Prevendo Desempenho dos Candidatos do ENEM Através de Dados Socioeconômicos

Bernardo Stearns[1], **Flavio Rangel**[1],
Fabrício Firmino[2], **Fabio Rangel**[12], **Jonice Oliveira**[12]

[1]Departamento de Ciência da Computação
Universidade Federal do Rio de Janeiro (UFRJ)
Rio de Janeiro – RJ – Brazil

[2]Programa de Pós-Graduação em Informática (PPGI)
Univesidade Federal do Rio de Janeiro
Rio de Janeiro, RJ – Brazil

{bernardo.stearns, flm.rangel, firminodefaria}@gmail.com

{fabiorangel, jonice}@ufrj.br

Abstract. *The present work analyzed the possibility to predict students performance based only in socioeconomic status. The dataset used in this work was extracted from the most important examn to join Brazilian Universities: National High School Examn (ENEM). The study compared the performance of two decision trees ensemble methods, in the task of predicting the scholar grade using socioeconomic data. The results show that socio-economic indicators can partly explain a bias in student scores.*

Resumo. *O presente artigo analisou a possibilidade de prever a performance de estudantes baseando-se apenas em suas informações socioeconômicas. O trabalho utilizou dados do exame mais importante para adentrar em universidades brasileiras: Exame Nacional do Ensino Médio (ENEM). O estudo comparou a capacidade de generalizar de dois métodos de agrupamento de árvores de decisão, na tarefa de regressão da nota por meio dos dados socioeconômicos. Os resultados apontaram que existe um viés significativo das características socioculturais dos alunos sobre as notas.*

1. Introdução

A mineração de dados educacionais é um campo emergente que consiste em analisar um grande volume de dados educacionais com objetivo de obter padrões não-triviais [Vahdat et al. 2015]. Esse campo utiliza técnicas de reconhecimento de padrão em contextos educacionais para criar módulos cognitivos de alta complexidade que possibilitem entender o processo de aprendizado, por exemplo, durante a prática de jogos eletrônicos [Lee et al. 2014]. Outro exemplo é a utilização de sistemas de recomendação para intensificar o aprendizado de estudantes durante o processo de estudo, informando quais tópicos eles precisam melhorar [Segal et al. 2014].

O Exame Nacional do Ensino Médio (ENEM) é uma política criada em 1998 pelo governo federal brasileiro com objetivo de avaliar a qualidade da educação das escolas de ensino médio e prover uma padronização do conhecimento. Muitas universidades

brasileiras adotaram a nota do ENEM como critério de admissão. Em 2014, o ENEM contou com mais de 9.5 milhões de inscritos. Todo aluno que se inscreve para fazer a prova precisa preencher um questionário socioeconômico. O dataset de cada ano é disponibilizado de forma gratuita para *download*[1].

O presente trabalho realizou um estudo sobre a capacidade de prever a nota de um aluno utilizando somente as informações do questionário socioeconômico como *features*. Para isso, foi necessário escolher um modelo de regressão capaz de trabalhar com um volume muito grande de dados. Árvores de Decisão são modelos capazes de atuar com sucesso nesse cenário [Hastie et al. 2005]. Além disso, modelos de árvores podem prover um entendimento das regras construídas para aquela regressão. Todo modelo que apresenta essa possibilidade é chamado de modelo *whitebox*, diferenciando-se de modelos *blackbox*, tal como Redes Neurais e Support Vector Machines [Satyanarayana et al. 2014] [Krishnaiah et al. 2014].

