Implementação de modelos preditivos para Sistema de Recomendação integrados à Serviços de Músicas *On-demand*

Ellaynne Christine Ribeiro de Moraes Sousa* e Edilberto Silva*

*Pós-Graduação, FACSENAC-DF, Brasília-DF [Março/2018]

ellaynnechris@gmail.com; prof.edilberto.silva@gmail.com

Abstract:.*Recommendation Systems have come up with the intention of solving a problem for users of social media applications: the overload of information that is available in used services, in which there is a vast amount of content. In this work we will present a solution to the challenge to create a logic of recommending songs for a streaming service. The steps to achieve this solution were based on the CRISP-DM methodology and some predictive models were developed using two different machine learning algorithms: Gradient Boosting Algorithm (GBM) and Deep Learning. The goal is to predict which songs a user might like or dislike. In the validation tests of the predictive models, at least one Area Under Curve (AUC) of 66% was obtained.*

Resumo. Os Sistemas de Recomendação surgiram com a intenção de resolver um problema para os usuários de aplicativos de mídias sociais: a sobrecarga de informações que estão disponíveis em serviços utilizados, nos quais há uma vasta quantidade de conteúdo. Neste trabalho será apresentada uma solução para o desafio de criar uma lógica de recomendação de músicas para um serviço de *streaming*. As etapas para alcançar essa solução foram baseadas na metodologia CRISP-DM e foram desenvolvidos alguns modelos preditivos utilizando dois diferentes algoritmos de aprendizado de máquina: o *Gradient Boosting Algorithm* (GBM) e o *Deep Learning*. O objetivo é prever de quais músicas um usuário poderá gostar ou não. Nos testes de validação dos modelos preditivos, se obteve pelo menos uma *Area Under Curve* (AUC) de 66%.

Palavras-chave: Mineração de Dados, Sistema de Recomendação, Aprendizado de Máquina, *Streaming* de músicas

1. Introdução

A evolução e a facilidade do acesso à internet trouxeram muitas mudanças e, dentre elas, tornou-se comum não mais armazenar uma infinidade de dados localmente, mas sim acessá-los na *web* quando for necessário. E, assim, aconteceu com os arquivos de músicas. Não é mais comum se comprar CDs [3] ou mesmo fazer *downloads* dessas mídias, já que há serviços que disponibilizam uma infinidade de músicas para se ouvir *online*.

Nasceram, então, as plataformas de transmissão de som instantânea, tais como Spotify, Deezer, Tidal e Google Music. Também chamadas de *streamings* ou serviços *on-demand*, essas plataformas permitem ouvir músicas sem precisar efetuar o *download* dos arquivos, dependendo apenas de uma conexão com a internet para ter à mão, onde e quando quiser, um enorme acervo de conteúdo musical [10].

Com tantas novidades surgindo a todo o momento no mundo e a grande quantidade de conteúdo disponível na *web*, os usuários se depararam com uma sobrecarga de informações, da qual ficou complicado recuperar o que realmente os interessava. Então, na década de 90, surgiram os primeiros Sistemas de Recomendação (SR) para ajudar os usuários com esse problema. Através de diversos algoritmos e abordagens, os SR filtram informações baseadas no perfil do usuário e sugerem aqueles itens mais relevantes ao seu interesse [11].

Dessa forma, tornou-se necessário que os *streamings* investissem em uma forma de agradar seus usuários: os ajudando a descobrir dentre tantas opções, quais músicas mais os agradavam. Por isso, a maioria dessas plataformas utiliza uma lógica de recomendação incorporado aos seus serviços para que possam agregar valor ao produto oferecido.

Nesse artigo, serão demonstradas algumas técnicas básicas para o desenvolvimento de um SR, cuja

proposta foi apresentada em uma das principais conferências de pesquisas envolvendo busca e mineração de dados, a *ACM International Conference on Web Search and Data Mining* (WSDM). Em sua 11ª conferência, a WSDM, em parceria com o Kaggle[1], lançaram um desafio para construir um sistema de recomendação de músicas usando o conjunto de dados da KKBox. A KKBox [12] é um serviço de transmissão de música asiático, que possui mais de 10 milhões de usuários que têm à disposição mais de 20 milhões de músicas.

O objetivo deste trabalho é dar uma noção introdutória de mineração de dados aplicada à sistemas de recomendação e mostrar como implementar modelos preditivos que ajudem na sugestão de músicas em um sistema de transmissão *on-demand*.

2. Trabalhos relacionados

Para que as recomendações tenham mais chances de acerto, é necessário conhecer o cenário onde os dados estão inseridos, a fim de identificar aquelas informações que mais influenciarão na lógica por trás do sistema. Por isso, os Sistemas de Recomendação possuem uma estrutura dividida em quatro processos: (i) identificação dos usuários, (ii) coleta de dados, (iii) estratégia de recomendação e (iv) visualização das recomendações [2].

