Análise de parâmetros na construção de florestas randômicas

André Filipe Caldas Laranjeira
Curso de Engenharia da computação
Universidade de Brasília
Matrícula: 16/0023777
Email: andrecaldaslaranjeira@gmail.com

Victor André Gris Costa
Curso de Engenharia da computação
Universidade de Brasília
Matrícula: 16/0019311
Email: victorandr98@gmail.com

I. Introdução

O problema a ser resolvido consiste na escolha de parâmetros para que um classificador implementado com florestas randômicas possa obter a melhor taxa de acerto ao aprender a classificar uma base de dados com 340 exemplos pertencentes a 30 classes e com 14 características cada um. Entre os parâmetros configurados com o intuito de atingir a melhor taxa de acerto se encontram a poda de árvore aplicada, a função de ganho utilizada, o número de características analisadas por nó, entre outros.

Para resolvermos este problema, utilizamos um programa escrito com a linguagem de programação Python para treinar várias configurações de florestas randômicas diferentes e comparar suas respectivas taxas de acerto. Para mensurar essa taxa, o conjunto de dados de 340 exemplos foi dividido em um subconjunto de 289 exemplos para treinamento (85% da base de dados) e um subconjunto de 51 exemplos para teste (15% da base de dados). Após essa divisão, foram calculadas a taxa de acerto para o conjunto de teste após cada modelo de floresta randômica ser treinado com o subconjunto de treinamento e a taxa de acerto obtida com a aplicação de validação cruzada com 10 dobras em cada modelo de floresta randômica utilizando-se o subconjunto de treinamento.

Este relatório tem o intuito de descrever melhor a configuração do programa escrito e alguns dos resultados observados. Uma breve descrição teórica acerca de florestas randômicas se encontra na seção *Introdução teórica*. Os detalhes acerca do programa implementado estão descritos na seção *Detalhes do programa*. Os resultados obtidos foram analisados com base na teoria de florestas randômicas na seção *Análise de resultados*. Por fim, uma conclusão geral acerca de quão bem o problema original foi resolvido foi feita na seção *Conclusão*.

II. Introdução teórica

Uma floresta randômica consiste em um conjunto de árvores de decisão montadas sobre conjuntos bootstrapped dos dados de treinamento. Diferente da técnica de bagging, a floresta randômica sempre escolhe a característica utilizada em um nó de decisão analisando um subconjunto de tamanho fixo de características selecionadas aleatoriamente especificamente para aquele nó de decisão de uma árvore de decisão. A previsão de classe para um exemplo é feita pela média das previsões realizadas em cada árvore pertencente à floresta randômica. [1]

A utilização de árvores de decisão montadas com conjuntos bootstrap e com seleção aleatória de características a serem consideradas a cada nó de decisão contribui para diminuir a variância dos resultados da floresta randômica e, assim, melhorar sua taxa de acerto. Ainda assim, como o modelo de árvore de decisão é um modelo propenso a sobreajuste e nossa base de dados não possui muitos exemplos, acreditamos que o modelo de florestas randômicas encontrará certas dificuldades no aprendizado da nossa base de dados. [1]

III. Detalhes do programa

Como dito na introdução, escrevemos um programa na linguagem de programação Python para testar o desempenho de várias configurações de florestas randômicas na classificação do conjunto de dados fornecidos. A métrica utilizada para essa avaliação foi a taxa de acerto do conjunto de teste e da validação de 10 dobras do conjunto de treinamento. Esta seção busca detalhar melhor a estrutura e o funcionamento desse programa.

