UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

A influência de dados correlacionados em modelos de Aprendizado de Máquina - Um estudo empírico

Erica da Silva Lopes

Dissertação de Mestrado do Programa de Mestrado Profissional em Matemática, Estatística e Computação Aplicadas à Indústria (MECAI)

ICMC USP
SÃO CARLOS

Erica da Silva Lopes

A influência de dados correlacionados em modelos de Aprendizado de Máquina - Um estudo empírico

Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Mestra – Mestrado Profissional em Matemática, Estatística e Computação Aplicadas à Indústria. *VERSÃO REVISADA*

Área de Concentração: Matemática, Estatística e Computação

Orientador: Prof. Dr. Adriano Kamimura Suzuki

USP – São Carlos
Maio de 2021

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

Lopes, Erica Silva
L864i A influência de dados correlacionados em modelos de Aprendizado de Máquina - Um estudo empírico / Erica Silva Lopes; orientador Adriano Kamimura Suzuki. -- São Carlos, 2021.
75 p.

Dissertação (Mestrado - Programa de Pós-Graduação em Mestrado Profissional em Matemática, Estatística e Computação Aplicadas à Indústria) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2021.

1. Aprendizado Estatístico. 2. Independência amostral. 3. Generalização. I. Suzuki, Adriano Kamimura, orient. II. Título.

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

Erica da Silva Lopes

The influence of correlated data on Machine Learning models - An empirical study

Dissertation submitted to the Instituto de Ciências Matemáticas e de Computação – ICMC-USP – in accordance with the requirements of the Professional Master's Program in Mathematics Statistics and Computing Applied to Industry, for the degree of Master in Science. *FINAL VERSION*

Concentration Area: Mathematics, Statistics and Computing

Advisor: Prof. Dr. Adriano Kamimura Suzuki

USP – São Carlos
May 2021

Este trabalho é dedicado a todos os questionadores e curiosos.
Aos que estão sempre antenados às publicações de pesquisadores e cientistas.
Aos que buscam melhores práticas, os que são incansáveis na busca pelo aprendizado.

AGRADECIMENTOS

Agradeço ao Prof. Dr. Adriano Kamimura Suzuki por toda paciência e parceria que teve comigo durante o desenvolvimento deste trabalho.

Agradeço à minha família por todo o apoio e incentivo nos momentos de dificuldade e por fazer parte de todos os momentos de alegrias e conquistas.

Agradeço o apoio do Itaú-Unibanco S.A, pela concessão de horas para o desenvolvimento deste estudo. Vale ressaltar que qualquer opinião ou conclusão deste estudo não reflete necessariamente as visões, políticas oficiais ou posicionamento do Itaú-Unibanco S.A.

“As invenções são, sobretudo,
o resultado de um trabalho de teimoso.”
(Santos Dumont)

RESUMO

LOPES, E. S. **A influência de dados correlacionados em modelos de Aprendizado de Máquina - Um estudo empírico**. 2021. 75 p. Dissertação (Mestrado – Mestrado Profissional em Matemática, Estatística e Computação Aplicadas à Indústria) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2021.

O uso de modelos de Aprendizado de Máquina tem sido difundido em diferentes áreas da indústria, seja para medir satisfação de marcas de acordo com comentários na *Internet*, ou para recomendar produtos, ou para avaliar o risco de crédito. Entretanto, muitos analistas associam esta disciplina exclusivamente à área da Ciência da Computação, desconsiderando conceitos estatísticos fundamentais para garantir o aprendizado, generalização, do modelo. Tendo em vista que a Teoria do Aprendizado Estatístico defende cinco premissas para a garantia da generalização, o objetivo deste estudo é avaliar empiricamente os efeitos ao desconsiderar uma das premissas, a independência entre as observações. Neste sentido, foram avaliadas duas bases de dados do setor bancário, ambas com dados coletados em uma janela temporal. As bases foram separadas em três subconjuntos (treino, validação e teste), em que os dois primeiros contêm observações coletadas na mesma janela temporal, porém a validação não foi usada no desenvolvimento do modelo. Observa-se que o teste contém informações novas não pertencentes a janela temporal da base de treino. O subconjunto de validação permite que o desempenho do modelo seja avaliado em dados que possuem características semelhantes aos utilizados no treino. Por outro lado, o teste permite a avaliação em um novo cenário, uma vez que o período de observação não foi incluído no treino do modelo. A técnica de Aprendizado de Máquina *Light Gradient Boosting Machine* foi usada para modelar cada uma das bases de treino. A performance dos modelos foi mensurada com a métrica AUC e comparada com os diferentes tipos de autocorrelação de cada base (dependência entre unidades amostrais e temporal). Os resultados mostram que a autocorrelação temporal, estatisticamente significante para os dois conjuntos de dados, influencia na queda de performance fora da janela temporal de desenvolvimento dos modelos (subconjuntos de teste). Por outro lado, para base de dados em que não há autocorrelação significativa entre as observações, o modelo ajustado apresentou bom desempenho para os dados de validação, diferente do que ocorreu com a base que possui autocorrelação significativa entre as observações. Sendo assim, há indícios de que ao se desconsiderar a premissa de independência no conjunto de dados a capacidade de aprendizado do modelo é prejudicada.

Palavras-chave: Independência amostral, Generalização, Aprendizado Estatístico, Aprendizado de Máquina.

ABSTRACT

LOPES, E. S. **The influence of correlated data on Machine Learning models - An empirical study**. 2021. 75 p. Dissertação (Mestrado – Mestrado Profissional em Matemática, Estatística e Computação Aplicadas à Indústria) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2021.

The use of Machine Learning models has been widespread in different areas of the industry, either to assess brand satisfaction according to comments on the internet, or to recommend products, or to assess credit risk. However, many people associate this subject exclusively to computer science area, disregarding fundamental statistical concepts to guarantee the learning, generalization, of the model. Bearing in mind that the Theory of Statistical Learning has five premises for a guarantee of generalization, the aim of this study is to empirically evaluate the effects by disregarding one of the premises, an independence among the necessary ones. In this sense, two databases of the banking sector were evaluated, both with data collected in a temporal window. The databases were divided into training, validation and testing, in which the first two were collected at the same time window, but the validation was not used in the development of the model. We note that the teste contains new information not belonging to the time frame of the training base. The Machine Learning textit Light Gradient Boosting Machine technique was used to model each of the training bases. The performance of the models was measured with the AUC metric and compared with the different types of autocorrelation for each base (dependence between sample and temporal units). The results showed that a temporal autocorrelation, statistically significant for the two data sets, influences the decrease in performance for the temporal window of development of the models (test subsets). On the other hand, for a database in which there is not autocorrelation between for a database in which there is not autocorrelation between the observations, the fitted model presented good performance for the validation data. Also, we note this not occur for the fitted model with the base that has significant autocorralation between the observations. Thus, there are indications that if the premise of independence in the data set is disregarded, the ability of the model to learn is impaired.

Keywords: Sampling independence, Generalization, Statistical learning, Machine learning.

LISTA DE ILUSTRAÇÕES

LISTA DE QUADROS

LISTA DE ALGORITMOS

SUMÁRIO

CAPÍTULO

1

INTRODUÇÃO

O setor bancário tem sido transformado pela Inteligência Artificial e o Aprendizado de Máquina. A aplicação de tecnologias e métodos regidos por esta área multidisciplinar da ciência contribui para o direcionamento de decisões mais assertivas, desenvolvimento de serviços sob medida para cada perfil de cliente e aprimorar o gerenciamento de riscos. A estimativa é que o uso desta ciência possa gerar mais de 250 bilhões de dólares no setor bancário, com maior contribuição para a gestão de risco, prevenção de fraudes e concessão de crédito (WALDRON, 2019).

Em análise de risco de crédito são utilizadas muitas técnicas de Aprendizado de Máquina, conforme exposto por Dumitrescu *et al.* (2018), dentre estas pode-se citar o K-ésimo Vizinho Mais Próximo (*k-nearest neighbors*, KNN), Redes Neurais, Árvores de Decisão, *Random Forest* e Máquina de Vetores de Suporte (*Support Vector Machine*, SVM). De modo geral, o interesse em ajustar um modelo de risco de crédito é atribuir uma nota ao cliente de acordo com a probabilidade de atrasar o valor emprestado ou não, dentro de uma janela de observação temporal.

Contudo, ao se modelar o atraso em uma janela temporal, tem-se por consequência, a existência de correlação temporal entre os dados, isto é, dependência temporal. Assim como em modelos estatísticos de Regressão Linear ou Logística (MORETTIN; BUSSAB, 2017), ao aplicar modelos de Aprendizado de Máquina, Vapnik (1999) defende cinco premissas para que haja de fato aprendizado, isto é, para que o modelo consiga generalizar os resultados para novos conjuntos de dados. Umas das premissas assumidas é a independência entre as observações. Partindo-se da ideia de que muitos analistas não verificam a premissa independência entre os dados no desenvolvimento de modelos de Aprendizado de Máquina, quais as possíveis consequências de não se considerar o rigor requerido pela teoria?

Muitas universidades abordam o Aprendizado de Máquina em cursos de Ciência da Computação, isso porque geralmente é estudado como parte da Inteligência Artificial. Essa abordagem direciona o foco para os algoritmos, porém, entender como os algoritmos funci-

onam requer forte conhecimento em Estatística e Matemática (MARSLAND, 2015). O não conhecimento da teoria do Aprendizado Estatístico faz com que aplicações de Aprendizado de Máquina ocorram, sem que haja garantia teórica de generalização e sem que se tenha consciência dos possíveis riscos. Por outro lado, alguns pesquisadores têm se dedicado a estudar formas de trabalhar com dados dependentes. Compondo este grupo, Steinwart, Hush e Scovel (2009) defendem a ideia de relaxar a premissa de independência dos dados, com o uso da técnica SVM. Os autores discutem e mostram a justificativa teórica da afirmação de que para qualquer processo de geração de dados, que satisfaça a Lei dos Grandes Números para Processo Estocástico, existe uma sequência de parâmetros de regularização tal que o SVM correspondente é consistente, isto é, garante o aprendizado.

Em contrapartida, Pagliosa e Mello (2017) acreditam que relaxar a premissa de independência não é o melhor caminho. Neste sentido, propõem o uso do Teorema de Imersão de Takens, uma ferramenta do Sistema Dinâmico que visa reconstruir séries temporais em um espaço de fase (TAKENS, 1981). O Teorema foi aplicado como função de kernel em séries temporais, com o intuito de mapear a dependência temporal das observações nos eixos de um espaço de fase, para que o modelo de Aprendizado de Máquina seja aplicado nesta transformação do conjunto de dados. No estudo foram usados três conjuntos de dados fictícios: o mapa logístico (usado para modelar a taxa de crescimento das populações), o sistema Lorenz (função que modela dados atmosféricos para apoiar a previsão do tempo) e o mapa de Hénon (representa a interseção de uma órbita periódica do espaço de fase de Lorenz com um certo subespaço de dimensão inferior, transversal ao fluxo do sistema, também conhecido como a seção de Poincaré). Os resultados do artigo mostram evidências de que houve aprendizado.

O presente trabalho tem o intuito de abordar esta discussão de maneira empírica e dar foco a um assunto tão pouco discutido na indústria. Para tal, foram considerados dois conjuntos de dados bancários: um sobre atraso em hipoteca residencial e outro sobre venda de depósito a longo prazo. Tratam-se de bases multivariadas (com mais um atributo, variável explicativa), cujos dados foram coletados em uma janela temporal e a unidade amostral é o indivíduo. Vale ressaltar que apesar de não ser uma regra, se a mesma unidade amostral estiver presente em mais de um período, então de fato existirá correlação entre as observações. Tendo em vista os diferentes tipos de autocorrelações dos dados de cada base, a mesma técnica da Aprendizado de Máquina (*Light Gradient Boosting Machine*) foi aplicada às duas bases, com o objetivo de avaliar o desempenho do modelo tanto para dados não usados no treino, mas pertencentes a mesma janela temporal de desenvolvimento do modelo, quanto para novos dados em outra janela temporal.

1.1 Objetivo Geral

Avaliar a influência da dependência, entre as observações e temporal, sobre o aprendizado do modelo em dois cenários (em novos dados com mesma correlação temporal e em novos dados em outra janela temporal) e para duas bases de dados diferentes.

1.2 Objetivos Específicos

Ajustar modelo de Aprendizado de Máquina a duas bases de dados (com observações temporais) e comparar a qualidade de ajuste com as seguinte métricas: sensibilidade, especificidade e a área sob a curva ROC (*Area Under the ROC Curve*, AUC). A comparação tem como objetivo avaliar se os modelos produzem resultados semelhantes ao desenvolvimento, quando aplicados a novos dados observados na mesma janela temporal do desenvolvimento e em nova janela.

