PUCRS

ESCOLA POLITÉCNICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DE DADOS – 2ª EDIÇÃO
CURSO DE ESPECIALIZAÇÃO EM CIÊNCIA DE DADOS – 2ª EDIÇÃO

FELIPE MÜLLER ALVES

MODELO DE PEOPLE & ANALYTICS PARA TOMADA DE DECISÃO EM GESTÃO DE CARREIRAS UTILIZANDO ÁRVORE DE DECISÃO

Porto Alegre
2022

PÓS-GRADUAÇÃO - *LATO SENSU*

Pontifícia Universidade Católica
do Rio Grande do Sul

Modelo de People & Analytics para tomada de decisão em gestão de carreiras utilizando Árvore de Decisão

Felipe Müller Alves
Porto Alegre, Brasil
felipe.germanisch@hotmail.com
Link para o código:
https://encurtador.com.br/xkePE

***Resumo*—A demanda por maior agilidade e dinamismo no mercado de trabalho, gerou diversas tecnologias capazes de auxiliar gestores durante a tomada de decisão sobre os colaboradores de sua equipe e seu destino dentro das organizações. Em resposta à esta demanda, surgem as práticas de People & Analytics, que visam expressar em números o comportamento humano e, dentro do contexto das organizações modernas, tornar o processo de tomada de decisão de gestores mais dinâmico e objetivo. Este trabalho propõe a criação de um modelo para decidir se um colaborador deve ser demitido, mantido, receber um aumento de salário ou ser promovido, de acordo com diversos indicadores, utilizando técnicas de *machine learning*, com o algoritmo de Árvore de decisão.**

Palavras chave - People & Analytics, Tomada de decisão, Gestão de Pessoas, Árvore de Decisão, Machine Learning

I. Introdução

People Analytics, também conhecido como análise de pessoas, é uma abordagem orientada a dados, que trata da coleta, organização e diagnóstico de dados sobre colaboradores e equipes de uma determina empresa ou organização, utilizando softwares que realizam o cruzamento de dados de diferentes fontes, no intuito de proporcionar um gerenciamento de recursos humanos mais eficiente [3].

Em linhas gerais, People Analytics é utilizado por gerentes e executivos que desejam tomar decisões mais assertivas, em relação aos seus colaboradores. Suas técnicas possibilitam um melhor acompanhamento de questões como engajamento, produtividade e a satisfação do público interno da empresa [3]. Dentro contexto, o presente trabalho apresenta um modelo que avalia os colaboradores sob diversas variáveis, principalmente referentes à produtividade, classificando os colaboradores e propondo a gestores tomar uma entre quatro decisões diferentes para cada colaborador da organização: Promover de cargo, conceder aumento salarial, manter como está ou demitir.

Para isso, foram gerados dados sintéticos que simulam pontuações que representariam colaboradores excelentes, medianos ou abaixo da média, que serviram para o treinamento do modelo, e aplicado posteriormente em dados de cerca de 1.000 colaboradores fictícios. Mesmo com a utilização de dados sintéticos, a ideia central é que gestores, ao coletar os mesmos dados propostos no presente trabalho, rodem o modelo nos dados de seus colaboradores, afim de obter o resultado classificatória desejado. Vale ressaltar que o modelo proposto não exclui uma análise qualitativa por parte dos colaboradores por parte dos gestores, mas sim uma análise quantitativa e objetiva, que serve de apoio à tomada de decisão, mas não como um julgamento final sobre a carreira do colaborador na organização.

II. Dados e Pré-processamento

A. *Dados*

Devido a questões de segurança e sigilo, nenhuma empresa aceitou fornecer os dados necessários para a realização do presente trabalho, e dados semelhantes não foram encontrados em nenhuma base de dados pública. Desta forma, para a realização do trabalho, foram utilizados dados gerados sinteticamente, dispostos em 11 colunas com dados aleatórios numéricos, sendo elas:

- **Salário;**
- **Dependentes;**
- **Faltas;**
- **Atrasos;**
- **Participação em atividades de T&D;**
- **Tarefas Concluídas;**
- **Metas alcançadas;**
- **Atestados apresentados;**
- **Cursos de aperfeiçoamento;**
- **Tickets internos atendidos;**
- **Avaliação da equipe;**
- **Avaliação do gestor.**

