Análise e Predição de Métodos de Pagamento nas Plataformas de *e-commerce* da Olist

Guilherme O. Soares[1], João Victor M. Gadelha[1]

[1]Instituto de Computação
Universidade Federal do Rio de Janeiro (UFRJ) – Rio de Janeiro, RJ – Brasil

guilhermevos@ic.ufrj.br, dlha@dcc.ufrj.br

Abstract. *This paper describes a process of analysis and predictions of consumer payment method behavior using as basis a public dataset from one of the biggest e-commerce players in Brazil. We delve deep into analysis and modeling of a particularly unbalanced set of data, hoping to potentially get insightful results and get a sense of the challenge in dealing with such data.*

Resumo. *Este artigo descreve processos de análise e predição da forma de pagamento preferida por usuários em compras online, usando como base um dataset público disponibilizado por uma das maiores empresas de e-commerce do Brasil, a Olist. É conduzido uma análise estatística básica e modelagem num dataset altamente desbalanceado, tentando obter resultados satisfatórios e ter uma ideia dos riscos associados à sua utilização.*

1. Introdução

Com a acessibilidade atual da internet não é surpresa que uma das principais atividades que se tornaram corriqueiras no dia-a-dia são o ato das compras. O *e-commerce* é presente em peso em todos os cantos da internet através de promoções, anúncios e preços competitivos. Destaca-se também a facilidade de entrada no mercado já que qualquer pessoa pode entrar numa plataforma e se tornar anunciante sem muitas dificuldades, eliminando a necessidade de existir uma loja física para o varejo.

Neste cenário achamos interessante investigar como os brasileiros realizam suas compras. Em especial qual forma de pagamento é mais comumente usada dependendo do valor da compra e da região.

Na plataforma *Kaggle* encontramos uma base publica de dados disponibilizada pela Olist, uma das maiores empresas de plataforma e-commerce do Brasil. Utilizando essa base de dados esperamos encontrar alguma relação entre o valor da compra e o meio de pagamento utilizado.

2. Dados Utilizados

Os dados disponibilizados via *Kaggle* vieram bem organizados cada um com um propósito específico. Um arquivo CSV diferente para cada: Pagamentos, Pedidos, Produtos, Vendedores, Localização, Clientes, Avaliações e Itens num Pedido. É possível facilmente mesclar as informações através dos *ID's* específicos de cada arquivo e uma figura do esquema também foi disponibilizada. Os dados disponíveis são do período de 2016 a 2018.

