UNIVERSIDADE FEDERAL DE SANTA CATARINA - UFSC
SISTEMAS DE INFORMAÇÃO - INE - CTC
DATA MINING - INE 5644

Trabalho prático de Data Mining

Entrega 3

Bruno Eduardo D'Angelo de Oliveira
Eduardo Demeneck

FLORIANÓPOLIS, 2016

1. Revendo os algoritmos selecionados anteriormente

Anteriormente foi decidido que o algoritmo *XGBoost* seria utilizado para se realizar a mineração de dados final para este trabalho. Contudo, com um pouco mais de atenção aos detalhes, percebeu-se que era necessário uma comparação com mais algoritmos, além do *Random Forests*, para se chegar a uma decisão satisfatória.

Ademais viu-se que os parâmetros selecionados para a mineração também necessitavam de uma revisão, pois especulava-se que os resultados obtidos deixariam a desejar, pelo fato das soluções de classificação não estarem sendo utilizadas em sua total capacidade.

Nesta seção tanto os algoritmos, quanto parâmetros para sua execução, serão revistos e comparados novamente.

a. XGBoost

O nome deste algoritmo é, em realidade, uma abreviação de *Extreme Gradient Boosting*, onde o termo *Gradient Boosting* se refere a técnica utilizada. Este algoritmo ganhou notoriedade por ter vencido o *Higgs Machine Learning Challenge* (KÉGL et al., 2014), e por sempre ter um bom resultado em diversas implementações em outras competições.

O *XGBoost* produz modelos preditivos através da combinação de diversos modelos mais fracos, os quais normalmente são árvores de decisão. Os modelos são criados em estágios, similar ao que ocorre em outros algoritmos deste tipo, e depois generalizados. Mais informações sobre a técnica de *Gradient Boosting* podem ser encontradas no trabalho do autor Friedman (2002), além de que é possível compreender melhor o *XGBoost* ao se ler o artigo do estudo que o criou (CHEN; GUESTRIN, 2016).

b. Random Forests

Random Forests é um conjunto de métodos para, principalmente, solucionar-se problemas de classificação e regressão. Esta técnica opera criando diversas árvores de decisão aleatórias e gerando as diversas saídas de cada árvore individual. Este algoritmo visa contrabalancear a tendência natural ao *overfitting* que as árvores de decisão apresentam. Dessa maneira a árvore com a menor variância é escolhida como a vencedora e é utilizada para predições.

A versão do algoritmo, escolhida para implementação neste trabalho, é discutida por (BREIMAN, 2001) em seu artigo, o qual visa uma melhora nas técnicas de *Random Forests* propostas inicialmente.

c. CART

O CART (Classification and Regression Trees) é um algoritmo de classificação baseado em árvores de decisão. Ele é muito similar ao C4.5 (QUINLAN, 1993), mas difere no fato de suportar classes numéricas e por não computar conjuntos de regras. Escolhemos este algoritmo pois, devido a simplicidade de uso, utilizamos a biblioteca scikit-learn (SCIKIT... 2016) e este é o algoritmo utilizado pela biblioteca para implementar árvores de decisão.

d. Revisão dos parâmetros de validação

Após melhor considerar os parâmetros propostos anteriormente, para a validação cruzada dos modelos, decidiu-se que era necessário mudar os valores. Assim sendo os valores de 1000 iterações e 10 partições dos dados (ao invés das 5 partições propostas em primeiro momento). Todos os modelos serão validados com estes parâmetros, e os resultados serão utilizados para selecionar o melhor entre os mesmos.

e. Comparação e seleção de algoritmos

Com os novos valores de entrada para a classificação e validação demos início ao processo de criação e treinamento de modelos, e depois comparamos seu desempenho (seguindo os parâmetros discutidos para tal na seção anterior). Abaixo pode se observar os resultados da validação cruzada de cada modelo gerado.

Algoritmo de classificação	**Precisão**	**Desvio padrão**
CART	58.17%	0.83%
Random Forest	61.22%	0.99%
XGBoost	64.91%	0.97%

Tabela 1: Resultados da validação cruzada aplicada.

Sendo assim o *XGBoost* foi escolhido para a mineração final, pois apresentou o melhor resultado nas validações. Porém é interessante estudarmos as diferenças entre importância de variáveis em cada algoritmo, pois pode se extrair alguma idéia para colunas novas, ou até mesmo se optar por retirar variáveis da mineração. Abaixo estão os gráficos de importância de cada algoritmo.