1.3 Organização do Trabalho

O Capítulo 2 é formado pela revisão bibliográfica, em que são apresentados os conceitos de Aprendizado de Máquina e da Teoria do Aprendizado Estatístico, bem como a técnica de modelagem usada no estudo de caso. No Capítulo 3, há apresentação sobre as bases de dados usadas no estudo e as análises descritivas. O Capítulo 4 é formado pela discussão sobres os métodos aplicados e os resultados obtidos. As consideração finais estão no Capítulo 5.

CAPÍTULO

2

REFERENCIAL TEÓRICO

Segundo Marsland (2015), Aprendizado de Máquina pode ser pensado como algoritmos que têm capacidade de aprenderem e se adaptarem a um conjunto de dados. Geralmente são usados em previsões ou para controlar ações de robôs, como por exemplo, conceder crédito a clientes de um banco, automatizar os sinais de trânsito de acordo com as condições da estrada, produzir estimativas financeiras, dentre outros.

Para um melhor entendimento sobre como ocorre o aprendizado, considere que um estudante esteja estudando para uma prova. Uma opção para ter bom desempenho seria decorar o conteúdo que será abordado. Porém, uma vez que o estudante não possui as questões da prova, seria necessário pensar em todas as possíveis perguntas e decorá-las, o que torna esta estratégia ineficiente. Uma boa estratégia, seria selecionar as ideias centrais e dedicar-se a entendê-las. De forma simplista, os algoritmos de Aprendizado de Máquina funcionam de maneira análoga. Genericamente, o conteúdo a ser estudado seriam as variáveis explicativas, e objetivo é acertar as questões da prova. A eficiência de estratégia de estudo seria avaliada de acordo com a nota na prova (LANTZ, 2013).

O Aprendizado de Máquina pode ser dividido em duas vertentes: não supervisionada e supervisionada. O aprendizado não supervisionado ocorre quando não se possui uma variável para guiar o aprendizado, neste caso, o algoritmo funciona de forma a encontrar semelhanças entre as observações, de acordo com as variáveis explicativas. O aprendizado supervisionado tem como objetivo aprender uma forma de mapear um conjunto de entrada a uma saída cujos valores corretos são fornecidos por um supervisor. O exemplo apresentado no parágrafo anterior, representa a vertente de aprendizado supervisionado. Essa abordagem permite que seja possível fazer previsões para casos não vistos com boa precisão (MELLO; PONTI, 2018).

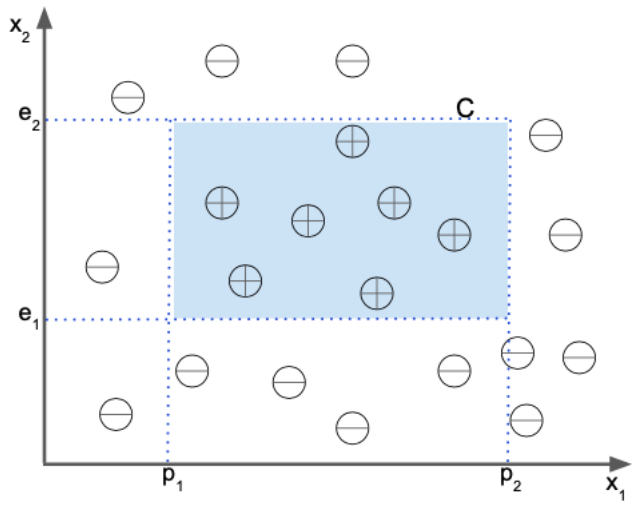
O foco desta dissertação é o aprendizado supervisionado, sobretudo a modelagem voltada para a classificação, uma vez que algoritmos com treino guiado pela variável de interesse permite avaliar a capacidade de generalização, isto é, o comportamento em dados novos.

2.1 Aprendizado de Máquina - Classificação

O aprendizado supervisionado pode ser usado com o objetivo de estimar uma variável contínua (são exemplos o salário, vendas e a temperatura) ou a classe da observação (por exemplo bons e maus pagadores, espécies de plantas, grupos socieconômicos). A classificação, cuja variável resposta (saída do modelo) possui apenas dois possíveis valores, é o caso mais simples (ALPAYDıN, 2010).

Para ilustrar o conceito, considere o cenário que se queira estimar a variável aleatória Y a partir do conjunto de variáveis $X = \{x_1, x_2\}$, em que Y é binário, isto é, pode assumir os valores 0 e 1, onde 0 denota a classe negativa e 1 a classe positiva. Cada observação é composta pelo par ordenado (x, y). O conjunto de dados com N observações é denotado por $D = \{x^t, y^t\}_{t=1}^{N}$. A Figura 1 é a representação cartesiana destes dados, em que cada ponto é a coordenada x_1^t, x_2^t e a classe Y é representada pelo tipo de ponto ($\oplus$ positiva, ou $\ominus$ negativa).

Figura 1 – Exemplo de classe de hipóteses



Fonte: Elaborada pelo autor.

Observe na Figura 1 que, a partir do conjunto de treino D, é possível determinar um retângulo C, delimitado por $(p_1 \leq x_1 \leq p_2)$ e $(e_1 \leq x_2 \leq e_2)$, em que observações da classe positiva estão contidas. O retângulo C é considerado a classe de hipóteses para mapear observações positivas. Seja H o conjunto de funções capazes de de estimar C, o objetivo do Aprendizado de Máquina é encontrar $h \in H$, Equação 2.1, que melhor se aproxima de C, através do conjunto D. Note que embora, visualmente, seja possível identificar a classe de hipóteses, os parâmetros da função $h(x)$ são desconhecidos.

$$h(x) = \begin{cases} 1 & \text{se } h \text{ classifica } x \text{ como um exemplo positivo,} \\ 0 & \text{caso contrário.} \end{cases} \tag{2.1}$$

É comum que a região C não seja conhecida, sendo assim, qualidade de ajuste de $h(x)$ é mensurada com o cálculo do erro empírico. O erro empírico, neste caso, é definido como a proporção de casos do conjunto de treino D, cujas previsões de h não correspondem a classe observada em D. A Equação 2.2 expressa matematicamente este erro, em que I é a função identidade, resulta 1 se $h(x^t) \neq y^t$ e 0 caso contrário.

$$E(h \mid D) = \frac{1}{N} \sum_{t=1}^{N} I(h(x^t) \neq y^t). \tag{2.2}$$

Logo, este problema consiste encontrar uma função $h \in H$ que melhor se aproxime da região C, isto é, que produza erro empírico próximo de zero. Neste caso, os parâmetros de h são os quatro vértices do retângulo $(p_1^h, p_2^h, e_1^h, e_2^h)$. Supondo que $X \in \mathbb{R}^2$, há infinitos pontos que definem a fronteira estimada de C e geram baixos valores de $E(h \mid D)$, entretanto, uma importante consideração é que o erro também seja baixo para observações não usadas no treino do modelo. Em outras palavras, $h(x)$ deve ser capaz de generalizar os resultados.

2.2 Teoria do Aprendizado Estatístico

Quando se trata de aprendizado supervisionado, há foco na preocupação com relação à capacidade de generalização do modelo proposto. Considerando o cenário em que o objetivo é classificar um conjunto de dados segundo as categorias da variável de interesse Y e variáveis explicativas X, em seu artigo, Vapnik (1999) expõe a Teoria do Aprendizado Estatístico que propõe pressupostos necessários para garantir a aprendizagem:

1. Nenhuma suposição é feita sobre a função de probabilidade conjunta $P(X \times Y)$;
2. Os exemplos devem ser amostrados de maneira independente;
3. As categorias da variável resposta podem assumir valores não determinísticos devido ao ruído e sobreposição de classes;
4. $P(X \times Y)$ está fixo (estático, então não muda ao longo tempo);
5. $P(X \times Y)$ é desconhecida durante o treino do modelo.

O primeiro pressuposto diz que não há restrição quanto ao tipo de distribuição de probabilidade conjunta dos dados. No segundo item, assumir independência entre as amostras é uma forma de garantir que a distribuição de probabilidade da amostra seja semelhante ao da população (universo). Com relação ao terceiro item, é comum que haja região de sobreposição das categorias da variável de interesse Y, ou por algum erro (que deveria ser pequeno) ou por características dos dados. O quarto item é uma implicação do segundo item, além disso se a probabilidade mudasse ao longo do tempo o conjunto de treino poderia não ser suficiente para

estimar $P(X \times Y)$. Quanto ao quinto item, os modelos puramente estatísticos, no geral, assumem alguma suposição sobre a distribuição dos dados, que reduz o problema a encontrar os parâmetros desta distribuição; ao não considerar a distribuição previamente, se a amostra for suficientemente grande é possível estimar $P(X \times Y)$ com erro controlado (MELLO; PONTI, 2018).

Para avaliar se o classificador $h(x^t)$ possui poder de generalização, deve-se considerar uma função de perda, Equação 2.3.Observa-se que a esperança desta função é denominada risco esperado, Equação 2.4, um escalar que quantifica as divergências entre os resultados observados e os inferidos ao aplicar $h(x), \forall x^t$, dado a probabilidade conjunta, $P(X \times Y)$, não conhecida.

$$I(h(x^t)) = \begin{cases} 0, & h(x^t) = y \\ 1, & h(x^t) \neq y \end{cases}, \tag{2.3}$$

$$R(h) := E(h \mid X). \tag{2.4}$$

Por outro lado, não se tem toda a população para realizar o cálculo do risco esperado. Sendo assim, considera-se o risco empírico, dado pela média da função de perda aplicada a cada observação do conjunto de treino D, Equação 2.5. De acordo com Mello e Ponti (2018), a avaliação a respeito da capacidade de generalização do modelo, classificador $h(x)$, pode ser realizada ao comparar o risco empírico, com o risco de um conjunto de dados não usado no desenvolvimento do modelo.

$$R(h)_{emp} = E(h \mid D) = \frac{1}{N} \sum_{t=1}^{N} I(h(x^t) \neq y^t). \tag{2.5}$$

$$G = \mid R(h)_{emp} - R(h) \mid . \tag{2.6}$$

Seja $R(h)$ a função de risco aplicada a dados não usados do desenvolvimento do modelo, se a diferença entre $R(h)_{emp}$ e $R(h)$, Equação 2.6, for pequena, a conclusão é que o modelo é capaz de generalizar o resultado, isto é, houve aprendizado. Por outro lado, Mello e Ponti (2018) observam que não necessariamente um modelo com boa capacidade de generalização é o modelo com menor erro. Sendo assim, é preciso encontrar o classificador que produza o menor erro e então, avaliar a generalização.

2.3 Gradiente descendente

Considere o caso de aprendizado supervisionado, em que o objetivo é estimar uma variável aleatória Y a partir de um conjunto de variáveis explicativas $X = \{x_1, x_2, x_3, \ldots, x_p\}$, também aleatórias. O objetivo deste tipo de aprendizado é encontrar a melhor função (ou

estimador) $h(X) = \widehat{Y}$, pertencente ao conjunto de todas as possíveis funções $h^* : X \rightarrow Y$. Entende-se por melhor função, a que minimiza o risco esperado. Em aprendizado de máquina, é comum a aplicação do método de gradiente descendente para encontrar o melhor estimador $h(X)$.

A definição matemática para o termo gradiente (*gradient*), diz respeito ao vetor das derivadas parciais no ponto $x_0 \in X$, também denominado gradiente de h em x_0 (Equação 2.7).

$$\bigtriangledown h = \left(\frac{\partial h}{\partial x_1} x_0, \frac{\partial h}{\partial x_2} x_0, \ldots, \frac{\partial h}{\partial x_p} x_0 \right). \tag{2.7}$$

Geometricamente, o gradiente indica a direção em que ocorre maior aumento no valor de uma função, sendo utilizado para encontrar pontos de máximos (ou mínimos) local ou global da função (GUIDORIZZI, 2002).