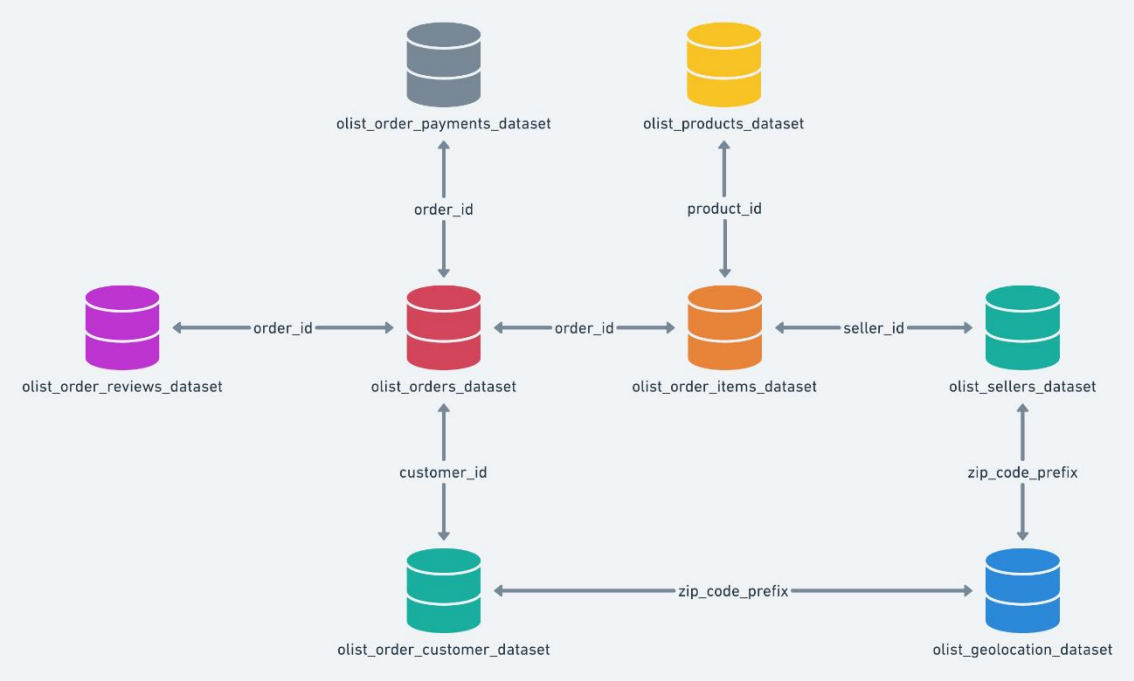

Figura 1. Esquema do Banco de Dados

Todos os dados sensíveis de identificação haviam sido previamente anonimizados e os datasets se encontravam completos e sem grandes erros. Foram usados na análise os arquivos referentes aos pagamentos, pedidos e consumidores, utilizando os devidos ID's para concatenação.

Obtivemos aproximadamente 100 mil entradas únicas de meio de pagamento entre: Cartão de Crédito, Boleto, Voucher e Débito. Haviam também 3 entradas aonde a forma de pagamento não havia sido especificada. Estas 3 entradas foram excluídas para análise.

Aparentemente o dataset na realidade foi criado com a finalidade de utilizar processamento natural de língua (NLP) para analisar o sentimento dos consumidores após as compras através das avaliações. Estamos utilizando o dataset de uma forma que ele não foi projetado para auxiliar.

3. Análise Inicial

Utilizando Python, juntamente com suas bibliotecas de dados mais populares como Pandas e Scikit conduzimos uma análise estatística básica inicial nos dados e descobrimos algumas métricas interessantes: A compra média do brasileiro custa em torno de 154 reais. O número médio de parcelas no cartão é perto de 3. Compras acima de mil reais representam por volta de 1% apenas dos dados.

Notamos também que, como a intenção é prever o tipo de pagamento utilizado, precisaríamos excluir a coluna referente ao número de parcelas, do Dataset. Várias outras colunas referentes a datas e outras informações irrelevantes ao estudo foram removidas. No fim ficamos apenas com o valor da compra e o estado aonde o comprador reside, esperando que exista alguma relação entre o local onde o comprador mora e os hábitos de gastos da população.

Alguns outros problemas ficaram aparentes durante a análise inicial. Primeiramente, o Dataset como um todo era extremamente desbalanceado: das 130 mil entradas tínhamos aproximadamente 80% das compras sendo realizadas no crédito. Outra questão eram os valores de compras acima de 2000 reais todas sendo possíveis de se considerar outliers já que representavam aproxidamente 0.5% dos dados. Como se trata de um estudo que toca em assuntos financeiros, achamos relevante manter os valores mais extremos no banco. O máximo registrado foi o valor de 14 mil reais.

