# Detecção de Fraude Financeira utilizando Ciência de Dados

**Victor M. Azevedo[1], Lucas B. Figueira[2]**

[1,2] Faculdade de Tecnologia de Ribeirão Preto (Fatec)
Ribeirão Preto – SP – Brasil

[1]victor.azevedo@fatec.sp.gov.br, [2]lucas.figueira@fatec.sp.gov.br

***Abstract.*** *This paper describes in a pratical way how Data Science combined with some machine learning algorithms help the identification of financial fraud. In this study, was used Artificial Neural Networks, Random Forest and XGB Trees. The research shows the importance of preprocessing, how the alogrithms work and wich one has obtained the best performance for the detection of fraud. For this, a synthetic dataset with about 6.4 million records was used.*

***Resumo.*** *O presente estudo demonstra de maneira prática como a Ciência de Dados, utilizando os diferentes tipos de algoritmos de aprendizado de máquina, colabora para a identificação de fraudes financeiras. Neste estudo, foram utilizados: Redes Neurais Artificiais, Floresta Aleatória (Random Forest) e XGB Trees. A pesquisa mostra a importância do pré-processamento, como os algoritmos atuam e qual obteve o melhor desempenho para a detecção de fraudes. Para este fim, utilizou-se um dataset sintético de aproximadamente 6.4 milhões de registros.*

## 1. Introdução

Fraude financeira possui diferentes significados, "financial fraud is a broad term with various potential meanings, but for our purposes it can be defined as the intentional use of ilegal methods or practices for the purpose of obtaining financial gain". (Zhou e Kapoor, 2011, apud West and Bhattacharya 2015). "Fraud has a large negative impact on business and society: credit card fraud alone accounts for billions of dollars of lost revenue each year." (Bhattacharya, Jha, Tharakunnel e Westland 2011, apud West e Bhattacharya, 2015).

A descoberta de fraudes financeiras, antes do advento da tecnologia, era um trabalho manual. Um dos primeiros trabalhos científicos realizados para a identificação de fraudes utilizando tecnologia foi feito em 1997 por Tom Fawcett e Foster Provost. De acordo com Fawcett e Provost (1997) o número de clonagens de celulares estava aumentando, trazendo prejuízos para as companhias telefônicas e para os clientes. Os registros de ligações telefônicas, por exemplo, eram analisados manualmente por um analista, o que por vezes gerava análises errôneas. Com estes dados, decidiram automatizar as análises, deixando-as mais assertivas.

Logo mais a frente, e com a tecnologia mais avançada, as técnicas de mineração de dados mostraram ser úteis por sua capacidade de identificar pequenas anomalias em grandes conjuntos de dados. (Ngai, Hu, Wong, Chen e Sun, 2011).

A Ciência de Dados, ou o cientista de dados, recentemente entrou à tona devido a Harvard Business Review chamá-la de a profissão mais sexy do século XXI. (Devenport, Patil, 2012). Porém, de acordo com Silveira (2016), a ciência de dados é uma área que já existe há 30 anos. O termo Big Data é proveniente da grande quantidade de dados, estruturados ou não estruturados, que são gerados diariamente. Data Science, ou Ciência de Dados, é o estudo acerca destes dados, incorporando técnicas e teorias de diversas áreas de conhecimento.

Conforme a tecnologia e o número de dados crescem abruptamente, o número de fraudes financeiras cresce em paralelo (Yeh; Lien, 2009, apud West e Bhattacharya, 2015).

A fraude financeira no setor de cartões de crédito, geram um custo muito grande ao comerciante, uma vez que acabam pagando custos de envio, devolução e custo administrativo. Além disso, perde-se a confiança do consumidor (Quah; Sriganesh, 2008, apud West e Bhattacharya, 2015).

De acordo com Diniz e Sousa (2017), o Brasil registra uma tentativa de fraude a cada 16,8 segundos. O terceiro segmento mais afetado pelas fraudes são os bancos e financeiras. Somente em 2016, o volume total de transações realizadas em cartões de débito e crédito no Brasil foi de R$1,14 trilhão. (Disponível em: https://istoe.com.br/volume-de-transacoes-com-cartoes-cresce-63-em-2016-para-r-114-tri-diz-abecs/ Acesso em jul.2020).