Na identificação dos usuários (i) são colhidas informações pessoais e comportamentais que podem ser relevantes; na coleta de dados (ii) são identificadas quais as preferências desses usuários. Já na etapa de estratégia da recomendação (iii), é escolhida a técnica que vai embasar o modelo de recomendação a ser desenvolvido. E, finalmente, na visualização das recomendações (iv), as sugestões de novos itens devem ser apresentadas aos usuários através de algum meio de interação como, por exemplo, e-mails, listas *online* de recomendações do serviço, etc.

A fase que requer maior esforço e empenho é a (iii), já que nela serão aplicadas e avaliadas as técnicas de criação de padrões que fornecerem um resultado mais assertivo. Existem várias estratégias utilizadas para SR, como a filtragem de informação, a personalização, a sumarização estatística, a individualização, a correlação e a mineração de dados. A mais comumente empregada nos SR é a filtragem, cujas subclassificações mais conhecidas são a filtragem colaborativa, a filtragem baseada em conteúdo e a filtragem híbrida [13].

Os recomendadores que utilizam a filtragem baseada em conteúdo analisam os itens que foram avaliados pelo usuário para lhe sugerir itens parecidos. [Pazzani, 1999 apud 11]. Já na abordagem da filtragem colaborativa, o princípio básico é identificar usuários com preferências semelhantes e desta forma apresentar os itens distintos bem avaliados a cada membro desse grupo. [Shardanand; Maes, 1995 apud 11]. E a filtragem híbrida visa uma complementação das duas abordagens anteriores de modo a corrigir falhas que existem nelas, ao mesmo tempo em que amplia as possibilidades do SR ao utilizar a lógica de duas vertentes [11].

Um bom exemplo de SR que está sempre inovando na sua forma de sugerir conteúdo aos seus usuários é a Netflix, um serviço de vídeos *on-demand*, onde podem ser encontrados filmes, séries, documentários e outros. Em janeiro de 2016, a Netflix já tinha alcançado mais de 130 países e para garantir a fidelização de seus assinantes de culturas tão variadas, investiu em novas formas de recomendação dos seus conteúdos. São utilizados vários algoritmos para realizar essa tarefa, como, por exemplo, o que leva em consideração fatores como o país e cultura nos quais os usuários estão inseridos e o que analisa as preferências pessoais, criando uma lista de recomendações única para cada telespectador [8].

Os SR podem ser inseridos em vários segmentos, de forma a agregar valor nas mais diferentes situações, como pode ser visto nos seguintes trabalhos correlatos da aplicação de recomendadores: [2], [11] e [13].

3. Metodologia CRISP-DM

Neste trabalho, que é baseado em Filtragem Colaborativa, foi utilizada a metodologia CRISP-DM (*Cross-Industry Standard Process for Data Mining*), já que ela tem o objetivo de orientar a condução de um projeto de Mineração de Dados.

Nela, são definidas seis etapas no ciclo de vida do projeto: Compreensão do Negócio, onde a meta é entender o contexto com o qual se está trabalhando e o problema que deve ser resolvido; Entendimento dos Dados, na qual requer uma avaliação e compreensão das informações disponíveis; Preparação dos Dados, onde se deve selecionar, ajustar e definir os formatos necessários dos atributos que serão utilizados; Modelagem, que é a etapa na qual serão aplicadas as técnicas de Mineração de Dados; Avaliação, que compreende a verificação da qualidade dos modelos gerados; e Desenvolvimento, que consiste na aplicação dos melhores modelos obtidos nas etapas anteriores e elaboração de um relatório apresentando os resultados alcançados [5].

[1] *O Kaggle é uma plataforma de competições voltada para aprendizado de máquina, na qual são postados desafios a serem resolvidos com modelagem preditiva e analítica dos dados disponibilizados. (Wikipédia, 2018)*

O computador utilizado para a execução deste trabalho possui um processador Intel Core i7-4500U, 8GB de memória RAM e 1TB de HD, no qual o sistema operacional utilizado é o Windows 8.1 de 64 bits. Para tratar os dados e implementar a modelagem optou-se por utilizar o RStudio[2] versão 1.1.383 integrado com o H2O[3] versão 3.16.0.4, que utiliza o Java JRE 1.8.0_141.

3.1. Compreensão do Negócio

O foco deste trabalho é voltado para a etapa de estratégia da recomendação (iii), que, como dito na seção 2, envolve a escolha da técnica que irá embasar o modelo de recomendação desenvolvido, uma vez que os dados e informações sobre os usuários já foram fornecidas pelo KKbox.