Os dados gerados nas 11 colunas, servem de base para que o modelo analise e classifique os colaboradores entre 4 decisões possíveis, para cada colaborador. Para isso, foram feitas quatro rodadas de geração de dados, uma para cada decisão (ou respostas categóricas):

- **Demissão.** Os primeiros dados gerados foram de colaboradores de baixo rendimento e que, no caso, seriam demitidos, de acordo com seus indicadores. Sendo assim, em cada uma das 11 colunas, os dados gerados refletem indicadores que indicam baixo rendimento do colaborador, como por exemplo, uma quantidade significativa de faltas, poucas metas alcançadas, má avaliação por parte de gestor e equipe, etc;
- **Manter.** Na segunda rodada de geração de dados, são gerados dados de colaboradores de rendimento baixo/médio, e que devem ser mantidos em seus respectivos cargos e salários. Os indicadores destes colaboradores tendem a ser melhores do que os indicadores apresentados pelos colaboradores a serem demitidos, com menos atrasos e mais participações em atividades de Treinamento & Desenvolvimento, por exemplo;
- **Conceder aumento.** Na terceira rodada, são gerados dados de colaboradores com desempenho médio/bom, e que são passíveis de receberem aumento salarial. Em geral, são colaboradores com boa pontuação em diversas colunas;
- **Promover.** Na última parte de geração de dados, são gerados dados de colaboradores de alto rendimento, e que elegíveis a serem promovidos pela organização. Estes profissionais possuem boa pontuação na maior parte das colunas, principalmente nos indicadores mais importantes;

É importante ressaltar que os dados gerados são totalmente subjetivos e de acordo com a vivência e experiência de mercado do

autor, a aplicação deste modelo para este tipo de tomada de decisão não deve ser absoluta, mas sim servir de base para que gestores tomem suas decisões, levando em conta também outros aspectos subjetivos em sua análise. Os dados reais nas organizações podem variar de forma significativa.

B. Pré-processamento dos dados

Para a elaboração do presente trabalho, foram realizadas determinadas etapas, que serviram de base para a preparação dos dados, e aplicação do algoritmo de árvore de decisão, além da análise de seu desempenho. As etapas são:

- **Importação das bibliotecas.** O primeiro passo foi a importação das bibliotecas necessárias para a geração dos dados e aplicação do modelo. No presente trabalho, foram utilizadas as bibliotecas *Pandas* e *NumPy*, para a geração dos dados. Já para aplicação da árvore, foram utilizadas ferramentas da biblioteca *Scikit-Learn*, ou *sklearn*. *Scikit Learn* é uma biblioteca em Python, que serve para aplicar modelos de *machine learning*, com diversas ferramentas e métodos já implementados, além de algoritmos e técnicas que tornam mais simples o trabalho do cientista de dados [2]. Foram importadas da biblioteca *sklearn.tree*, através do comando *import*, a *DecisionTreeClassifier*, que é o classificador da árvore de decisão, da *sklearn.model_selection*, a *train_test_split*, utilizada para dividir a base de dados entre dados de teste e dados de treino do modelo, e do *sklearn*, o *metrics*, utilizado para avaliar o desempenho do modelo, de forma rápida e simples. Além da importação das bibliotecas necessários, também é definido a quantidade de objetos (colaboradores) que serão gerados para cada uma das categorias de classificação.

- **Estabelecimento de parâmetros.** Nesta etapa, são definidos valores mínimos e máximos, para cada uma das 11 colunas que serão geradas no banco de dados. Esta etapa é repetida 4 vezes, uma para cada categoria de classificação, ou seja, são estabelecidos valores mínimos e máximos, em cada uma das 11 colunas e para cada tipo de colaborador: os que serão demitidos, os que serão mantidos, os que receberão aumento e os que serão promovidos. Os valores dos parâmetros utilizados foram estabelecidos de forma subjetiva pelo autor, entretanto, a lógica para o estabelecimento dos mesmos obedece à regra de que os parâmetros não são mutuamente excludentes, do contrário, o modelo apresentaria *overfitting*. Por exemplo, na coluna "avaliação do gestor", os dados gerados para os colaboradores de baixo rendimento e que deveriam ser demitidos, foram gerados de forma aleatória entre 0 e 5, pois o autor entende que dificilmente um colaborador de baixo rendimento teria uma nota do gestor acima de 5, já para os colaboradores de alto rendimento, e que são classificados para promoção, foram gerados dados aleatórios entre 3 e 10, pois o autor também entende que dificilmente um colaborador que seria promovido, teria uma avaliação menor que 3 do próprio gestor. A mesma lógica foi utilizada em todas as colunas. Desta forma, mesmos os dados se sobrepondo, ainda assim o modelo consegue classificar uma decisão para cada colaborador.