Levando em conta o volume total de transações realizadas e a quantidade de ocorrências de fraude que ocorrem por segundo no Brasil, há uma chance de que entre estas transações há alguma relacionada a fraude.

O projeto em questão irá beneficiar os três pilares de interesse: empresa, funcionários e sociedade. Com a detecção automática de fraudes financeiras, as empresas poderiam evitar a perda de receita, uma vez que fraudes financeiras geram perda de receitas e até mesmo perda de clientes. (Quah; Sriganesh, 2008, apud West e Bhattacharya, 2015). Os funcionários também passariam a se beneficiar, uma vez que, caso não haja um SAD (Sistema de Apoio a Decisão) para auxiliá-lo nas detecções, o mesmo fará análises ineficazes, podendo comprometer a segurança da empresa. A sociedade se beneficiaria como um todo, uma vez que a taxa de fraudes diminuiria, aumentando a confiabilidade na segurança do mercado.

O presente trabalho será dividido em: **Materiais e Métodos**, em que será apresentado as técnicas de pré-processamento do *dataset* e os algoritmos de aprendizado de máquina (*Multilayer perceptron*, *Random Forest* e *XGB Trees)*. **Resultados e discussão**, em que será apresentado e explicado os resultados dos algoritmos citados no item anterior.

## 2. Materiais e Métodos

Para este trabalho, o objeto de estudo foi um conjunto de dados *(dataset)* de aproximadamente 6.4 milhões de registros gerados por um simulador chamado *PaySim*. Este simulador utiliza dados agregados de um *dataset* privado. O *dataset* em questão foi obtido pela plataforma *Kaggle*, a qual é denominada a maior comunidade de ciência de dados do mundo. (Kaggle. **Kaggle:** Your Machine Learning and Data Science Community, c2020. Página inicial. Disponível em: https://www.kaggle.com/)

Todo o estudo foi realizado na plataforma *Google Colab* utilizando a linguagem de programação Python versão 3.6.9.

## 2.1. O *Dataset*

O *Dataset* utilizado neste estudo possui várias transações que foram geradas sinteticamente em uma simulação de 30 dias. Cada transação possui diferentes dados, tais como:

a) unidade de tempo. No caso deste *dataset* foi mensurado em "*step",* que representa 1 hora do dia, ou seja, o *dataset* possui 744 *steps* que correspondem a 30 dias (744 horas);

b) tipo de transação, tais como: *cash-in* (depósito), *cash-out (*saque), *debit* (débito), *payment* (pagamento) e *transfer* (transferência);

c) *amount:* valor da transação;

d) *nameOrig* e *nameDest*: quem iniciou a transação e quem a recebeu;

e) *oldBalanceOrig, newBalanceOrig, oldBalanceDest e newBalanceDest*: saldo antigo e novo de quem originou a transação e do destinatário.

f) *isFraud*: este campo foi imprescindível para a análise ocorrer. Estas transações são feitas por agentes fraudulentos dentro da simulação. Neste *dataset* o comportamento dos agentes visa lucrar assumindo o controle ou as contas dos clientes, tentando esvaziar os fundos transferindo para outra conta e depois sacando o dinheiro do sistema.

g) *isFlaggedFraud*: o modelo de negócios do simulador visa controlar transferências massivas de uma conta para outra e sinaliza tentativas ilegais. Uma tentativa ilegal do *dataset* é a tentativa de transferir mais de 200.000 em uma única transação.

Na etapa de pré-processamento, é realizada a análise de todos os dados citados acima, gerando conhecimento necessário para definir quais dados realmente serão significantes para o modelo ou não.

## 2.2. Pré-processamento

O pré-processamento de dados consiste em todas as ações tomadas antes do início do processo de análise de dados real. O principal objetivo de qualquer análise de dados é descobrir conhecimentos que serão usados para a resolução de problemas ou tomada de decisões. Entretanto, problemas com os dados podem impedir isto. Na maioria das vezes, imperfeições com os dados não são notadas até a análise dos dados começar. (Famili, 1997).