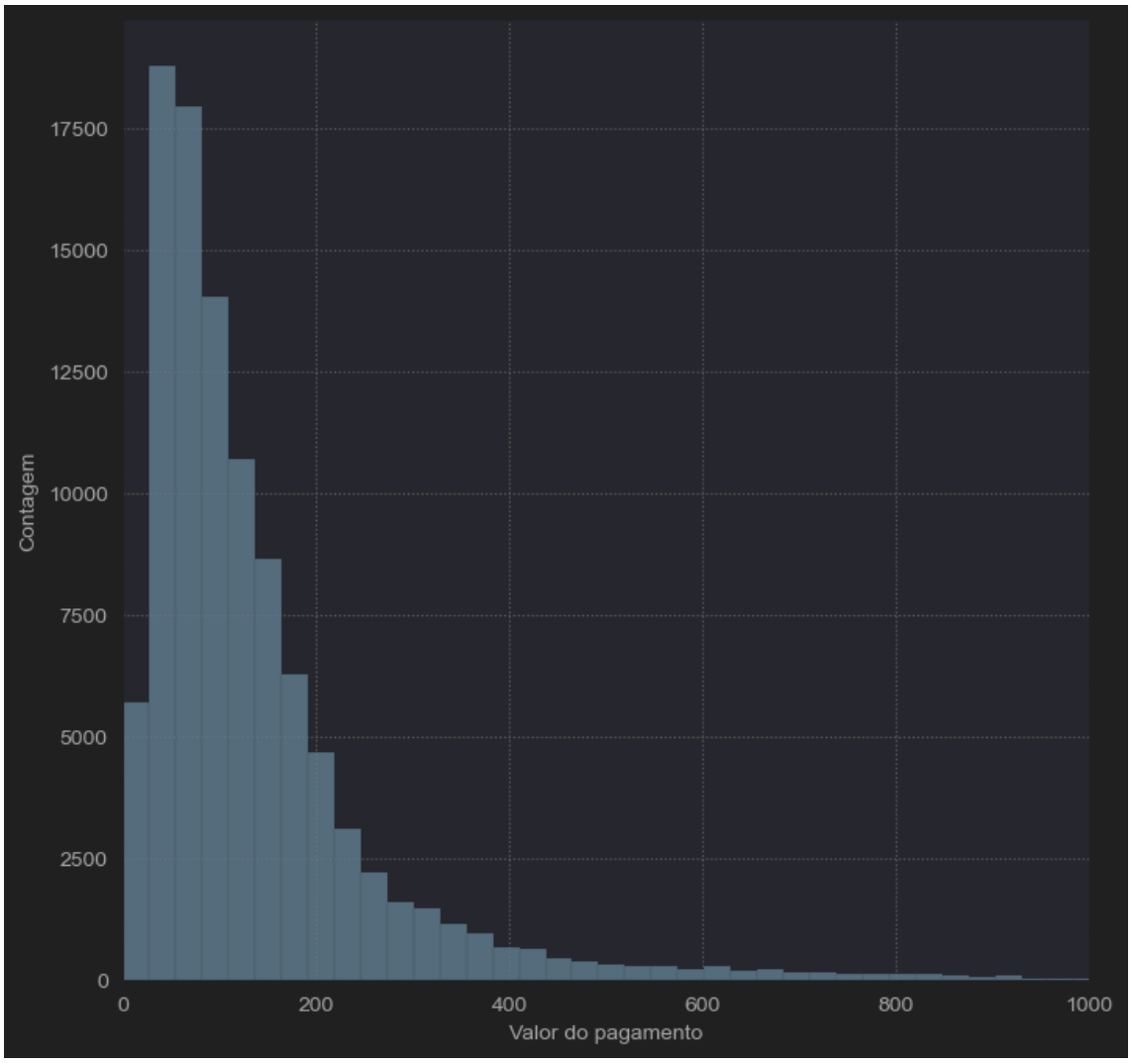

Figura 2. Distribuição dos valores dos pagamentos

Utilizados o método *get_dummies* da biblioteca Pandas para rapidamente fazer o encoding da nossa única feature categórica, os Estados. Após este processo podemos finalmente começar com os processos de aprendizado de máquina.

4. Aprendizado de Máquina

Sendo a intenção principal do estudo entender se existe relação entre o valor de uma compra e o método de pagamento, tomamos como opção métodos de classificação. Como a análise inicial demonstrou que o Dataset é extremamente desbalanceado optamos por utilizar métodos de classificação com complexidade ascendente. Também separamos 20% dos dados para testes utilizando o método *train_test_split* do Scikit. Todos os modelos de aprendizado de máquina utilizados também são da mesma biblioteca. Foi utilizado o valor 64 para os atributos *random_state* ao longo dos testes. Escolher um valor garante a reprodutibilidade dos resultados.

4.1. Regressão Logística

A forma mais básica de classificador foi utilizada primeiro como referência. É um modelo linear que tenta diminuir o erro entre as classificações reais e as previstas. Levando em consideração as limitações do modelo em si, e a complexidade do dataset escolhido, deduzimos que o desempenho não seria satisfatório. O modelo foi utilizado com os valores iniciais padrão.