A título de simplificação, considere agora que as possíveis funções $h^*(X)$ possam ser escritas de forma genérica, conforme a Equação 2.8:

$$h^*(X) = X\beta + \varepsilon, \tag{2.8}$$

em que $\beta = \{\beta_1, \beta_2, \beta_3, \ldots, \beta_p\}$ são os coeficientes de X, ε o erro aleatório e h uma função diferenciável. Então, x_0 é ponto critico (mínimo ou máximo) local se para qualquer $x_i \in \{x_1, x_2, x_3, \cdots, x_p\}$, a igualdade da Equação 2.9 for satisfeita (GUIDORIZZI, 2002).

$$\frac{\partial h}{\partial x_i} x_0 = 0. \tag{2.9}$$

Partindo-se destes conceitos matemáticos, o método de gradiente descendente é proposto como solução iterativa para encontrar os valores do vetor de coeficientes $\beta = \{\beta_1, \beta_2, \ldots, \beta_p\}$. O algoritmo de implementação consiste em selecionar um valor inicial para o vetor β^0, e a partir deste atualizar os valores β^k de maneira sucessiva, em que $k = 1, 2, 3, \ldots, t$, representa as iterações realizadas, isto é, até que seja encontrado o mínimo local ou até que atinja o valor máximo de k (Equação 2.10). Note que em cada atualização de β^k é considerada a direção oposta ao gradiente de h em β^{k-1}, além disso há o elemento α que determina o tamanho do passo dado em direção ao ponto de mínimo local, também conhecido com taxa de aprendizado (ANDRYCHOWICZ *et al.*, 2016).

$$\beta^k = \beta^{k-1} - \alpha \bigtriangledown h(\beta^{k-1}). \tag{2.10}$$

2.4 Árvore de decisão

Na área de Ciência da Computação, uma Árvore é um grafo definido como um conjunto de elementos denominados nós. O primeiro, de cima para baixo, é chamado de nó raiz. Este estabelece uma estrutura hierárquica sobre os nós subsequentes, denominados filhos. Em Aprendizado de Máquina, o conceito de Árvore pode ser usado em algoritmos supervisionados com o intuito de criar um modelo regressivo, quando a variável de interesse é contínua, ou um modelo de classificação, quando a variável de interesse é categórica (MARSLAND, 2015). O foco deste estudo está em algoritmos de classificação.

Algoritmos classificadores baseados em Árvores de Decisão são modelos construídos em uma estrutura de árvore. Em cada nó é tomada uma decisão lógica, com relação a uma das variáveis explicativas. Estes nós, denominados nós internos, se dividem em ramos. Os ramos podem formar mais nós de decisão, ou pode ser um nó terminal, também chamado de folha. Os nós folha indicam a possível categoria das observações, dado o caminho seguido nos nós internos (LANTZ, 2013). A Figura 2 representa um exemplo de Árvore de decisão, proposto por Quinlan (1986), aplicada a um conjunto de dados em que as variáveis explicativas são: Expectativa, Umidade, Vento e temperatura. A variável de interesse (ou variável resposta) é decidir se dadas as combinações de condições climáticas a decisão é jogar Tênis ou não. O nó raiz é a Expectativa, os nós internos são Umidade, Vento e temperatura. As extremidades são os nó folhas, que contem a decisão final de jogar ou não.

Figura 2 – Exemplo de Árvore de Decisão

Fonte: Elaborada pelo autor.

Note que, dada uma base de dados, é possível construir diferentes árvores. De fato algumas árvores irão proporcionar maior precisão do que outras, porém fazer busca por "força bruta" para encontrar a árvore mais eficiente é computacionalmente inviável, uma vez que o espaço de busca é exponencial sobre a quantidade de variáveis explicativas. Para suprir este problema, há alguns algoritmos para encontrar uma árvore que apresente resultados satisfatórios. Todos apresentam solução "gulosa", no sentido de selecionar em cada nó variáveis explicativas que implicam em solução localmente ótimas (TAN *et al.*, 2019). Antes de ilustrarmos o funcionamento de um destes algoritmos, é preciso definir o conceito de ganho de informação.

A decisão de qual variável será associada a cada nó não é realizada de forma aleatória. Cada nó é construído de forma a minimizar a impureza dos dados, impureza aqui é no sentido de conter mais de uma categoria da variável de interesse nos ramos dos nós. Para quantificar a impureza usa-se a medida de entropia, Equação 2.11, em que $p(i \mid t)$ é a proporção da k-ésima categoria (da variável de interesse) no nó t (LANTZ, 2013).

$$Entropia(t) = -\sum_{i=1}^{k} p(i \mid t) log_2 p(i \mid t). \tag{2.11}$$

A partir da medida de entropia defini-se a medida de ganho de informação. O ganho de informação é obtido através da diferença de entropia entre o nó pai e seus filhos (estes ponderados pela quantidade de elementos que contém. Na Equação 2.12, N é o total de observações no nó Pai e N_i é a quantidade de informação no i-ésimo nó filho (LANTZ, 2013).

$$Ganho = Entropia(Pai) - \sum_{i=1}^{j} \frac{N_i}{N} Entropia(Filho_i). \tag{2.12}$$

Agora que foi definido a medida de ganho de informação, pode-se descrever o funcionamento do algoritmo mais usado em Árvore de Decisão, segundo Marsland (2015). O algoritmo ID3 (*Iterative Dichotomizer*) é considerado "guloso" de cima para baixo, sendo que variáveis selecionadas em nós anteriores não são consideradas. Em cada etapa, é selecionada a melhor variável de acordo com o ganho de informação, quanto maior a medida, melhor. O Algoritmo 1 expõe o funcionamento do ID3. A variável do primeiro nó é a que apresenta maior ganho de informação, e as seguintes seguem o mesmo processo de escolha, de forma recursiva, até que se obtenha pureza (apenas uma categoria na folha) ou até que se verifique todos os atributos.

Algoritmo 1 – Algoritmo ID3

Input: D ▷ Base de dados

procedimento ID3(D)
  $Arvore = \{\}$
  se D é puro **então**
    pare
  fim se
  para todo variavel $\in D$ **faça**
    Calcule o ganho de informacao
  fim para
  melhor_variavel = Variavel com maior ganho de informacao
  $Arvore$ = Cria o nó raiz com *melhor_variavel*
  D_v = cria um conjunto de dados sem a variavel *melhor_variavel*
  para todo D_v **faça**
    $Arvore_v$ = ID3(D_v)
    Agrega $Arvore_v$ em $Arvore$
  fim para
  retorna $Arvore$
fim procedimento

2.5 *Gradient boosting decision tree*

Ao abordar o conceito de *Gradient boosting* aplicado à árvores de decisão, o primeiro passo é definir o conceito de *ensemble* e *boosting*.

O método de *ensemble* em aprendizado de máquina consiste em combinar previsões de vários modelos, para obter um modelo final mais assertivo, isto é, robusto. Há duas formas de aplicar o *ensemble*: através de abordagem paralela ou sequencial. Na abordagem paralela, são treinados vários modelos simultaneamente (e com diferentes amostras aleatórias dos dados), sendo o resultado final a combinação dos modelos treinados, por exemplo, a resposta média para variáveis de interesse contínuas ou votação, para variáveis categóricas. Na abordagem sequencial, os modelos são treinados sequencialmente, mas ao invés de considerar amostras aleatórias, os dados são selecionados de forma a reduzir o erro cometido pelo modelo anterior. O *boosting* é uma forma de abordagem sequencial de *ensemble*, em que o procedimento de treino dos modelos consiste em ajustar modelos voltados para as observações cujo o modelo anterior errou (BüHLMANN, 2012).

Seja $D = \{(x_i, y_i)_{i=1}^n\}$ um conjunto de dados com n observações, onde $x \in \mathbb{R}^m$ e $F^* : \mathbb{R}^m \to \mathbb{R}$ é o conjunto de funções que mapeiam x em y (um escalar). O *Gradient Boosting Decision Tree*, também conhecido por GBDT, é um método de *ensemble* via *boosting* que integra M árvores de decisão de forma aditiva, conforme Equação 2.13. O primeiro modelo $f_0(x)$ é a primeira árvore ajustada e $\{f_k(x)\}_1^M$ são modelos incrementais que visam melhorar os erros

cometidos pelos modelos anteriores.

$$F(x) = \sum_{k=0}^{M} f_k(x). \tag{2.13}$$

Em outras palavras, a ideia é propor um modelo base (weak learner) e o aprimorar (ou "ensinar") a cada iteração, em que os ajustes seguintes F_k são calculadas com base no gradiente da função de risco esperado do modelo anterior $R(F_{k-1})$, conforme exposto na Equação 2.14, em que $E(L(y, F_{k-1}(x))$ é o valor esperado da função de perda L aplicada no modelo anterior F_{k-1}, $F_{k-1}(x) = \sum_{i=0}^{k-1} F_i(x)$ e ρ_k é o tamanho do passo em direção oposta ao gradiente de $R(F_{k-1})$, definido em Equação 2.15.

$$F_k(X) = -\rho_k \bigtriangledown R(F_{k-1}(X)) = -\rho_k \left[\frac{\partial E(L(y, F_{k-1}(X)) \mid X)}{\partial F_{k-1}(X)} \right], \tag{2.14}$$

$$\rho_k = \arg\min_{\rho} (\rho_{k-1} F_{k-1}(X) - \rho_k F_k(X)) \tag{2.15}$$

Neste contexto, considera-se como modelo base a árvore de classificação binária com $y = \{-1, 1\}$, expressa na Equação 2.16. O termo $R_{j_i}^J$ diz respeito à regiões disjuntas cuja a união cobre todo o espaço dos valores de x. Cada região é representada pelo j-ésimo nó terminal da árvore. O valor b_j representa o valor de y mais frequente na região R_j e I é o indicador que assume o valor 1 se $(x \in R_j)$ e 0, caso contrário.

$$h(x; \{b_j, R_j\}_1^J) = \sum_{j=1}^{J} b_j I(x \in R_j). \tag{2.16}$$

Segundo Friedman (2001), a atualização descrita na equação Equação 2.14 pode ser reescrita conforme Equação 2.17:

$$F_k(X) = F_{k-1}(x) - \sum_{j=1}^{J} \gamma_{jk} I(x \in R_{jk}), \tag{2.17}$$

em que γ_{jk} por meio da função de perda L aplicada ao modelo anterior, Equação 2.18.

$$\{\gamma_{jk}\}_1^J = \arg\min_{\{\gamma_j\}_1^J} \sum_{i=1}^{N} L \left(y_i, F_{k-1}(x) - \sum_{j=1}^{J} \gamma_j I(x \in R_{jk}) \right). \tag{2.18}$$

Observa-se que se o objetivo é a classificação binária, a função de perda pode ser escrita como a função log-verossimilhança negativa (Equação 2.19a), dependente da função $G(x)$ exposta em Equação 2.19b.

$$L(y, G(x)) = \log[1 + \exp(-2yG(x))], \tag{2.19a}$$

$$G(x) = \frac{1}{2}\log\left[\frac{P(y=1 \mid x)}{P(y=-1 \mid x)}\right]. \tag{2.19b}$$

Sendo assim, no artigo de Friedman (2001) é sugerida a adaptação para o uso em árvores de decisão, Equação 2.20, pautada na deparabilidade de cada nó terminal da região R_{jk}.

$$\gamma_{jk} = \arg\min_{\gamma} \sum_{x_i \in R_{jk}} \log[1+\exp(-2y_i(F_{k-1}(x_i)+\gamma))]. \tag{2.20}$$

Para simplificar o cálculo de γ_{jk}, é possível aplicar o método de Newton–Raphson e obter:

$$\gamma_{jk} = \frac{\sum_{x_i \in R_{jk}} \hat{y}_i}{\sum_{x_i \in R_{jk}} \mid \hat{y}_i \mid (2- \mid \hat{y}_i \mid)},$$

em que

$$\hat{y}_i = -\left[\frac{\partial L(y_i, F_{k-1}(x_i))}{\partial F_{k-1}(x_i)}\right] = \frac{2y_i}{1+\exp(2y_i F_{k-1}(x_i))}.$$

Desta forma o modelo final $F(x)$ é uma aproximação da razão de chances, devido a Equação 2.19b. A probabilidade de y dado x é calculada com Equação 2.21.

$$\begin{aligned} p_+(x) &= \hat{P}(y=1 \mid x) = \frac{1}{1+\exp(-2F(x))}, \\ p_-(x) &= \hat{P}(y=-1 \mid x) = \frac{1}{1+\exp(2F(x))}. \end{aligned} \tag{2.21}$$

O procedimento descrito nesta seção, por der resumido conforme Algoritmo 2.

Algoritmo 2 – Algoritmo GBDT

Input: $D = \{(x_i, y_i)_{i=1}^n\}$
procedimento GBDT(D)
  para todo variavel $\in D$ **faça**
    $\hat{y}_i = 2y_i/(1+exp(2y_i F_{k-1}(x_i)))$, com $i = 1, 2, \cdots, N$
    $\{R_{jk}\}_1^J$ = j-ésimo nó terminal da $arvore(\hat{y}_i, x_i)_1^N)$
    $\gamma_{jk} = \sum_{x_i \in R_{jk}} \hat{y}_i / (\sum_{x_i \in R_{jk}} \mid \hat{y}_i \mid (2- \mid \hat{y}_i \mid))$
    $F_k(X) = F_{k-1}(x) - \sum_{j=1}^J \gamma_{jk} I(x \in R_{jk})$
  fim para
fim procedimento

2.5.1 Light Gradient Boosting Machines

O *Light Gradient Boosting Machines*, ou LGBM, é um aprimoramento do GBDT, proposto por Ke *et al.* (2017), que consiste em dois pontos de otimizações no processo de construção das árvores que compõem o modelo final.