Existem problemas com dados do mundo real e geralmente é necessária alguma forma de pré-processamento para analisá-los de forma inteligente. Além de resolver problemas, tais como dados corrompidos, irrelevantes ou faltando atributos nos *datasets*, pode-se aprender mais a respeito da natureza dos dados, ou alterar a estrutura destes (ex., níveis de granularidade), para prepará-los para um processo mais eficiente de análise. (Famili, 1997)

### 2.2.1. Fluxo

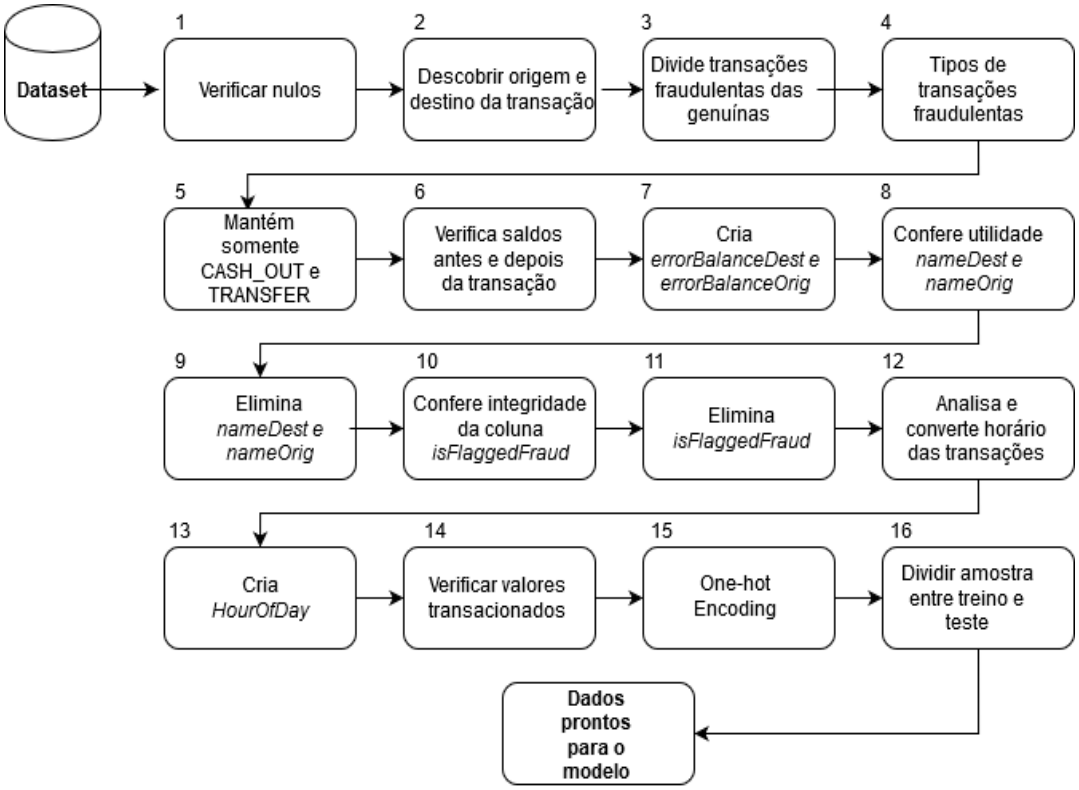


**Figura 1. Fluxo do pré-processamento até obter dados prontos para o modelo**
Fonte: O autor (2020)

Conforme análise do fluxo mostrado na Figura 1, percebe-se que, além de ter sido feito um grande estudo sobre o *dataset* em si, com ele alguns atributos foram retirados, pois não apresentavam uma real importância para o modelo e em contrapartida novos atributos foram criados. Este processo de limpeza, estudo e tratamento dos dados foi de suma importância para o resultado do modelo.

