UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Câmpus de Presidente Prudente

LEONARDO EVANGELISTA DE ABREU

PEOPLE ANALYTICS: USO DE ÁRVORES DE DECISÃO NA RETENÇÃO DE TALENTOS

PRESIDENTE PRUDENTE

2022

LEONARDO EVANGELISTA DE ABREU

PEOPLE ANALYTICS: USO DE ÁRVORES DE DECISÃO NA RETENÇÃO DE TALENTOS

Relatório Final para Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Estatística da FCT/Unesp para aproveitamento na disciplina Trabalho de Conclusão de Curso.

Orientador: Prof. Dr. Klaus Schlunzen Junior

PRESIDENTE PRUDENTE

2022

A162p

Abreu, Leonardo Evangelista

People Analytics: uso de Árvores de Decisão na retenção de talentos / Leonardo Evangelista Abreu. -- Presidente Prudente, 2022

38 f.

Trabalho de conclusão de curso (Bacharelado - Estatística) - Universidade Estadual Paulista (Unesp), Faculdade de Ciências e Tecnologia, Presidente Prudente

Orientador: Klaus Schlunzen Junior

1. Árvores de Decisão. 2. Retenção de Talentos. 3. Gestão de Pessoas. 4. Recursos Humanos. 5. People Analytics. I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Biblioteca da Faculdade de Ciências e Tecnologia, Presidente Prudente. Dados fornecidos pelo autor(a).

Essa ficha não pode ser modificada.

TERMO DE APROVAÇÃO

Leonardo Evangelista de Abreu

PEOPLE ANALYTICS: USO DE ÁRVORES DE DECISÃO NA RETENÇÃO DE TALENTOS

Relatório de Final de Trabalho de Conclusão de Curso aprovado como requisito para obtenção de créditos na disciplina Trabalho de Conclusão do curso de graduação em Estatística da Faculdade de Ciências e Tecnologia da Unesp, pela seguinte banca examinadora:

Orientador: _____

Prof. Dr. Klaus Schlunzen Junior
Departamento de Estatística

_____

Profa. Dra. Miriam Rodrigues Silvestre

Departamento de Estatística

_____

Prof. Dr. José Gilberto Spasiani Rinaldi

Departamento de Estatística

Presidente Prudente, __ de Fevereiro de 2022.

RESUMO

Este trabalho foi desenvolvido com uma base de dados disponível pelo Kaggle e tem como objetivo identificar, traçar o perfil e prever, por meio do uso de árvores de decisão, quais pessoas têm tendência a se desligar de uma empresa a partir de suas características pessoais e profissionais. A ideia do projeto consiste em mostrar que o uso de ferramentas estatísticas na área de Gestão de Pessoas pode agregar muito valor de forma prática e simples, ainda mais nessa área que a análise de dados não é muito frequente. Embora a base de dados seja desbalanceada e possui muitos valores faltantes, para fins práticos desconsiderou-se as observações que possuíam pelo menos um dado faltante e não foi usado nenhuma técnica para equilibrar as classes de interesse. Para lidar com as variáveis categóricas foram criadas variáveis dummy pois o sckit-learn, biblioteca usada para modelagem no Python, não suporta dados categóricos. Foram feitos dois modelos, o primeiro cresceu exaustivamente, ficando muito específico com problemas de sobreajuste, enquanto o segundo foi podado usando o critério de custo de complexidade. Através de métricas estatísticas, tais como Recall, Precisão e F1-Score, checou-se a performance dos modelos avaliando se possuem boa capacidade discriminante. Embora alguns dados sobre os colaboradores possam ser difíceis de serem obtidos, as árvores de decisão demonstram serem bem interessantes para auxiliar na retenção de talentos.

Palavras-chave: Árvores de Decisão; Retenção de Talentos; Recursos Humanos; Gestão de Pessoas.

ABSTRACT

This work was developed with a database available by Kaggle and aims to identify, profile and predict, through the use of decision trees, which people tend to leave a company based on their personal characteristics and professionals. The idea of the project is to show that the use of statistical tools in the area of People Management can add a lot of value in a practical and simple way, especially in this area where data analysis is not very frequent. Although the database is unbalanced and has many missing values, for practical purposes, observations that had at least one missing data were disregarded and no technique was used to balance the classes of interest. To deal with category variables, dummy variables were created because sckit-learn, the library used for modeling in Python, does not support categorical data. Two models were made, the first grew exhaustively, becoming very specific with overfitting problems, while the second was pruned using the complexity cost criterion. Through statistical metrics, such as Recall, Precision and F1-Score, the performance of the models was checked, evaluating whether they have good discriminating capacity. Although some data about employees can be difficult to obtain, decision trees prove to be very interesting to help retain talent.

Keywords: Decision Trees; Retaining talent; Human Resources; People management.

Lista de Tabelas

Lista de Figuras

Sumário

1 Introdução

O capital humano de uma empresa é o principal e mais valioso ativo que ela pode ter, uma vez que funcionários habilidosos e competentes terão a capacidade de inovar e gerar mais valor para sua organização, fazendo-a atingir seus objetivos (CHIAVENATO, 2008). No entanto, manter seus colaboradores produtivos a todo momento não é uma tarefa tão fácil. É preciso mantê-los motivados, e como a motivação vem do interior de cada um, é necessário que a organização tenha práticas de incentivo para que assim o colaborador se sinta inspirado a dar o seu melhor. Como exemplos dessas práticas tem-se bonificações, treinamentos, perspectivas de carreira e alinhamento entre interesses pessoais e profissionais. Para que isso aconteça, é fundamental a atuação do departamento de gestão de pessoas, que age estrategicamente garantindo o sucesso do negócio.

Fazer gestão de pessoas é ser capaz de alinhar os objetivos do colaborador com a empresa (FISCHER, 2002) e buscar o constante desenvolvimento, engajamento e desempenho do membro, possibilitando o crescimento e contribuição do mesmo frente a organização. Com o intuito de ajudar o departamento de gestão de pessoas a cumprir com todas essas funções, surge, então, a necessidade da coleta, organização e análise de dados dos seus colaboradores, conhecido como método de gestão *People Analytics*.

People Analytics é um método de gestão orientado a dados para que, com boas análises, auxilie em tomadas de decisões e se tenha uma visão mais estratégica da função do colaborador. Sua aplicação é diversa: muito usada em recrutamento e seleção, ajudando a definir as reais necessidades da companhia; no traçar de perfis adequados ao cargo e valores da empresa e saber quais candidatos se assemelham a esses perfis; no desempenho do time, entendendo as motivações do colaborador e agir para promover ações que o mantenha engajado; na retenção de talentos, prevendo quais potenciais candidatos/colaboradores permanecerão ou abandonarão a empresa; e entre outros tipos de aplicação.

Ao contratar uma pessoa espera-se, normalmente, que os resultados venham ao longo do tempo, pois, por mais habilidosa que seja, ainda não se sabe muito sobre o negócio e é preciso gastar recursos, como tempo e dinheiro, para formá-la e desenvolvê-la. Uma saída precoce certamente é prejudicial a empresa já que a possibilidade de retorno sobre o investimento é menor devido a sua permanência de curto período. Um colaborador mais experiente saberá lidar melhor com os problemas, trazer soluções mais aplicáveis e agir com mais independência. Portanto, uma contratação é sempre um investimento de longo prazo e por isso é muito importante que a organização evite a saída de bons colaboradores e entrada daqueles com baixa adesão a sua cultura e valores.

