Modelos de predição da nota média de participantes do ENEM

Giovani de A. Valdrighi
FGV EMAp
giovani.valdrighi@gmail.com

RESUMO

O ENEM é uma prova que ocorre anualmente e possibilita a entrada de muitos brasileiros em cursos de ensino superior. No entanto, pelo seu caráter competitivo, obter um bom resultado pode ser um obstáculo maior para determinados grupos da sociedade. Este trabalho se propõe a desenvolver modelos de predição da nota média através de informações básicas e socio-econômicas dos participantes. Os modelos considerados são modelos baseados em árvores, e os resultados obtidos permitiram uma a identificação de fatores importantes para a obtenção de um bom resultado na prova.

Index Terms: Predição—Educação—Modelos de árvore—;

1 INTRODUÇÃO

O Exame Nacional do Ensino Médio (ENEM) é realizado anualmente desde 1998 com o objetivo de analisar o conhecimento de alunos que concluíram a educação básica. Alguns anos após sua criação, o ENEM já se tornou uma avaliação extremamente importante para o país, em 2001 quando alunos de escolas públicas ganharam a isenção do pagamento da prova, foram 1,6 milhões de participantes. Pela sua metodologia e ampla acessibilidade, o ENEM se tornou instrumento para a distribuição de bolsas de ensino superior. Criado em 2004, o Programa Universidade para Todos (ProUni) realiza concessão de bolsas de estudos integrais ou parciais de ensino superior, e utiliza da nota obtida no exame como critério de seleção. O Sistema de Seleção Unificada (SISU), criado em 2009, é um portal que auxilia os participantes do ENEM a se inscreverem nas diversas instituições de ensino superior públicas que utilizam da prova como ingresso, substituindo o vestibular.

Por se tornar a mais comum e mais acessível forma de ingresso em cursos de graduação, um bom resultado no ENEM se tornou o principal objetivo de muitos estudantes, em 2020 foram ao todo 5,8 milhões de inscrições, peça chave na garantia de uma carreira de sucesso e melhores condições de vida. No entanto, grupos específicos da sociedade podem possuir dificuldades no acesso a um estudo básico de qualidade, tornando a competição com demais estudantes extremamente desfavorável.

A prova é aplicada anualmente para alunos que ainda estejam cursando o ensino médio (classificados como *treineiros*) ou qualquer pessoa que já tenha concluído o ensino médio. Em 2020 começou a ser oferecida uma versão digital, mas nos últimos anos a prova foi aplicada presencialmente em escolas em todo o país dividida em dois dias de aplicação. Ela é composta por questões de 4 áreas (linguagens, matemática, ciências humanas, ciências exatas) com 90 questões cada e uma redação. Em cada uma dessas 5 formas de avaliação, o aluno pode obter uma nota de 0 a 1000. A pontuação é computada utilizando a teoria de resposta ao item (TRI), cada questão valerá um valor de pontos que é definido considerando a dificuldade da questão, a possibilidade de acerto com chute, e quão bem a questão distingue participantes que conhecem o conteúdo dos que não conhecem.

O objetivo desde trabalho é desenvolver modelos capazes de predizer para cada participante a nota média obtida utilizando das informações disponibilizadas no momento de inscrição, e com este modelo preditivo, ser capaz de gerar conclusões a cerca de quais são os fatores que se relacionam com o desempenho do aluno.

2 METODOLOGIA

2.1 Dados

O Ministério da Educação garante o acesso a informação através da disponibilidade dos dados sobre diferentes exames aplicados no país, em particular, anualmente são disponibilizados microdados sobre os participantes do ENEM, isto é, dados individuais coletados no momento de inscrição de cada um dos participantes. Os dados disponíveis podem ser separados nas seguintes categorias: informações do participante; informações sobre a escola em que o participante se formou; informações sobre a escola em que o participante fez a prova; informações sobre necessidade de atendimento especial; resposta de 27 questões socio-econômicas.

Os dados apresentam poucos valores ausentes, concentrados em sua maioria nas informações sobre a escola em que o participante se formou, algo que possivelmente ocorre pois participantes já não possuem mais as informações sobre a instituição de ensino em que se formaram.