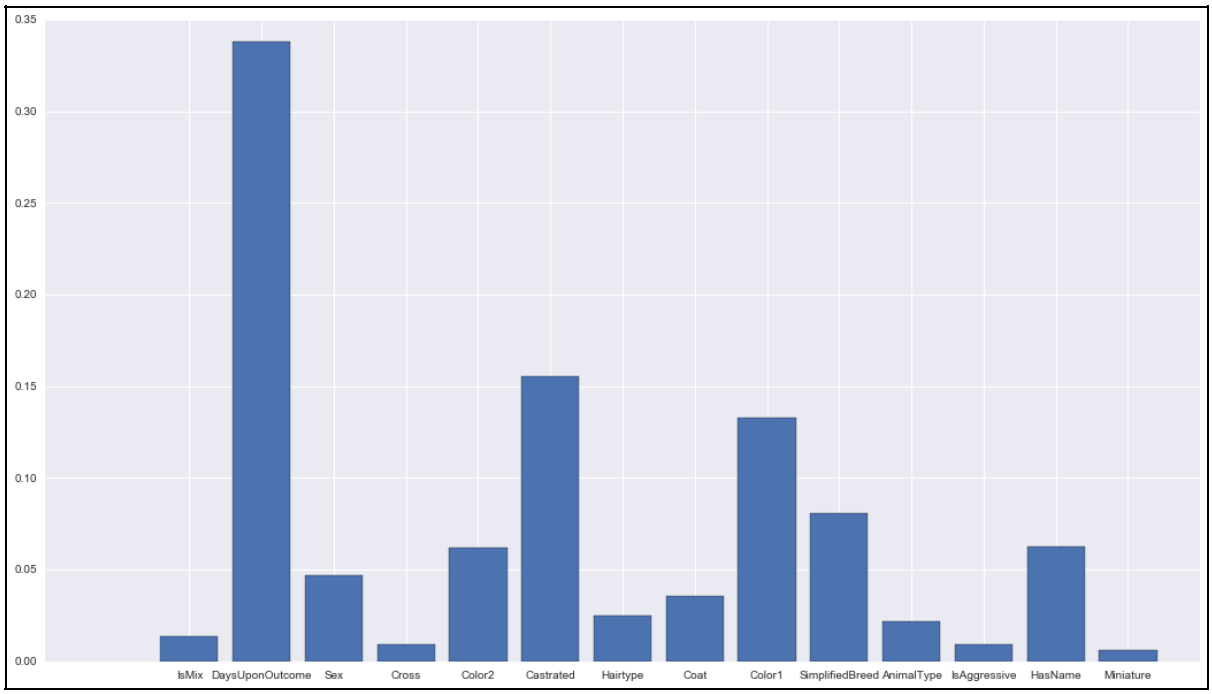

Figura 1: Gráfico de importância do algoritmo *Random Forest*.