Seguindo esse contexto, o objetivo desse projeto é mostrar um caso do uso de Árvores de Decisão na retenção de talentos para identificar e traçar o perfil e prever quais pessoas possivelmente sairão da empresa a partir de suas características pessoais e profissionais, ajudando a área de gestão de pessoas a tomar ações antecipadas para evitar que o colaborador se desligue.

2 Metodologia

O estudo e aplicação de Árvores de Decisão foi feito em uma base de dados usada para fins de aprendizagem e prática preditiva na área de Gestão de Pessoas fornecida por Janata Hack (JANATAHACK, 2020) e disponibilizada na plataforma de *Data Science Kaggle* (KAGGLE, 2020) por Möbius (MÖBIUS, 2020).

Os dados, como cidade, gênero, grau de instrução, tempo de experiência, entre outras variáveis, foram projetados para entender os motivos que levam uma pessoa a trocar de emprego e ajudar a área de Gestão de Pessoas/Recursos Humanos a tomar decisões.

O banco de dados contém mais de 19 mil observações, com maioria de variáveis categóricas (nominais, ordinais, binárias), é composto por informações demográficas, educacionais e de experiências dos candidatos que se inscreveram em cursos ministrados por uma empresa de *Big Data* e *Data Science.*

Apesar de serem dados de pessoas que se inscreveram no curso da empresa e não os dados dos seus próprios colaboradores, ela poderá usá-los para entender os motivos que levam seus membros a se desligar e procurar por um novo emprego.

Então, primeiro é necessário dividir o banco de dados em treino e teste e depois construir o modelo de Árvores de Decisão no conjunto de treinamento, sendo possível verificar sua capacidade de classificação e previsão no conjunto de teste.

A partir da seleção dos atributos com maior capacidade discriminante que compõe as regras do modelo de Árvore de Decisão, seria possível classificar e observar o trajeto de perfil de cada candidato e calcular as probabilidades de evasão. Dessa forma, a empresa saberia lidar melhor com seus colaboradores traçando planos para agir antes que eles se desligassem.

A manipulação dos dados e as análises e modelagem estatística foram feitas por meio das linguagens de programação Python, utilizando principalmente os pacotes Pandas, para manipulação do banco de dados,

NumPy, para operações matemáticas com vetores e matrizes e Scikit-Learn para a parte de modelagem com Árvores de Decisão.

2.1 Árvores de Decisão

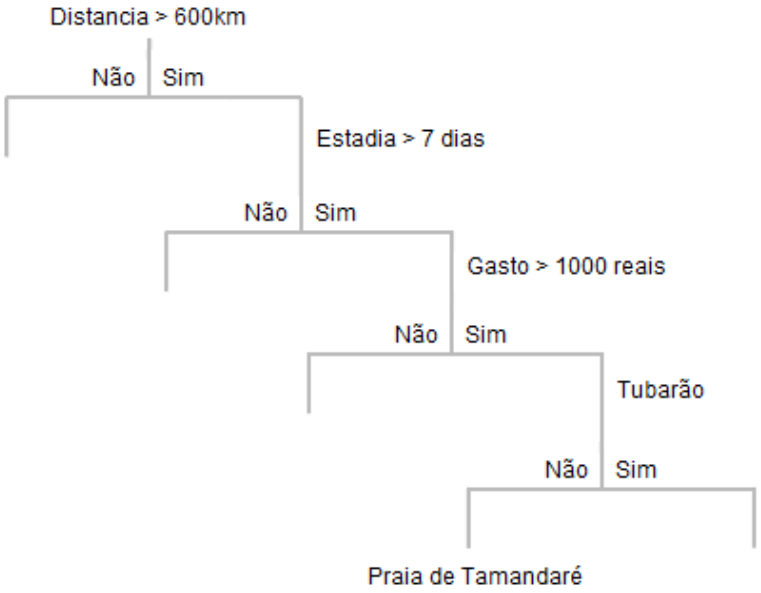
A primeira aparição do conceito de árvores de decisão foi no livro *Machine Learning* em 1975 por J. Ross Quinlan. No entanto, o progresso dessa ideia só aconteceu em 1986 quando Quinlan desenvolveu o primeiro algoritmo de árvores de decisão chamado ID3 (*Iterative Dichotomiser 3)* com a finalidade de determinar quais campos de informação eram importantes e quais deles se relacionavam com o problema (SIMÕES, 2008). Posteriormente, esse algoritmo foi evoluindo e outros também foram desenvolvidos como o C4.5, C5.0, CART (*Classification and Regression Trees*), entre outros.

Uma árvore de decisão tem esse nome devido ao seu formato visual ser parecido com o de uma árvore e que, de forma bem intuitiva e de fácil interpretação, ilustra uma espécie de trajeto a ser percorrido até a tomada de decisão. Por isso é tão comum ver sua empregabilidade em diversos campos, podendo ser aplicado desde decisões no cotidiano, até em problemas mais complexos em diferentes áreas de estudo. A própria terminologia das árvores de decisão foi adotada de forma análoga a terminologia de uma árvore. A árvore de decisão é sempre lida descendentemente, com o atributo mais importante contido no primeiro nó, chamado de nó raiz. Os demais atributos são nós subsequentes, nomeados como subnós ou nós internos, e os nós terminais, que apresentam a decisão a ser tomada, são chamados de folhas. A poda é uma técnica que reduz o tamanho da árvore removendo os subnós menos importantes, deixando-a menos complexa e evitando problemas de *overfitting* ou sobreajuste.

Suponha que você está pensando em ir para a praia com uma turma de amigos, mas não sabem em qual ir. Para isso, vocês recorrem a árvore de decisão para ter uma melhor ideia. Contida no nó raiz, a primeira pergunta é: Querem uma praia a mais de 600km? Sim, então segue para o nó filho a direita. Há disponibilidade de estadia para mais de 7 dias? Sim! Logo, o próximo é: Cada integrante da turma pode desembolsar mais de 1000 reais? O pessoal está bem

de dinheiro, a resposta é sim. Querem poder entrar na água do mar sem risco de tubarão? Sim! Desse modo, chegaram ao nó folha Praia de Tamandaré, como ilustra a Figura 1.