Iremos utilizar os dados do ano de 2019 dos participantes do estado de São Paulo. Algumas etapas de limpeza são necessárias para facilitar a análise de dados e modelagem. As etapas são:

- Participantes que não comparecem em pelo menos um dos dois dias de prova são removidos; não possuímos interesse em desenvolver um modelo para predizer a nota desses alunos pois pela ausência já é fixo que a nota será 0. Com essa alteração, as variáveis que indicam presença também podem ser removidas pois não são mais úteis.

- Como desejamos predizer a nota média dos alunos, não devemos utilizar as notas em cada uma das provas como preditores, pois possuem relação direta e não conseguiremos um modelo com características interessantes. As notas de cada uma das áreas são removidas.

- O ano de coleta dos dados e o número de inscrição do participante são removidos visto que evidentemente não possuem relação com a nota obtida.

- Localização da residência do participante, localização da escola em que cursou ensino médio e localização da escola em que realizou a prova são removidas. Tais variáveis apresentam valores nominais e dificilmente podem ser implementadas em modelos pela alta quantidade de valores assumidos, o que geraria um alto custo computacional ao serem consideradas como categorias.

- Entre as colunas restantes, as que apresentam dados ausentes são (e as respectivas porcentagens de ausência): se a escola em que cursou o ensino médio é pública ou privada (43%); se a escola em que cursou o ensino médio é municipal, estadual ou federal (70%); se a escola em que cursou o ensino médio

é urbana ou rural (70%); se a escola em que cursou o ensino médio ainda está em funcionamento (70%). Tais colunas não serão usadas em modelos pela grande quantidade de dados ausentes.

2.2 Análise exploratória dos dados

Foi produzida uma análise exploratória dos dados para obtermos uma intuição inicial sobre o comportamento dos dados e as relação dos preditores com a variável predita.

Inicialmente iremos analisar as variáveis sexo, cor/raça, quantidade de computadores em casa e faixa de renda. Para cada uma dessas variáveis, iremos exibir um gráfico com a densidade das notas médias para cada uma das categorias que a variável pode assumir. Com essa visualização, seremos capazes de identificar se a distribuição da nota média é independente da variável analisada, ou se para diferentes valores assumidos a nota média possui uma distribuição distinta. Tal visualização está presente na Figura 3.

No primeiro gráfico podemos observar que participantes do sexo feminino possuem notas um pouco menores; no segundo as menores notas são de participantes da categoria preta, parda e indígena; no terceiro identificamos quanto maior o número de computadores em casa maior a nota; e no último gráfico quanto maior a renda maior a nota. Dessa forma, todas as variáveis observadas apresentam relação com a distribuição da nota média.

Podemos gerar outra visualização similar, desta vez analisando se as informações sobre a escola em que o participante realizou a prova possui possuem relação com a nota média, as variáveis são a dependência administrativa da escola, a localização e a situação de funcionamento. Com essa segunda visualização (Fig. 4) somo capazes de ver que participantes que realizaram provas em escolas federais e privadas obtiveram maiores notas, mas no entanto, com as outras visualização uma relação entre as variáveis e a nota média não é tão clara, a menos que participantes cuja escola esteja extinta possuem notas maiores.

2.3 Modelos

É apresentado os 3 modelos que serão utilizados para o desenvolvimento de um modelo preditivo.

2.3.1 Árvores de decisão

Árvores de decisão são modelos não-paramétricos de aprendizado supervisionado que podem ser utilizados para classificação e regressão. Através de um conjunto de regras (decisões) utiliza das variáveis observadas para realizar uma predição. A observação percorre a árvore através de partições, em cada uma das partições ela pode seguir para um dos dois galhos a depender de seu valor e da regra da partição, até chegar em uma das folhas, e todas que pertencerem a esta folha possuem a mesma predição. Uma das vantagens da árvores de decisão é que se trata de um modelo simples e que permite interpretabilidade.