Com essas informações, o objetivo é desenvolver um modelo de recomendação para o serviço de *streaming* de músicas, cuja principal tarefa é classificar o resultado da análise dos dados em falso ou verdadeiro. Esse modelo, chamado de modelo preditivo, é um algoritmo de aprendizado de máquina supervisionado[4] que irá criar uma lógica, utilizando os atributos do conjunto de dados, para tentar prever quais itens de uma lista o usuário irá gostar, para que assim o KKbox possa sugeri-lo ao ouvinte.

3.2. Entendimento dos Dados

No Kaggle, foram disponibilizados seis arquivos (Apêndice A) em formato .CSV a serem utilizados para realizar o treinamento do algoritmo de aprendizado de máquina e realizar o teste do modelo preditivo, o qual deve ser enviado para validação na própria plataforma.

Os arquivos *train*, *songs*, *members* e *song_extra_info* têm os dados para realizar o treinamento. Além do atributo *target*, que guarda a classificação de uma música pelo usuário (*target* = 1, gostou; *target* = 0, não gostou), há atributos como gênero, nome do artista, compositor e idioma da música, e idade, país e sexo do assinante, dentre outras.

O arquivo *test* possui uma estrutura parecida com a do *train*, porém sem o atributo *target*, já que o seu valor será definido pelo algoritmo de classificação. O resultado da predição, isto é, se o usuário irá ou não gostar de uma música, deverá seguir a estrutura do exemplo disponibilizado de submissão (*sample_submission*), no qual deve conter apenas o *id* da linha e a resposta para a *target*.

Ao analisar os dados contidos nos arquivos descritos acima, foi identificado que alguns atributos possuíam *outliers*, isto é, alguns valores são discrepantes da maioria dos valores apresentados. O atributo *bd* (idade) é um exemplo disso. Além de *outliers*, haviam *missing values* (valores nulos ou em branco) também.

Para realizar o treinamento do algoritmo responsável pela criação do modelo preditivo, foram utilizadas todas os registros presentes no arquivo *train*, o qual possui mais de 7.000.000 de linhas.

3.3. Preparação dos Dados

A primeira tarefa foi dividir (*split*), em linhas individuais, as músicas que possuem mais de uma classificação de gênero. Em seguida, os valores das variáveis *source_system_tab, source_screen_name, source_type* e *gender* foram convertidos para o tipo inteiro e a *target* foi convertida para o tipo categórico (*factor*), com o objetivo de melhorar a performance na execução do treinamento do algoritmo.

Visando uma melhor organização das informações, foram criadas faixas de idade, utilizando um critério aleatório de variância, para o atributo *bd* (idade), conforme especificado na Tabela 1.

Tabela 1: Tabela de faixas de idade

ID	Faixa
1	Menor que 0 (NA)
2	Entre 0 e 14
3	Entre 15 e 18
4	Entre 19 e 22
5	Entre 23 e 26
6	Entre 27 e 30
7	Entre 31 e 34
8	Entre 35 e 38
9	Entre 39 e 42
10	Entre 43 e 46
11	Entre 47 e 50
12	Maior que 51

Fonte: os autores

Logo após essas conversões de tipos dos atributos, foi realizada a unificação (*merge*) dos arquivos *train*, *songs* e *members* em um único *dataset*, no qual foram mantidos apenas alguns atributos devido à limitação da máquina utilizada para realizar o processamento dos dados. O *dataset* utilizado para realizar o treinamento

[2] *O RStudio (www.rstudio.com) é um ambiente de desenvolvimento integrado (IDE) para a leitura da linguagem R, uma linguagem de programação para gráficos e cálculos estatísticos.*

[3] *O H2O (www.h2o.ai) é uma plataforma voltada para aprendizagem de máquina que pode ser integrada a diversos outros ambientes.*

[4] *O aprendizado de máquina supervisionado utiliza os dados de um conjunto de observações ou exemplos de classes já definidas para realizar o treinamento do algoritmo, com o qual é possível encontrar uma hipótese que classifique novas situações em algum grupo de classes já estabelecidas [Brink; Richards, 2014 apud 1].*

do modelo preditivo ficou composto dos atributos que estão listados na Tabela 2.

Todos esses ajustes também foram aplicados no *dataset* de teste, já que as suas variáveis devem ser as mesmas do treinamento, inclusive seus tipos.

Ademais, foi especificado na função de importação dos arquivos para o RStudio que os valores em branco (*missing values*) deveriam assumir um valor indicativo: 'NA'. Apesar de ser uma boa prática omitir os valores inválidos, optou-se por não fazê-lo, pois isso haveria uma baixa na quantidade de registros da base de treinamento.