Um modelo de Árvore de Decisão, quando utilizado sozinho, é considerado um algoritmo preditivo "fraco"(*weak learner*). Uma possível solução é combinar diversos modelos de árvore, criando um classificador com alto poder preditivo [Woźniak et al. 2014]. Este trabalho comparou duas técnicas de *boosting* para agregar Árvores de Decisão: Gradient Boosting [Chen and Guestrin 2016] e AdaBoost [Freund and Schapire 1997]. Embora esses agregadores melhorem a capacidade da regressão, o número de hiperparâmetros se torna maior, tornando mais difícil encontrar um conjunto que defina um modelo ótimo, tanto para as árvores quanto para as técnicas de *Boosting*. A tarefa de encontrar esse conjunto é conhecida como Model Selection [Bergstra and Bengio 2012]. Para contornar esse problema, utilizou-se uma heurística de otimização baseada em enxame de partículas chamada PSO (Particle Swarm Optimization) [Kennedy 2011].

O trabalho então pode ser dividido em duas etapas: otimização de hiperparâmetros e validação do experimento. Na primeira etapa, criou-se uma função objetivo utilizando duas métricas para avaliar a regressão: MAPE (Mean Absolute Percentage Error) e R Squared (R^2). Na segunda, aplicou-se 10-Fold Cross Validation utilizando como métricas: MAE (Mean Absolute Error) e R^2. A comparação da performance dos regressores foi feita utilizando Paired t-Test sobre as amostras obtidas através dos folds da validação. Os resultados mostraram que os indicadores socioeconômicos podem explicar um viés na pontuação dos alunos, sendo possível prever a nota do aluno com um MAE de 65.9 pontos, enquanto o MAE da média é de 97.27. Alguns resultados e a metodologia apresentada neste trabalho foram publicados no *25th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, e podem se encontrados em [Stearns et al. 2017].

2. Metodologia

O presente trabalho propõe a utilização de métodos de aprendizado de máquina na tarefa de regressão das notas do exame de Matemática do ENEM utilizando como entrada informações socioeconômicas. O exame de Matemática foi escolhido devido a alta variância das notas. A regressão é considerada possível se o modelo preditivo consegue prever a nota melhor do que a média. Para isso é necessário um teste estatístico cuja

[1]http://dados.gov.br/dataset/microdados-do-exame-nacional-do-ensino-medio-enem

hipótese nula admite que a regressão pela média e utilizando o modelo sejam equivalentes. As observações serão obtidas por 10-Fold Cross Validation, utilizando como métricas MAE e R^2.

2.1. Dataset

O dataset utilizado possui aproximadamente 6.11 gigabytes, provido pelo INEP (Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira), referente ao exame do ENEM de 2014. O dataset é dividido em três partes:

- A primeira parte contém informações preenchidas pelo estudante ao fazer o registro para a prova. Consiste nas seguintes características: idade, gênero, estado civil, etnia, grau de escolaridade, tipo da escola (pública ou privada), estado de residência, região de residência.
- A segunda parte possui respostas à perguntas também respondidas no ato da inscrição:
  - Qual a escolaridade dos seus pais?
  - Qual é a renda da sua família?
  - Quantas pessoas moram na sua casa?
  - Possui acesso a televisão ou internet na sua casa?
  - Você trabalha?
  - Onde localiza-se sua residência?
- A última parte é referente a performance do estudante em cada disciplina, que consiste nos seguintes exames: Matemática, História & Geografia, Língua Portuguesa, Física & Química e Redação.

2.2. Preprocessamento

O preprocessamento aplicado no dataset pode ser resumido em: (i) Filtro de Instâncias; (ii) *Feature Selection*; (iii) *Feature Engineering*; (iv) *Data Normalization*. Tais passos foram aplicados nessa sequência visando padronizar os dados.

2.2.1. Filtro de Instâncias

Devido a presença de estudante no dataset que faltaram o exame, foi necessário remover essas instâncias. Deve-se ter em mente que o trabalho não visa prever os alunos que deixariam de comparecer ao exame.

2.2.2. Feature Selection

Essa etapa selecionou um conjunto de *features* para o modelo de aprendizado. Esse conjunto de *features* precisam compor um espaço vetorial cuja dimensão seja razoável. Dessa forma, incluir alguns itens do dataset, como por exemplo escola onde estudou, poderia levar a um espaço vetorial de dimensão imensa (número de diferentes escolas), e possivelmente atribuindo um viés ao modelo de aprendizado para alguns estudantes específicos, incapacitando o modelo de generalizar.