Em GBDT durante a construção das árvores, observações com gradientes diferentes desempenham papéis diferentes no cálculo do ganho de informação. Conforme visto na Equação 2.12, observações com maiores erros (gradientes maiores) contribuirão mais para o ganho de informação. Logo, se uma observação estiver associada a um pequeno gradiente, o erro de treinamento para essa observação é pequeno e já está bem treinado. Neste sentido, a ideia é descartar essas observações de dados com pequenos gradientes. Porém, ao realizar tal procedimento a distribuição dos dados seria alterada, o que prejudicaria a precisão do modelo. Para evitar este problema, a proposta do artigo é a aplicação da técnica Amostragem unilateral baseada em gradiente (*Gradient-based One-Side Sampling*, GOSS). O funcionamento é descrito abaixo:

1. Ordena as observações de acordo com o valor absoluto de seus gradientes;
2. Seleciona as primeiras $a \times 100\%$ das observações;
3. Amostra aleatoriamente $b \times 100\%$ das observações restantes;
4. Aplica o peso constante $w = (1-a)/b$ na amostra, para assim calcular ganho de informação com os dados descritos nos itens 2 e 3.

A amostragem via GOSS tem como escopo manter todas as observações com gradientes grandes e realizar amostragem aleatória nas observações com gradientes baixos. Para minimizar a influência na distribuição dos dados, ao computar o ganho de informação, é atribuído um peso constante para as instâncias de dados com pequenos gradientes. Assim, o foco está em observações com maiores erros sem alterar muito a distribuição original do dados.

A segunda melhoria proposta é o Pacote de recursos exclusivos (*Exclusive Feature Bundling*, EFB). No geral o conjunto de variáveis explicativas x (recursos) são esparsos. Em seu artigo, Ke *et al.* (2017) discute o fato de que no espaço em que x, o conjunto de variáveis explicativas, é esparso, muitas destas variáveis são mutuamente exclusivas, isto é, nunca assumem valores diferentes de zero simultaneamente. Sendo assim, a ideia é agrupar com segurança variáveis exclusivas em um único grupo de variáveis (EFB). Desta forma, a complexidade para criar quebras das variáveis que definem as folhas das árvores muda de $O(\#dados \times \#variveis)$ para $O(\#dados \times \#EFB)$, sendo enquanto $\#EFB \ll \#variveis$.

CAPÍTULO

3

SOBRE AS BASES DE DADOS

No setor bancário, é comum que modelos sejam desenvolvidos com o intuito de realizar concessões de crédito de maneira mais assertiva. Cada produto (financiamento imobiliário, empréstimos pessoal, cartão de crédito, dentre outros) é concedido de acordo com a pontuação de crédito (*Credit Score*) atribuído ao cliente via modelo. Apesar de o modelo ser desenvolvido em nível granular cliente/produto, no geral as análises são realizadas considerando a proporção/taxa de maus pagadores ao longos dos meses, uma vez que o modelo é construído para reduzir este número (BENZSCHAWEL, 2012). Neste sentido, as análises apresentadas nas próximas seções serão guiadas na visão período a período.

3.1 Hipotecas residenciais

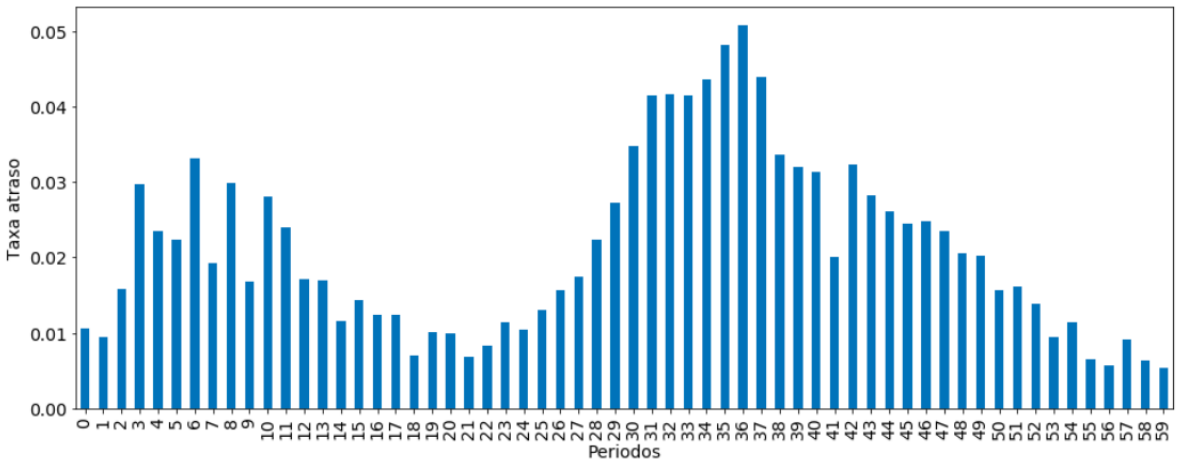
Esta base é formada com dados de hipotecas residenciais de 50 mil mutuários dos EUA, observados ao longo de 60 períodos. Vale ressaltar que não necessariamente um mutuário está presente durante todos os períodos, isso porque pode ter ocorrido transferência de empréstimos entre bancos e investidores como na securitização ou pode haver censura do empréstimo à medida que vence ou o mutuário refinancia. O conjunto de dados foi originado a partir de uma amostra aleatória de empréstimos hipotecários coletados das carteiras de securitização de títulos lastreados em hipotecas residenciais (*Residential Mortgage-Backed Securities*, RMBS) dos EUA. A base está disponível em Analytics (2016).

3.1.1 Análise descritiva

Há no total 622.489 observações na base de dados. A variável de interesse (default_time) indica se há atraso no período de observação, sendo assim assume o domínio 1 para atraso e 0 caso contrário. No total há 2.466 observações em que a variável resposta é igual a 1, logo a taxa de atraso total é 0,0270. Ao analisar a taxa periodicamente, Figura 3, verifica-se que há

movimento de ciclo temporal, em que no segundo ciclo ocorre aumento no patamar da taxa de atraso.

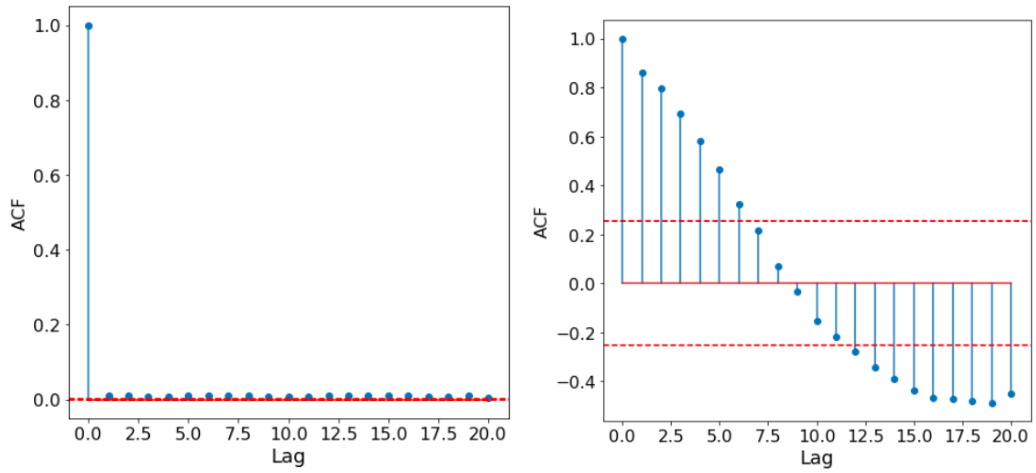
Figura 3 – Taxa de atraso ao logo dos períodos



Fonte: Elaborada pelo autor.

Quando se trata de modelagem de dados, uma premissa importante é a independência entre as observações. A base de dados foi ordenada por período para avaliar a autocorrelação entre as observações (linhas) com relação à variável de interesse, registrada na Figura 4a. Note que ao considerar as informações desagregadas, a autocorrelação relativa ao atraso não é estatisticamente significante. Entretanto, quando considerada a base em que o atraso médio é agregado periodicamente, isto é, a taxa de atraso, verifica-se que há correlação, Figura 4b. As 6 primeiras *lags* estão fora do intervalo de confiança, indicando que a taxa de atraso em um instante t sofre influencia dos 6 períodos anteriores.

Figura 4 – Gráficos de autocorrelação (ACF), para observações da base de hipotecas



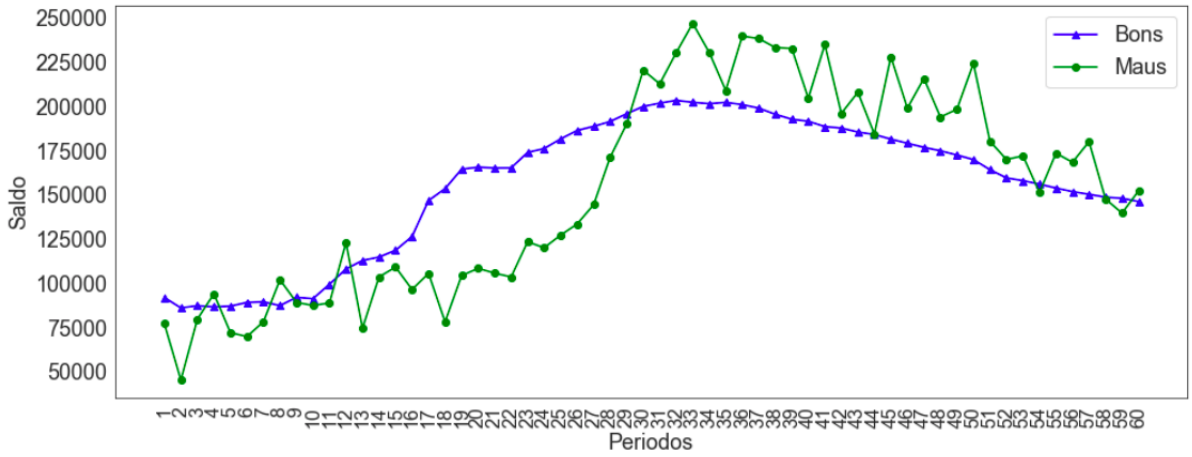
(a) ACF por observação

(b) ACF por taxa de atraso

Fonte: Elaborada pelo autor.

Das 14 variáveis disponíveis na base de dados, optou-se por analisar 4 delas, conforme discutido nos próximos parágrafos. A Figura 5 representa o saldo da hipoteca, isto é, o valor devido no período observado. De acordo com o exposto por Segal (2019), os pagamentos de hipotecas durante os primeiros anos consistem, principalmente, em pagamentos de juros, enquanto os pagamentos posteriores consistem principalmente em amortizar o valor emprestado (principal). Este fato justifica a tendência crescente das curvas. Além disso, note que nos períodos iniciais o saldo mediano devido é maior para bons pagadores (curva azul), quando comparado com as pessoas que atrasaram o empréstimo (curva verde).

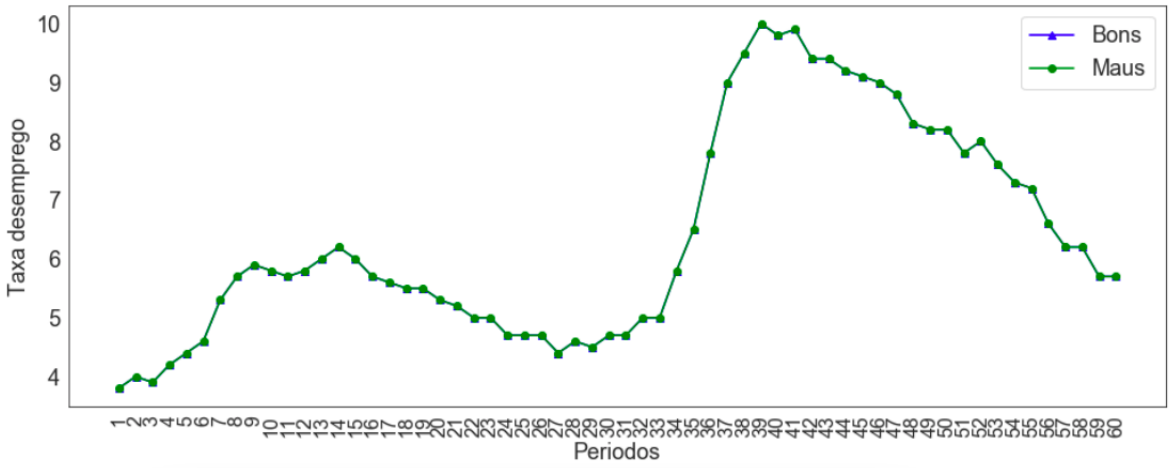
Figura 5 – Distribuição da mediana do saldo de hipoteca pendente, ao longo dos períodos, para bons e maus pagadores



Fonte: Elaborada pelo autor.