Tabela 2: *Dataset* de treinamento

Atributo	Tipo
source_system_tab	numérico
source_screen_name	numérico
source_type	numérico
song_length	numérico
genre_ids	numérico
artist_name	numérico
language	numérico
city	numérico
bd	numérico
gender	numérico
target	categórica

Fonte: os autores

3.4. Modelagem

Durante a etapa de modelagem foram realizados inúmeros testes e análises dos resultados dos mesmos. Aqui, serão apresentados o melhor modelo encontrado para dois algoritmos utilizados: o GBM (*Gradient Boosting Algorithm*) e o *Deep Learning*, do pacote do H2O[5].

Os parâmetros passados para os algoritmos foram definidos manualmente e ajustados com base em tentativas e erros, devido ao curto prazo entre a data de ingresso no desafio e a data final da competição. Os parâmetros utilizados foram escolhidos com base no controle de memória alocada durante o processamento, já que o computador apresentou estouro de memória muitas vezes.

Em todos os modelos, utilizou-se o recurso de *cross-validation* (também conhecido como *k-folds* em algumas literaturas), que consiste em dividir o conjunto de dados e realizar o treinamento e validação com essas partes, alternando entre elas, aumentando, assim, as possibilidades de aprendizado do algoritmo.

[5] *Para maiores informações sobre os algoritmos e demais funções, consultar a documentação do H2O (http://docs.h2o.ai/h2o/latest-stable/h2o-docs/index.html).*

3.4.1. GBM

O GBM é um algoritmo de aprendizagem supervisionada baseado em árvores de decisão utilizado em problemas de classificação e regressão. Segundo [4], uma árvore de decisão divide o universo do problema sucessivamente em subconjuntos (criando nós contendo os testes para cada situação) até que todos estes estejam classificados em uma classe ou até que uma classe não possua requisitos para ser novamente dividida (neste caso, gerando uma folha contendo a classe mais significativa).

Muitos ajustes podem ser realizados na forma como o algoritmo irá criar as árvores de decisão e assim obter um melhor resultado. Há vários parâmetros de entrada que definirão propriedades de "podagem" das árvores, como os descritos na Tabela 3, mas que se não forem definidos, assumirão valores *default*, com exceção dos três primeiros parâmetros, que são obrigatórios. Assim, o primeiro modelo encontrado, utilizando o GBM, foi definido com as propriedades listadas na Tabela 3.

Tabela 3: Valores passados o gbm

Parâmetro	Definição	Valor
x*	Vetor que contém os nomes das variáveis independentes do modelo	Variáveis mostradas na Tabela 2
y*	Nome do atributo alvo do modelo	Target
Training_frame*	Um objeto de dados do H2O (H2OFrame) que contém as variáveis do modelo	*Dataset* da Tabela 2
Nfolds	Partes em que o *dataset* de treinamento será divido para o *cross-validation*	5
Ntrees	Define o número de árvores criadas	200
Max_depth	Número máximo da profundidade das árvores.	50
Balance_classes	Faz o balanceamento de classes caso os dados estejam com subamostragem ou desbalanceados.	True

* Parâmetros obrigatórios

Fonte: os autores

Na figura 1, pode-se visualizar o gráfico, gerado pelo H2O, que mostra a importância das variáveis utilizadas na criação do modelo GBM. Nela, a variável de maior importância é a *source_type*, através da qual a árvore de decisões deve iniciar a validação dos dados.

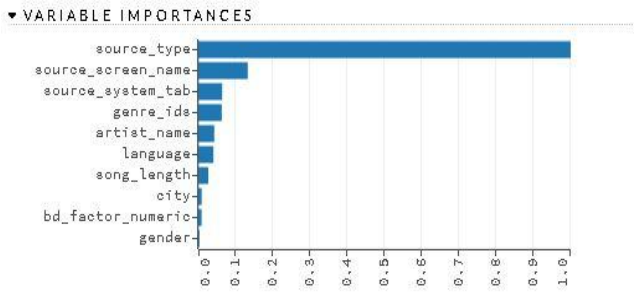

Figura 1: Importância das variáveis do modelo GBM

3.4.2. Deep Learning

O *Deep Learning* é baseado em Redes Neurais Artificiais (RNA), que são algoritmos inspirados nas ligações neurológicas que constituem o cérebro dos seres vivos. Muitos autores consideram as redes neurais como métodos estatísticos paramétricos já que o treino de uma rede neural significa encontrar valores apropriados para pesos (parâmetros) [6].

Assim como no GBM, o algoritmo *Deep Learning* também aceita muitos parâmetros de entrada e o modelo foi parametrizado como mostra a Tabela 4.

E na figura 2, é mostrada a importância das variáveis do modelo *Deep Learning*, na qual a variável de maior importância também é a *source_type*.