2.2.3. Feature Engineering

Essa etapa consistiu em transformações das *features* de forma a permitir que sejam utilizadas pelos algoritmos de aprendizado, ou mesmo melhorar o desempenho deles. Um passo importante dessa etapa se resume em mapear variáveis categóricas do dataset para valores numéricos, por meio de um encode binário. Cada possível valor para uma determinada feature categórica se torna uma nova feature.

2.2.4. Data Normalization

Data Normalization é um tipo de preprocessamento para variáveis numéricas em um vetor de dados. A heterogeneidade das *features* torna difícil uma análise estatística [Uragun and Rajan 2011]. Essa etapa usou a normalização Min-Max, que usa a informação de valor máximo e mínimo para cada feature, alterando seus valores. A Equação 1 apresenta como é obtido o valor v' a partir do valor v para a feature A. A variável u é o maior valor da feature A, enquanto l é o menor.

$$v' = \frac{v - min_A}{(max_A - min_A)}(u - l) + l \tag{1}$$

2.3. Algoritmos de Aprendizado

Dois métodos de agrupamento de Árvores de Decisão foram comparados nesse trabalho: AdaBoost e Gradient Boosting. Boosting refere-se à um método efetivo de produzir modelos acurados. A escolha em utilizar Árvores de Decisão se deve ao fato de conseguirem lidar bem com grande volume de dados, além de serem modelos *whitebox*, provendo regras que geraram o modelo treinado.

2.3.1. AdaBoost

AdaBoost é um algoritmo baseado em Boosting. Os detalhes da teoria do AdaBoost estão presentes em [Freund and Schapire 1997]. Um regressor AdaBoost é um meta-estimador que utiliza múltiplas árvores combinadas (”weak learners”) para produzir um modelo melhor. Cada árvore é ajustada utilizando parte do dataset. O peso de cada instância de árvore é ajustado baseado no erro da predição. A Equação 2 apresenta um AdaBoost com T regressores, cada $h_t(x)$ representa uma árvore de regressão, onde α_t é o peso associado.

$$H(x) = \sum_{t=1}^{T} \alpha_t h_t(x) \tag{2}$$

O AdaBoost garante que o agrupamento será melhor contanto que a performance de cada modelo seja melhor que o aleatório, sendo assim a função de perda será exponencial [Lafferty 2002].

2.3.2. Gradient Boosting

A implementação do Gradient Boosting utilizada nesse trabalho se chama XGBoost. XGBoost é uma melhoria do algoritmo padrão de Gradient Boosting [Friedman 2001]. Como qualquer outro método de Boosting, ele combina "weak learners"para produzir um melhor. O método utiliza os preditores (árvores) que minimizem a função de perda escolhida. A função de perda é composta por dois fatores: uma taxa de erro calculada sobre a validação e um fator de regularização. XGBoost é escalável e seus detalhes estão descritos em [Chen and Guestrin 2016].

2.4. Metricas de Avaliação

As seguintes métricas foram utilizadas para avaliar a performance dos regressores: Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE) e R-Squared (R^2). Nas equações a seguir, y_i é o valor correto para o exemplo i, enquanto $\hat{y}_i$ é a predição para o mesmo exemplo.

MAE calcula a distância absoluta entre o valor correto e a predição, para todos os exemplos, calculando uma média ao final. A Equação 3 apresenta o MAE.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{3}$$

MAPE mede o erro médio em termos percentuais. A principal utilização dessa métrica no trabalho é a composição de uma função objetivo para a otimização de hiperparâmetros que inclua uma métrica de erro e o R^2. Como o resultado do MAPE se encontra no intervalo $[0, 1]$, escolheu-se essa métrica para a composição. Sua fórmula está descrita na Equação 4.