Figura 1 – Exemplo de uso de Árvore de Decisão para escolher qual praia ir



Fonte: Elaborado pelo autor

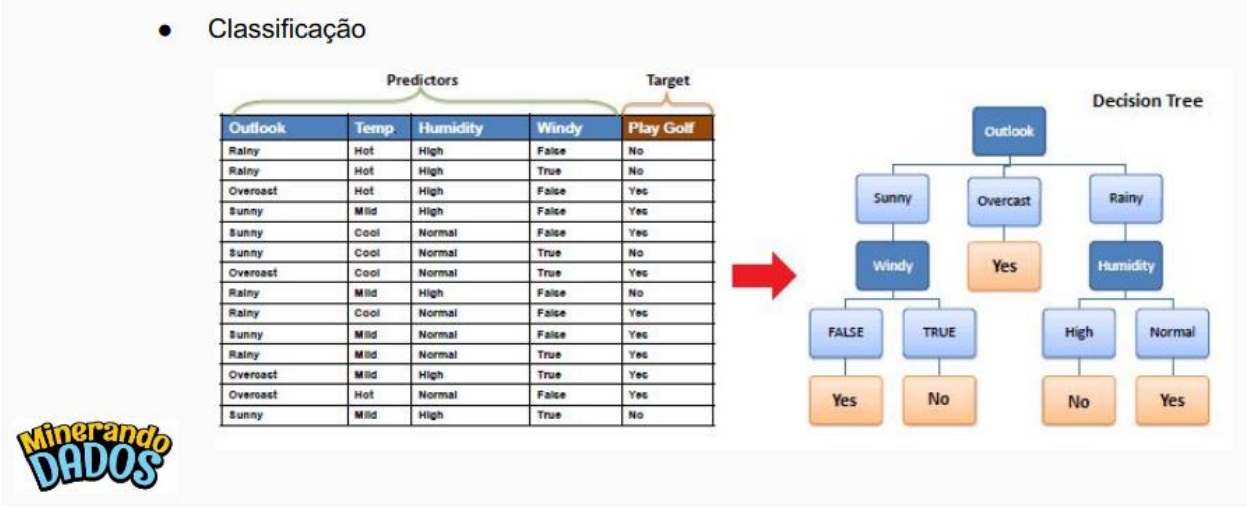
A representação gráfica de uma árvore de decisão, de forma computacional e matemática a partir de dados, é obtida por meio de algoritmos de árvore de aprendizado de máquina supervisionados não-paramétricos, que podem ser aplicados tanto na resolução de problemas de classificação, quanto nos de regressão, ou seja, os algoritmos de árvore de decisão trata-se de predizer o valor de um atributo de interesse categórico ou numérico (variável resposta, alvo ou *target*, por isso modelo supervisionado) a partir de um outro conjunto de atributos, sem antes ter a necessidade de assumir algo sobre os dados (não-paramétrico). Por isso, a ideia por trás desses algoritmos se baseia em encontrar o atributo unido a uma regra (nó), que melhor divide os dados em grupos homogêneos, resultando em subconjuntos com maior pureza.

Para chegar no formato da árvore de decisão, inicialmente é necessário aplicar o algoritmo numa amostra de treinamento, onde é construído o modelo

e, por sua vez, as regras que melhor discriminam os atributos. Dessa forma, para verificar seu desempenho, aplica-se o modelo construído em uma nova amostra com dados diferentes dos que foram usados na de treinamento, chamada de base de teste, onde é possível avaliar o modelo pelas métricas de performance, e se há possíveis ajustes a serem feitos, como por exemplo, árvores muito extensas em que é necessário fazer a poda para evitar *overfitting*. No fim, se estiver tudo certo com o modelo, ele poderá ser aplicado em novos casos em que o resultado (variável resposta) é desconhecido.

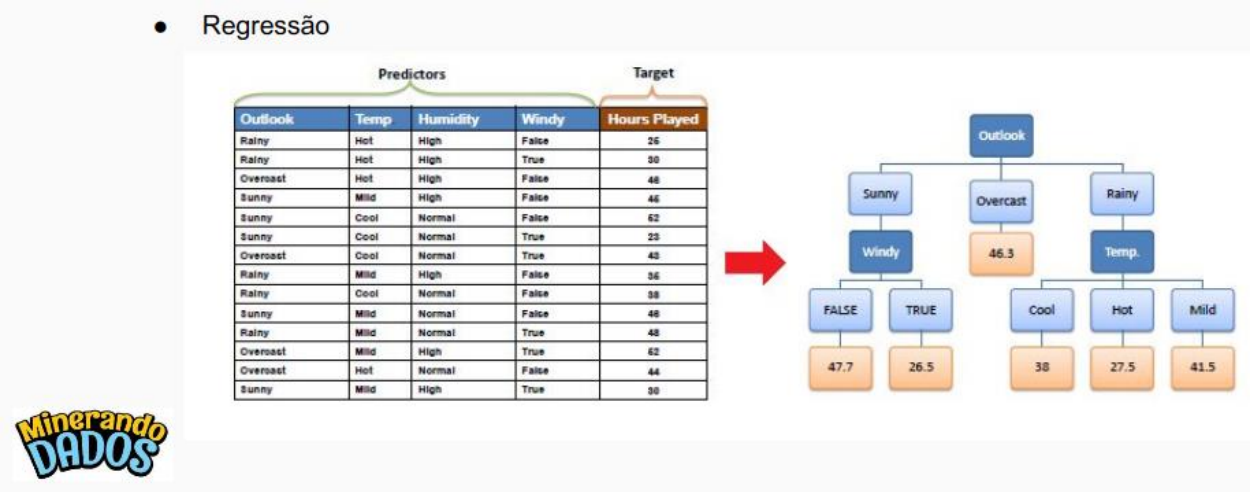
As árvores de decisão podem ser usadas para auxiliar em problemas de classificação, que são para prever a categoria de uma observação, ou regressão, que são para situações que se quer estimar um valor numérico, conforme os exemplos das Figuras 2 e 3. Como o objetivo desse estudo busca analisar e entender o perfil de pessoas que procuram por um novo emprego, trata-se de um problema de classificação.

Figura 2 – Exemplo de árvore de classificação para jogar golfe



Fonte: SANTANA (mineirandodados.com.br, 2020)

Figura 3 – Exemplo de árvore de regressão para estimar horas jogadas



Fonte: SANTANA (mineirandodados.com.br, 2020)

No entanto, nem todos os algoritmos de árvore de decisão conseguem auxiliar na resolução de ambos os casos, como é o caso do ID3, usado apenas em problemas de classificação. Veja na sequência alguns deles e suas características.

2.1.1 Algoritmos de árvore

Foram desenvolvidos diversos algoritmos de árvore de decisão desde a criação do primeiro em 1986. Nessa seção, são apresentados três tipos diferentes: o pioneiro ID3 (*Iterative Dichotomiser* 3), seu sucessor C4.5, e um dos mais usados atualmente CART (*Classification and Regression Trees*).

2.1.1.1 ID3 (*Iterative Dichotomiser* 3)

O ID3 é um algoritmo de classificação e foi desenvolvido em 1986 por Ross Quinlan aceitando apenas dados categóricos. O algoritmo possui uma estratégia gananciosa na hora de escolher o melhor atributo em cada iteração, selecionando aquele que tem maior ganho de informação ou menor entropia, ou seja, maior pureza. Seu nome é devido ao algoritmo dividir repetidamente os nós

em dois ou mais grupos em cada etapa, criando múltiplos caminhos. Por isso, as árvores construídas a partir do ID3 geralmente são simples e grandes, garantindo o melhor caminho, mas não necessariamente a melhor árvore (QUINLAN, 1986).