A. Especificações técnicas do programa

- Linguagem de programação utilizada: Python 3.7.3
- Pacotes utilizados:
  - numpy (1.15.4)
  - opencv-python (4.0.0.21)
  - pandas (0.24.2)
  - scikit-learn (0.20.3)
- Tempo médio de execução do programa: 20 minutos
- Repositório do programa: https://github.com/AndreLaranjeira/Machine_learning-Random_forests

B. Carregamento de dados

O arquivo de dados com a classe e as características de cada exemplo foi baixado do link https://archive.ics.uci.edu/ml/machine-learning-databases/00288/. Como o arquivo de dados baixado já estava no formato csv e os exemplos nele estavam devidamente organizados, utilizamos uma função do pacote *pandas* para extrair os dados do arquivo csv para uma lista da linguagem Python. Então separamos as colunas de características da coluna com o rôtulo de classe para cada

item na lista, obtendo uma lista com as características de cada exemplo e uma lista com os rôtulos de cada exemplo. Por fim, utilizamos uma função do pacote *scikit-learn* para separar essas listas em um conjunto de treinamento e em um conjunto de dados.

C. Parâmetros analisados

Os parâmetros analisados por nosso programa na configuração de florestas randômicas foram a poda das árvores, a função de ganho e a quantidade de características analisadas a cada nó de decisão da árvore.

1) Poda das árvores: A pode de uma árvore de decisão envolve a utilização de algum critério para parar a construção da árvore antecipadamente ou remover galhos posteriormente a construção da árvore. O intuito da poda é tentar diminuir o fator de sobreajuste presente inerente a árvores de decisão ao forçar o modelo a ignorar parte da informação fornecida pelo conjunto de treinamento. [1]

Dentre as formas de poda analisadas em nosso programa se encontram limitar a altura da árvore, limitar a quantidade de nós folha na árvore, impor um limite mínimo de exemplos para que a árvore crie um nó de decisão e impor um limite mínimo de exemplos para que árvore forme um nó folha.

2) Função de ganho: A função de ganho é a função utilizada para definir qual característica do conjunto de dados analisado deve ser empregada em um nó de decisão da árvore para fornecer o maior ganho de pureza aos conjuntos de dados de cada nó. A pureza de um conjunto de dados mede quão bem os dados desse conjunto pertencem à mesma classe, de forma que quanto maior o ganho de pureza por nó de decisão, mais eficiente uma árvore de decisão é em separar os dados pertencentes a diferentes classes. [1]

Dentre as funções de dados analisadas em nosso programa se encontram o índice de gini e a função de entropia.

3) Quantidade de características analisadas a cada nó de decisão: Como dito na *Introdução teórica*, a floresta randômica sempre analisa um subconjunto de características aleatório ao decidir qual características utilizar em um nó de decisão. Essa prática tem o intuito de permitir que características menos dominantes também possam gerar contribuições significativas na classificação dos exemplos. [1]

Em nosso programa, testamos a quantidade de características analisadas como sendo a raiz quadrada da quantidade total de características, o logarítmo na base 2 da quantidade total de características e todas as características (essa técnica é conhecida como bagging, e não mais como florestas randômicas).

D. Formato dos testes

Por não ser possível fazer todas as combinações possíveis, decidimos fazer 5 gráficos com a evolução da acurácia relativa aos valores do parâmetro. Cada gráfico conteve 6 curvas, 1 para cada combinação de função de ganho (gini ou entropia) com quantidade de *features* escolhidas por nó de decisão (raiz quadrada, logarítimo base 2 ou todas as *features*). Cada gráfico foi criado utilizando 5 a 20 amostras de estimadores de florestas randômicas com os parâmetros crescendo. Anotou-se para todos os testes a taxa de erro de teste e a taxa de erro da aplicação de validação cruzada 10 dobras sobre o conjunto de treinamento.

IV. Análise de resultados

A. Sem a utilização de poda com variação do número de árvores

Com o intuito de se criar um parâmetro de comparação para os tipos de corte utilizados, testou-se a criação de florestas randômicas sem nenhum parâmetro de corte e variando-se o número de árvores de decisão treinadas. Os resultados obtidos foram registrados na figura 1.