Como esperado, devido a grande densidade de entradas no cartão de crédito, o modelo chutou todas as entradas como tal. Uma precisão de 100% para cartão de crédito e 0% para todos os outros tipos. Não havia altas expectativas para o desempenho do classificador mais básico desde o começo então logo partimos para o próximo modelo.

payment_type	count
credit_card	15339
boleto	3948
voucher	1168
debit_card	322

0	count
credit_card	20777

Figura 3. Densidade real do conjunto de teste (esquerda) versus as predições do modelo de regressão logística simples (direita).

4.2. Classificador de Máquina de Vetores de Suporte (SVM)

Este classificador tenta criar “zonas” de classificação fazendo cálculos avançados de álgebra linear. Num caso com apenas dois classificadores por exemplo, após rodar o algoritmo o SVM faria uma reta com distancia equidistante aos dois pontos mais próximos das retas. Tudo acima ou abaixo da reta é da mesma classe. No nosso caso com várias classes o SVM faria várias zonas.

Infelizmente, os resultados com SVM foram ainda mais estranhos: a maioria dos dados foi classificado como Voucher. Uma dedução razoável do porque o algoritmo falhou neste caso pode se dar ao fato dos valores acima de 2000 também serem em maioria compras no crédito. Neste caso a linha que serviria de separação entre crédito e as outras classes pode ter ficado quase horizontal, para poder englobar os valores extremos. Incrivelmente, crédito foi a classe menos escolhida. Sendo SVM também inviável para a o cenário atual partidos para o próximo modelo.

payment_type	count
credit_card	15339
boleto	3948
voucher	1168
debit_card	322

0	count
voucher	16229
debit_card	2243
boleto	2028
credit_card	277

Figura 4. Densidade real do conjunto de teste (esquerda) versus as predições do modelo classificador SVM (direita).

4.3. Classificador de Floresta Aleatória (*Random Forest Classifier*)

Como tentativa final utilizamos o *Random Forest Classifier*, modelo que gera várias árvores de decisão e aplica um sistema de votação entre as diferentes instâncias.

Apesar da densidade ainda deixar a desejar esse foi o modelo mais "balanceado" dentre todos os construídos. A distribuição de escolhas é muita mais próxima do conjunto de teste.

payment_type	count
credit_card	15339
boleto	3948
voucher	1168
debit_card	322

0	count
credit_card	17153
boleto	2689
voucher	753
debit_card	182

Figura 5. Densidade real do conjunto de teste (esquerda) versus as predições do modelo classificador *Random Forest* (direita).

Sendo o modelo com o melhor desempenho até o momento, analisamos sua matriz de confusão. Temos aproximadamente 80% das instâncias de cartão de crédito sendo corretamente identificadas. Boleto sendo a segunda mais bem identificada com aproximadamente 25%. Voucher vem em terceiro com aproximadamente 3% e débito com apenas um acerto o que é essencialmente 0%.

Ponderamos sobre como melhorar os resultados prosseguindo com Random Forest. Já que de todos os modelos foi o com melhor desempenho.