Dessa forma, se aplicar o ID3 em problemas que possuem variáveis numéricas, será criado um nó para cada valor, tornando a árvore gigantesca.

2.1.1.2 C4.5

C4.5 é uma evolução do ID3, apresentado pelo mesmo autor, no qual não se tem mais a restrição de trabalhar apenas com dados categóricos. Dessa forma são feitas partições nos atributos contínuos e representados em um conjunto discreto de intervalos. Vale destacar que a partir dele pode-se trabalhar com dados faltantes, os quais não são usados em cálculos de ganho de informação e entropia. Foi introduzido o método de poda, que acontece durante a criação das próprias regras da árvore e após a construção de todo modelo para lidar com problemas de sobreajuste.

2.1.1.2 CART (*Classification and Regression Trees*)

Muito semelhante ao C4.5 e tendo seu nome usado como referência para as árvores de decisão (Árvores de Classificação e Regressão), CART é um algoritmo caracterizado pela divisão recursiva binária, em que cada nó é sempre dividido em exatamente outros dois nós. Isso acontece de forma recursiva em cada novo nó. As árvores binárias são construídas usando a impureza de Gini como métrica para avaliar a divisão de um nó e selecionar os atributos com maior pureza.

Como o objetivo do CART é minimizar a função custo (Índice de Gini, que será descrito mais adiante) em cada nó, a seleção de atributos é feita de forma gananciosa de modo que diferentes pontos de divisão sejam considerados e testados, selecionando a divisão com menor custo.

Diferente dos outros, esse algoritmo consegue trabalhar com problemas de regressão. Nos casos de regressão, o CART não calcula conjuntos de regras,

encontra divisões que minimizam o erro quadrático de predição e a estimativa mostrada na folha é baseada na média ponderada do nó (SINGH e GUPTA, 2014).

Como o CART usa do método recursivo, definir um critério de parada pode ser interessante para que assim a árvore não fique muito grande e específica para o conjunto de treinamento. Desse jeito, provavelmente não terá um bom desempenho no conjunto de teste. Uma forma de fazer isso é definir um número mínimo de observações para se chegar a uma decisão. Ou seja, se a quantidade de observações até a folha for inferior ao valor escolhido, aquela ramificação é descartada. Outra forma é definir qual será o tamanho máximo que a árvore poderá assumir do nó raiz até a folha, conhecido como profundidade máxima.

Outro meio de ajudar no desempenho da árvore é fazendo a poda de alguns ramos após a árvore já ter sido treinada de modo a aumentar o poder de generalização. Quanto mais divisões uma árvore tem, mais complexa fica sua interpretabilidade e dificulta que outros dados (do mesmo problema de negócio) se adequem devido sua tamanha especificidade.

2.1.2 Algoritmo escolhido

O algoritmo usado para a criação das árvores de decisão nesse estudo será o CART, já que um bom pacote para modelagem no Python é o scikit-learn, que usa uma versão otimizada do CART para construção das árvores.

2.1.3 Métricas de seleção de atributos

Para se chegar numa árvore de decisão ótima precisa-se testar todos os caminhos possíveis, e isso computacionalmente pode ser inviável. Dessa forma, os algoritmos procuram fazer escolhas de locais ótimos para encontrar o melhor atributo e a regra com maior pureza.

As medidas de impureza, tal como Entropia, Razão de Ganho, Índice Gini, ajudam a encontrar o melhor ponto de corte para determinado nó, selecionando aquele com maior ganho de informação entre todos os possíveis candidatos.

2.1.3.1 Entropia e Ganho de Informação

Entropia

Nas árvores de decisão, entropia é uma forma de medir a impureza de cada conjunto. É um conceito matemático que mede a aleatoriedade ou incerteza. O nível de entropia varia de 0 a 1, quanto mais próximo de zero, mais puro é o conjunto (menos aleatoriedade), e quanto mais próximo de um, maior aleatoriedade (vários elementos distintos).

Dado um conjunto S de s amostras com m classes distintas C_i (i = 1,..., m), a entropia é dada por:

$$Entropia(S) = -\sum_{i=1}^{m} p_i \, log_2(p_i),$$

onde p_i é a probabilidade de um elemento pertencer a classe C_i que é calculada pela proporção s_i/s.

Ganho de Informação

Depois de calculada a entropia do conjunto, mede-se o ganho de informação para selecionar o melhor atributo para o nó, em que é calculada a diferença entre a entropia antes da divisão e a entropia média após a divisão do conjunto. O Ganho de Informação ou Ganho representa a quantidade de informação esperada ao fazer a divisão. Esse cálculo é feito para cada atributo em cada nó da árvore e aquele com maior ganho de informação (ou menor entropia) é escolhido.

Matematicamente, o ganho de informação é dado por:

$$Ganho(S, A) = Entropia(S) - \sum_{j=1}^{m} \frac{|S_j|}{|S|} Entropia(S_j),$$

onde A é o atributo a ser testado, S_j é a soma de todos os valores do nó, S é o número total de amostras, e Entropia(S_j) é a entropia do nó atual.

Para ilustrar as métricas de seleção de atributos em uma árvore de decisão, o site vooo.pro (2016) usa o exemplo de 30 alunos de uma escola em que se deseja prever quem jogará tênis no intervalo através de três variáveis: Sexo (masculino ou feminino), Classe (IX ou X) e Altura (160cm a 180cm), e que desses 30 alunos, 15 jogam tênis no intervalo.

Figura 4 - Imagem com a distribuição de cada variável do exemplo

Fonte: ADMINVOOO (vooo.pro, 2016)

Para exemplificar os cálculos usou-se duas variáveis: Sexo e Classe. O cálculo da entropia e ganho de informação se dá por:

1. Calcular a entropia do nó pai:

$$Entropia(S) = -\left(\frac{15}{30}\right) log_2\left(\frac{15}{30}\right) - \left(\frac{15}{30}\right) log_2\left(\frac{15}{30}\right) = 1$$

2. Calcular a entropia de cada nó individual e calcular a média ponderada de todos os subnós disponíveis na divisão:

2.1. Entropia para Sexo:

- Entropia para o subnó feminino:

$$Entropia(SexoF) = -\left(\frac{2}{10}\right) log_2\left(\frac{2}{10}\right) - \left(\frac{8}{10}\right) log_2\left(\frac{8}{10}\right) = 0{,}72$$

- Entropia para o subnó masculino:

$$Entropia(SexoM) = -\left(\frac{13}{20}\right) log_2\left(\frac{13}{20}\right) - \left(\frac{7}{20}\right) log_2\left(\frac{7}{20}\right) = 0{,}93$$

- Entropia da Divisão por Sexo:

$$Entropia(Sexo) = \left(\frac{10}{30}\right) * 0{,}72 + \left(\frac{20}{30}\right) * 0{,}93 = 0{,}86$$

2.2. Entropia para Classe:

- Entropia para o subnó Classe IX:

$$Entropia(ClasseIX) = -\left(\frac{6}{14}\right) log_2\left(\frac{6}{14}\right) - \left(\frac{8}{14}\right) log_2\left(\frac{8}{14}\right) = 0{,}99$$