No entanto, obter o número de partições e e as regras específicas que garantem o menor erro de predição é um problema *NP-hard* (precisa de muito tempo para ser resolvido), desse modo, algoritmos de aprendizado utilizam de métodos gulosos para construir o modelo. Iterativamente o modelo calcula a melhor partição para o nó atual da árvore, para isto, ele considera cada uma das variáveis dos dados, e para cada uma dessas variáveis, todas as possíveis separações dessa variável, e entre todas as possibilidade a partição escolhida é aquela que minimiza uma métrica preditiva, como o erro quadrático médio (*mse*).

Com este algoritmo, caso a árvore não possua critérios de parada, ela irá crescer (em quantidade de decisões/partições) até que cada uma das folhas possua apenas uma observação, no entanto, isso irá gerar um *overfitting* nos dados de treinamento, com o modelo sendo incapaz de generalizar. Para lidar com isto, o algoritmo de treinamento possui hiper-parâmetros de parada. Podemos limitar a profundidade máxima que a árvore pode alcançar (número de regras sequenciais); o número mínimo de amostras para realizar uma partição; e o número mínimo de amostras em cada uma das folhas.

Para lidar com isso, iremos realizar um afinamento desses hiperparâmetros criando modelos para um conjunto hiper-parâmetros e selecionar aquele que gerará o menor R^2. Os valores considerados serão (10, 15, 20) para a profundidade máxima, (20, 30, 40) para o número mínimo de amostras para partição e (30, 45, 60) para o número mínimo de amostras em cada folha.

Será utilizada a classe *Decision Tree* oferecida pela biblioteca *Scikit-learn*, no entanto, tal implementação não possui suporte para variáveis categóricas (que são frequentes em nossos dados), e ao utilizar *one hot encoding* teremos uma aproximação do que seria o modelo treinado com variáveis categóricas, pois tal ajuste não lidará com o fato de que uma variável categórica pode assumir apenas uma categoria. Iremos gerar um modelo apenas utilizando as variáveis numéricas e um modelo utilizando as numéricas e as categóricas com *one hot encoding*.

2.3.2 CatBoost

Uma das principais limitações das árvores de decisões é a sua facilidade para resultar em um *overfitting*, para lidar com isso existem métodos que combinam os resultados de diversas árvores construídas de formas distintas para obter uma generalização. O método chamado *boosting* utiliza diferentes modelos de previsão que são treinados sequencialmente utilizando a informação obtida no treinamento do modelo anterior, aqui estaremos utilizando a biblioteca *CatBoost* [3], que disponibiliza um algoritmo de treinamento *gradient boosting* com árvores de decisão com suporte para variáveis numéricas e categóricas.

Para minimizar uma função de perda, o método *gradient boosting* utiliza de um modelo preditivo fraco, no caso, árvores de decisão com pequeno número de regras (baixa profundidade), e em sequência outras árvores são treinadas, não para predizer a variável objetivo, mas para predizer o erro que a árvore anterior gerou, após isso os resultados da nova árvore são somados com os resultados da árvore anterior (com pesos), e uma subsequente árvore será treinada com o novo resíduo gerado pela combinação de árvores.

Os hiper-parâmetros que irão passar por afinamento é o número de árvores a ser treinado (*iterations*); a taxa de aprendizado que atribui um peso para cada uma das árvores; e a profundidade de cada uma das árvores. Os valores avaliados serão (400, 500, 600) para o número de árvores, (0.1, 0.2, 0.5) para a taxa de aprendizado e (6, 10, 14) para a profundidade.

2.3.3 Distributed Random Forest

De forma similar, o modelo floresta aleatória cria uma floresta de árvores de decisão, no entanto, utilizando um método *bagging*, cada uma das árvores é criada utilizando um subconjunto amostrado dos dados, e além disso, cada árvore é criada com um subconjunto dos preditores. Dessa forma, lidamos com um problema das árvores de decisão que é sua alta variância através da criação de diversas árvores não correlacionadas (pela utilização de subconjunto de preditores) e utilizamos a média da predição das árvores como a predição.