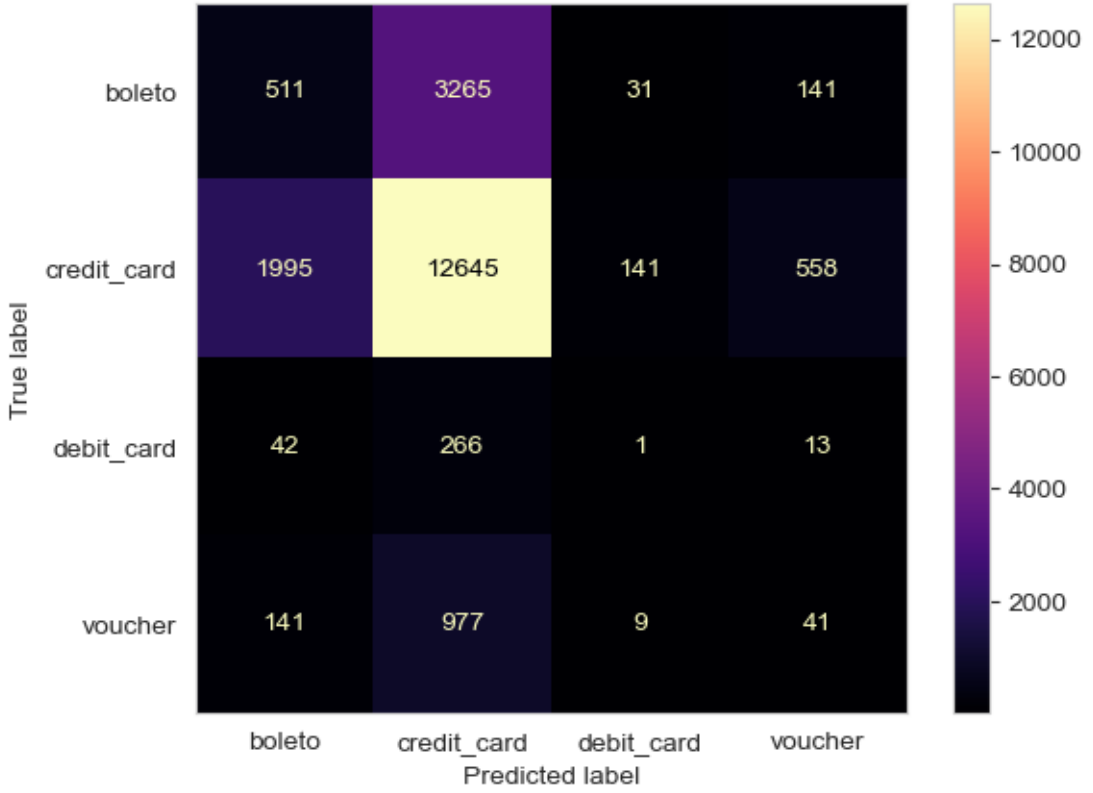

Figura 6. Matriz de confusão gerada pelo modelo *Random Forest*.

4.4. Feature Engineering

Se os valores de compra e os estados não são suficientes para uma predição precisa, retornamos nossa atenção ao Dataset para averiguar se existiria alguma informação que daria para inferir das informações disponíveis. Como a tabela de consumidores possui

cidade, estado e CEP, achamos que seria razoável incluir aos dados a informação se um determinado consumir mora ou não numa capital.

O senso comum dita que pessoas nas capitais têm podem de compra maior. Portanto, possivelmente aconteceria algum efeito na modelagem. Utilizamos um dicionário em Python onde a chave é a sigla do Estado e o valor é a capital do respectivo estado. Criamos uma função que pega linha por linha do dataset e checa se o valor da chave no atributo estado da linha é igual ao nome da capital no atributo cidade da linha. O vetor de zeros e uns, um representando "Sim", foi concatenado as outras features.

Refizemos a modelagem e obtivemos outra matriz de confusão bem semelhante. Fora uma ligeira melhora no reconhecimento de pagamentos feitos via Boleto, o desempenho do modelo em geral foi **pior** do que sem a informação da capital.