- Entropia para o subnó Classe X:

$$Entropia(ClasseX) = -\left(\frac{9}{16}\right) log_2\left(\frac{9}{16}\right) - \left(\frac{7}{16}\right) log_2\left(\frac{7}{16}\right) = 0{,}99$$

- Entropia da Divisão por Classe:

$$Entropia(Classe) = \left(\frac{14}{30}\right) * 0{,}99 + \left(\frac{16}{30}\right) * 0{,}99 = 0{,}99$$

Apenas com o cálculo da entropia para cada variável é possível saber que a divisão por Sexo seria o primeiro nó escolhido da árvore, já que apresenta a menor entropia. Mas para saber quanta informação se obtém ao fazer essas divisões, calcula-se o ganho de informação:

$$Ganho(Sexo) = Entropia(S) - Entropia(Sexo) = 1 - 0{,}86 = 0{,}14$$

$$Ganho(Classe) = Entropia(S) - Entropia(Classe) = 1 - 0{,}99 = 0{,}01$$

Logo, como o ganho de informação fazendo a divisão por Sexo é maior, seleciona-se esse recurso. Dessa forma, é possível interpretar que é reduzida a incerteza no resultado da previsão em 0,14 (bits).

2.1.3.2 Índice de Gini

O Índice de Gini é outra métrica de impureza que mede a heterogeneidade dos dados sendo utilizado para decidir qual o ponto de corte ideal nas divisões dos nós. A impureza de Gini é dada por:

$$Gini(S) = 1 - \sum_{i=1}^{m} {p_i}^2,$$

onde p_i é a probabilidade de uma tupla em S pertencer a classe C_i.

O índice de Gini considera divisões binárias para cada atributo, em que se pode calcular a soma ponderada da impureza de cada divisão. Se a divisão binária de um atributo particionar os dados em S_1 e S_2, o índice de Gini de S será:

$$Gini(S, A) = \frac{|S_1|}{|S|} Gini(S_1) + \frac{|S_2|}{|S|} Gini(S_2)$$

Quanto menor o valor do Índice de Gini maior a homogeneidade do nó. Se o nó é considerado puro, a impureza de Gini é igual zero. O atributo de divisão selecionado é aquele com menor Índice Gini.

Para ilustrar o cálculo do Índice de Gini, usou-se o exemplo mostrado acima:

1. Calcular o Índice Gini para os subnós e calcular sua média ponderada:

1.1. Índice de Gini para Sexo:

- Gini para subnó feminino:

$$Gini(SexoF) = 1 - \left(\frac{2}{10}\right)^2 - \left(\frac{8}{10}\right)^2 = 0{,}32$$

- Gini para subnó masculino:

$$Gini(SexoM) = 1 - \left(\frac{13}{20}\right)^2 - \left(\frac{7}{20}\right)^2 = 0{,}46$$

- Gini ponderado da Divisão por Sexo:

$$Gini(Sexo) = \left(\frac{10}{30}\right) * 0{,}32 + \left(\frac{20}{30}\right) * 0{,}46 = 0{,}41$$

1.2. Índice de Gini para Classe:

- Gini para subnó Classe IX:

$$Gini(ClasseIX) = 1 - \left(\frac{6}{14}\right)^2 - \left(\frac{8}{14}\right)^2 = 0{,}49$$

- Gini para subnó Classe X:

$$Gini(ClasseX) = 1 - \left(\frac{9}{16}\right)^2 - \left(\frac{7}{16}\right)^2 = 0{,}49$$

- Gini ponderado da Divisão por Classe:

$$Gini(Classe) = \left(\frac{14}{30}\right) * 0{,}49 + \left(\frac{16}{30}\right) * 0{,}49 = 0{,}49$$

Após os cálculos observa-se que o menor índice de Gini é na divisão por Sexo, portanto, igual aconteceu no cálculo da entropia, a variável de Sexo será escolhida para o nó principal da árvore.

3 Métricas de Performance do Modelo

Para avaliar a capacidade discriminatória de um modelo de classificação, após treinado e aplicado no conjunto de teste, utiliza-se métricas que são calculadas a partir de uma matriz de confusão.

Uma matriz de confusão permite visualizar de forma simples se as observações do conjunto de teste foram classificadas corretamente ou erroneamente. Sendo assim é possível calcular métricas que informam o quanto o modelo está acertando ou se está errando mais do que devia.

Tabela 1 - Matriz de Confusão

	Valor Predito		
Valor Real		Y = 1	Y = 0
	Y = 1	Verdadeiro Positivo (VP)	Falso Negativo (FN)
	Y = 0	Falso Positivo (FP)	Verdadeiro Negativo (VN)

Fonte: Elaborada pelo autor

Sendo as definições das medidas da Tabela 1:

- Verdadeiro Positivo (VP): classe de interesse classificada corretamente;
- Falso Positivo (FP): classe de não interesse classificada como classe de interesse;
- Verdadeiro Negativo (VN): classe de não interesse classificada corretamente;
- Falso Negativo (FN): classe de interesse classificada como classe de não interesse.

Algumas métricas que se pode calcular com essas medidas da matriz de confusão são: Acurácia, Precisão, Recall e F1-Score.

Acurácia

A acurácia indica a proporção do quanto o modelo classificou corretamente, ou seja, de todas as previsões, quantos porcento o modelo acertou. Apesar de ser uma métrica usual e de fácil interpretação, ela pode não ser a mais adequada e deve-se tomar cuidado ao levá-la em consideração em casos que os dados são desbalanceados, pois o modelo pode aprender a classificar melhor a classe que mais aparece do que a classe que está em menor proporção, dessa forma inflando o valor da acurácia. Sua fórmula é dada por:

$$Acurácia = \frac{VP + VN}{VP + VN + FP + FN}$$

Precisão

A precisão é uma métrica que quantifica a capacidade do modelo em não classificar como positiva uma observação que na realidade é negativa, ou seja, de todas as previsões positivas, quantas estão corretas. Sua fórmula é dada por:

$$Precisão = \frac{VP}{VP + FP}$$

Recall

O recall indica quanto o modelo consegue classificar todas as observações positivas corretamente, ou seja, de todas as observações que na realidade são positivas, quantas foram previstas corretamente. É uma métrica bem forte para situações em que se quer que o modelo preveja todos os eventos de interesse corretamente e que seja prejudicial não os prever. Sua fórmula é dada por:

$$Recall = \frac{VP}{VP + FN}$$

F1-Score

O F1-Score é uma métrica que une precisão e recall e é definida pela média harmônica entre as duas. Embora seja menos interpretável, o F1-Score resume bem a qualidade do modelo. Sua fórmula é dada por:

$$F1 = \frac{2 * precisão * recall}{precisão + recall}$$

4 Base de dados, desenvolvimento do modelo e resultados

4.1 Descrição dos dados

O conjunto de dados é composto por 13 variáveis que caracterizam cada inscrito, mais um último valor que corresponde a variável objetivo binária, que indica se a pessoa está procurando por um novo emprego ou não. Os atributos de cada inscrição são:

- enrollee_id: ID exclusivo para candidato;
- city: código da cidade (categórica);
- city_development_index: índice de desenvolvimento da cidade (escala) (numérica);
- gender: gênero do candidato (categórica);
- experience_relevent: experiência relevante do candidato (categórica);
- enrolled_university: tipo de curso universitário matriculado (se houver) (categórica);
- education_level: nível educacional do candidato (categórica);
- major_discipline: disciplina principal de educação do candidato (categórica);
- experience: experiência total do candidato em anos (categórica);
- company_size: nº de funcionários na empresa do empregador atual (categórica);
- company_type: tipo de empregador atual (categórica);
- last_new_job: diferença em anos entre o trabalho anterior e o trabalho atual (categórica);

- training_hours: horas de treinamento completo (numérica);
- target: 0 – Não procura mudar de emprego, 1 – Procura mudar de emprego.