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{\hat{y}_i}\right| \tag{4}$$

R^2 calcula a razão entre a variância do estimador e a variância das observações. Seu valor está no intervalo $[0, 1]$, onde 1 representa a melhor performance. Na Equação 5, o R^2 é apresentado e $\bar{y}$ é o valor médio para y.

$$R^2 = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{5}$$

2.5. Otimização de Hiper-Parâmetros

Otimização de hiper-parâmetros é o problema de escolher um conjunto de hiperparâmetros para um algoritmo de aprendizado [Bergstra and Bengio 2012]. Como o conjunto define um modelo, essa tarefa também é conhecida como seleção de modelo. Para essa etapa, é necessário um dataset e uma métrica de avaliação. Como os parâmetros do problema apresentado são todos numéricos, em um espaço vetorial, Particle Swarm Optimization (PSO) foi utilizado. Para essa otimização utilizou-se 30% do dataset original. Essa amostra do dataset não foi utilizada na validação para evitar contaminação.

No PSO, um número de partículas é posicionada no espaço de busca do problema ou função, e cada partícula avalia individualmente a função objetivo de acordo com seu posicionamento [Kennedy 2011]. Depois disso, cada partícula se move de acordo com três vetores: velocidade, *pbest* e *gbest*. O *pbest* é o vetor obtido pela diferença entre a sua posição e a melhor posição que a partícula visitou. O *gbest* é obtido pela diferença entre a sua posição e a melhor posição já visitada por alguma partícula. Cada vetor possui um peso que é um parâmetro do algoritmo. No presente trabalho, cada vetor possuiu o mesmo peso (0.5) e utilizou-se 100 partículas. O PSO foi utilizado em sua forma canônica.

A função objetivo dessa otimização foi construída utilizando o MAPE e o R^2. Para cada partítula, a média e o desvio padrão foi calculado repetindo a função apresentada na Equação 6 30 vezes. Para cada repetição foi feita uma amostra de tamanho 1052. Metade foi utilizado para treino e metade para avaliar. Essa amostragem considera um intervalo de confiança de 95%, com erro de 5%, do dataset original. Ao final, o resultado da função objetivo era a média das 30 execuções somando o desvio padrão, em um problema de minimização. Na Equação 6, X é a amostra. Essa função objetivo é uma das contribuições desse trabalho, cuja saída se encontra no intervalo $[0, +\infty)$, composta por duas métricas diferentes.

$$function(X) = 1 + MAPE(X) - R^2(X) \tag{6}$$

Para o XGBoost (Gradient Boosting), os hiper-parâmetros otimizados foram: *maximum depth*, *learning rate*, *minimum child weight*, *gamma*, *sub-sample size*, *column sample by tree*, e *number of rounds*. Onde *gamma* é o peso do termo de regularização. Para o AdaBoost otimizou-se os seguintes hiper-parâmetros: *maximum depth*, *number of estimators*, e *learning rate*.

2.6. Validação

A validação foi aplicada para avaliar os modelos utilizando 70% do dataset. Note que essa amostra não foi utilizada na etapa de otimização de hiper-parâmetros. Este trabalho utilizou k-Fold Cross Validation [Refaeilzadeh et al. 2009] com $k = 10$. Utilizando esse processo de validação, o dataset é dividido em 10 partes. Então, cada parte é utilizada como teste enquanto as outras 9 são usadas para treinar o modelo. O processo é repetido 10 vezes, trocando a parte utilizada como teste. Essa validação foi utilizada para calcular o resultado final da predição, após a etapa de otimização de hiper-parâmetros.

3. Resultados dos Experimentos

Primeiramente, a seleção dos hiper-parâmetros foi conduzida utilizando o PSO para os dois modelos de Boosting. Os parâmetros encontrados para ambos modelos estão descridos na Tabela 1 and Tabela 2.

A imagem da função objetivo é apresentada na Figura 1. Embora os valores dessa imagem possuem um pequeno decaimento ao longo das iterações, é importante perceber que esse valor é uma porcentagem, e uma pequena variação pode representar uma grande diferença no resultado da predição.