Alguns pontos apresentados na Figura 1 foram essenciais e merecem destaque: no item 3, realizou-se a divisão do *dataset*. Um *dataset* somente com transações fraudulentas – marcadas como *isFraud* – e outro com transações genuínas. Após esta divisão, no item 4, realizou-se um estudo com os dois *datasets*, sendo que o que continha somente fraudes, apresentava somente operações de '*CASH_OUT*' (saque) e '*TRANSFER*' (transferência). Desta forma, como a investigação trata de transações fraudulentas, os outros tipos de operações foram excluídos do *dataset* de transações genuínas. No item 6, verificou-se os saldos, tanto de origem quanto de destino da operação para garantir que o resultado do saldo final fosse condizente com a operação realizada. Por exemplo: em uma transação de transferência de $181, garantir que o saldo final de quem enviou fosse subtraído pelo valor enviado, e o saldo do destinatário fosse somado com o valor recebido. Porém, a análise mostrou que em todos os registros havia erro no saldo, ou no saldo da pessoa que originou ou no saldo do destinatário. Desta forma, criou-se dois atributos chamados '*errorBalanceOrig*' (saldo de origem errado) e '*errorBalanceDest*' (saldo do destinatário errado). Outro ponto deste estudo foi observado: cerca de 57% das transações fraudulentas, apresentavam erro no saldo do destinatário. Por outro lado, cerca de 90% das transações genuínas apresentavam erro no saldo de origem.

No item 12, analisou-se o atributo "*step"* do *dataset*. Em um primeiro momento, foi analisada a comparação entre o número de transações genuínas e fraudulentas por *step*. Após, foi feita a conversão para dias da semana e, posteriormente, a conversão para hora do dia. Nesta última, percebeu-se que as transações fraudulentas mantêm um ritmo alto, enquanto as genuínas, entre 0h e 09h, são realizadas pouca ou nenhuma transação. Este fator foi determinante para a inclusão do atributo *HourOfDay* no *dataset*.

### 2.2.2. *One-hot encoding*

Segundo Artasanchez & Joshi (2020), *one-hot encoding* é uma técnica frequentemente usada no pré-processamento. Alguns algoritmos de *machine learning* não conseguem lidar com atributos categóricos, portanto o *one-hot encoding* é uma maneira de converter esses atributos categóricos em atributos numéricos.

### 2.2.3. *Cross Validation* e *Standardization*

Segundo Refaeilzadeh (2009), o *cross-validation* é um método estatístico para avaliar e comparar os algoritmos de aprendizado, dividindo os dados em dois segmentos: um usado para aprender ou treinar um modelo e o outro usado para validar o modelo. *Standardization* inclui o desvio padrão como parte de seu cálculo, desta forma, minimiza e suaviza o efeito de *outliers*. (Artasanches and Joshi, 2020)

## 2.3. Os Modelos de *Machine Learning*

### 2.3.1. Redes Neurais Artificiais

Um dos modelos utilizados no projeto, foi o *Multilayer Perceptron* que é um tipo de Rede Neural Artificial. Segundo Braga (2007), as redes neurais artificiais são inspiradas no funcionamento dos neurônios biológicos. De acordo com o autor, cada neurônio, chamado de nodos aqui, calculam determinadas funções matemáticas, normalmente não lineares, e cada um é interligado por muitas conexões, geralmente unidirecionais. Cada conexão está associada a um peso, o qual armazena o conhecimento representado no modelo e serve para ponderar a entrada recebida por um neurônio da rede. O autor também cita que o procedimento usual na resolução de problemas passa primeiro por uma aprendizagem, em que um conjunto de exemplos é apresentado para a rede e, consequentemente, características são extraídas para representar a informação fornecida. Estas características serão usadas posteriormente para gerar respostas para o problema. No presente trabalho, estes exemplos podem ser identificados como as transações marcadas como fraudulentas no *dataset*.

Conforme mostra a Figura 2, Braga (2007) diz que as unidades intermediárias funcionam como detectores de características. Elas geram uma codificação interna dos padrões de entrada, que é utilizado para a definição da saída da rede.