A taxa de desemprego, Figura 6 não diferencia bons e maus pagadores, pois as curvas se sobrepõem. Entretanto essa variável explica a tendência do LTV (*Loan-To-Value*), que, aparentemente, tende a discriminar os dois grupos.

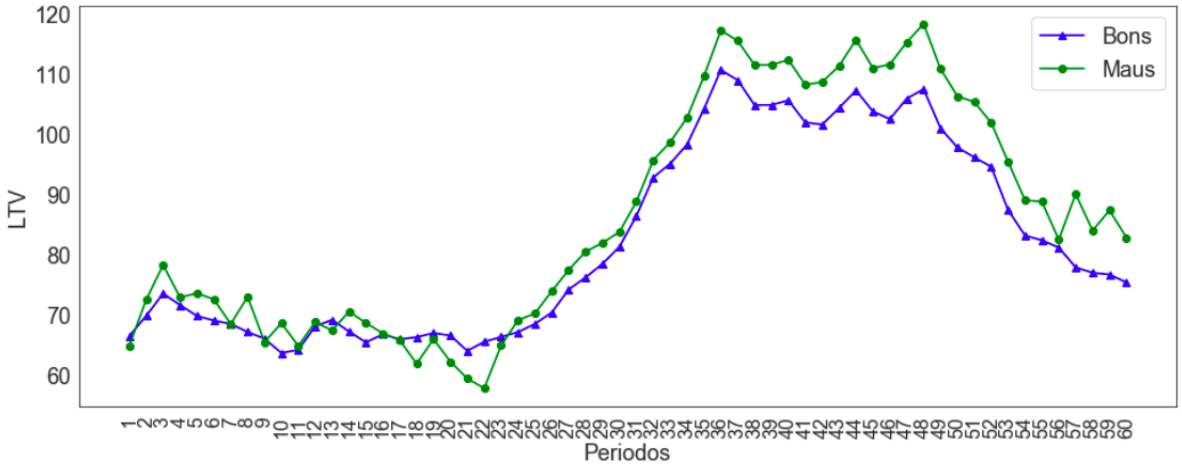
Figura 6 – Distribuição da taxa de desemprego, ao longo dos períodos, para bons e maus pagadores



Fonte: Elaborada pelo autor.

O coeficiente empréstimo-valor (*Loan-To-Value*, LTV) é uma avaliação do risco de crédito usada por credores e instituições financeiras na concessão de uma hipoteca. Altos índices de LTV indicam alto risco oferecido pelo cliente para honrar o empréstimo (HAYES, 2020). A Figura 7 mostra que a partir do vigésimo quarto período, estes índices possuem tendência crescente, voltando a decair a partir do período 50. Observe que os valores medianos do LTV por período tem o mesmo movimento que a taxa de desemprego (Figura 6), o que pode explicar o aumento do índice LTV.

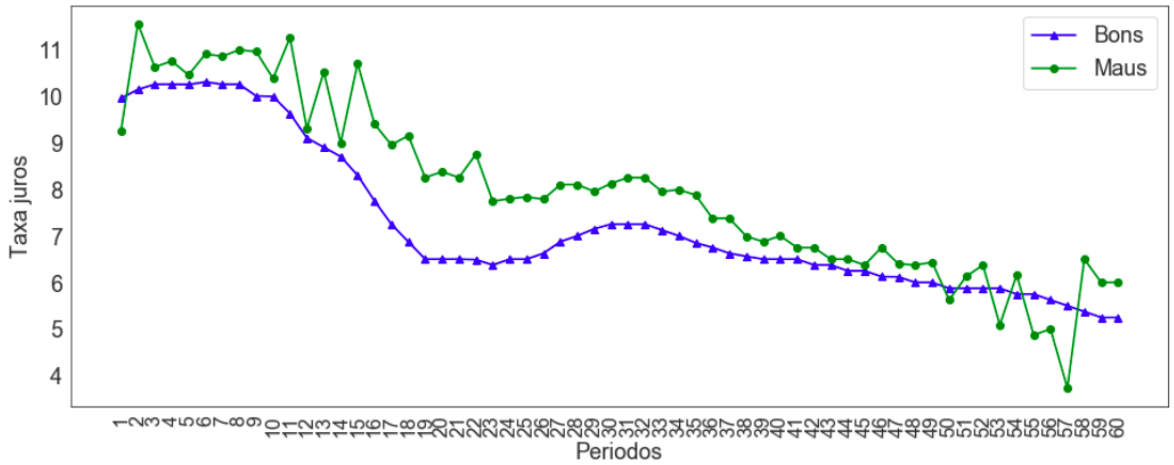
Figura 7 – Distribuição da mediana do coeficiente empréstimo-valor, ao longo dos períodos, para bons e maus pagadores



Fonte: Elaborada pelo autor.

Com relação à taxa de juros, na Figura 8 observa-se que a tendência é de queda ao longo dos períodos, movimento contrário ao da taxa de desemprego. De fato, a correlação dessas variáveis (considerando a mediana por safra) é de -0,4937, isto é, relativamente forte e negativa (Figura 9b).

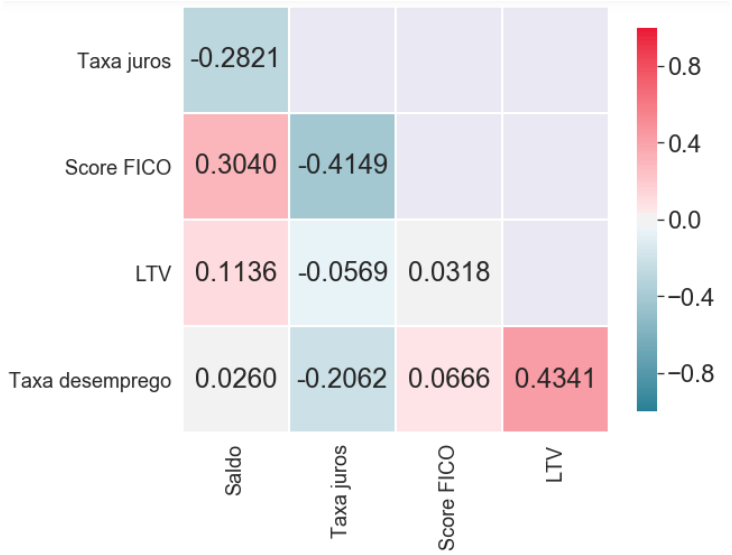
Figura 8 – Distribuição da taxa de juros, ao longo dos períodos, para bons e maus pagadores



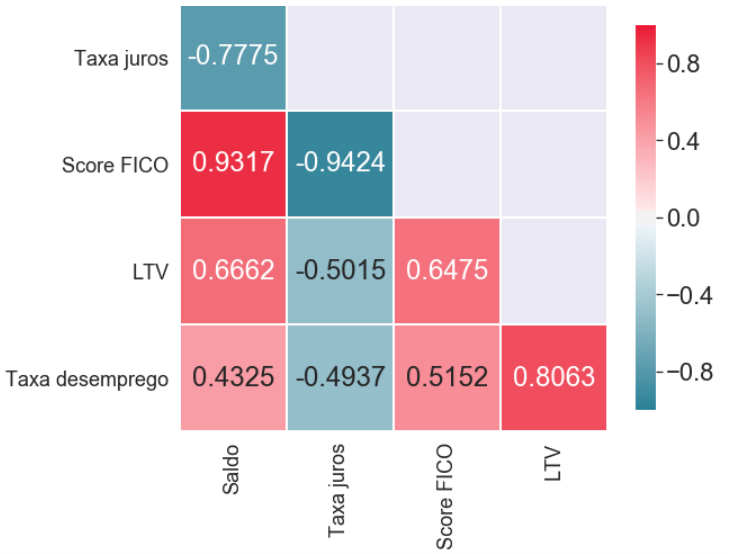
Fonte: Elaborada pelo autor.

Por outro lado, a correlação entre essas variáveis diminui, de forma significativa, quando calculada com os dados desagregados, isto é, cliente a cliente por período (Figura 9a).

Figura 9 – Correlação entre as vaiáveis, base de hipoteca



(a) Correlação dados desagregados



(b) Correlação mediana por período

Fonte: Elaborada pelo autor.

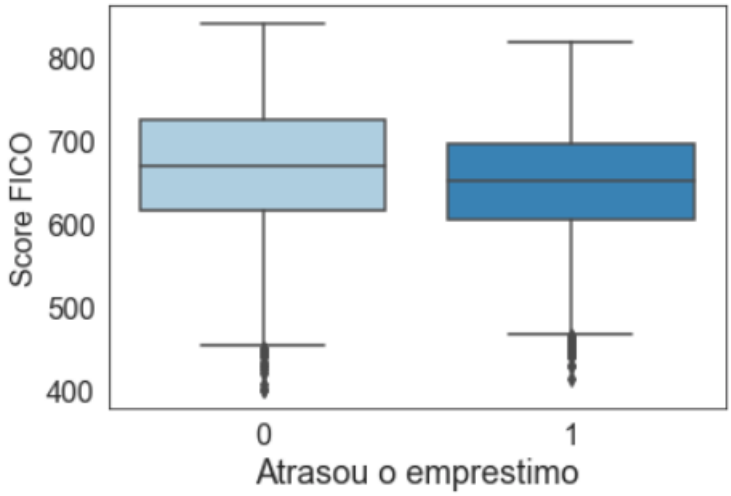
Na base de dados há uma variável referente ao *Score* FICO no momento da concessão do empréstimo. Trata-se de uma pontuação de risco de crédito, atribuída para cada pessoa, com o intuito de identificar bons e maus pagadores. Este é um score amplamente usado por credores e financeiras com o objetivo de tomar decisões mais assertivas (FICO, 2020). A Figura 10 foi construída considerando cada cliente e seu respectivo *Score* no momento da contratação da hipoteca. Note que a distribuição dos valores entre bons pagadores (0) e maus pagadores (1) é semelhante, isto é, a diferença entre os dois grupos é sutil. As informações do Quadro 1 mostram

que (com exceção do valor mínimo do *Score*) todas as estatísticas apontam que maus pagadores (atraso igual a sim) possuem valores menores para esta variável. O teste de hipótese para média do *Score* FICO resulta em $p - valor < 0.0001$, logo esta diferença é estatisticamente significante com nível de confiança de 1%.

Quadro 1 – Medidas resumo *Score* FICO

Atraso	Mínimo	Média	Mediana	Máximo	Desvio Padrão
Sim	413	649	651	818	67
Não	400	667	669	840	74

Figura 10 – Boxplot *Score* FICO



Fonte: Elaborada pelo autor.

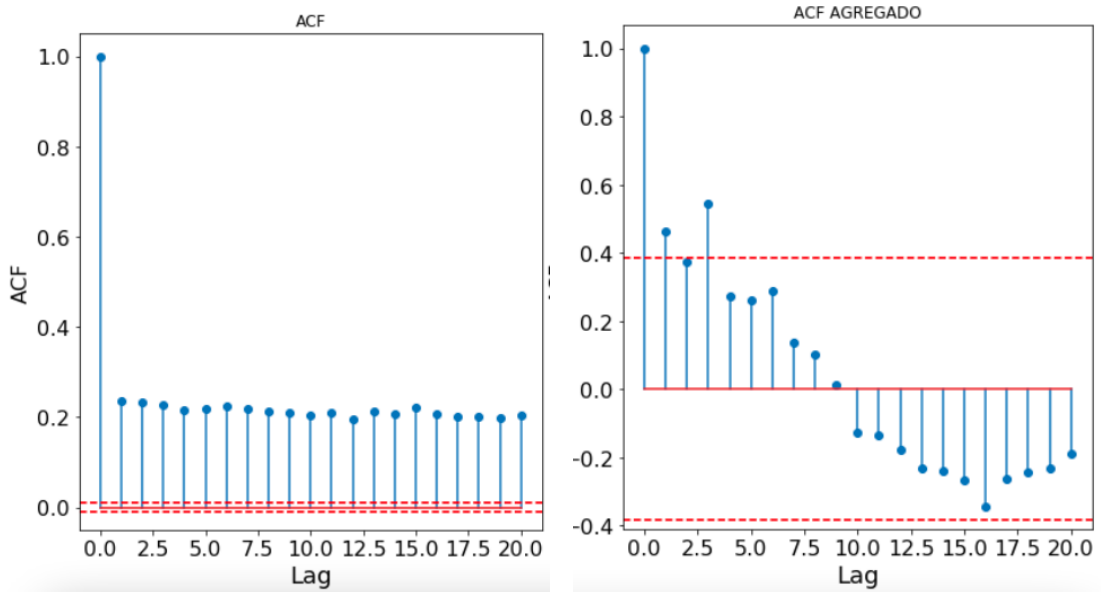
3.2 *Marketing* bancário

O *telemarketing* é definido como uma forma de vender, solicitar ou promover um produto ou serviço pelo telefone. O ato de contactar pessoas/clientes por telefone é uma estrategia típica para aprimorar os negócios. Trata-se de uma forma de realizar *marketing* direto ao público alvo de maneira econômica e flexível, no sentido que a abordagem é feita para o indivíduo (VENCO, 2010). O conjunto de dados, disponível em Repository (2020), é referente a campanha de vendas do produto "depósito de longo prazo" através de *telemarketing*, realizada por um banco português. A abordagem dos clientes é realizada de maneira ativa, quando os funcionários do banco entram em contato com o cliente para vender o produto, e passiva, quando o cliente entra em contato por qualquer motivo e o funcionário aproveita para oferecer o produto. Os dados foram coletados entre maio de 2008 e junho de 2013, totalizando 52.944 contatos telefônicos. A análise descritiva abaixo será guiada para a variável de interesse, isto é, se o cliente contratou ou não o produto vendido.