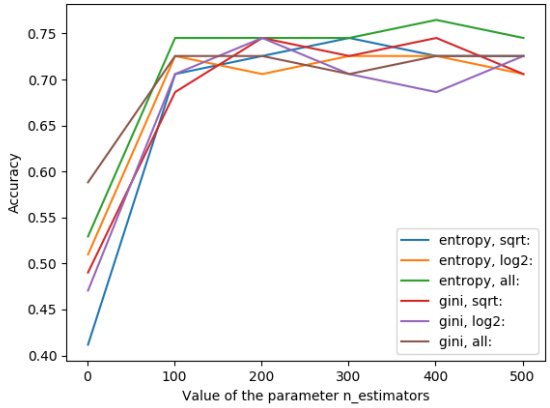

Figura 1. Resultados obtidos para sem a utilização de poda e com variação do número de árvores utilizadas.

1) Resultados observados: Analisando os resultados obtidos, percebemos que o aumento do número de árvores na floresta randômica gerou um aumento na taxa de acerto de todos os modelos até que o número de árvores atingiu 100. A partir desse ponto, alguns modelos apresentaram melhoras, estagnação ou piora conforme o número de árvores aumentou.

Além disso, percebemos, analisando a imagem 1, que o desempenho das florestas randômicas sem nenhuma poda variou de forma relativamente volátil para algumas categorias com a variação do número de árvores utilizadas, indicando, possivelmente, que uma maior quantidade de exemplos de treinamento e teste são necessários para tornar o modelo robusto a variações individuais dos exemplos.

Nesta categoria, a floresta randômica com 401 árvores que utiliza a entropia como função de ganho e não limita o número de parâmetros analisados pelos nós de decisão (bagging) obteve o melhor resultado com uma taxa de acerto de 76.4706% para os exemplos de teste e uma taxa de acerto de 73.7069% com desvio padrão de (6.91%).

B. Poda por altura

O primeiro treinamento com poda consiste em uma poda por altura. A poda por altura consiste em terminar a expensão de cada árvore na floresta randômica até um certo nível de altura. Os resultados obtidos foram registrados na figura 2.

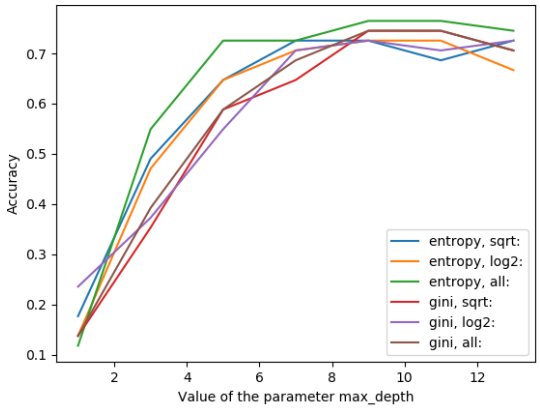

Figura 2. Resultados obtidos para poda por altura.

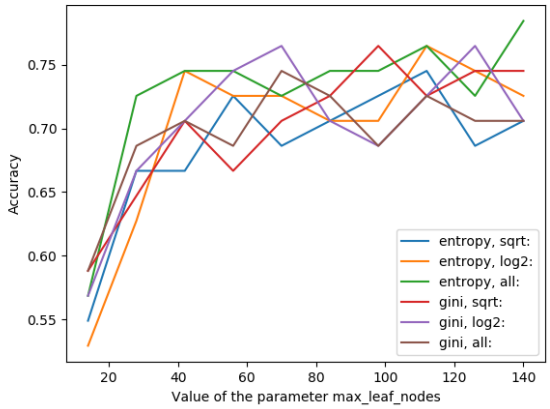

Figura 3. Resultados obtidos para poda por quantidade total de nós folha.