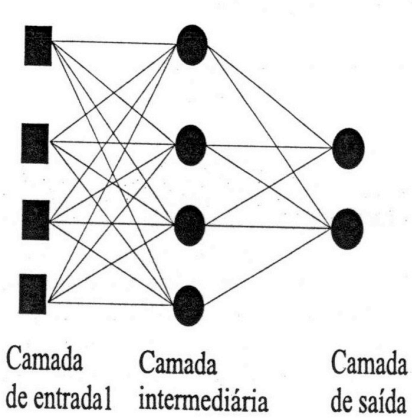


**Figura 2. Exemplo de uma rede MLP com camada intermediária (BRAGA, 2007)**

### 2.3.2. Floresta Aleatória *(Random Forest)*

*Random Forest* é um conjunto de árvores de decisão. A popularidade de árvores de decisão surgiu da facilidade de uso, da flexibilidade em termos de lidar com vários tipos de atributos e da fácil interpretação. Porém, modelos de apenas uma árvore podem ser instáveis e muito sensíveis a dados específicos. (Breiman, 2001, apud Bhattacharya, 2011).

Conjuntos de árvores de decisão possuem um bom desempenho quando seus membros individuais são diferentes e, florestas aleatórias, obtêm a variação entre árvores individuais usando duas fontes de aleatoriedade: primeiro, cada árvore é construída utilizando amostras de treino separadas; na segunda etapa, na construção das árvores individuais, apenas alguns atributos escolhidos aleatoriamente são considerados em cada nó. Assim, florestas aleatórias combinam os conceitos de *"bagging"*, no qual modelos individuais em um conjunto são desenvolvidos através de amostragem utilizando os dados de treinamento, e o método de subespaço aleatório, no qual cada árvore do conjunto é construída a partir de um subconjunto de atributos. (Bhattacharya, 2011).

Didaticamente, o *dataset* é dividido em sub *datasets*: um de teste e outro de treinamento. Após, a árvore de decisão é construída do *dataset* de treinamento. A taxa de erros será calculada utilizando o *dataset* de teste. A partir disso, a árvore cresce a partir de amostras retiradas do *dataset* de treino. O *dataset* original de treinamento é mantido e utilizado para selecionar a melhor sub-árvore podada. Este procedimento é repetido *k* vezes resultando em árvores de classificação (1 até *k*). Para cada elemento do *dataset* de teste, compara-se a classe prevista com a classe verdadeira. A proporção de vezes que a classe estimada difere da verdadeira, é a taxa de erro de classificação "*bagging"*. A divisão aleatória nos *datasets* de teste e treino é repetida. O resultado consiste nas taxas médias de erro sobre as árvores.

### 2.3.3. *eXtreme Gradient Boost Trees (XGB Trees)*

*Gradient Boosting* é uma poderosa técnica de *machine learning* utilizada para regressão, classificação e problemas de ranqueamento. *XGB* significa *eXtreme Gradient Boosting*, uma das implementações do conceito de *gradient boosting*. O diferencial do *XGB* é que é implementado um modelo mais regularizado para controlar o *over-fitting* e alcançar

uma melhor performance (Yang, 2019), desta forma o *XGBoost* foi construído justamente para otimizar o uso de memória e explorar o poder computacional. (Dhieb, 2019)

A principal ideia do *boosting* é construir sub-árvores sequenciais a partir de uma árvore original de modo que cada árvore subsequente reduza os erros da árvore anterior. (Chen, 2019). Ou seja, as árvores são construídas uma de cada vez e cada nova árvore ajuda a corrigir os erros da árvore anterior. Porém, como as árvores são geradas sequencialmente, o treinamento leva mais tempo.

## 3. Resultados e discussão

**Tabela 1. Resultados obtidos nos três modelos**

| | *TN* | *TP* | *FN* | *FP* | *acc* | *AUC* | *tempo execução* |
|---|---|---|---|---|---|---|---|
| MLP | 690512 | 1589 | 468 | 34 | 99,93 | 0,8862 | 126,93 |
| RF | 690561 | 2034 | 6 | 2 | 99,99 | 0,9985 | 94,15 |
| XGB | 690363 | 2036 | 4 | 200 | 99,97 | 0,9989 | 365,08 |

Fonte: O autor (2020)

Observando a Tabela 1, ao analisar os dados do *Multilayer perceptron* (MLP), apesar do ótimo resultado obtido, as redes neurais artificiais aplicadas com dados complexos e desbalanceados, tendem a ficar enviesados pela classe majoritária. (Castro, Braga, 2013). No presente estudo, essa classe é a de transações genuínas. MLP foi executado com um número de camadas e de nós de 11, que é o número de colunas do *dataset* final. O número de épocas foi de 1000.