A base de dados possui 19158 inscrições, sendo que 4777 são de pessoas que estão procurando por emprego. As variáveis *enrollee_id* e *city* não serão usadas para modelar o problema em questão por não apresentar significado plausível a procura de emprego. De toda a base, existem 20733 campos nulos e, para fins práticos, esses valores faltantes serão retirados da base.

4.2 Dados faltantes

Ainda que nessa base de dados haja grande ausência de dados e por mais que a técnica de Árvores de Decisão apresente potencial para lidar com essas situações emergentes, o pacote do python scikit-learn, usado para construção do modelo, possui problemas com dados faltantes. Como o intuito do projeto é ilustrar a aplicação da Estatística de forma usual em uma área que ainda não faz seu uso efetivo, optou-se por retirar todas as inscrições que possuía pelo menos um dado faltante. Portanto a base de dados foi reduzida de 19 mil observações para quase 9 mil, conforme ilustra a Tabela 2.

Tabela 2 - Quantidade de dados e proporções de classes das bases

Base	Quantidade de Observações	Quantidade de pessoas procurando emprego (variável alvo)	Proporção de pessoas procurando emprego
Base de dados original	19158	4777	25%
Base de dados sem dados faltantes	8955	1483	17%

Fonte: elaborada pelo autor

4.3 Base desbalanceada

Na Tabela 2, nota-se que as bases estão desbalanceadas, isto é, as proporções das categorias da variável resposta estão diferentes e bem discrepantes. Na base sem dados faltantes, 17% das pessoas estão procurando emprego, enquanto 83% não.

Essa diferença faz com que se tenha mais informações a respeito da categoria mais recorrente e menos das minoritárias. Nisso, alguns tipos de modelos de Aprendizagem de Máquina podem não lidar tão bem com essa situação, que é o caso dos algoritmos de Árvores de Decisão, já que o algoritmo procura por padrões frequentes, logo as classes desfavorecidas não influenciam do jeito adequado a formação desses padrões.

Por isso que, como comentado na seção 3, a Acurácia levaria a uma falsa avaliação do modelo pois preveria corretamente muitas observações da classe de não interesse sendo que o objetivo é prever aqueles que estão procurando um novo emprego. Dessa forma, outras métricas como Precisão e Recall (e F1-Score, por ser a média harmônica entre elas) se fazem mais úteis para avaliar o modelo já que distinguem as classes de interesse e não interesse.

Essa discrepância entre as classes pode representar a real distribuição de pessoas que procuram por emprego, principalmente num cenário de inscrições para um curso de Data Science, no qual as pessoas estão procurando se capacitar.

Técnicas como undersampling e oversampling, que igualam as quantidades das classes da variável resposta, sendo que a primeira reduz as observações da classe maioritária, e a segunda replica as observações da classe minoritária, podem ser usadas para tratar o desbalanceamento. No entanto, para fins práticos, optou-se por preservar as proporções das classes.

4.4 Variáveis Dummy

Com o avanço em melhorias nos algoritmos de Árvores de Decisão, conseguiu-se fazer uso de dados categóricos e contínuos, inclusive trabalhar com dados categóricos com mais de duas classes utilizando-se de divisões binárias nos nós (CART). No entanto, as Árvores de Decisão construídas por

meio da biblioteca do scikit-learn no python, que usam o algoritmo CART, não suportam dados categóricos, e a grande maioria das variáveis do conjunto de dados desse estudo são categóricas. Então, para conseguir usá-las foi necessário primeiro transformá-las em dados numéricos utilizando-se do conceito de variável dummy.

As variáveis dummy são variáveis binárias (0 ou 1) criadas para representar uma variável com duas ou mais categorias. Para uma variável categórica de n categorias, serão criadas n-1 variáveis dummy. As dummy são representadas por 1 se a característica estiver presente e por 0 se não estiver.

Dessa forma, as 9 variáveis categóricas explicativas úteis da base foram transformadas em dummy, totalizando em 36 variáveis explicativas, sendo 34 variáveis dummy e 2 variáveis numéricas.

Como exemplo, no caso da variável *gender* existem 3 categorias diferentes: Male, Female e Other. Dessa forma, para representar essas 3 categorias, foram criadas 2 variáveis dummy *gender_Male* e *gender_Other*. Então, se o inscrito for um homem, a variável *gender_Male* estará com 1 e a *gender_Other* com 0. Caso o inscrito tenha se identificado como outro sexo, a variável *gender_Male* fica com 0 e a *gender_Other* com 1. Por fim, se o inscrito se identificar como mulher ambas as variáveis ficam com 0. Veja na Figura 5 abaixo:

Figura 5 – Exemplo de variável dummy com a variável *gender*

gender_Male	gender_Other
1	0
0	1
0	0

Fonte: elaborada pelo autor

4.5 Ajuste do modelo e Resultados

Para a construção do modelo, foi separado um conjunto de 70% das observações da base para compor a base de treinamento e foram utilizadas as 36 variáveis explicativas em um modelo de árvore de decisão definida com as

configurações padrão do scikit-learn (seu padrão usa o Índice Gini para escolher os nós e fazer as divisões).

Após treinado, foi aplicado o modelo na base de teste (os 30% dos dados restantes) para verificar seu desempenho. Assim, pode-se notar na Tabela 3 o quão performático foi o modelo.

Tabela 3 - Resultados das métricas do modelo na base de teste

Métricas modelo 1			
Acurácia	Precisão	Recall	F1-Score
0,77	0,36	0,37	0,36

Fonte: elaborado pelo autor

Por mais que o valor da acurácia seja alto, com uma taxa de acerto geral de 77%, ela não é uma boa métrica para avaliar esse caso de dados desbalanceados, como dito anteriormente. Para isso, avalia-se as outras métricas como Precisão, Recall e F1-Score, que no caso do primeiro modelo estão bem baixas. Com essa precisão tem-se que de todas as previsões que o modelo fez de pessoas que estão procurando emprego, apenas 36% de fato estão. Enquanto no recall, de todas as pessoas que efetivamente estão procurando emprego, o modelo acertou apenas 37%.