1) Resultados observados: Analisando os resultados obtidos na figura 2, percebemos que uma poda de altura relativamente pequena (cortar poucos níveis da árvore) gera um ganho na taxa de acerto da maioria dos modelos treinados, enquanto uma poda de altura grande (cortar muitos níveis da árvore) diminui rapidamente a taxa de acerto do modelo. Isso é consistente com a teoria estudada, pois uma pequena poda cumpre o papel de generalizar mais as árvores da floresta randômica, retirando nós formados por sobreajuste, enquanto uma grando pode passa a retirar informações gerais do modelo, diminuindo a taxa de acerto. [1]

É interessante constatar que, mesmo em um modelo de floresta randômica gerado com pouquíssimos exemplos de treinamento, a utlização de uma poda de altura ainda foi uma ferramenta útil para melhorar a taxa de acerto.

Nesta categoria, a floresta randômica com uma poda de altura que limita às árvores de decisão a 9 níveis de nós, que utiliza a entropia como função de ganho e não limita o número de parâmetros analisados pelos nós de decisão (bagging) obteve o melhor resultado com uma taxa de acerto de 76.4706% para os exemplos de teste e uma taxa de acerto de 72.3030% com desvio padrão de (8.3571%), gerando com apenas 100 árvores resultados compatíveis com a utilização de 401 árvores de decisão.

C. Poda por quantidade total de nós folha

O segundo treinamento com poda consiste em uma poda por quantidade total de nós folha. A poda por quantidade total de nós folha consiste na geração de árvores com um número limitado de nós folha, sendo que os melhores nós folha (em termos de redução de impureza relativa) são gerados primeiro. Os resultados obtidos foram registrados na figura 3.

1) Resultados observados: Analisando os resultados obtidos na figura 3, percebemos, primeiramente, que os dados foram muito ruidosos, indicando, mais uma vez, o impacto causado pela baixa quantidade de exemplos utilizada na construção de florestas randômicas e dificultando a retirada de conclusões.

Apesar disso, podemos constatar que a utilização de uma poda pequena por quantidade total de nós folha (uma grande quantidade total de nós folha) foi benéfica para a maioria dos modelos de florestas randômicas, gerando modelos com uma menor taxa de sobreajuste, mas que a maioria dos modelos apresentava uma rápida queda na taxa de acerto para podas maiores (uma pequena quantidade total de nós folha), pois informações inerentes ao modelo passaram a ser descartadas. Percebemos que, diferente da poda por altura, a poda por quantidade total de nós folha é mais sensível à quantidade de exemplos presentes na base de dados e menos previsível. [1]

Nesta categoria, a floresta randômica com uma poda de quantidade total de nós folha que limita às árvores de decisão a 140 nós folha, que utiliza a entropia como função de ganho e não limita o número de parâmetros analisados pelos nós de decisão (bagging) obteve o melhor resultado com uma taxa de acerto de 78.4314% para os exemplos de teste e uma taxa de acerto de 72.6724% com desvio padrão de (8.1969%), gerando com apenas 100 árvores resultados que superaram a utilização de 401 árvores de decisão.

D. Poda por quantidade mínima de exemplos para um nó de decisão

O terceiro treinamento com poda consiste em uma poda por quantidade mínima de exemplos em nó de decisão. A poda por quantidade mínima de exemplos para um nó de decisão consiste na geração de árvores com um limite mínimo de exemplos para ocorrer alguma decisão, por exemplo se o mínimo for 4 uma decisão só será adicionada se ainda houver 4 exemplos nele, dividindo em 3 para um lado e 1 para o outro ou 2 para cada. Os resultados obtidos foram registrados na figura 4.

1) Resultados observados: Analisando os resultados obtidos na figura 4, percebemos, primeiramente, que os dados foram muito ruidosos, indicando, mais uma vez, o impacto causado pela baixa quantidade de exemplos utilizada na cons-

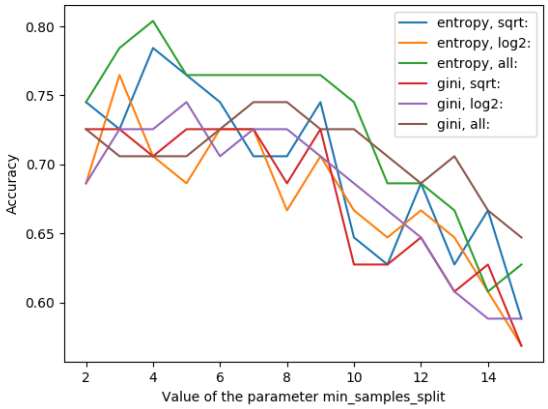

Figura 4. Resultados obtidos para poda por quantidade mínima de exemplos para um nó de decisão.