Após a otimização de hiper-parâmetros, 70% do dataset foi utilizado para aplicação do 10-Fold Cross Validation. Um teste estatístico Paired t-test foi aplicado para

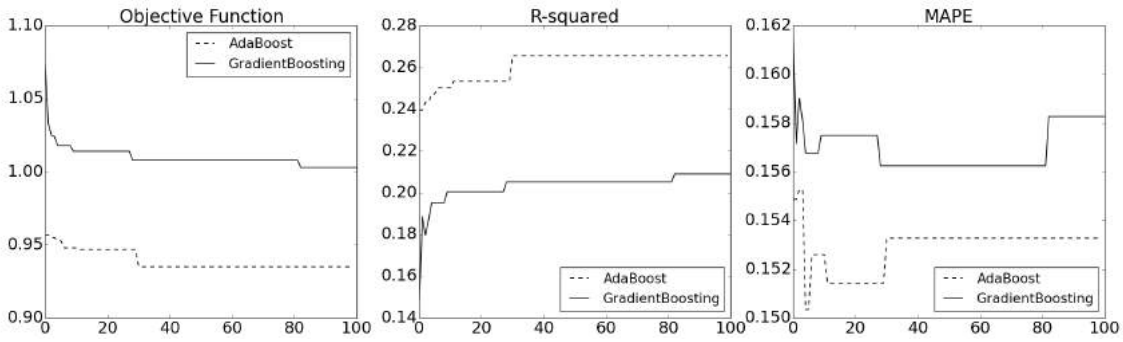
Tabela 1. XGBoost Parameters

Hyper-Parameter	Value
maximum depth	14
learning rate	0.1
minimum child weight	4
gamma	5.03
subsample size	0.899
column sample by tree	0.875
number of rounds	228

Tabela 2. AdaBoost Parameters

Hyper-Parameter	Value
maximum depth	5
learning rate	0.27
number of estimators	221

Figura 1. Busca pelo conjunto de Hiper-Parâmetros utilizando PSO [Stearns et al. 2017].



obter relevância estatística sobre a comparação dos resultados. A hipótese nula afirma que a média das observações dos resultados de uma determinada métricas para ambos modelos é igual. Cada valor médio dos folds para cada métrica está presente na Tabela 3. É possível perceber que o Gradiente Boosting obteve resultado superior em ambas métricas. Além disso, ambas métricas conseguem prever o resultado melhor que o valor médio, que possui MAE de 90.27 pontos, representando que existe informação presente nos dados socioeconômicos que permite uma predição. Além disso, prever pela média representa um R^2 zero.

Tabela 3. Resultado do 10-Fold Cross Validation.

	Gradient Boosting	AdaBoost	p-valor
MAE	65.90 ± 0.11	72.66 ± 1.37	1.27×10^{-7}
R^2	0.35 ± 0.0014	0.18 ± 0.0354	1.34×10^{-7}

Como o XGBoost apresentou o melhor resultado e possui uma implementação de importância de *features* baseando-se na quantidade de vezes que uma *features* foi utilizada para um nó de decisão da árvore, aproveitou-se para analisar quais foram as informações foram mais relevantes para o modelo. A Tabela 4 apresenta um rank com as 10 *features*

mais importantes para o modelo.

Tabela 4. Ranking das Features

Rank	Feature
1	Longitude
2	Latitude
3	Idade
4	Motivo de Realizar o ENEM: Ingressar no Ensino Privado
5	Ano de Conclusão do Ensino Médio
6	Renda Mensal Familiar
7	Motivo de Realizar o Enem: Financiamento do FIES
8	Quantidade de Pessoas que Moram na mesma Residência
9	Motivo de Realizar o ENEM: Aumentar Possibilidade de Emprego
10	Idade que Começou a Exercer Atividade Remunerada