3.2.1 Análise descritiva

O primeiro ponto relevante é a autocorrelação entre as observações da base de dados. Para calcular a autocorrelação dos dados desagregados (comparando observação a observação) as observações foram ordenadas apenas pela variável temporal. O cálculo da correlação foi realizado sobre a variável resposta (contratação ou não do depósito a longo prazo). Observa-se na Figura 11a que os dados, quando comparados um a um, possuem alta correlação. Com relação à autocorrelação da taxa mensal de vendas, Figura 11b, a venda de um dado mês é significativamente influenciada pela venda do mês anterior.

Figura 11 – Autocorrelação (ACF) para a variável de contratação de depósito a longo prazo

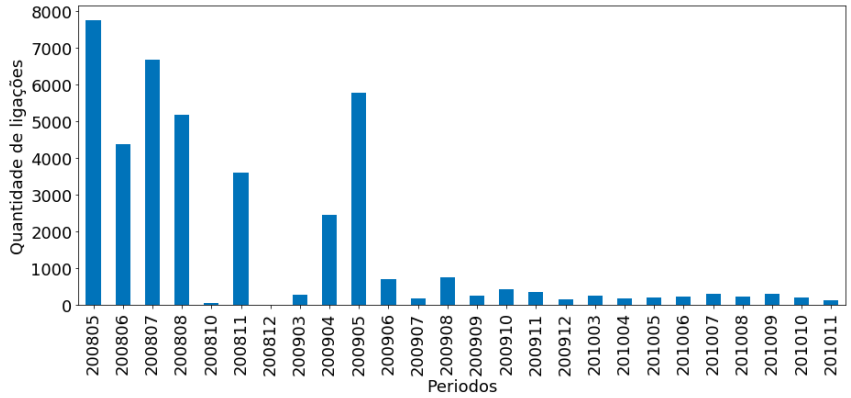


(a) ACF dados desagregados (b) ACF taxa de vendas mensal

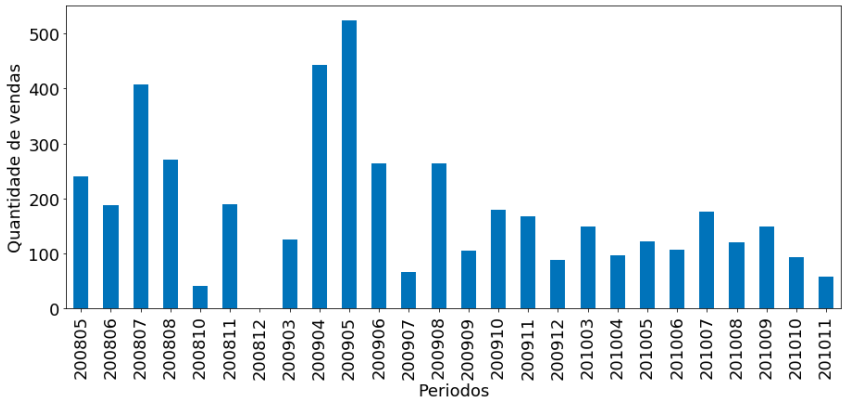
Fonte: Elaborada pelo autor.

Ao avaliar a quantidade de ligações e a taxa de vendas mensais (Figura 12), observa-se que há nítida quebra de tendência nos dados após abril de 2009. A quantidade de ligações realizadas por mês, que apresentava tendência de queda entre maio de 2008 e abril de 2009, ganha estabilidade nos meses subsequentes em um patamar mais baixo (Figura 12a). Embora a quantidade de vendas também tenha reduzido (Figura 12b), tornou-se mais estável. Note que nos últimos meses observados a taxa de vendas é maior, quando comparada com os primeiros meses (Figura 12c). Com estas informações, pode-se pressupor que, aparentemente houve mudança na estrategia de vendas via *telemarketing*, considerando que foram realizadas menos ligações, porém mais assertivas, guardadas as proporções.

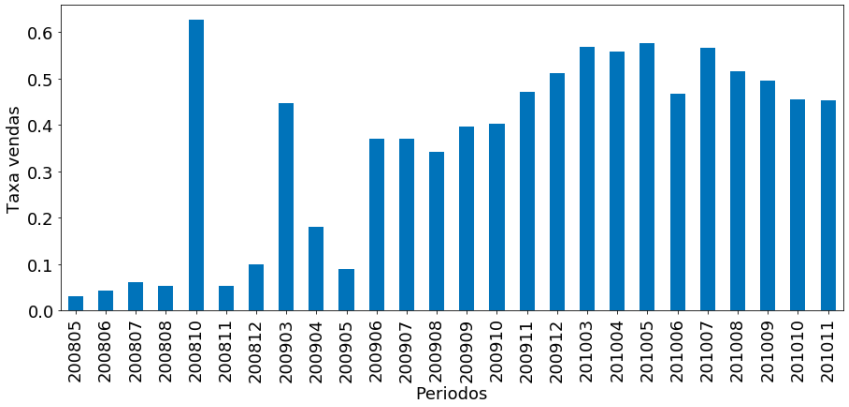
Figura 12 – Gráficos de quantidades de ligações, vendas e taxa de vendas, em cada mês



(a) Quantidade de ligações, por mês



(b) Quantidade de vendas, por mês



(c) Taxa de vendas, por mês

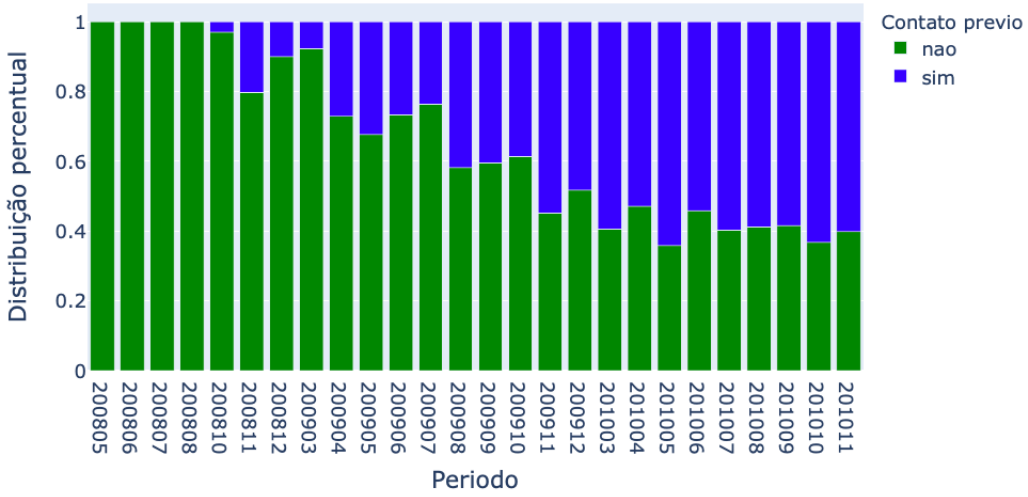
Fonte: Elaborada pelo autor.

Os dados mostram que a chance de cliente contatado por celular comprar o produto é 3 vezes maior, quando comparado com cliente acionado por telefone fixo. Além disso, clientes que recebem muitas ligações, tendem a não contratar o produto. O percentual de clientes que receberam até 3 ligações e contrataram o produto é 12%, esse número cai para 7% quando a quantidade de ligações é superior a 3. Outro número interessante é que a quantidade de clientes que possuía algum atraso bancário e contratou a oferta foi, exatamente, 3 (o que representa

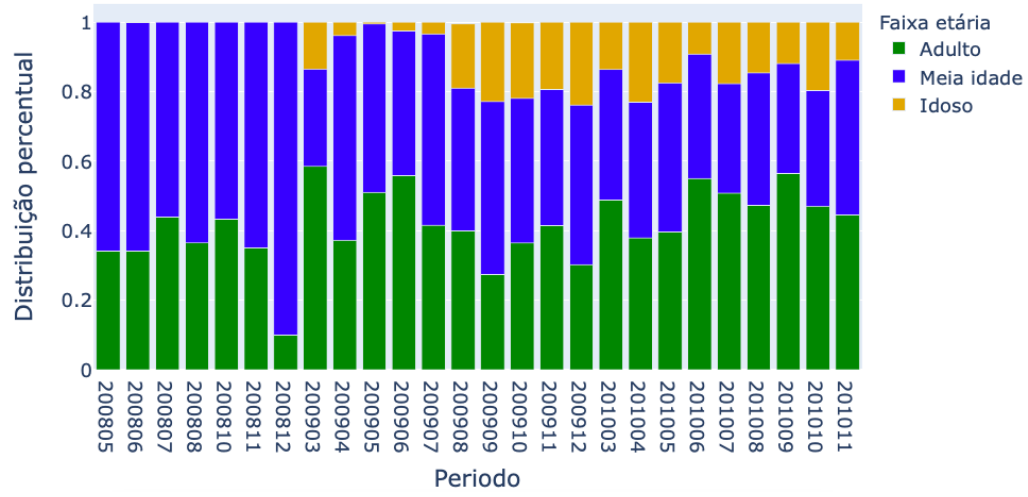
aproximadamente 0% deste público). Por outro lado, dentre os clientes que não possuíam atraso, o percentual de aquisição do produto foi 13%.

Corroborando o pressuposto de mudança de estratégia, observe na Figura 13a que nos primeiros meses, para a maioria dos clientes, a ligação registrada nos dados era a primeira recebida. Mas nos últimos meses, esse percentual cai. Indicando, que além de reduzirem a quantidade de ligações realizadas, houve foco em alguns clientes. Já na Figura 13b, é exposto que o grupo de pessoas idosas (com idade maior que 61 anos) tornou-se alvo das campanhas de *telemarketing*.

Figura 13 – Gráficos distribuição das variáveis contato prévio e faixa etária



(a) Proporção de clientes que receberam contato prévio, por mês



(b) Distribuição faixa etária clientes, por mês

Fonte: Elaborada pelo autor.

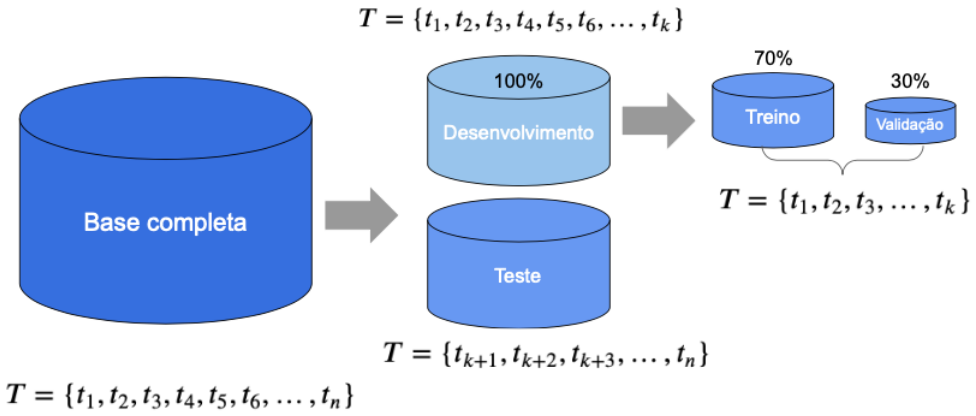
CAPÍTULO

4

AJUSTE DE MODELOS E RESULTADOS

Neste capítulo é apresentado o precedimento de modelagem dos dados, bem com a discussão dos resultados. Antes do ajuste dos modelos, cada base de dados foi dividida em três partes: treino, validação e teste. Para cada base, B, foi definido com conjunto de períodos (ou meses), P, em que o modelo foi desenvolvido. Desta forma, as bases foram divididas em duas B_1 e B_2, em que, considerando a variável temporal $T = \{t_1, t_2, t_3, \ldots, t_n\}$, $B_1 = \{B \mid T \subset P\}$ e $B_2 = \{B \mid T \not\subset P\}$. O subconjunto de dados B_2 é o teste. Em B_1 foi realizado uma amostra aleatória em que 70% dos dados foram separados para treino e 30% para validação (Figura 14). O objetivo de dividir a bases desta forma é avaliar a capacidade de generalização dos modelos em dois cenários: o primeiro é quando as observações não foram usadas para o treino, mas pertence ao período de desenvolvimento (validação); o segundo é para observações não presentes no treino e no período de desenvolvimento. Avaliar os resultados sob estas perspectivas traz a ideia, mesmo que empírica, dos efeitos das correlações intra-amostral (validação) e temporal (teste).

