Disponível em: https://www.gov.br/inep/pt-br/centrais-de-conteudo/noticias/enem/3-9-milhoes-estao-inscritos-no-enem-2023. Acesso em: 25 jun. 2025.