A Floresta Aleatória (RF) foi executada com um número de 15 estimadores. Os estimadores significam a quantidade de árvores na floresta. Analisando o *accuracy* e a AUC, nota-se o ótimo desempenho do modelo.

O XGB, foi executado com uma profundidade máxima da árvore de 3. Também foi realizado um tratamento de pesos para *dataset* desbalanceado. O tempo total de execução foi de 365.08 segundos. Conforme dito no item 2.3.3, XGBoost possui um maior tempo de execução. De todos os modelos, este foi o que mais levou tempo para ser executado. A AUC do XGBoost foi levemente maior se comparada com a AUC da Floresta Aleatória.

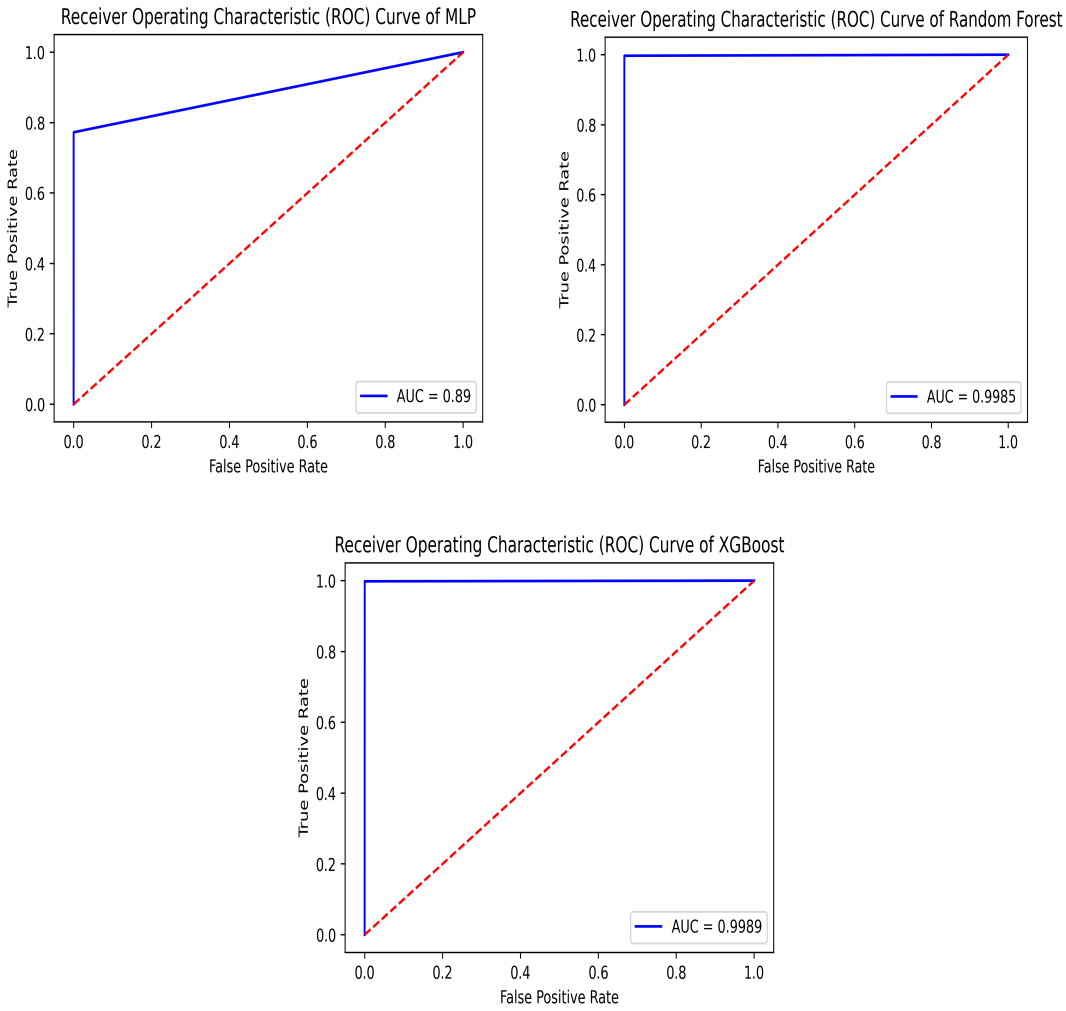


**Figura 3. AUC dos três modelos executados**

Fonte: O autor (2020)

Após toda as análises e comparando os modelos, nota-se que os que obtiveram o melhor resultado foram Floresta Aleatória e XGBoost com apenas algumas diferenças cruciais entre os dois.