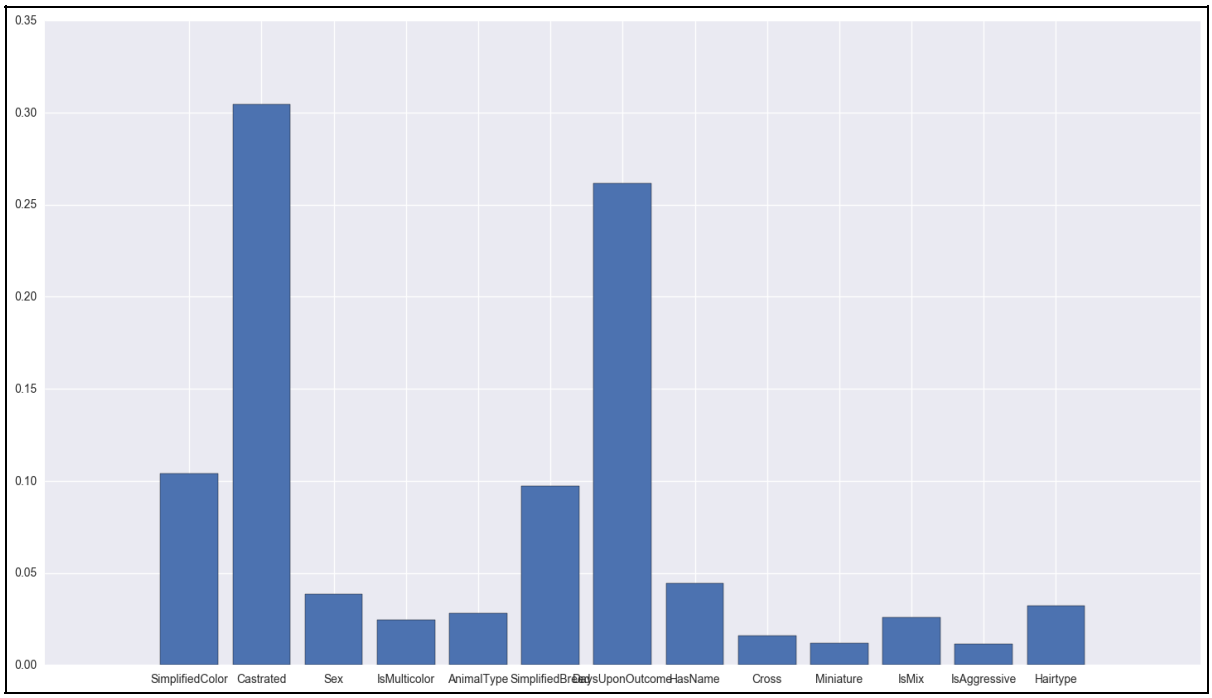

Figura 2: Gráfico de importância do algoritmo *CART*.

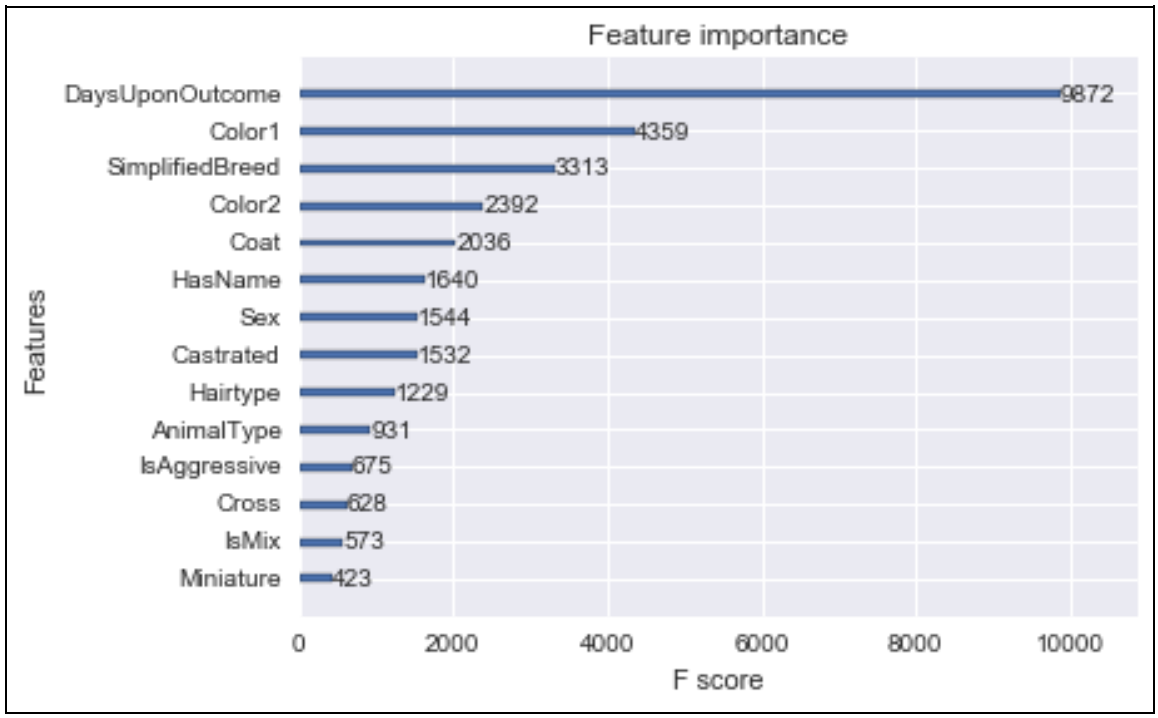

Figura 3: Gráfico de importância do algoritmo *XGBoost*.

Ao estudar estes gráficos é possível notar que os algoritmos priorizaram variáveis diferentes, contudo concordam que a idade, do animal em questão, detêm o maior peso para a classificação neste estudo. Este fato se deve, muito provavelmente, a propensão natural das pessoas a escolherem animais mais novos, enquanto animais mais velhos enfrentam dificuldades ao tentar achar um lar.