- **Geração dos dados.** Após o estabelecimento dos parâmetros, são gerados os dados para cada uma das colunas, utilizando o método *np.random.randint*, tendo como parâmetro as variáveis atribuídas para cada valor mínimo e máximo em cada coluna. Cada coluna de dados é gerada separadamente e depois concatenada com *np.concat*, onde são colocadas em ordem. Por último, é acrescentada a coluna com a variável resposta que categoriza a decisão para o colaborador. Por exemplo, para os colaboradores de baixo rendimento, são estabelecidos os parâmetros para cada coluna, são geradas e concatenadas as 11 colunas, e por último, é acrescentada a coluna categórica "demissão". Desta forma, é estabelecido o perfil do colaborador passível de demissão, com base nos dados gerados.

- **Concatenação das bases de dados geradas.** Após 4 rodadas de geração de dados, e a definição de perfil de colaborador e sua respectiva categoria de classificação, os 4 *dataframes* gerados são concatenados, formando apenas uma base de dados consolidada e pronta para a aplicação da *Decision Tree*.

Em todos os quatro conjuntos de dados, foram gerados 150 objetos, ou seja, 150 colaboradores, totalizando 600 colaboradores, uma quantidade razoável de colaboradores para uma média/grande empresa. Para efeitos de avaliação deste trabalho, na geração dos dados foi utilizado o comando *np.random.seed*, no intuito de que se obtenha sempre os mesmos números aleatórios gerados.

III. O ALGORITMO

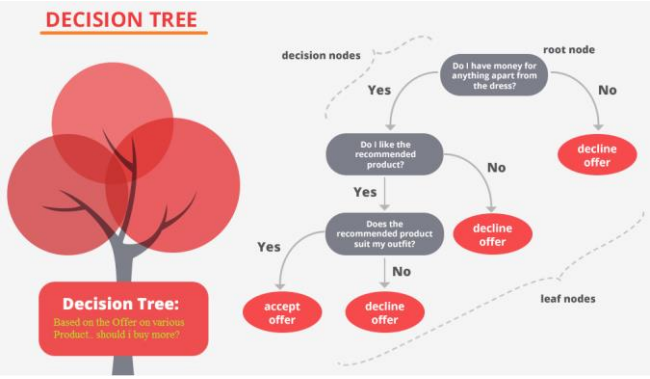
A árvore de decisão, ou *Decision Tree*, é um algoritmo que possui especial destaque, em meio aos outros algoritmos de aprendizado de máquina, devido à sua estrutura visual. Tal fenômeno ocorre devido à sua similaridade com um fluxograma, onde cada parte do mesmo são facilmente visualizadas e compreendidas. Desta forma, a árvore de decisão se mostra um algoritmo com resultados satisfatórios para um melhor entendimento sobre o funcionamento do aprendizado de máquina [1].

Pode ser o algoritmo mais conhecido de aprendizado de máquina, desconsiderando as redes neurais. Entretanto, ainda que as redes neurais sejam muito conhecidas, poucos sabem como elas de fato funcionam. A árvore de decisão carrega a ideia de tomada de decisão, que tem firme presença em diversas áreas, porém, é de mais fácil entendimento [1].

A. O que é uma Decision Tree?

A árvore de decisão é um algoritmo de *machine learning*, fortemente utilizado para classificar objetos. Sendo assim, pode ser aplicado para prever categorias discretas, mas também pode servir para previsão de valores numéricos (regressão) [1].

Da mesma forma que o fluxograma, a árvore de decisão possui nós (também conhecidos como *decision nodes*), que possuem relação um com o outro de forma hierárquica, sendo eles o nó-raiz (*root node*), o mais importante de todos, que é um atributo da base de dados, e os nós-folha (*leaf nodes*), que contém as categorias discretas finais que são geradas como respostas [1].



Fonte da imagem: Data Science Foundation.