Um dos pontos chaves do resultado e da escolha do melhor modelo para o problema em questão, foi o número de transações genuínas marcadas como fraudulentas.

O modelo de Floresta Aleatória classificou apenas 2 transações neste cenário. Por outro lado, XGBoost classificou 200. Ao analisar todo o cenário e regra de negócio, este número implicaria um custo maior, não só para o modelo em si, mas também para os estabelecimentos.

Comparando a AUC entre os dois, a diferença é mínima e XGBoost obteve um desempenho ligeiramente maior. Porém, no geral, o modelo de Floresta Aleatória foi o mais performático e o que obteve o melhor desempenho de todos os modelos.

XGBoost classificou um número considerável de falsos-positivos. Desta forma, imaginando um cenário de um *dataset* maior e até mesmo com situações do dia a dia, haveria um grande custo com essas transações genuínas que foram marcadas como fraudulentas. Provavelmente geraria um grande desgaste no cliente e um enorme custo na resolução do problema.

Outro ponto interessante, conforme apresentado na Figura 4, foi analisar a *feature selection* do modelo, isto é: identificar o melhor atributo e qual obteve a melhor performance de aprendizado, menor custo computacional e deixou o modelo mais interpretável. (Wang, 2017). Com isto, identificamos a importância de um bom pré-processamento dos dados antes de executar os modelos pretendidos.

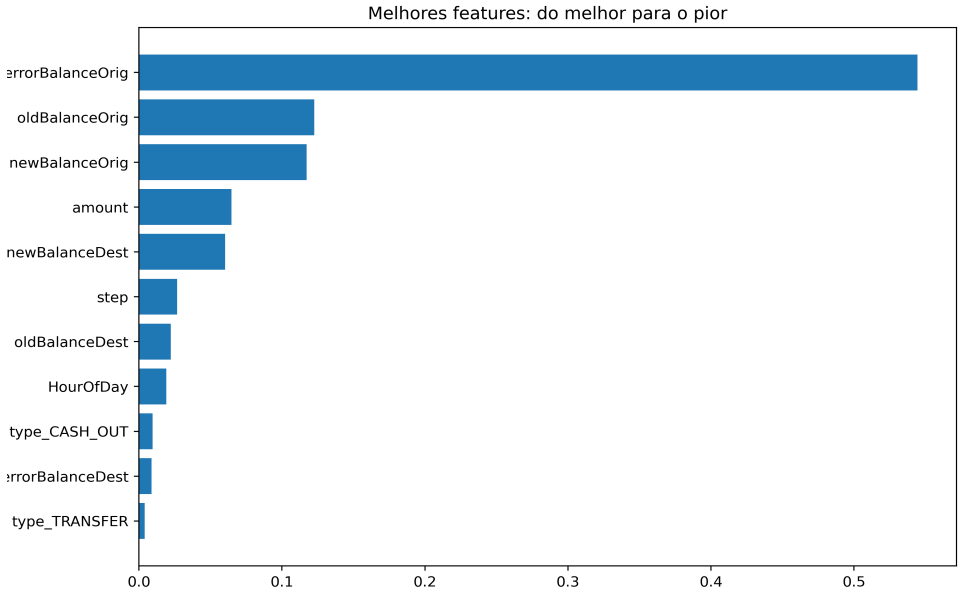


**Figura 4. *Feature selection* da Floresta Aleatória**

Fonte: O autor (2020)

## 4. Conclusão

Com o presente trabalho, conclui-se que para *datasets* desbalanceados, o modelo de Floresta Aleatória obteve um ótimo desempenho. Ao final, vê-se a importância de um pré-processamento completo e minucioso dos dados. Ao analisar o *feature selection* do modelo, foi notado que o atributo decisivo foi o inserido na etapa de pré-processamento. Apesar do XGBoost apresentar um bom desempenho, em uma análise final, mais voltada a negócio do que computacional, mostrou que não era o suficiente. Ou seja, além da abordagem computacional, é preciso ter um conhecimento não só dos dados, mas de negócio também.