A plataforma de aprendizado de máquina *H2O.ai* oferece a implementação de um algoritmo de treinamento de *Distributed Random Trees* com suporte para variáveis numéricas e categóricas e adaptações para lidar com grandes quantidades de dados.

Os hiper-parâmetros que serão utilizados para o afinamento são o número de árvores a serem treinadas e profundidade máxima de cada uma dessas árvores. Os valores a serem avaliados serão (25, 50, 75) para o número de árvores e (5, 10, 15) para a profundidade máxima.

2.4 Treinamento

Como já apresentado nesta seção, serão avaliados 3 modelos distintos, e para cada será realizado um processo de afinamento dos hiper-parâmetros. O professo de avaliação ocorrerá da seguinte forma: os dados serão separados em treino (80%) e teste (20%), os dados de treino serão utilizados para escolher os melhores hiper-parâmetros, dado uma seleção de hiper-parâmetros possíveis, para cada combinação será realizado um processo de validação cruzada (*cross validattion*) utilizando 3 separações com os dados de treino, os melhores hiper-parâmetros serão aqueles que apresentarem o melhor R^2 médio nas 3 separações da validação cruzada, por fim, os melhores parâmetros serão avaliados nos dados de treino.

3 Resultados

Os resultados dos modelos com os melhores hiper-parâmetros se encontram na Tabela 1. Como podemos ver, os valores de R^2 obtido variam em torno de 0.2 e 0.3, mostrando modelos com não tanto poder preditivo. Notamos também que um modelo simples como a árvore de decisão já foi capaz de apresentar um resultado próximo ao de modelos mais complexos, obtendo até mesmo melhor resultados do que o modelo CatBoost.

Na Figura 1, apresentamos uma visualização da nota observada contra a nota prevista para os dados de treinamento e teste, notamos que os pontos estão distribuídos ao redor da reta identidade, no entanto, quando a nota média observada é maior de 700 pontos, nosso modelo tendeu a predizer valores menores.

Podemos também obter a importância de cada uma das variáveis, a biblioteca *H2O.ai* dispõe um cálculo de importância de variáveis através da influência relativa: se a variável foi selecionada para fazer uma partição, e o quanto essa partição diminuiu o erro (em todas as árvores), baseado no artigo de Friedman [1]. na Figure 2 temos a importância normalizada das 15 mais importantes variáveis, é importante notar que a variável mais importante para a predição é a faixa de renda, seguida por diversas variáveis que indicam o nível social do participante e da sua família. Um resultado interessante é que a quantidade de banheiros na casa se mostrou como um bom preditor para a nota média, o que pode parecer contra-intuitivo. Deve ser lembrado que tais resultados não representam relações de causalidade, isto é, se um preditor tem relação com a variável predita, ele não necessariamente é a causa da variável predita.

Além disso, ao olharmos as variáveis menos importantes, temos que todas aquelas que indicam a necessidade de atendimento especial apresentaram baixíssima importância normalizada (menor do que 0.001), o que indica que a distribuição da nota média não apresenta grande diferença entre participantes com necessidades especiais e os demais.

Modelo	Parâmetros	R^2
Árvore de decisão (dados numéricos)	profundidade máx.: 15; amostras mín. partição: 30; amostras mín. folhas: 60	0.250
Árvore de decisão	profundidade máx.: 15; amostras mín. partição: 40; amostras mín. folhas: 60	0.325
CatBoost	núm. de árvores: 650; taxa de aprendizado: 0.5; profundidade: 10	0.302
Distributed Random Forest	núm. de árvores: 75; profundidade: 15	0.374

Tabela 1: Resultados para os melhores hiper-parâmetros de cada modelo.

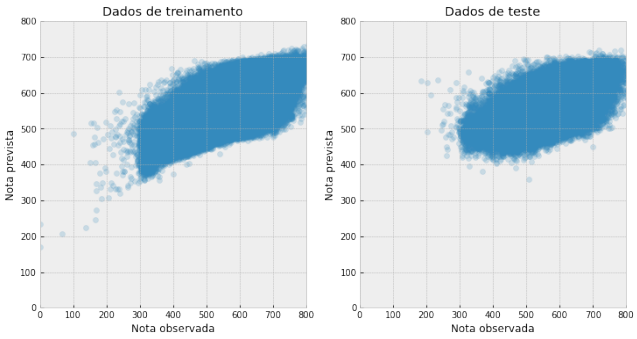

Figura 1: Resultado de predição.