Em meio às conexões entre os diferentes *nodes*, existem regras do tipo "*if-then*". Quando o algoritmo se depara com um *node*, ele se pergunta a respeito de uma determinada regra, um condicional, algo como "se número de faltas de um colaborador analisado é menor do 20?". Caso seja menor, então o algoritmo vai para um lado da árvore, caso contrário, vai para o outro lado. No node seguinte, continua da mesma forma. A *Decision Tree* segue o chamado "recursivo", repetindo o mesmo padrão, conforme se aprofunda em todas as etapas, como se uma determinada função chamasse a si mesma como uma segunda função a ser executada paralelamente, onde a primeira é necessária para chegar à sua resposta. A principal atuação da *Decision Tree* é encontrar os nodes que serão alocados

em cada uma das posições, qual será o *root node*, depois qual será o da esquerda, o da direita, e assim sucessivamente [1].

Para que o algoritmo funcione corretamente, são necessários determinados cálculos. Uma forma comumente aceita por analistas é utilizar o ganho de informações e entropia. Tais variáveis são referentes à falta de organização e uniformidade presente nos dados. Quanto maior a entropia, mais bagunçados os dados se apresentam, de modo simetricamente oposto, quanto menor a entropia, mais organizados e homogêneos está a base de dados [1]. Para estabelecer as posições, é necessário realizar o cálculo da entropia das variáveis respostas e o ganho de informação dos atributos dos dados. Aquele que tiver o maior ganho de informação dentre os diferentes atributos, é o *root node*. Já para definir os lados direito e esquerdo do nó, são realizados mais cálculos de entropia e ganho de informação com a base de dados que satisfaça a condição que encaminha para a direita ou esquerda. Para que se forme a árvore, são necessárias as condições. Com base nas condições, a base de dados é dividida em diferentes caminhos, de acordo com a análise dos objetos que atendam a condição [1].

Por exemplo: caso uma condição seja "atrasos > 10" para o lado esquerdo, haverá apenas registros da base de dados que possuem atrasos maior que 10, e para o lado esquerdo, apenas registros de atrasos menor que 10. O ganho de informação é calculado partindo deste pressuposto. Caso seja analisado um determinado atributo, e os registros da base de dados para cada lado são similares ou homogêneos, é obtido um maior ganho de informação, pois é possível saber que, caso seja atendida certa condição, é mais provável que seja encontrada a resposta esperada, ou que se esteja mais próxima de sua descoberta [1].

B. *Para que serve este algoritmo em machine learning?*

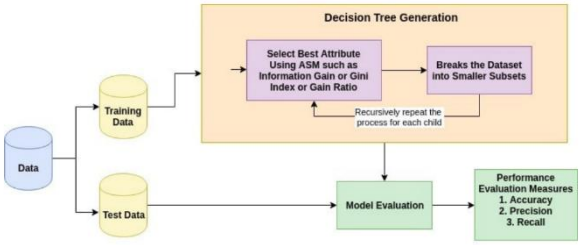
Conforme destacado anteriormente, as árvores de decisão são aplicadas para classificação e regressão, que são tarefas de *machine learning*. Por ser mais versátil, a *Decision Tree* acaba sendo amplamente utilizada, sendo muito popular entre cientistas de dados, em especial quando comparado com algoritmos mais simples, como é o caso do algoritmo *Naive-Bayes* [1].

Outra utilidade importante da *Decision Tree* é quando ocorre a necessidade de resolver um determinado problema com diferentes rótulos, em que existem diferentes e diversas categorias de classificação. Alguns algoritmos se mostram um tanto problemáticos e complexos para resolver este tipo de problema, já a *Decision Tree* apresenta maior facilidade. Uma grande vantagem da árvore de decisão é que não é necessário se ocupar demasiadamente do tratamento dos dados para que o algoritmo funcione corretamente, pois os valores que destoam muito dos demais ou valores ausentes, não influenciam negativamente seu desempenho, sendo assim, bastam mínimas etapas de pré-processamento para funcionar, não sendo necessário sequer a conversão de dados para valores numéricos, uma vez que a *Decision Tree* também lida de forma satisfatória com dados categóricos ou nominais [1].

Quando o assunto for resolver problemas que envolvem tanto regressão quanto classificação, a *Decision Tree* também é indicada, pois também é possível utilizar uma *Random Forest*, ou floresta aleatória, que é um conjunto de árvores treinadas, que resultam em uma predição a respeito dos dados, classificando-os em uma determinada categoria. Desta forma, é realizada uma avaliação da variável resposta, ou categoria, que mais se destacou nos resultados das árvores para que se chegue a uma decisão absoluta. Com códigos simples, analistas obtém uma visualização de uma *Decision Tree*, devido a isso, é possível ver, de forma prática e rápida, como este algoritmo funciona [1].