De maneira não surpreendente a raça dos animais também afeta de maneira significativa seu destino. De posse do conhecimento do peso que esta variável apresenta, talvez possa se realizar mais alguma adição aos dados baseada nessa coluna, como as raças mais populares nos Estados Unidos.

Contrariamente a última observação, a cor da pelagem dos animais parece apresentar um peso mais significativo do que o esperado. Testes revelaram que ao livrar-se dessa variável os algoritmos tendem a perder precisão, mas talvez um estudo, e tratamento, mais minucioso deste atributo seja necessário, dependendo dos resultados finais da mineração.

2. Descrição do processo de mineração

O processo de mineração se iniciou com a criação do modelo *XGBoost* a ser utilizado, para isso foram usados os parâmetros base, salvo pela taxa de aprendizado e o método de avaliação do modelo, que foram inicializados com os valores de, respectivamente, 5% e *logarithmic loss*.

Também foi decidido que se deveria calcular as probabilidades de cada classe, ao invés do método comum de se tentar inferir com 100% de certeza apenas uma classe. Essa decisão foi tomada em vista da nota de avaliação da competição ser dada pela função

logarithmic loss (o que é comum em competições do Kaggle). O que ocorre é que este método de análise é muito severo com predições de 100% de certeza (apenas uma classe prevista) que estejam enganadas, fazendo com que calcular todas as probabilidades seja mais atraente para este cenário.

Durante todo o processo de mineração foram realizadas diversas tentativas de melhorar o resultado final, tanto por ajustes nos parâmetros de entrada, quanto experimentando retirar certas variáveis que não apresentavam significância elevada nos gráficos gerados (figura 3). Contudo abandonar variáveis se provou infrutífero, reduzindo a eficácia do modelo em até 12%, por fim decidiu-se que era mais interessante realizar a previsão com todos os dados disponíveis até o momento, já que, de qualquer maneira, temos poucos dados disponíveis para trabalhar.

3. Resultados obtidos

Após ser gerado o documento CSV com as probabilidades de cada animal, foi gerada matriz de confusão dos dados de treinamento. Isso foi realizado para se poder interpretar de uma maneira melhor o funcionamento e eficácia do modelo *XGBoost* criado.

No gráfico em questão cada possível desfecho foi plotado, com o eixo X indicando o ID do animal e o eixo Y indicando a probabilidade prevista, ademais as classes reais e antecipadas são visíveis