Tabela 4: Valores passados para o *deep learning*

Parâmetro	Definição	Valor
x*	Vetor que contém os nomes das variáveis independentes do modelo	Variáveis mostradas na Tabela 2
y*	Nome do atributo alvo do modelo	Target
Training_frame *	Um objeto de dados do H2O (H2OFrame) que contém as variáveis do modelo	*Dataset* da Tabela 2
Nfolds	Partes em que o *dataset* de treinamento será divido para o *cross-validation*	5
Epoch	Especifica o número de vezes para iterar (transmitir) o conjunto de dados	100
Hidden	Matriz que especifica os tamanhos da camada oculta da RNA	c(100,100)
Seed	Especifica a lógica de geração de números aleatórios	"1122"

* Parâmetros obrigatórios

Fonte: dos autores

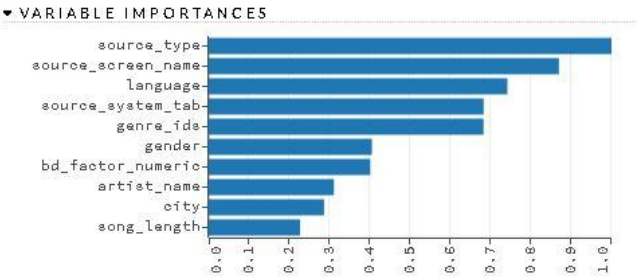

Figura 1: Importância das variáveis do modelo GBM

3.5. Avaliação e Desenvolvimento

Neste tópico serão avaliados os modelos desenvolvidos, bem como apresentados os resultados obtidos com eles.

Há várias métricas para realizar a avaliação de um modelo preditivo, tais como a Acurácia, a Curva ROC e a AUC. [Mikhail; Ackermann, 1976 apud 7] descreve que a acurácia reflete a proximidade de uma grandeza

estatística ao valor do parâmetro para o qual ela foi estimada. Em outras palavras, ela mede a "distância" entre os resultados obtidos e os valores esperados.

A curva ou gráfico ROC (*Receiver Operating Characteristic*) é baseado na probabilidade de identificação de verdadeiros positivos (eixo y) e de falsos negativos (eixo x) [9]. Para cada predição realizada pelo modelo, é realizado um cálculo de porcentagem de verdadeiros e falsos negativos em cima da matriz de confusão[6] para formar os pontos do gráfico. Quanto mais a curva se aproximar do eixo y e do topo (ponto 1), maior a taxa de probabilidade de acerto.

A partir de uma curva ROC, é calculada a AUC (*Area Under Curve*), cuja área abaixo da linha desenhada é uma fração da área de um quadrado de lado um e "é numericamente igual à probabilidade de dados dois exemplos de classes distintas, o exemplo positivo seja ordenado primeiramente que um exemplo negativo" [Egan, 1975 apud 1].

3.5.1. GBM

O modelo criado a partir do algoritmo GBM e validado a partir do recurso de *cross-validation* obteve uma acurácia de 58% (a acurácia e demais métricas podem ser vistos no Apêndice B), o que quer dizer que o modelo acertou 58% dos casos. A sua curva ROC pode ser visualizada na Figura 1 (linha azul), na qual também já é mostrada a AUC, que foi igual à 0.671679.

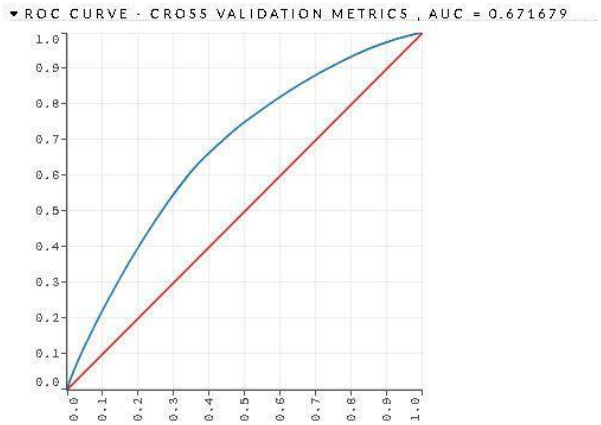

Figura 3: Curva ROC do modelo GBM

Ao realizar a predição dos dados do *dataset* de teste, o resultado da submissão apurada pelo Kaggle que gerou um *score*[7] de 0,57.

3.5.2. Deep Learning

Para o modelo *Deep Learning* a acurácia foi de 57% (Apêndice C) e a sua curva ROC e AUC (66%) são mostrados na Figura 2.

Para os dados preditos com esse modelo, o Kaggle gerou um *score* de 0.53.