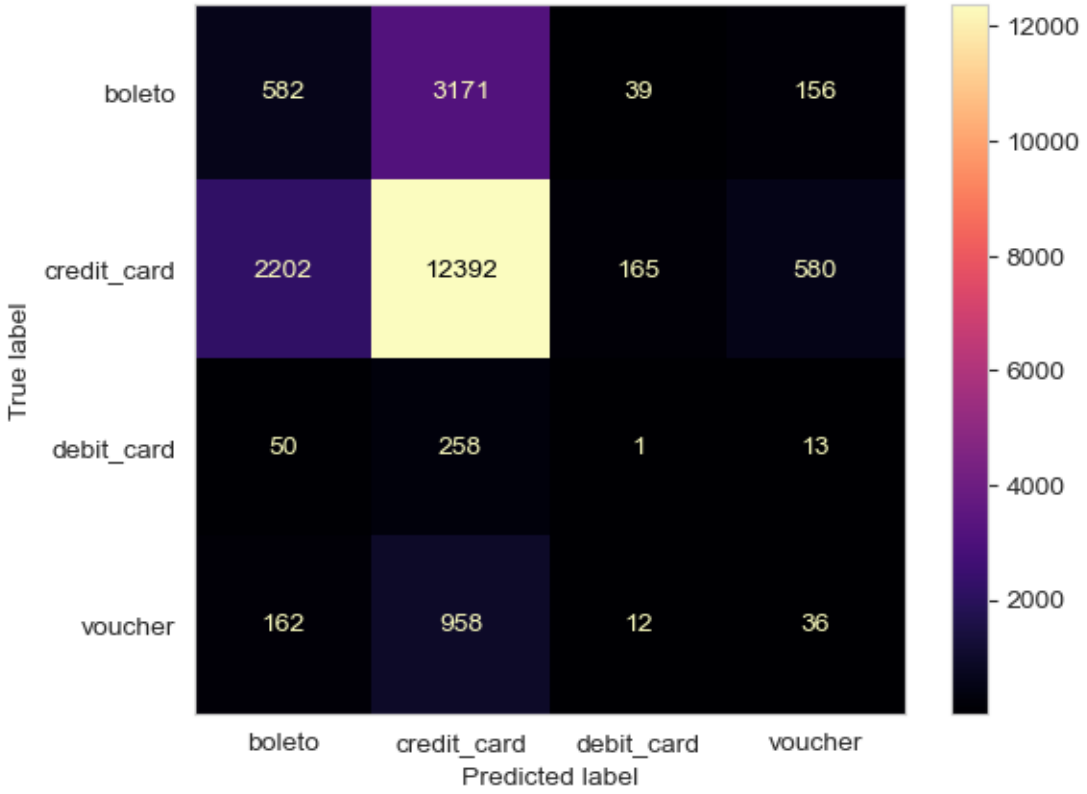

Figura 7. Matriz de confusão gerada pelo modelo *Random Forest com informação sobre a Capital*.

A conclusão lógica é que não existe forte correlação entre a cidade ser uma capital e o meio de pagamento utilizado. Sem muitas opções, a única forma restante de tentar otimizar o modelo é via força bruta.

4.5. Grid Search

Grid Search é um método avançado da biblioteca do Scikit que permite que o usuário escolha, entre outras coisas, um modelo, uma função para classificar o modelo, e um dicionário com várias chaves, cada uma referente a um parâmetro que pode ser alterado no modelo selecionado.

O *Grid Search* iterativamente cria um modelo diferente com cada combinação possível de parâmetros e guarda sua nota. Neste experimento, focamos nos parâmetros *n_estimators* e *Bootstrap.* O parâmetro *n_estimators* diz respeito à quantas árvores diferentes criar por floresta, *Bootstrap,* é um valor booleano que simboliza se as amostras num treinamento podem ser feitas com ou sem reposição.

Também utilizamos validação cruzada para tentar diminuir o erro das estimativas. Por se tratar de um método de força bruta mais a adição de validação cruzada era esperado que o treinamento desta vez fosse consideravelmente mais longo. O algoritmo *Grid Search* demorou perto de 37 minutos para terminar, numa máquina com processador de 4 núcleos e sem placa gráfica. Com 6 possíveis valores para o número de estimadores e 2 possíveis para o *Bootstrap,* foram testados 12 modelos diferentes.

A função que dá a nota do modelo utilizada foi a precisão balanceada, que faz uma media dos acertos levando em consideração o tamanho relativo de cada classe. O modelo que saiu vencedor foi o modelo onde o número de estimadores foi igual a 10 e com *Bootstrap* ativado.

Construindo o modelo com os parâmetros indicados pelo *Grid Search* partimos para a análise de mais uma matriz de confusão.

De imediato é possível ver que a classificação dos Boletos é melhor em relação ao primeiro Random Forest, e a classificação do Cartão de Crédito é melhor em relação ao segundo modelo. Por isso que o sistema de classificação baseado em precisão balanceada recomendou os parâmetros específicos. No geral, porém, o ganho na classificação é mínimo. No fim das contas os três modelos são, para efeitos práticos, equivalentes.

5. Conclusões

Lidar com o Dataset desbalanceado se provou um verdadeiro desafio, e reforça a ideia que na área de dados o processo de obtenção, limpeza e validação dos dados realmente representa 80% do trabalho.