IV. TREINANDO E TESTANDO A *DECISION TREE*

Uma vez gerados os dados e estabelecido o *dataframe*, os dados são separados entre dados de treino e dados de teste, utilizando-se o método train_test_split(). Dentro deste método, são utilizados como parâmetros a coluna contendo as variáveis respostas categóricas, onde também é utilizando um comando *.drop* para tirá-la do *dataframe*, enquanto se treina o modelo, conforme modelo a seguir:



Fonte: NumPy Ninja

Além de informar a coluna contendo as respostas categóricas, também é utilizado um parâmetro chamado *test_size*, que possibilita ao analista definir um tamanho em porcentagem, de quantos objetos serão utilizados para treino e quantos serão utilizados para teste. No presente trabalho, foi utilizado *test_size* = 0.3, onde 70% dos dados foram utilizados para treino e 30% para teste, ou seja, o modelo da *Decision Tree* aprende com 70% dos dados e testa seu aprendizado nos outros 30%.

Após treinar e testar a *Decision Tree*, também foi utilizado um atributo chamado *.features_importances_*, que analisa e indica quais colunas possuem maior importância para a análise do modelo, conforme demonstrado abaixo:

```
clf.feature_importances_
```

```
array([0.01872319, 0.01102195, 0.02658328, 0.05729264, 0.2573651 ,
       0.03590096, 0.03995707, 0.14807079, 0.08456046, 0.11058156,
       0.20436274, 0.00558027])
```

Conforme observado, as colunas 5 (25% de importância), 8 (14% de importância), 10 (11% de importância) e 11 (20% de importância), respectivamente com os dados sobre a participação em atividades de Treinamento & Desenvolvimento, atestados apresentados, tickets internos atendidos e avaliação da equipe, possuem maior peso e juntas explicam aproximadamente 70% do modelo.

Por fim, com o atributo *.predict()*, que utiliza como parâmetro os dados de teste, é executada a devida predição da categoria para a classificação de cada objeto. É nesta parte em que efetivamente o modelo prevê a categoria resposta, ou seja, classifica os colaboradores a serem demitidos, mantidos, receberem aumento ou serem promovidos.

V. AVALIAÇÃO DA *DECISION TREE*

Após o treinamento e teste do modelo de árvore de decisão, é realizada a avaliação de desempenho do modelo, utilizando o método *metrics* da biblioteca *sklearn*, com o atributo *.classification_report()*, que possui como parâmetros os dados de teste e o resultado da predição aplicada anteriormente, é gerada uma matriz de confusão, contendo a pontuação atingida pelo modelo, conforme mostrado abaixo:

	precision	recall	f1-score	support
Dar aumento	0.85	0.62	0.72	53
Demitir	0.97	1.00	0.99	39
Manter	0.67	0.86	0.75	37
Promover	0.94	0.98	0.96	51
accuracy			0.86	180
macro avg	0.86	0.87	0.85	180
weighted avg	0.86	0.86	0.85	180

A precisão é um dos indicadores mais utilizados para a avaliação de desempenho de modelos de classificação. Em linhas gerais, este indicador calcula a soma dos exemplos classificados de forma correta como positivos (verdadeiros positivos), dividida pelo total de objetos classificados como verdadeiros positivos + falsos

positivos. A métrica de precisão enfatiza os erros de classificação por falso positivo, ou seja, indica quanto dos objetos classificados como positivos, são de fato, positivos [4].

No presente trabalho, o modelo de *Decision Tree* apresentou precisão satisfatória. Dentre os colaboradores classificados para serem promovidos, o modelo executou a classificação corretamente em 94% dos casos, ou seja, a cada 100 colaboradores indicados anteriormente para serem promovidos (positivo), o modelo acertou em 94% dos casos durante os testes. Na classificação de colaboradores para demissão, a precisão foi ainda maior, de 97%.

Diferentemente do indicador de precisão, o *recall*, ou revocação, também considerado como taxa de sensibilidade, enfatiza para os erros de classificação de falso negativo [4]. Ou seja, o *recall* calcula a quantidade de objetos classificados como positivo, que realmente são positivos + falsos negativos, e divide pelo número total de exemplos positivos, por exemplo, verdadeiros positivos / total de positivos [5]. O modelo de árvore de decisão apresentou overfitting nos casos de colaboradores classificados para demissão, obtendo uma pontuação de recall de 100%, já para os colaboradores classificados para receber aumento, o *recall* foi de 62%.