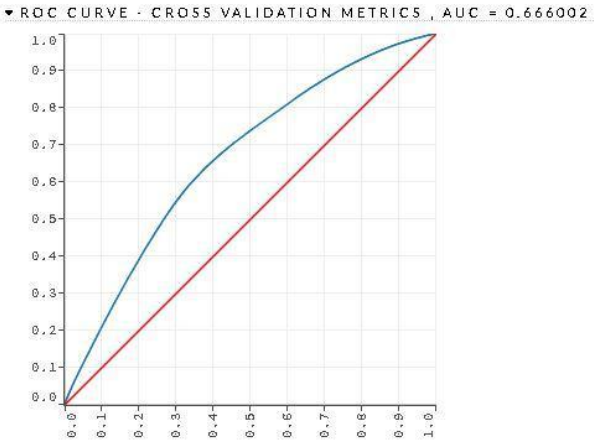

Figura 4: Curva ROC do modelo *Deep Learning*

4. Conclusão

No trabalho aqui mostrado foram definidos conceitos de Sistemas de Recomendação (SR) e como iniciar seu desenvolvimento utilizando técnicas de Aprendizado de Máquina com a IDE RStudio integrada ao H2O. Foram citados dois algoritmos de classificação, um baseado em árvores de decisão (GBM) e outro em redes neurais artificiais (*Deep Learning*), utilizados para criar modelos preditivos para classificar a probabilidade de um usuário gostar ou não de uma música presente na lista fornecida pelo KKBox.

A etapa de Preparação dos Dados e Modelagem, da metodologia CRISP-DM, foram repetidas diversas vezes até chegar ao que aqui foi apresentado. Na preparação de dados foram aplicadas apenas algumas técnicas de ajuste dos dados, já que a máquina utilizada não suportou processamento de mais dados. Poder-se-ia ter convertido os valores do atributo *artist_name* em *hash* MD5 (uma forma de criptografia) para facilitar o processamento do algoritmo, por exemplo.

Na modelagem aconteceu o mesmo: algumas parametrizações não puderam ser aplicadas ao

[6] *Em uma matriz de confusão pode ser visualizado o cruzamento dos valores reais com os valores de saída do modelo de predição. Nela são vistos os quantitativos de verdadeiros positivos, falsos positivos, falso negativo e verdadeiro negativo [9].*

[7] *O score gerado pelo Kaggle é baseado na AUC de 50% dos exemplos disponibilizados no arquivo de teste e os outros 50% são de outros exemplos desconhecidos pelos competidores.*

algoritmo ou os valores dos parâmetros não puderam ser testados com uma maior variância, pois havia estouro de memória da máquina. Com isso, pode-se deduzir que se for utilizada uma máquina com melhores recursos, há como alcançar resultados melhores na predição.

Embora os algoritmos utilizados possuam metodologias diferentes de classificação dos dados, ambos obtiveram a mesma variável (*source_type*) como a mais relevante na criação do modelo preditivo, bem como valores bem próximos para a AUC (*Area Under Curve*): o GBM gerou uma área de 67% e o Deep Learning, 66% nas validações locais.

Os *scores* apresentados pelo Kaggle, que também são baseados na AUC, geraram os valores de 57% e 53% para os resultados classificados pelos algoritmos GBM e *Deep Learning*, respectivamente.

Mesmo não alcançando um bom percentual de AUC, uma métrica chamada *recall*[8], que pode ser visualizada nos apêndices B e C, revelou que a classificação de verdadeiros negativos teve um acerto maior que 90% em ambos os modelos, o que significa que o modelo acertou bastante as músicas que não irão agradar determinado usuário. Essa é uma informação importante a ser utilizada na hora de selecionar músicas a serem recomendadas ao usuário.

Por fim, o trabalho desenvolvido foi de muita importância para um aprendizado prático inicial sobre mineração de dados, aprendizado de máquina e sistemas de recomendação, assuntos esses que estão ganhando lugar no mercado de Tecnologia da Informação (TI), contribuindo com a filtragem de informações e busca por conhecimentos que agregam valores às empresas.

5. Referências

[1]. CARVALHO, Hialo Muniz. **Aprendizado de Máquina voltado para Mineração de Dados: Árvores de Decisão**. Brasília-DF, 2014.

[2]. CARVALHO, L. A. M. C; NUNES, M. A. S. N. **Uso da Personalidade na modelagem de usuário e suas aplicações em Sistemas de Recomendação: survey 2011**. São Cristóvão-SE, 2013.

[3]. Época Negócios. 2018. Disponível em: <https://epocanegocios.globo.com/Tecnologia/noticia/2017/05/epoca-negocios-consumo-de-musica-por-streaming-volta-a-crescer.html>. Acessado em: 24/02/2018.