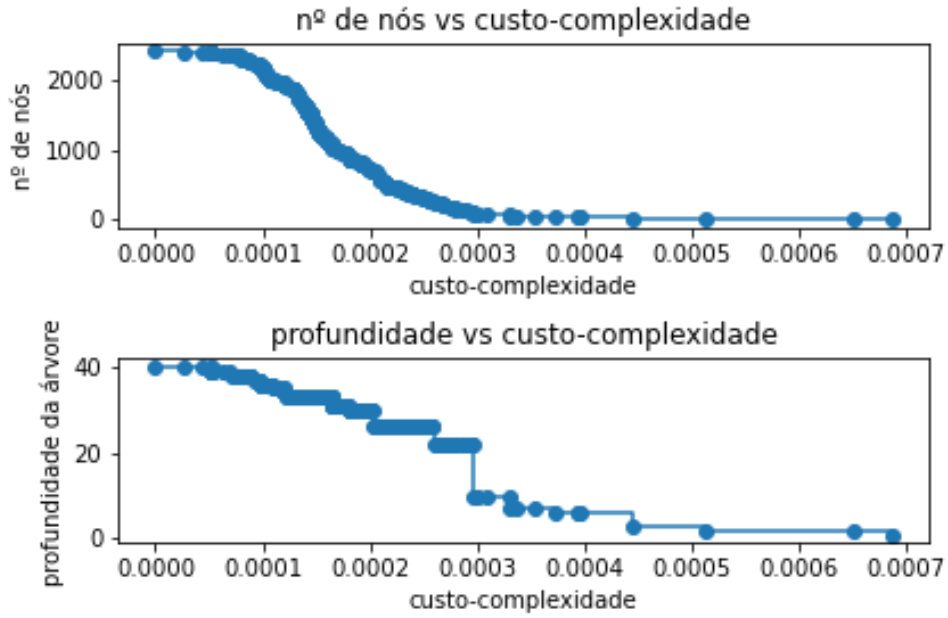
Um dos motivos que possam ter levado ao baixo desempenho do modelo é a sua tamanha especificidade, já que como as configurações padrão do scikit-learn para as árvores de decisão não limita o crescimento da árvore, foram feitos recursivamente vários nós para explicar a procura por emprego, e como há uma grande quantidade de variáveis, a árvore tornou-se gigantesca com o total de 2423 nós. Com essa quantidade de nós, é inviável interpretá-la, principalmente para a ocasião desse estudo em que se quer entender usualmente os motivos que levam a pessoa a procurar por um novo emprego e prever se de fato ela está procurando por um. Além disso, pela árvore ser muito específica, faz com que ela sofra de *overfiting*. Portanto, nesse caso é necessário que a árvore seja podada.

4.5.6 Critério de Poda

Uma opção para achar o ponto de poda ideal é se baseando no Custo de Complexidade de Poda. Esse parâmetro controla a complexidade do modelo, ou seja, quanto menor seu valor, menos a árvore será penalizada, portanto, maior será o tamanho da árvore.

Então, inicialmente é gerado uma árvore com o parâmetro de custo-complexidade definido em zero, resultando em uma árvore de tamanho máximo. A Figura 6 mostra a relação de cada etapa do tamanho da árvore com seus respectivos custos-complexidade.

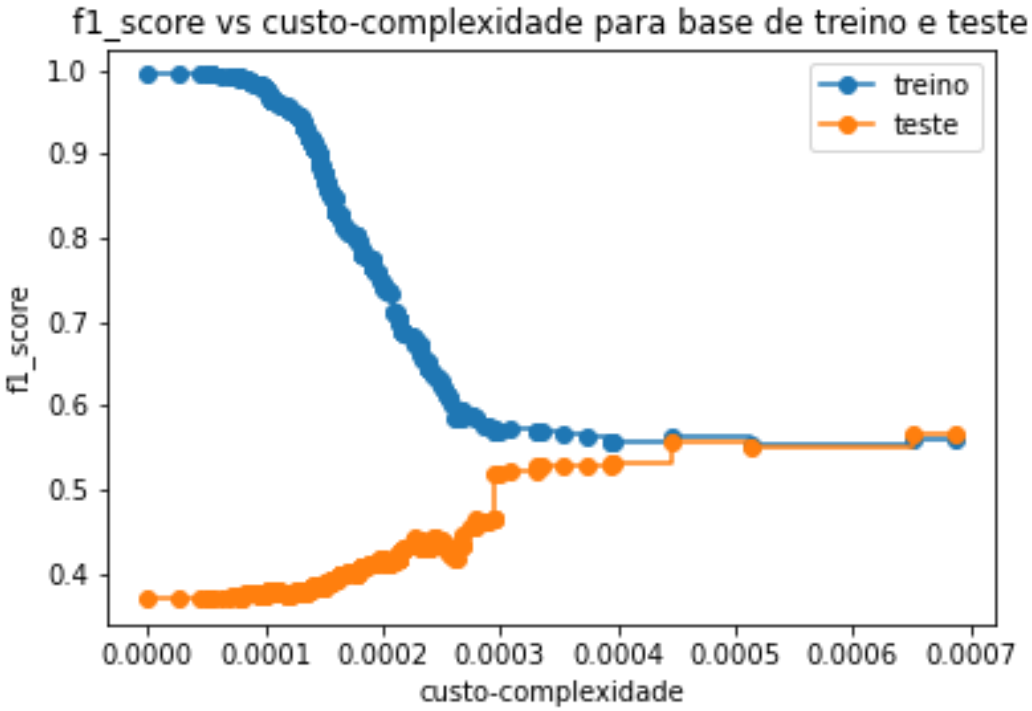
Figura 6 - Variação de nós e profundidade para cada custo-complexidade



Fonte: elaborada pelo autor

Para esse estudo, a árvore de tamanho ideal é aquela que maximiza o F1-Score, portanto para achar o custo-complexidade que maximiza a métrica é criada uma árvore para cada custo-complexidade - obtidos da árvore de tamanho máximo - e selecionar o custo-complexidade da árvore que tem o maior valor de F1-Score, conforme ilustra a Figura 7.

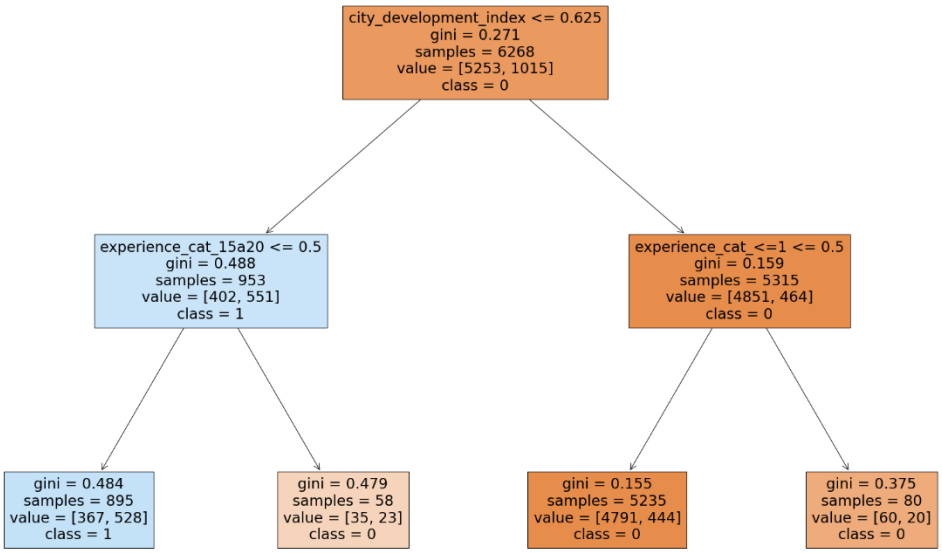
Figura 7 - Variação do F1-Score nas bases de treino e teste para cada árvore gerada para cada custo-complexidade