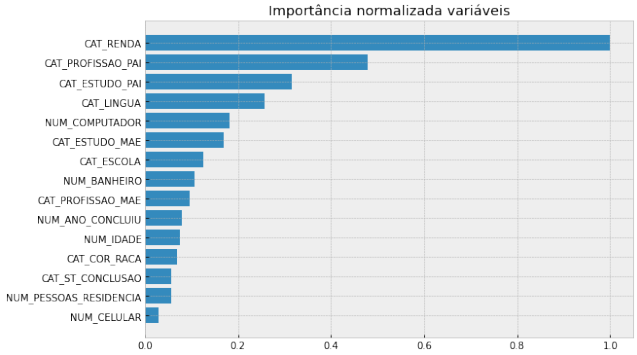

Figura 2: Importância das variáveis do modelo de *Distributed Random Trees*.

4 Conclusão

Com o trabalho apresentando desenvolvemos 4 modelos distintos para a predição da nota média de participantes do ENEM, obtendo com o modelo mais preciso, *Distributed Random Forest*, um R^2 de 0.374, um valor que apesar de baixo, apresenta resultados interessantes por tratar da predição de uma variável com alta variância e que possui inúmeros aspectos sociais e cognitivos que a influenciam. Além disso, fomos capazes de identificar relações entre as variáveis preditoras e a predita, com a análise da importância de variáveis de nosso modelo, vemos que a condição econômica do participante e a experiência estudantil e profissional dos pais possui alto poder de predição da nota média.

Tal modelagem ainda pode ser desenvolvida, uma alternativa é passar a utilizar dados de vários anos e de vários estados para o treinamento, e pela grande quantidade de dados disponíveis, utilizar de redes neurais para obter modelos com maiores R^2 e utilizar técnicas de interpretabilidade de redes neurais para obter também um resultado de análise de variáveis importantes.

Referências

[1] J. H. Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pp. 1189–1232, 2001.

[2] G. James, D. Witten, T. Hastie, and R. Tibshirani. *An introduction to statistical learning*, vol. 112. Springer, 2013.

[3] L. Prokhorenkova, G. Gusev, A. Vorobev, A. Veronika Dorogush, and A. Gulin. CatBoost: unbiased boosting with categorical features. *arXiv e-prints*, p. arXiv:1706.09516, June 2017.