Figura 14 – Esquema divisão da base de dados em treino, validação e teste



Fonte: Elaborada pelo autor.

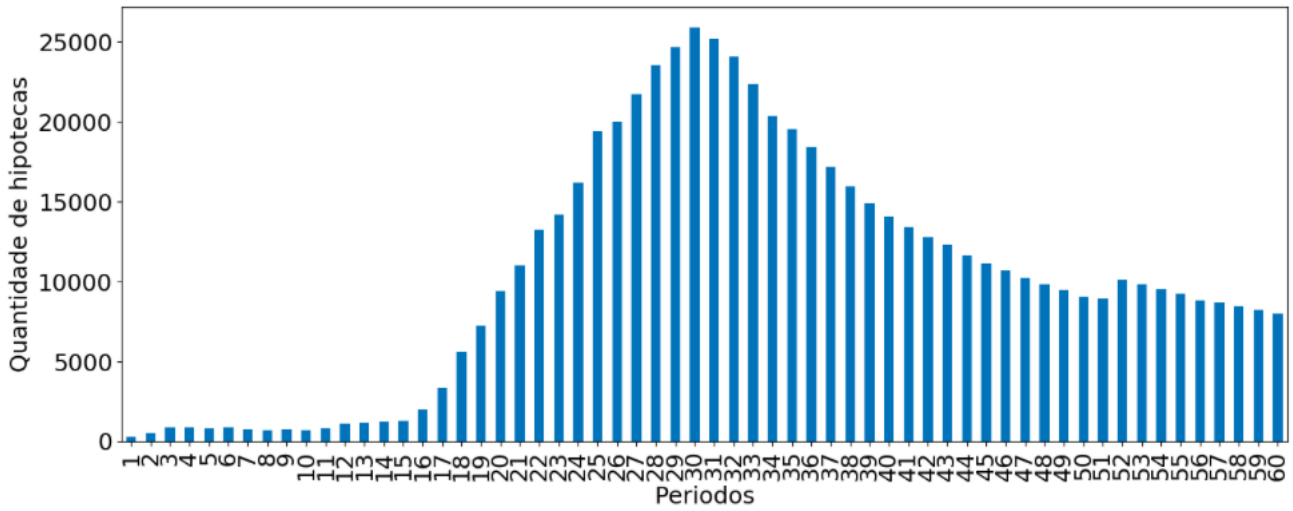
4.1 Base de dados hipotecas residenciais

Conforme mencionado na seção Seção 3.1, a base é composta por dados observados ao longo de 60 períodos. Considerando o comportamento das variáveis explicativas, em que nos últimos períodos as curvas medianas de bons e maus pagadores tendem a se encontrarem (o que indica menor discriminação), o estudo será realizado considerando apenas os primeiros 36 períodos. Sendo assim, tem-se as seguintes divisões do conjunto de dados:

- Treino: amostra aleatória de 70% conjunto de dados, em que período é menor ou igual a 30;
- Validação: são os 30% restantes do conjunto de dados (cujo período é menor ou igual a 30), após a amostragem do treino;
- Teste: todas as amostras em que o período é maior que 30 e menor ou igual a 36.

Observe no Quadro 2 que a base é desbalanceada, isto é, a quantidade de clientes que atrasaram (maus pagadores) é muito inferior à quantidade de clientes que que pagaram em dia. Nos subconjuntos de treino e validação, apenas 1,5% dos clientes atrasaram, enquanto que no subconjunto de teste, este percentual é de 4,1%. A proximidade entre a taxa de maus pagadores nas bases de treino e validação, é consequência da amostragem aleatória dento do mesmo intervalo de tempo. O volume de hipoteca mês a mês não tem distribuição uniforme (Figura 15), entre os períodos 15 e 30 há tendência crescente e a partir deste período, tende a cair.

Figura 15 – Volume de hipoteca, por período



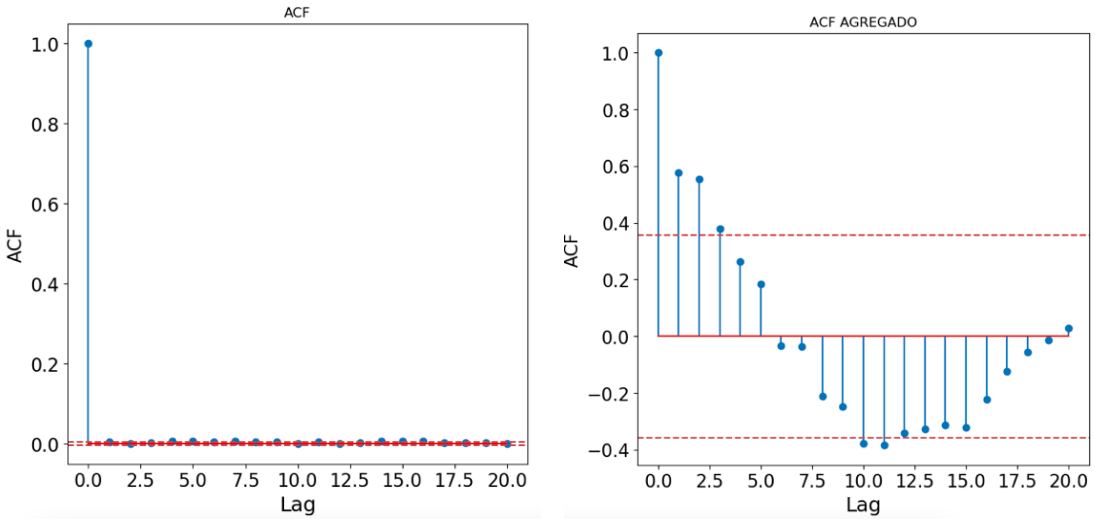
Fonte: Elaborada pelo autor.

Quadro 2 – Divisão base de dados hipotecas

Subconjunto	Períodos	N	Maus	Taxa maus
Treino	[1; 30]	161.037	2.466	0,0153
Validação	[1; 30]	69.016	1.055	0,0153
Teste	(30;36]	129.961	5.388	0,0415

Considerando-se que houve uma seleção dos períodos de desenvolvimento do modelo (treino, validação e teste), é interessante reavaliar a autocorrelação dos dados neste recorte (primeiros 36 períodos) da base de dados, Figura 16. Note que com relação aos dados desagregados, isto é, observação a observação, ainda há baixa autocorrelação (Figura 16a), quando comparada com a base completa (Figura 9a). A diferença entre a base completa e o recorte realizado está na avaliação da autocorrelação da taxa de atraso por período. Se com a base completa a taxa de atraso em um dado período tinha dependência de até 6 períodos anteriores (Figura 9b), no subconjunto de dados usado no desenvolvimento há influência significativa dos três meses anteriores (Figura 16b).

Figura 16 – Gráficos de autocorrelação (ACF), para observações da base de hipotecas e nos períodos desenvolvimento do modelo



(a) Autocorrelação dados desagregados (b) Autocorrelação taxa atraso por período

Fonte: Elaborada pelo autor.

Com o subconjunto de treino, foi ajustado o modelo com o uso da técnica de classificação LightGBM (*Light Gradient Boosting Machine*), com taxa de aprendizado igual a 0,0010, profundidade máxima das árvores igual a 4 e 600 estimadores com amostragem de 70% da base de treino. Além destes hiper-paramentos, devido a baixa taxa de atraso, realizou-se balanceamento, ou seja, amostragem ponderada, em que observações com variável resposta 1 tem peso 64 vezes maior, quando comparada com resposta 0.

Para avaliar a acurácia, sensibilidade e especificidade do modelo, foi considerado o limiar

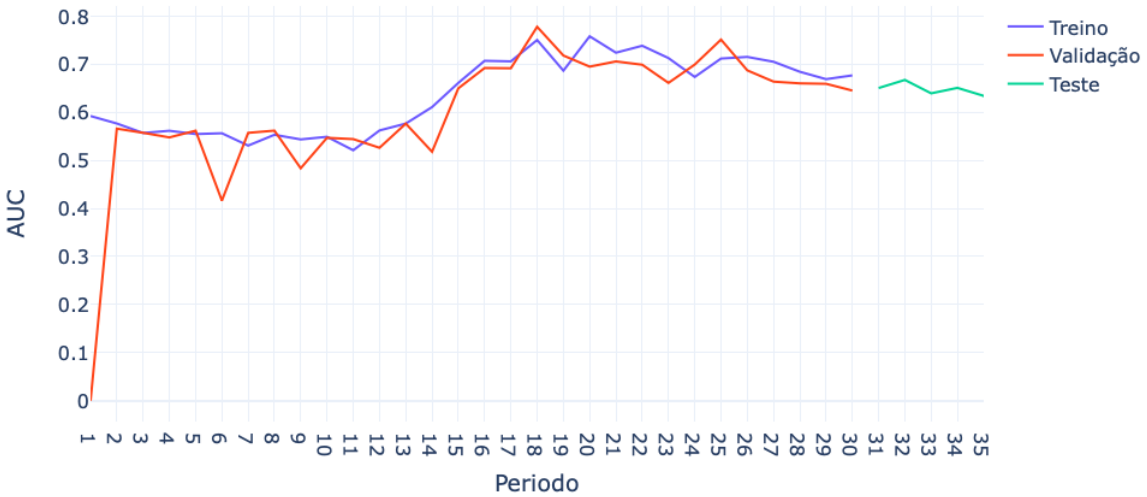
igual a 0,5000. Sendo assim, para observações em que a probabilidade de atraso for maior ou igual ao limiar, então o valor inferido é 1 (atraso) e zero, caso contrário. No geral, o modelo tem boa sensibilidade, isto é, boa capacidade em predizer maus pagadores. Porém, a performance é menor sob a ótica de identificar bons pagadores.

Quadro 3 – Resultado modelo para base de dados hipotecas

Subconjunto	Acurácia	Sensibilidade	Especificidade	AUC
Treino	0,5697	0,8386	0,5655	0,7661
Validação	0,5673	0,8038	0,5636	0,7532
Teste	0,5302	0,7506	0,5207	0,6917

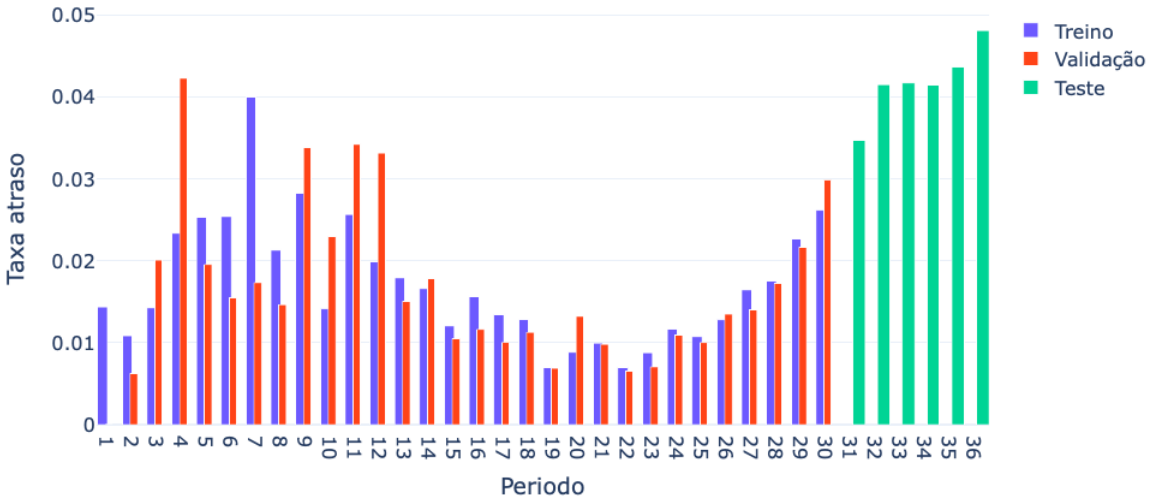
No Quadro 3, note que as métricas consolidadas para os subconjunto de treino e validação são próximas, porém quando comparadas aos dados de teste a distância torna-se maior. Quando a métrica AUC é calculada período a período (Figura 17), as curvas de treino e validação estão muito próximas (em alguns pontos, sobrepostas) com exceção do primeiro período em que a quantidade de maus na base de validação é zero, conforme mostrado em Figura 18. Em contrapartida no período de teste há queda no patamar de AUC, no último ponto do treino a métrica tem valor 0,6768, contra 0,6342 no ultimo período do teste, queda de 0,0426 de AUC.