Talvez com mais informações fosse possível refinar os modelos. Alguns destaques seriam por exemplo: idade do comprador, já que menores não podem

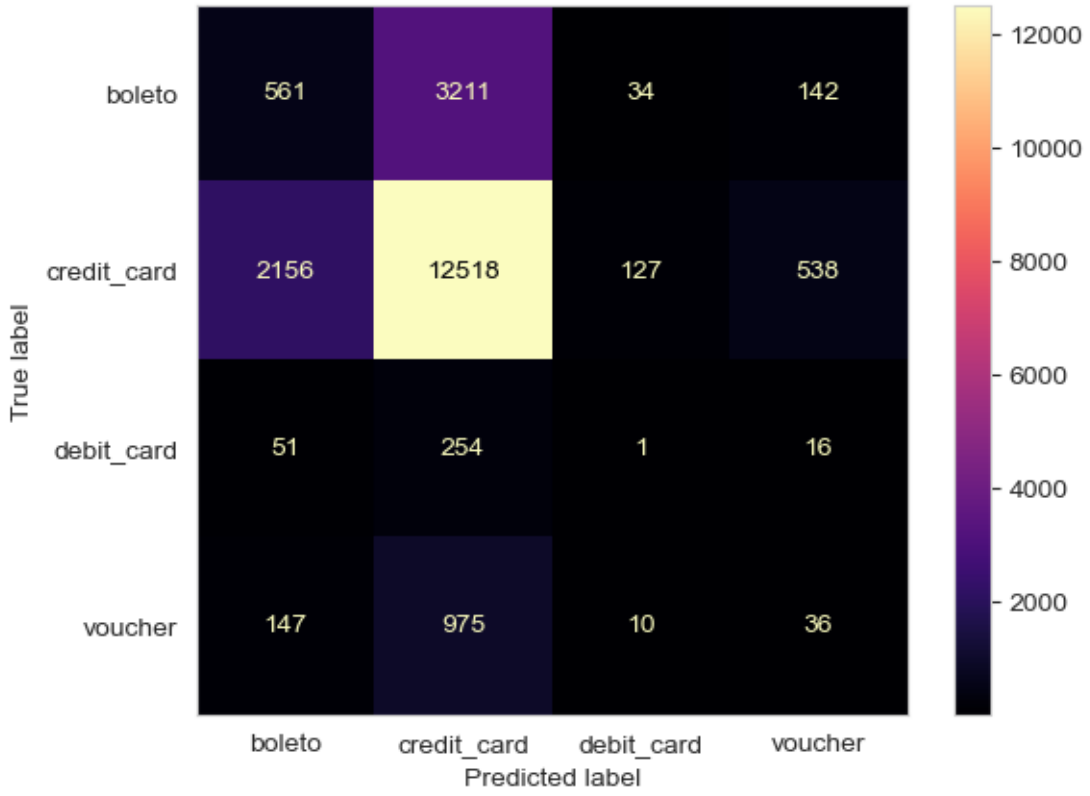

Figura 8. Matriz de confusão gerada pelo modelo *Random Forest* utilizando os parâmetros ótimos recomendados pelo *Grid Search*.

legalmente trabalhar e pessoas mais velhas tendem a ter alguma forma de renda fixa e/ou aposentadoria. Junto com a idade talvez o tipo de produto comprado também agregasse alguma coisa: existem certos produtos que são mais comprados por certas faixas etárias, especialmente roupas.

Outro ponto que chama atenção é o fato de só aproximadamente 1.5% das compras terem sido feitas no Débito. Em proporção, para cada 50 compras no crédito somente uma é realizada no debito.

Outras utilizações desta mesma base de dados no *Kaggle* tentam usar o modelo para prever o preço de certos produtos, fazer análise de sentimento, treinar visualizações e fazer analises geográficas da distribuição do mercado. Acreditamos que para tentar prever o meio de pagamento tenha sido a primeira tentativa.

Referências

Kaggle, plataforma de aprendizado e competições de ciência de dados

https://www.kaggle.com

Brazilian E-Commerce Public Dataset by Olist, Dataset Utilizado

https://www.kaggle.com/datasets/olistbr/brazilian-ecommerce

Python

https://www.python.org/

Scikit

https://scikit-learn.org/

Pandas

https://pandas.pydata.org/

Notebook Jupyter com o código:

https://github.com/GuilhermeVOS/DataScience/blob/master/Intro%20Aprendizado%20de%20Maquina/OlistPagamento.ipynb