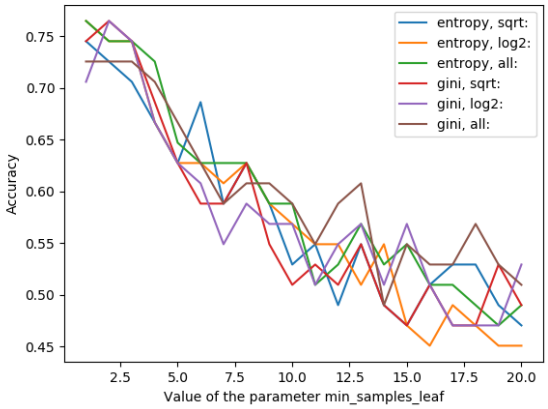

Figura 5. Resultados obtidos para poda por quantidade mínima de exemplos para um nó folha.

trução de florestas randômicas e dificultando a retirada de conclusões.

Apesar disso, podemos constatar que a utilização de uma poda pequena, mas não muito, por quantidade mínima de exemplos por nó de decisão (mínimo de 3 a 5) foi benéfica para a maioria dos modelos de florestas randômicas, gerando modelos com uma menor taxa de sobreajuste, mas que a maioria dos modelos apresentava uma rápida queda na taxa de acerto para podas maiores (mínimo maior), pois informações inerentes ao modelo passaram a ser descartadas.

Nesta categoria, a floresta randômica com uma poda de mínimo de exemplos por nós de decisão que limita as árvores de decisão a no mínimo 4 exemplos por nó de decisão, que utiliza a entropia como função de ganho e não limita o número de parâmetros analisados pelos nós de decisão (bagging) obteve o melhor resultado com uma taxa de acerto de 80.3922% para os exemplos de teste e uma taxa de acerto de 72.6478% com desvio padrão de (5.9427%), gerando com apenas 100 árvores resultados que superaram a utilização de 401 árvores de decisão.

E. Poda por quantidade mínima de exemplos para um nó folha

O quarto treinamento com poda consiste em uma poda por quantidade mínima de exemplos em nó folha. A poda por quantidade mínima de exemplos para um nó folha consiste na geração de árvores com um limite mínimo de exemplos em uma folha, caso uma divisão gere folhas com menos exemplos que o mínimo, a divisão não será feita. Os resultados obtidos foram registrados na figura 5.

1) Resultados observados: Analisando os resultados obtidos na figura 5, percebemos, primeiramente, que os dados foram muito ruidosos para podas maiores, indicando, mais uma vez, o impacto causado pela baixa quantidade de exemplos utilizada na construção de florestas randômicas e dificultando a retirada de conclusões.

Apesar disso, podemos constatar que a utilização de uma poda pequena ou nenhuma poda por quantidade mínima de exemplos por nó folha (mínimo de 1 a 3) foi benéfica para a maioria dos modelos de florestas randômicas, gerando modelos com uma menor taxa de sobreajuste, mas que a maioria dos modelos apresentava uma rápida queda na taxa de acerto para podas maiores (mínimo maior), pois informações inerentes ao modelo passaram a ser descartadas.