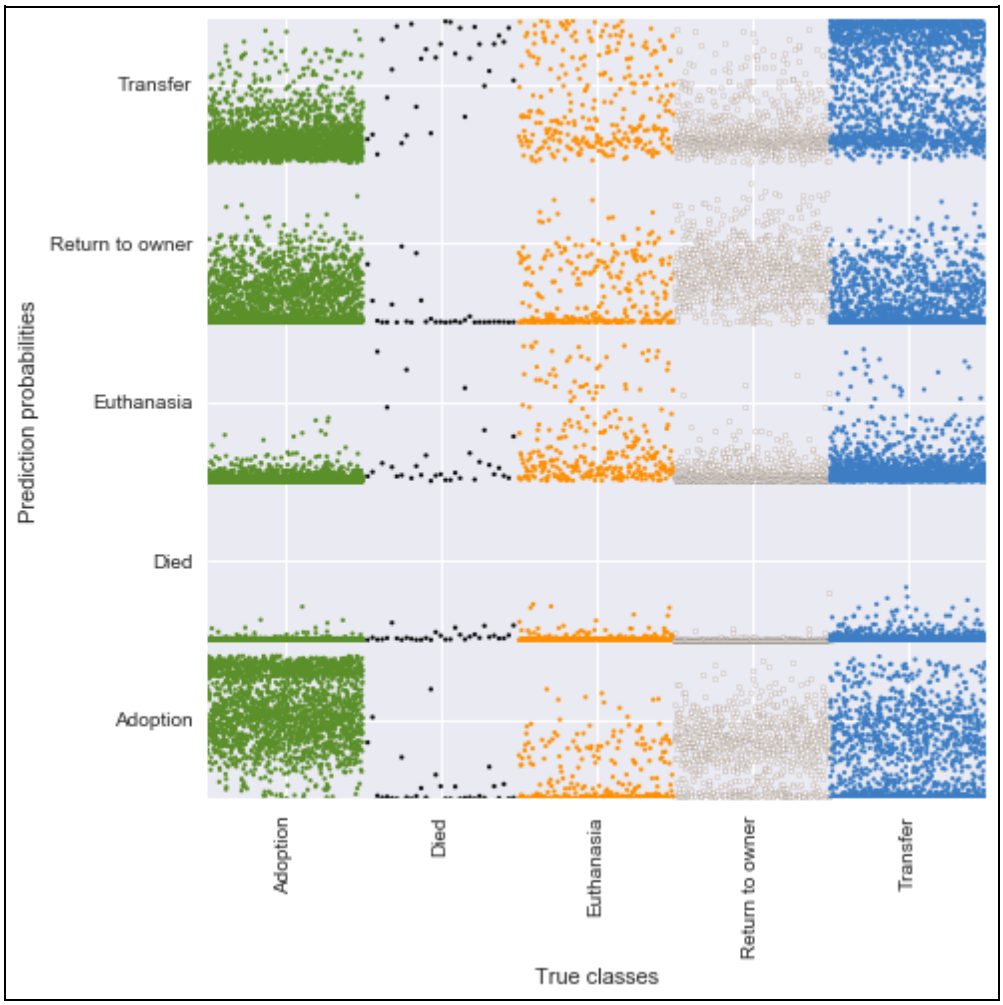

Figura 4: Matriz de confusão das predições de probabilidade.

No gráfico acima fica evidente a dificuldade em distinguir adoções de transferências, bem como o fato de que cães que faleceram (naturalmente ou por eutanásia) não estão sendo classificados corretamente. Em contrapartida o modelo consegue predizer com certa precisão casos em que o animal é retornado ao seu dono.

Quando enviado para avaliação na plataforma, o arquivo com as predições recebeu uma nota de 0.83 (mantendo em mente que notas menores são melhores), o que o deixa entre os 45% melhores modelos preditivos. Abaixo pode-se ter uma noção do resultado final do arquivo CSV que foi submetido para avaliação.

	A	B	C	D	E	F
1	ID	Adoption	Died	Euthanasia	Return_to_owner	Transfer
2	1	0.0641491786	0.0031673198	0.0416796543	0.2447276413	0.6462761760
3	2	0.4779222012	0.0009778949	0.0993446782	0.3042537272	0.1175014526
4	3	0.4655559063	0.0028254564	0.0182368737	0.1228968129	0.3904849887
5	4	0.1168542653	0.0007213215	0.0196369365	0.1598293781	0.7029580474
6	5	0.4799399972	0.0004262954	0.0038278915	0.3714353442	0.1443704218
7	6	0.3643431962	0.0009971960	0.0157012325	0.4456235170	0.1733348966
8	7	0.4618112445	0.0033249089	0.2125054449	0.1255601496	0.1967982352
9	8	0.7327679992	0.0021206518	0.0033746550	0.0490404703	0.2126962245

Figura 5: Parte do arquivo submetido para avaliação.

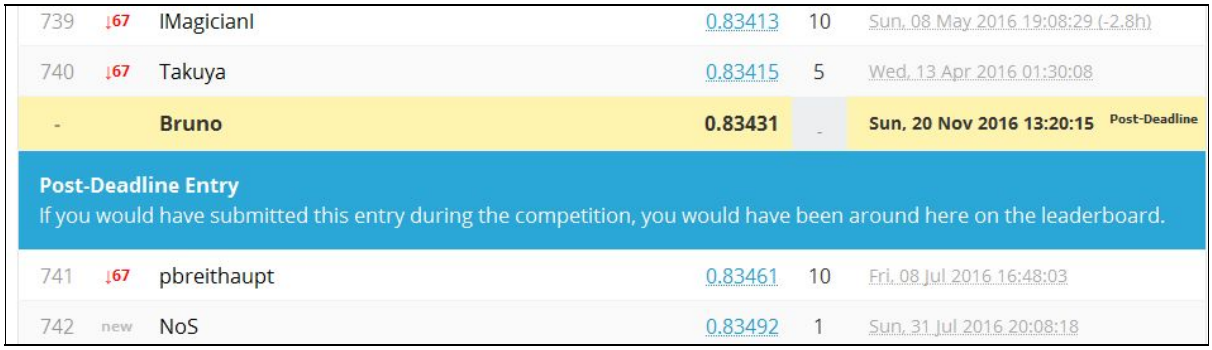

Figura 6: Resultado da submissão do arquivo com as predições.

a. Justificativa para os resultados encontrados

Apesar dos resultados obtidos não serem ruins, é possível refinar o pré-processamento de algumas variáveis deste estudo. Atributos como a raça poderiam ter sido trabalhados de forma a se criar uma coluna contendo a popularidade de cada raça nos Estados Unidos, além do que já foi realizado neste trabalho.