A Apêndice

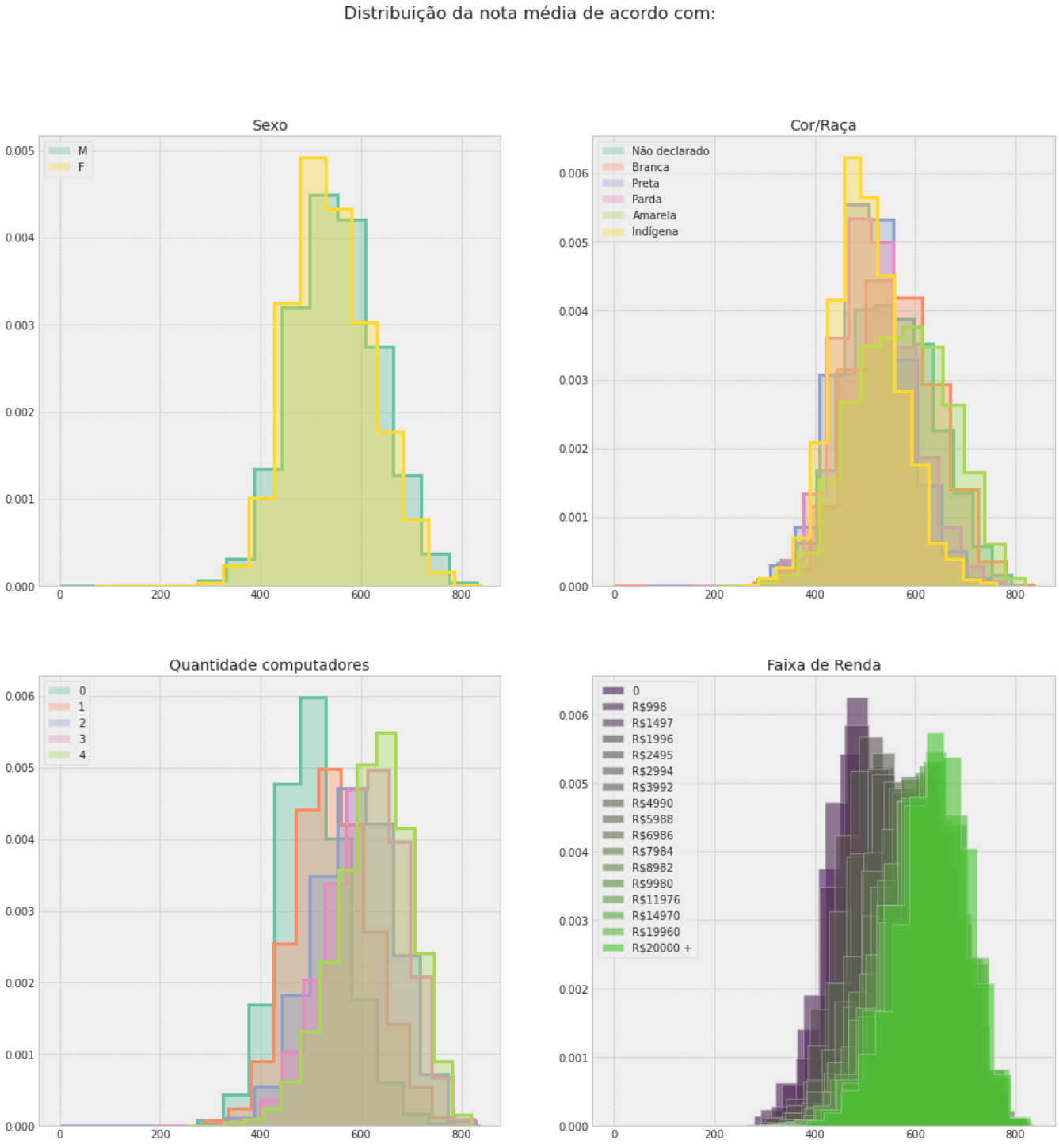

Figura 3: Gráfico de distribuição da nota média condicionada a diferentes variáveis do participante.