Fonte: elaborada pelo autor

Pode-se notar na Figura 7 que quando o custo-complexidade é definido como zero, a árvore se sobreajusta, fazendo com que o F1-Score na base de treinamento seja de 100%, e pelo modelo estar muito específico, sua performance quando aplicado na base de teste não é tão boa, com um F1-Score de 37%. Assim, enquanto o custo-complexidade aumenta, mais a árvore é podada, criando assim uma árvore de decisão que generaliza melhor. Dessa forma, a árvore foi podada de modo que o F1-Score fosse o máximo observado, resultando em uma árvore com nível de profundidade de 2 e o custo-complexidade igual a 0,00065.

Figura 8 - Imagem da árvore gerada após a poda



Fonte: elaborada pelo autor

Após treinada novamente, agora em um nível de profundidade de 2 e um custo-complexidade de 0,00065, e aplicada no conjunto de teste, nota-se na Tabela 4 a performance melhorada e uma árvore interpretável.

Tabela 4 - Resultados das métricas do modelo podado na base de teste

Métricas modelo 2			
Acurácia	Precisão	Recall	F1-Score
0,86	0,62	0,50	0,55

Fonte: elaborado pelo autor

Ainda que a performance geral, principalmente do F1-Score não seja tão alta, pode-se dizer, que devido à complexidade do problema, obteve-se bons resultados.

4.6 Interpretação do modelo

Na Tabela 5, são apresentados os resultados da importância das 36 variáveis preditoras na geração do modelo de Árvore de Decisão. A variável *city_development_index* foi a que teve maior importância com 97,89%, o que mostra uma forte correlação com a variável *target*. Outras variáveis como *experience_cat<=1* e *experience_cat<=15a20* tiveram uma baixíssima correlação com a variável alvo, enquanto as demais ficaram com importância zero, indicando que não são importantes para predizer a procura por emprego.

Tabela 5 - Importância das variáveis para predizer a procura por emprego

Variáveis	Importância (%)
city_development_index	97,89%
experience_cat<=1	1,08%
experience_cat<=15a20	1,02%
Demais variáveis	0%

Fonte: elaborado pelo autor

Com as regras geradas, após a execução do algoritmo de árvore de decisão, é possível ver como o inscrito será classificado de acordo com as especificações de cada regra. Nota-se na Figura 8, que as duas variáveis usadas para compor as regras do modelo foram *city_development_index* e *experience*, sendo a de índice de desenvolvimento da cidade a mais forte para explicar a procura por emprego.

Interpreta-se a árvore apresentada na Figura 8 da seguinte forma: se a pessoa mora em uma cidade com o índice de desenvolvimento inferior a 0,625 e ela possuir menos de 15 anos de experiência de trabalho, a árvore já a classifica como procurando emprego, caso contrário, se a pessoa não atender exclusivamente esses dois critérios a pessoa é declarada como não procurando emprego.

Pela variável *city_development_index* ser a mais forte, mostra que pessoas que moram em cidades com baixo desenvolvimento tendem a procurar mais por emprego. Isso pode dizer que a cidade não oferece boas condições de

vida e por isso as pessoas querem mudar de emprego para se mudar de cidade. Assim, a empresa precisa se atentar nas cidades em que seus colaboradores moram para tentar oferecer condições melhores, até mesmo para que se mudem de cidade.

5 Conclusão

Apesar das dificuldades enfrentadas como a base ser desbalanceada, grande ausência de valores e a performance não tão elevada, o modelo de árvore de decisão estimado apresentou resultados coerentes e boa capacidade de discriminação das pessoas que procuram ou não por emprego, visto que procurar por um emprego novo não se resume apenas em características demográficas, perfil acadêmico e profissional.

O modelo de árvore de decisão pode ser melhorado se conseguir treiná-lo em uma base de dados em que a variável alvo tenha proporções parecidas. No entanto, para esse tipo de problema do estudo, é natural que menos pessoas estejam procurando trocar de emprego, e por outro lado, pode ser difícil obter os tipos de dados usados neste trabalho.

No mais, a técnica de árvore de decisão conseguiu cumprir bem o papel de entender o perfil de uma pessoa que está procurando por emprego e ilustrar isso de forma intuitiva e fácil para quem precisa tomar ações para que evite o colaborador se desligue.

Referências

ADMINVOOO. **Um tutorial completo sobre modelagem baseada em árvores de decisão (códigos R e Python)**. 2016. Disponível em: https://www.vooo.pro/insights/um-tutorial-completo-sobre-a-modelagem-baseada-em-tree-arvore-do-zero-em-r-python/. Acesso em: 17 jan. 2022.

CHIAVENATO, Idalberto. **Gestão de Pessoas**. 3. ed. Rio de Janeiro: Campus, 2008.

FISCHER, Rosa Maria, **Mudança e Transformação Organizacional**, In: FLEURY, M., (org.), As pessoas na organização. São Paulo: Editora Gente, 2002.

JANATAHACK. **JanataHack: RH Analytics**. 2020. Disponível em: https://datahack.analyticsvidhya.com/contest/janatahack-hr-analytics/. Acesso em: 10 mar. 2021.

KAGGLE. Disponível em: https://www.kaggle.com. Acesso em: 10 mar. 2021.

Möbius. **HR Analytics: Job Change of Data Scientists**. 2020. Disponível em: https://www.kaggle.com/arashnic/hr-analytics-job-change-of-data-scientists. Acesso em: 10 mar. 2021.

QUINLAN, J.R. **Induction of Decision Trees**. Machine Learning, 1, 81-106. 1986. Disponível em: http://dx.doi.org/10.1007/BF00116251. Acesso em: 18 ago. 2021.

SANTANA. **Árvore de Decisão (Projeto passo a passo)**. 2020. Disponível em: https://minerandodados.com.br/arvores-de-decisao-conceitos-e-aplicacoes/. Acesso em: 15 ago. 2021.

SIMÕES, Adriana Carla Araújo; Crispim Vasconcelos, Germano. **Mineração de dados baseada em árvores de decisão para análise do perfil de contribuintes**. 2008. Dissertação (Mestrado). Programa de Pós-Graduação em Ciência da Computação, Universidade Federal de Pernambuco, Recife, 2008. Disponível em: https://repositorio.ufpe.br/handle/123456789/1476. Acesso em: 08 maio 2021.

SINGH, Sonia; GUPTA, Priyanka. **Comparative study ID3, CART and C4.5 Decision Tree Algorithm: A Survey**. 2014. International Journal of Advanced Information Science and Technology (IJAIST), Vol.27, No.27, July 2014. Disponível em: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.685.4929&rep=rep1&type=pdf. Acesso em: 12 set. 2021.