[4]. FONSECA, J. M. M. R. **Indução de Árvores de Decisão: HistClass – Proposta de um algoritmo não paramétrico**. Lisboa, 1994.

[5]. GOLDSCHMIDT, Ronaldo; PASSOS, Emmanuel; BEZERRA, Eduardo. **Data Mining: conceitos, técnicas, algoritmos, orientações e aplicações**. Editora Elsevier. Rio de Janeiro, 2015

[6]. MONARD, Maria Carolina. et al. **Uma Introdução ao Aprendizado Simbólico de Máquina por Exemplos**. 1997

[7]. MONICO, João Francisco Galera; PÓZ, Aluir Porfírio Dal; GALO, Maurício; SANTOS, Marcelo **Carvalho dos; OLIVEIRA, Leonardo Castro de. Acurácia e precisão: revendo os conceitos de forma acurada**. Curitiba-PR, 2009.

[8]. Netflix Media Center. **Recomendações: uma abordagem global.** 2016. Disponível em: <https://media.netflix.com/pt_br/company-blog/a-global-approach-to-ecommendations>. Acessado em: 13 de fev. 2018.

[9]. PRATI, R. C.; BATISTA, G. E. A. P. A.; MONARD, M. C. **Curvas ROC para avaliação de classificadores**.

[10]. SANTOS, Mylena Ceribelle Gadelha; RAMOS, Rebecca Costa; RIOS, José Riverson Araújo Cysne. **Aplicativos de música: o Spotify, as mudanças no mercado fonográfico e os filtros-bolha**. Fortaleza-CE, 2016.

[11]. SILVA, Rafael Glauber Nascimento e. **Sistema de Recomendação baseado em conteúdo textual: avaliação e comparação**. Salvador-BA, 2014.

[12]. Wikipédia. 2018. Disponível em: <https://en.wikipedia.org/wiki/KKBox>. Acessado em: 11 de fev. 2018.

[13]. ZANONA, Arthur Felipe. **Sistema de recomendação de roteiros turísticos para plataforma móvel usando localização global**. São José - SC, 2013.

[8] O *recall* se refere à especificidade do modelo preditivo, cujo o objetivo é mensurar a taxa de acerto na classificação dos verdadeiros negativos.

Apêndice A
Arquivos de Dados KKBox

Train	
msno	ID do usuário
song_id	ID da música
source_system_tab	o nome da guia onde o evento foi acionado. As abas do sistema são usadas para categorizar as funções de aplicativos móveis do KKBOX. Por exemplo, a guia *my library* contém funções para manipular o armazenamento local, e a guia *search* contém funções relacionadas à pesquisa.
source_screen_name	nome do *layout* que um usuário vê.
source_type	um ponto de entrada que um usuário reproduz pela primeira vez em aplicativos móveis. Um ponto de entrada pode ser *album*, *online-playlist*, *song*, etc.
target	esta é o atributo alvo. *target*=1significa que o usuário curtiu a música. Caso contrário, *target*=0.

Test	
id	ID da linha (será usado para submissão)
msno	ID do usuário
song_id	ID da música
source_system_tab	o nome da guia onde o evento foi acionado. As abas do sistema são usadas para categorizar as funções de aplicativos móveis do KKBOX. Por exemplo, a guia *my library* contém funções para manipular o armazenamento local, e a guia *search* contém funções relacionadas à pesquisa.
source_screen_name	nome do *layout* que um usuário vê.
source_type	um ponto de entrada que um usuário reproduz pela primeira vez em aplicativos móveis. Um ponto de entrada pode ser *album*, *online*-playlist, *song* etc.

Songs	
song_id	ID da música
song_length	comprimento da música em ms
genre_ids	categoria de gênero. Algumas músicas têm vários gêneros e são separadas por "|"
artist_name	nome do artista
composer	compositor
lyricist	escritor da letra da música
language	idioma da música

Members	
msno	ID do usuário
city	cidade do usuário
bd	Idade do usuário. Nota: esta coluna possui valores *outliers*, use seu julgamento.
gender	gênero
registered_via	método de registro
registration_init_time	data de registro
expiration_date	data de cancelamento da conta

Song Extra Info	
song_id	ID da música
song_name	nome da música
isrc	código de gravação padrão internacional

Sample Submission	
id	idêntico ao ID do arquivo *test*
target	resultado da predição do *test*