Nesta categoria, as florestas randômicas com uma poda de mínimo de exemplos por nós folha que limita as árvores de decisão a no mínimo 2 exemplos por nó folha, que utilizam a entropia como função de ganho e limitam o número de parâmetros analisados pelos nós de decisão a raiz quadrada do total ou logaritmo base 2 do total (bagging) obtiveram os melhores resultados com uma taxa de acerto de 76.4706% para os exemplos de teste. A taxa de acerto para raiz quadrada é 71.9704% com desvio padrão de (4.9871%). A taxa de acerto para logaritmo base 2 é 73.7069% com desvio padrão de (5.3594%). Ambas geraram com apenas 100 árvores resultados que superaram a utilização de 401 árvores de decisão.

F. Função de ganho

Ao longo dos teste realizados anteriormente para funções de poda, foram testados modelos que utilizavam a função de ganho de entropia e modelos que utilizavam a função de ganho do índice de gini. Essa seção sumariza os resultados observados agregando-se os testes realizados e comparando as funções de ganho utilizadas.

1) Resultados observados: Em geral, os modelos que utilizaram a entropia como função de ganho apresentaram melhores resultados que aqueles que utilizaram o índice de Gini como função de ganho. De acordo com a teoria estudada, essas funções de ganho deveriam fornecer valores parecidos numericamente, [1] mas há uma considerável diferença entre as duas funções de ganho dependendo da situação.

Acredita-se que a função de ganho de entropia seja mais adequada para modelos com uma baixa quantidade de informações, de forma que a diminuta natureza da base de dados aprendida beneficie a função de ganho de entropia.

G. Quantidade de características analisadas a cada nó de decisão

Ao longo dos teste realizados anteriormente para funções de poda, foram testados modelos que limitavam a quantidade de características analisadas em cada nó e modelos que não a limitavam. Essa seção sumariza os resultados observados agregando-se os testes realizados e comparando as quantidades utilizadas.

1) Resultados observados: Em geral, os modelos que utilizaram a entropia e não limitaram as características analisadas apresentaram melhores resultados que os outros. Não houve um claro segundo ganhador.

Acredita-se que quando há menos informações é melhor usar todas as características possíveis. Nota-se isso ao observar que limitar gerava resultados piores que usar tudo.

V. Conclusão

Nas análise realizadas com tipos de poda, funções de ganho e quantidade de características utilizadas por nó de decisão, observamos variações notáveis na taxa de ganho, indicando que as características alteradas influenciavam a taxa de acerto do modelo. Apesar disso, houve certo ruído nos dados coletados, indicando a dificuldade de se trabalhar com uma base de dados pequena, em especial para modelos que possuem uma tendência a sobreajuste, como árvores de decisões. Acreditamos que a utilização de uma base de dados mais robusta nos permitiria visualizar com mais clareza o impacto que cada mudança de característica exerceu na taxa de acertos do modelo aprendido, gerando, como consequência, melhores conclusões.

Em nossos testes, nossos melhores resultados orbitaram em torno de 76% e não ultrapassaram a marca dos 81%, indicando que os modelos gerados com florestas randômicas são úteis, mas ainda podem melhorar. Mais uma vez, acreditamos que isso seja consequência do tamanho da base de dados utilizada e da natureza do modelo de florestas randômicas. A utilização de uma base de dados maior ou mesmo de outra forma de aprendizado de máquina mais adequada para bases de dados pequenas talvez possibilitassem a obtenção de uma melhor taxa de acerto.

Entre as características analisadas, foi constatado que, para a base de dados utilizada, a função de ganho de entropia, a utilização de uma pequena poda e a análise de todas as características a cada nó de decisão forneceram os melhores resultados. A melhor taxa de acerto obtida entre todos os modelos foi o que utilizou todas as características nos nós, com a função de ganho de entropia e fez a poda por mínimo de exemplos por nó de decisão. A taxa de acerto para os exemplos de teste foi de 80.3922% e a taxa de acerto foi de 72.6478% com desvio padrão de (5.9427%).

Referências

[1] G. James, D. Witten, T. Hastie and R. Tibshirani, “An introduction to statistical learning with applications in R”, Springer, 7th printing, pp.303–323, 2017