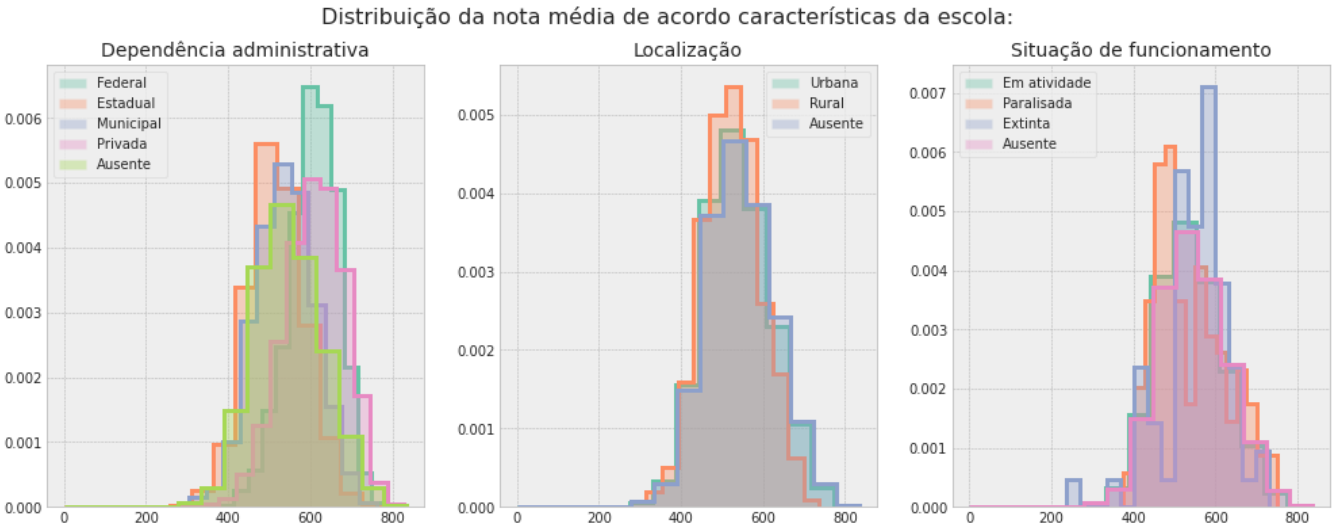

Figura 4: Gráfico de distribuição da nota média condicionada a diferentes variáveis da escola de aplicação da prova.

Profundidade máx.	Amostras. mín. partição	Amostras mín. folhas	R^2
10	20	30	0.249
10	20	45	0.249
10	20	60	0.249
10	30	30	0.249
10	30	45	0.249
10	30	60	0.249
10	40	30	0.249
10	40	45	0.249
10	40	60	0.249
15	20	30	0.247
15	20	45	0.249
15	20	60	0.251
15	30	30	0.247
15	30	45	0.249
15	30	60	0.251
15	40	30	0.247
15	40	45	0.249
15	40	60	0.251
20	20	30	0.240
20	20	45	0.246
20	20	60	0.250
20	30	30	0.240
20	30	45	0.246
20	30	60	0.250
20	40	30	0.240
20	40	45	0.246
20	40	60	0.250

Tabela 2: Resultados árvore de decisão com dados numéricos.

Profundidade máx.	Amostras. mín. partição	Amostras mín. folhas	R^2
10	20	30	0.317
10	20	45	0.317
10	20	60	0.318
10	30	30	0.317
10	30	45	0.317
10	30	60	0.318
10	40	30	0.317
10	40	45	0.317
10	40	60	0.318
15	20	30	0.317
15	20	45	0.322
15	20	60	0.324
15	30	30	0.317
15	30	45	0.322
15	30	60	0.324
15	40	30	0.317
15	40	45	0.322
15	40	60	0.324
20	20	30	0.305
20	20	45	0.315
20	20	60	0.321
20	30	30	0.305
20	30	45	0.315
20	30	60	0.321
20	40	30	0.305
20	40	45	0.315
20	40	60	0.321

Tabela 3: Resultados árvore de decisão com dados numéricos e categóricos.

Iterações	Taxa aprendizado	Profundidade	R^2
400	0.1	6	0.188
400	0.1	10	0.215
400	0.1	14	0.257
400	0.2	6	0.262
400	0.2	10	0.285
400	0.2	14	0.252
400	0.5	6	0.293
400	0.5	10	0.296
400	0.5	14	0.290
500	0.1	6	0.228
500	0.1	10	0.278
500	0.1	14	0.262
500	0.2	6	0.278
500	0.2	10	0.296
500	0.2	14	0.287
500	0.5	6	0.293
500	0.5	10	0.303
500	0.5	14	0.296
650	0.1	6	0.242
650	0.1	10	0.271
650	0.1	14	0.277
650	0.2	6	0.290
650	0.2	10	0.299
650	0.2	14	0.305
650	0.5	6	0.304
650	0.5	10	0.308
650	0.5	14	0.305

Tabela 4: Resultados do modelo CatBoost.

Núm. árvores	Profundadide máx.	R^2
25	15	0.359
25	20	0.337
25	25	0.309
50	15	0.366
25	5	0.316
25	10	0.354
25	15	0.359
50	5	0.320
50	10	0.356
75	5	0.321
75	10	0.357
75	15	0.368
100	10	0.357

Tabela 5: Resultados do modelo Distributed Random Tree.