Apêndice B
Métricas do modelo GBM

▾ OUTPUT · CROSS-VALIDATION METRICS SUMMARY

	mean	sd	cv_1_valid	cv_2_valid	cv_3_valid	cv_4_valid	cv_5_valid
accuracy	0.58033055	0.0023347843	0.5761629	0.5841789	0.5839128	0.5772539	0.5801442
auc	0.6716835	2.5147927E-4	0.6711025	0.6717748	0.6718794	0.67151016	0.67215055
err	0.41966945	0.0023347843	0.4238371	0.4158211	0.4160872	0.4227461	0.41985574
err_count	652289.2	3786.3394	659040.0	645934.0	646840.0	657549.0	652083.0
f0point5	0.5970902	0.001324629	0.59512836	0.5995639	0.59889877	0.5950313	0.59682846
f1	0.6854142	2.801919E-4	0.6859594	0.6857781	0.6851226	0.68490577	0.685305
f2	0.80443	0.0024498329	0.80951023	0.8009507	0.8003495	0.80676013	0.80457944
lift_top_group	1.5480651	0.006099857	1.5322062	1.555204	1.5463442	1.5557499	1.550821
logloss	0.647065	1.2386618E-4	0.6473707	0.64699996	0.64694387	0.6471353	0.64687496
max_per_class_error	0.75245947	0.009929446	0.7720223	0.7375329	0.7363669	0.7634715	0.7529038
mcc	0.2101953	0.0026197506	0.20520386	0.21406758	0.21445774	0.20694833	0.210299
mean_per_class_accuracy	0.5786548	0.0023848298	0.57397753	0.58220047	0.5825261	0.57592744	0.57864267
mean_per_class_error	0.42134514	0.0023848298	0.42602244	0.41779953	0.41747388	0.42407256	0.4213573
mse	0.22767654	5.9463826E-5	0.22782375	0.22764732	0.22761899	0.22770832	0.22758427
precision	0.5498577	0.0018639421	0.5468541	0.5531994	0.55254	0.54716474	0.54953027
r2	0.08926966	2.4139148E-4	0.088668644	0.08937584	0.08950683	0.089152776	0.08964422
recall	0.9097692	0.0051612225	0.9199774	0.90193385	0.90141916	0.9153264	0.91018915
rmse	0.4771546	6.230656E-5	0.47730887	0.477124	0.47709432	0.47718793	0.47705793
specificity	0.24754052	0.009929446	0.22797771	0.26246712	0.26363307	0.23652849	0.2470962

Apêndice C
Métricas do modelo *Deep Learning*

▾ OUTPUT · CROSS-VALIDATION METRICS SUMMARY

	mean	sd	cv_1_valid	cv_2_valid	cv_3_valid	cv_4_valid	cv_5_valid
accuracy	0.57438016	0.0013839335	0.5734066	0.57694316	0.5737973	0.57155424	0.57619953
auc	0.66617715	6.145864E-4	0.6665481	0.6648937	0.66755015	0.6659236	0.66597027
err	0.42561984	0.0013839335	0.42659342	0.42305684	0.42620274	0.4284458	0.42380044
err_count	661535.2	2013.3701	662445.0	658081.0	662484.0	665911.0	658755.0
f0point5	0.59371966	6.689694E-4	0.59312713	0.5949127	0.5934659	0.5924112	0.5946811
f1	0.6847744	2.5461777E-4	0.68523246	0.6844204	0.6851937	0.6845257	0.6844996
f2	0.8088264	0.0016741479	0.81120205	0.8056318	0.81046087	0.81056035	0.8062768
lift_top_group	1.4867254	0.014453222	1.5016384	1.5003242	1.5078827	1.4613893	1.4623927
logloss	0.64894545	4.2160368E-4	0.6485179	0.64942145	0.6480626	0.64970946	0.6490158
max_per_class_error	0.77469504	0.0062590274	0.78087115	0.7625804	0.7789331	0.78522605	0.7658643
mcc	0.2021335	0.0019031456	0.20281897	0.20496628	0.20239106	0.19704166	0.20344953
mean_per_class_accuracy	0.5726214	0.0015144722	0.57181764	0.57545197	0.57200843	0.5693745	0.57445437
mean_per_class_error	0.42737862	0.0015144722	0.42818236	0.42454803	0.42799157	0.43062553	0.4255456
mse	0.22856154	1.8767559E-4	0.22836636	0.22877915	0.22816727	0.2288927	0.22860219
precision	0.54537547	9.926029E-4	0.54434824	0.54720426	0.5448403	0.54364055	0.546844
r2	0.08573013	7.5249316E-4	0.08651599	0.084865585	0.08730724	0.084394604	0.085567206
recall	0.9199378	0.0033847594	0.9245064	0.9134844	0.92294997	0.92397505	0.91477305
rmse	0.47808102	1.9629355E-4	0.47787693	0.47830865	0.47766858	0.47842732	0.4781236
specificity	0.22530499	0.0062590274	0.21912883	0.23741958	0.22106689	0.21477392	0.2341357