Por fim, um dos principais indicadores na avaliação de modelos de *machine learning* é a acurácia, que informa quantos objetos foram classificados corretamente, de forma geral. A acurácia calcula o total de verdadeiros positivos + verdadeiros, dividido pelo total de verdadeiros positivos + verdadeiros negativos + falsos positivos + verdadeiros negativos. Esta métrica também pode ser entendida como a media aritmética entre os indicadores de precisão [4].

No presente trabalho, o modelo de *Decision Tree* apresentou uma pontuação satisfatória de 86% de acurácia, ou seja, de forma geral, o modelo acertou na classificação em 86% dos casos, o que denota um ótimo desempenho do modelo na recomendação das ações a serem tomadas para cada colaborador.

Conclusões

O propósito deste trabalho foi a criança de um modelo de *machine learning*, utilizando o algoritmo de árvore de decisão, para uma tarefa de classificação de colaboradores, a fim de colaborar com o processo de tomada de decisão de gestores e departamentos de recursos humanos, em relação aos seus profissionais e equipes. A área de gestão de pessoas das grandes organizações é uma das mais complexas, pois envolve decisões que podem ser muito subjetivas, referentes ao destino dos colaboradores. É salutar lembrar que os dados gerados são muito subjetivos, totalmente passíveis de discussão e tomados como base de acordo com a visão e experiência profissional do autor. Sob hipótese alguma é recomendado que gestores utilizem as ferramentas de *People & Analytics*, como o modelo proposto neste trabalho, como solução final em seus processos de tomada de decisão sobre seus colaboradores. A proposta aqui é de tornar este processo de decisão mais objetivo e dinâmico, mas que ainda assim, o gestor faça suas considerações pessoas e suas ponderações em cada caso.

A partir da construção do presente modelo de *Decision Tree*, também é possível realizar diversos outros experimentos a fim de aprimorar o modelo e analisar diferentes resultados de classificações. É possível, por exemplo, eliminar as colunas que não são importantes e rodar o modelo apenas utilizando as colunas importantes, com base no resultado do uso do atributo *.feature_importances()*. Outro experimento interessante de ser feito é aplicar o banco de dados gerados em outros modelos de machine learning e realizar uma comparação entre diferentes algoritmos, no presente caso foi utilizado o algoritmo de árvore de decisão devido à sua boa aplicação para saídas discretas, mas é possível tentar outros, no intuito de avaliar se a *Decision Tree* é de fato o melhor algoritmo para este tipo de análise.

Referências

[1] Sacramento, Gabriel, "Árvore de Decisão: Entenda este algoritmo de Machine Learning". < https://blog.somostera.com/data-science/arvores-de-decisao#:~:text=Uma%20%C3%A1rvore%20de%20decis%C3%A3o%20%C3%A9,para%20classifica%C3%A7%C3%A3o%20e%20para%20regress%C3%A3o.&text=%C3%89%20um%20algoritmo%20que%20segue,em%20novos%20n%C3%ADveis%20de%20profundidade>

[2] Santana, Felipe, "Café com Código #06: Introdução a Machine Learning com Scikit-Learn". < https://minerandodados.com.br/cafe-com-codigo-06-introducao-machine-learning-com-scikit-learn/ >

[3] Dias, Mariana, "People Analytics: o que é, benefícios e como aplicar no RH". < https://www.gupy.io/blog/people-analytics >

[4] Kunumi, "Métricas de Avaliação em Machine Learning: Classificação". < https://medium.com/kunumi/m%C3%A9tricas-de-avalia%C3%A7%C3%A3o-em-machine-learning-classifica%C3%A7%C3%A3o-49340dcdb198 >

[5] Filho, Mário, "As Métricas Mais Populares para Avaliar Modelos de Machine Learning". < https://www.mariofilho.com/as-metricas-mais-populares-para-avaliar-modelos-de-machine-learning/ >

Pontifícia Universidade Católica do Rio Grande do Sul
Pró-Reitoria de Graduação
Av. Ipiranga, 6681 - Prédio 1 - 3º. andar
Porto Alegre - RS - Brasil
Fone: (51) 3320-3500 - Fax: (51) 3339-1564
E-mail: prograd@pucrs.br
Site: www.pucrs.br