Os próximos passos a seguir são a respeito de verificar o desempenho dos modelos em dados balanceados ou até mesmo outro *dataset* sobre fraude e, também, verificar se Floresta Aleatória continua sendo a melhor opção.

## 5. Referências Bibliográficas

Artasanchez, A., & Joshi, P. (2020). "Artificial intelligence with Python: Your complete guide to building intelligent apps using Python 3.x and TensorFlow 2". Birmingham: Packt Publishing.

Bhattacharyya, Siddhartha & Jha, Sanjeev & Tharakunnel, Kurian & Westland, J.. (2011). "Data mining for credit card fraud: A comparative study. Decision Support Systems. 50. p. 602-613.

Braga, A. D., Carvalho, A. C. P. L., & Ludermir, T. B. (2007). "Redes neurais artificiais: Teoria e aplicações." Rio de Janeiro: LTC Editora.

Castro, Cristiano & Braga, Antônio. (2013). "Novel Cost-Sensitive Approach to Improve the Multilayer Perceptron Performance on Imbalanced Data." Neural Networks and Learning Systems, IEEE Transactions on. 24. p. 888-899.

Chen, Minghua & Liu, Qunying & Chen, Shuheng & Liu, Yicen & Zhang, Changhua & Liu, Ruihua. (2019). "XGBoost-Based Algorithm Interpretation and Application on Post-Fault Transient Stability Status Prediction of Power System." IEEE Access. vol. 7, p. 13149-13158.

Dhieb, Najmeddine & Ghazzai, Hakim & Besbes, Hichem & Massoud, Yehia. (2019). "Extreme Gradient Boosting Machine Learning Algorithm For Safe Auto Insurance Operations." 1-5.

Famili, Fazel & Shen, Wei-min & Weber, Richard & Simoudis, Evangelos. (1997) "Data Preprocessing and Intelligent Data Analysis" Intell. Data Anal. 1. p. 3-23.

Fawcett, P., Provost F. (1997) "Adaptive Fraud Detection", In: Data Mining and Knowledge Discovery 1, pages 291-316.

Liu H. (2011) "Feature Selection." In: Sammut C., Webb G.I. (eds) Encyclopedia of Machine Learning. Springer, Boston, MA

Ngai, E.W.T. & Hu, Yong & Wong, Y.H. & Chen, Yijun & Sun, Xin. (2011) "The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature.", In: Decision Support Systems. 50. p. 559-569.

Quah, J. T. S. and Sriganesh, M. (2008) "Real-time credit card fraud detection using Computational Intelligence", In: Expert Systems with Applications, n.35, p. 1721-1732, Singapore.

Refaeilzadeh P., Tang L., Liu H. (2009) "Cross-Validation." In: LIU L., ÖZSU M.T. (eds) Encyclopedia of Database Systems. Springer, Boston, MA.

Ting K.M. (2017) "Confusion Matrix." In: Sammut C., Webb G.I. (eds) Encyclopedia of Machine Learning and Data Mining. Springer, Boston, MA

West, Jarrod & Bhattacharya, Maumita (2015) "Intelligent Financial Fraud Detection: A Comprehensive Review." Computers & Security. 57, p. 47-66.

Yang, Xulei & Wang, Li & Niu, Xuetong. (2019). "A Comparison Study of Credit Card Fraud Detection: Supervised versus Unsupervised."

Yeh, Ivy & Lien, Che-Hui. (2009) "The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients.", Expert Systems with Applications. 36. p. 2473-2480.

Zhou, Wei & Kapoor, Gaurav. (2011) "Detecting evolutionary financial statement fraud." Decision Support Systems. 50. p. 570-575.