4. Conclusão e Trabalhos Futuros

O presente trabalho trouxe a proposta de avaliar dois métodos de regressão baseados em Boosting de árvores de decisão, para prever a nota de um estudante no ENEM baseando-se somente no questionário socioeconômico. Os resultados apontaram que existe um viés desses dados sobre a nota. Dois modelos foram utilizados para a tarefa de regressão: AdaBoost e Gradient Boosting. Para avaliar os modelos, três diferentes métricas foram utilizadas: MAE, MAPE e R^2. Uma etapa de otimização de hiper-parâmetros foi realizada utilizando uma heurística de otimização baseada em enxame de partículas: PSO. Os resultados apresentaram melhor resultado para o Gradient Boosting, com alta relevância estatística. O trabalho também apresenta um rank das *features* mais importantes nesse primeiro momento, algo que pode ser utilizado como base para entender o viés na nota.

Alguns resultados dessa pesquisa foram publicados no *25th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, cujo qualis é B1, e podem se encontrados em [Stearns et al. 2017]. Resultados parciais foram apresentados na XXXVIII Jornada Giulio Massarani de Iniciação Científica, Tecnológica, Artística e Cultural (JICTAC 2016) da UFRJ. E esse trabalho serve como base para estudos científicos na área de educação e sociedade, com possibilidade de aplicações para melhorar o sistema de ensino brasileiro.

Contribuições do Bernardo Stearns

Embora o aluno Bernardo Stearns estivesse na graduação no decorrer do presente trabalho, ele foi responsável por todo o desenvolvimento prático desta pesquisa, tendo trabalhado em todas as etapas, desde o processamento dos dados à apresentação dos resultados. Os demais pesquisadores auxiliaram o aluno na elaboração da metodologia da pesquisa e na escrita do presente trabalho.

Referências

Bergstra, J. and Bengio, Y. (2012). Random search for hyper-parameter optimization. *The Journal of Machine Learning Research*, 13(1):281–305.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. *CoRR*, abs/1603.02754.

Freund, Y. and Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119 – 139.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232.

Hastie, T., Tibshirani, R., Friedman, J., and Franklin, J. (2005). The elements of statistical learning: data mining, inference and prediction. *The Mathematical Intelligencer*, 27(2):83–85.

Kennedy, J. (2011). Particle swarm optimization. In *Encyclopedia of machine learning*, pages 760–766. Springer.

Krishnaiah, V., Narsimha, G., and Chandra, N. S. (2014). Survey of classification techniques in data mining. *International Journal of Computer Science and Engineering*, 2.

Lafferty, G. L. J. (2002). Boosting and maximum likelihood for exponential models. *Advances in neural information processing systems*, 14:447.

Lee, S. J., Liu, Y.-E., and Popovic, Z. (2014). Learning individual behavior in an educational game: A data-driven approach. In *Educational Data Mining 2014*.

Refaeilzadeh, P., Tang, L., and Liu, H. (2009). Cross-validation. In *Encyclopedia of database systems*, pages 532–538. Springer.

Satyanarayana, N., Ramalingaswamy, C., and Ramadevi, Y. (2014). Survey of classification techniques in data mining. *International Journal of Innovative Science, Engineering & Technology*, 1.

Segal, A., Katzir, Z., Gal, K., Shani, G., and Shapira, B. (2014). Edurank: A collaborative filtering approach to personalization in e-learning. In *Educational Data Mining 2014*.

Stearns, B., Rangel, F., Rangel, F., Firmino, F., and Oliveira, J. (2017). Scholar performance prediction using boosted regression trees techniques. In *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)*. Citeseer.

Uragun, B. and Rajan, R. (2011). Developing an appropriate data normalization method. In *Machine Learning and Applications and Workshops (ICMLA), 2011 10th International Conference on*, volume 2, pages 195–199. IEEE.

Vahdat, M., Ghio, A., Oneto, L., Anguita, D., Funk, M., and Rauterberg, M. (2015). Advances in learning analytics and educational data mining. *Proc. of ESANN2015*, pages 297–306.

Woźniak, M., Graña, M., and Corchado, E. (2014). A survey of multiple classifier systems as hybrid systems. *Information Fusion*, 16:3–17.