Os problemas encontrados com o modelo, como a dificuldade em discernir entre adoções e transferências e classificar corretamente animais que morreram, vem do fato de terem sido disponibilizados poucos dados, tanto em quantidade de variáveis quanto em número de registros, o que dificulta a geração de um modelo mais confiável. Por outro lado essas adversidades poderiam ser remediadas ao se utilizar as colunas *OutcomeSubtype* e *DateTime* (é possível usar a data para se distinguir transferências das demais situações, pois estas ocorrem em massa em um dia, e não aos poucos durante o decorrer do tempo), mas as mesmas continham dados que não estariam disponíveis em situações reais de uso do modelo de classificação, por isso as variáveis foram descartadas em etapas prévias do projeto.

Além disso algumas equipes no Kaggle decidiram ignorar o arquivo fornecido na competição e baixar os dados diretamente do site do abrigo de animais, o que na nossa visão subverte o propósito da competição e, por consequência, deste trabalho. Sendo assim optamos por não seguir essa estratégia, mesmo que os resultados possam ser melhorados dessa maneira.

Também é válido lembrar que diversos modelos nesta competição foram baseados em exploits - como pode se confirmar nos fórums do Kaggle dedicados a competição em questão

- e, caso estes fossem desclassificados, o posicionamento deste trabalho veria uma melhora considerável.

4. Conclusão

Apesar da nota obtida na entrega do arquivo contendo as predições poderia ser melhorada usando técnicas como junção de modelos (KAGGLE... 2015), ou se realizando mais testes com os parâmetros de entrada do *XGBoost*.

Os resultados obtidos foram satisfatórios, já que os dados fornecidos eram limitados, e com algumas colunas sendo ignoradas (por não representarem dados que estariam disponíveis em uma situação real). Porém mais fontes externas podem ser adicionadas aos dados base, assim aumentando a eficiência dos modelos preditivos selecionados.

Ao final do trabalho a compreensão dos autores, sobre o processo completo de análise dos dados, foi aprofundado e desafiado, fato que proporcionou um crescimento pessoal valioso na área de análise de dados para os integrantes da equipe. Sendo assim, os autores deste trabalho se dão por satisfeitos com o que foi desenvolvido neste trabalho, e se sentem preparados para no futuro aplicar, de maneira melhor e mais completa, os conhecimentos desenvolvidos.

5. Referências

KÉGL, Balázs et al. **The Higgs Machine Learning Challenge.** 2014. Disponível em: <https://higgsml.lal.in2p3.fr/>. Acesso em: 10 nov. 2016.

QUINLAN, J. R.. **Programs for Machine Learning**. San Mateo, Ca: Morgan Kaufmann Publishers, 1993.

1.10. Decision Trees — scikit-learn 0.18.1 documentation. Disponível em: <http://scikit-learn.org/stable/modules/tree.html>. Acesso em: 15 nov. 2016.

FRIEDMAN, Jerome H.. Greedy function approximation: A gradient boosting machine. **The Annals Of Statistics.** [s.l.], p. 1189-1232. 08 fev. 2002. Disponível em: <http://projecteuclid.org/euclid.aos/1013203451>. Acesso em: 10 nov. 2016.

CHEN, Tianqi; GUESTRIN, Carlos. XGBoost. **Proceedings Of The 22nd Acm Sigkdd International Conference On Knowledge Discovery And Data Mining - Kdd '16,** [s.l.], p.1-13, 2016. Association for Computing Machinery (ACM). http://dx.doi.org/10.1145/2939672.2939785.

BREIMAN, Leo. Random Forests. **Machine Learning,** [s.l.], v. 45, n. 1, p.5-32, 2001. Springer Nature. http://dx.doi.org/10.1023/a:1010933404324.

KAGGLE Ensembling Guide. 2015. Disponível em: <http://mlwave.com/kaggle-ensembling-guide/>. Acesso em: 11 nov. 2016.

SCIKIT Learn. Disponível em: <http://scikit-learn.org/stable/#>. Acesso em: 11 nov. 2016.