Figura 17 – AUC do modelo para os dados de hipoteca, por período, para os subconjuntos de treino, validação e teste



Fonte: Elaborada pelo autor.

Figura 18 – Taxa de atraso, por período, para os subconjuntos de treino, validação e teste



Fonte: Elaborada pelo autor.

O código, em linguagem Python, usado no desenvolvimento do modelo para este conjunto de dados, está disponível em Apêndice A.

4.2 Base de dados *marketing* bancário

Conforme exposto em Figura 12a, a quantidade média de ligações realizadas entre maio de 2008 e 2009 é 3.622, contra 310 referente a junho de 2009 a novembro de 2011, o que indica mudança de comportamento na série histórica. Por este motivo, optou-se por trabalhar com os dados datados a partir de julho de 2009, em que:

- Treino: compreende dados de julho de 2009 a junho de 2010, inclusive. Desta janela temporal, foi retirada amostra aleatória de 70% conjunto de dados.
- Validação: são os 30% complementares ao conjunto de treino, considerando a mesma janela temporal;
- Teste: todas as observações obtidas de julho a novembro de 2011.

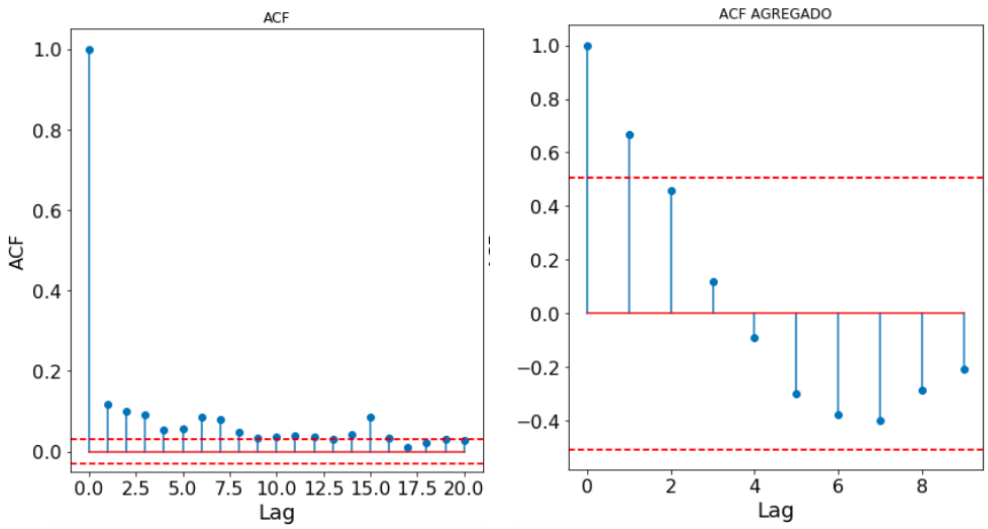
Quadro 4 – Divisão base de dados *marketing* bancário

Subconjunto	Períodos	N	Vendas	Taxa vendas
Treino	[julho/2009; junho/2010]	2.149	951	0,4425
Validação	[julho/2009; junho/2010]	921	397	0,4311
Teste	[julho/2010; novembro/2010]	1.179	597	0,4936

No Quadro 4, observa-se que embora a quantidade de vendas efetivas não represente exatamente 50% dos contatos telefônicos realizados, esta base de dados possui volume de sucessos superior a 43%, o que garante representatividade à quantidade de vendas realizadas.

Comparando a autocorrelação nos meses selecionados para o desenvolvimento do modelo (treino, validação e teste), exposto na Figura 19, com a autocorrelação da base inteira (Figura 11), observa-se que são semelhantes. As observações ainda possuem correlações significativas entre si (Figura 19a) e a taxa de vendas no mês anterior influencia da mesma forma as vendas no mês atual (Figura 19b).

Figura 19 – Gráficos de autocorrelação (ACF), para observações da base de hipotecas e nos períodos desenvolvimento do modelo



(a) Autocorrelação dados desagregados (b) Autocorrelação taxa de vendas por mês

Fonte: Elaborada pelo autor.

Análogo ao ajuste realizado com a base de hipotecas, foi ajustado ao conjunto de treino o modelo de classificação LightGBM, com taxa de aprendizado igual a 0,1000, profundidade máxima das árvores igual a 3 e 1.000 estimadores com amostragem de 50% da base de treino. Para o cálculo das métricas de avaliação do modelo, foi considerado o limiar igual a 0,5000, isto é, para probabilidade de venda maior ou igual ao limiar, então o valor inferido é 1 (atraso) e zero, caso contrário.

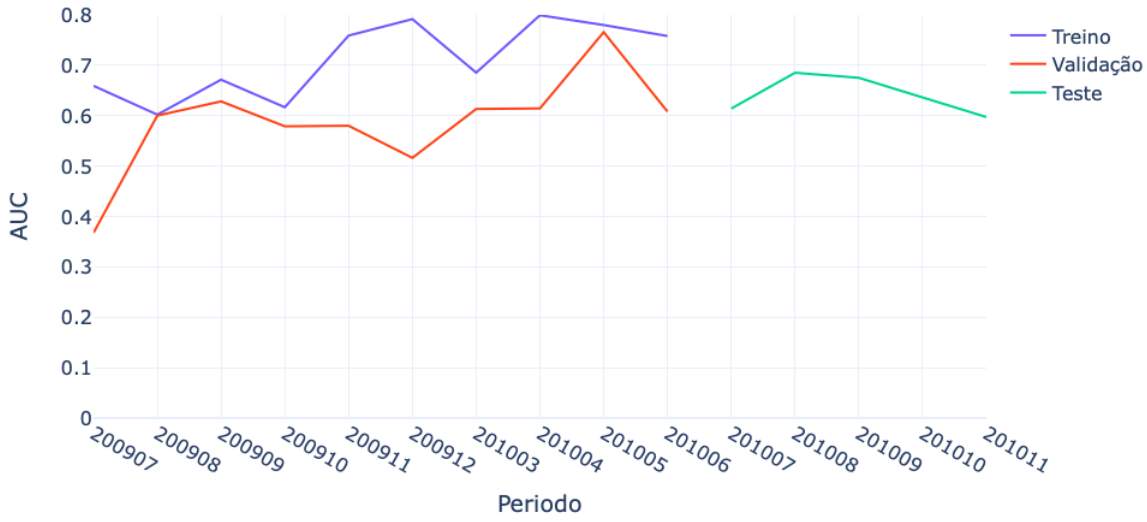
Quadro 5 – Métricas do modelo para base de dados *marketing* bancário

Subconjunto	Acurácia	Sensibilidade	Especificidade	AUC
Treino	0,7282	0,5836	0,8431	0,7974
Validação	0,6352	0,4584	0,7691	0,6207
Teste	0,6429	0,5209	0,7680	0,6612

Os resultados mostram que para as métricas avaliadas de forma consolidada, Quadro 5, o desempenho na base de treino é superior ao observado nas bases de validação e teste. Além disso,

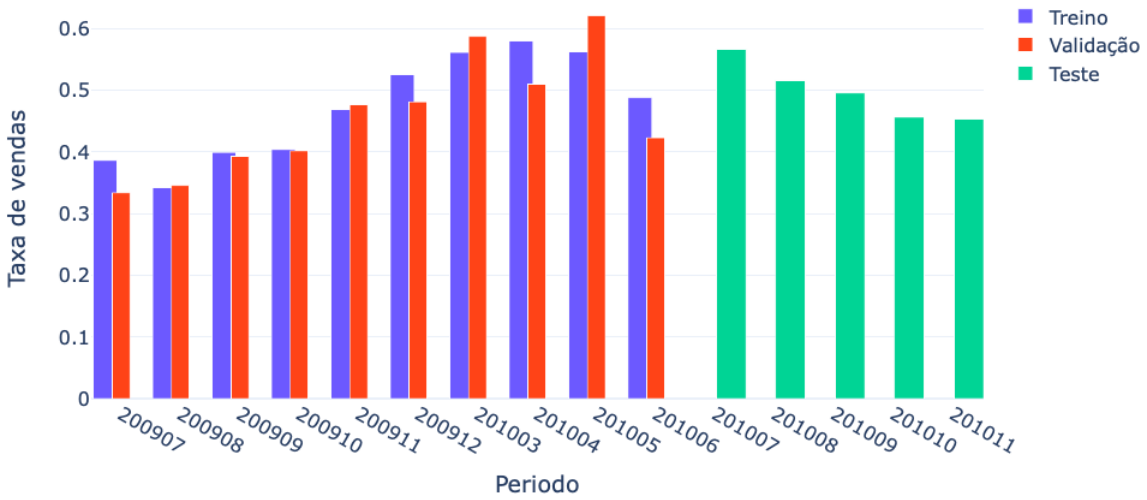
na Figura 20 verifica-se que, quando avaliado mês a mês, o AUC no conjunto de treino é sempre superior ao das bases de validação e teste, mostrando-se baixa capacidade de generalização tanto no conjunto com condições temporais semelhantes, quanto em observações pertencentes aos meses não utilizados no treino do modelo. Com relação à taxa de vendas, Figura 21, observa-se que os patamares mensais nas bases de treino e validação são próximos. Mesmo quando comparados os patamares da taxa de vendas entre os três conjuntos de dados, não há mudanças bruscas.

Figura 20 – AUC do modelo para os dados de vendas de depósito a longo prazo, por período, para os subconjuntos de treino, validação e teste



Fonte: Elaborada pelo autor.

Figura 21 – Taxa de vendas, por período, para os subconjuntos de treino, validação e teste



Fonte: Elaborada pelo autor.

O código, em linguagem Python, usado no desenvolvimento do modelo para este conjunto de dados, está disponível em Apêndice B.

CAPÍTULO

5

CONCLUSÃO

As bases de dados estudadas neste trabalho foram retiradas de situações reais e cotidianas do setor bancário. Cada uma delas possui características próprias, não apenas por tratarem de assuntos distintos, uma sobre atrasos e hipoteca imobiliária e outra de vendas de depósito a longo prazo. As análises descritivas mostraram que a base de hipotecas possui correlação não significativa entre as amostras, o que não acontece com a base de vendas de depósitos a longo prazo. Este é o cerne da discussão proposta neste trabalho. Ambas as bases foram avaliadas sob as mesmas perspectivas de técnicas de modelagem e métricas de desempenho, com o intuito de identificar, de forma empírica, os efeitos da dependência entre as observações de cada amostra.

Os resultados mostram que o modelo aplicado a base de hipoteca teve maior capacidade de generalização para observações não usadas no treino do modelo, mas capturadas na mesma janela temporal do conjunto de treino. A diferença absoluta entre AUC no treino e validação é de 0,0129. Mesmo quando a métrica é calculada em cada período, em vários pontos as métricas são iguais ou próximas. Por outro lado, quando se trata de observação que não pertence à janela temporal do treino, há queda na performance do AUC (a diferença entre treino e teste é 0,0745).

Sobre a base de vendas de depósito a longo prazo, a diferença entre o AUC do treino e da validação (observações coletadas nos mesmos meses) é 0,1766, o que representa queda de 22% na performance. Este desempenho indica que há dificuldade em generalização para observações coletadas em mesmas condições temporais. Quando a comparação é realizada entre o conjunto de treino e o de teste, a diferença de AUC é 0,1362. Sendo assim, o modelo ajustado a este conjunto de dados, apresentou baixa capacidade de generalização, também, para observações coletadas na janela temporal não usada no treino do modelo.

No que tange a comparação entre as duas bases de dados, ambas originadas de dados temporais, quando avaliadas as variáveis explicativas agregadas ao longo do tempo, a distribuição não é constante, refletindo a dependência temporal. Esta dependência também pode ser avaliada quando considerado o cálculo da autocorrelação da média (taxa) mensal de cada uma delas. A

principal diferença entre as bases é a autocorrelação entre cada observação (dados desagregados). O fato da base com autocorreção amostral não significativa (hipoteca residencial) generalizar melhor para o conjunto de validação pode estar, fortemente, correlacionado com a premissa de independência entre as amostras para garantia do aprendizado (generalização) do modelo, proposta por Vapnik (1999) e também defendida na Estatística (MORETTIN; BUSSAB, 2017). Vale ressaltar que os modelos ajustados ao conjuntos de dados, não foram capazes de estabelecer generalização para dados observados fora da janela temporal de desenvolvimento, considerando a queda de performance entre treino e teste. Esta efeito pode estar relacionado a autocorrelação significativa das variáveis resposta média por período, o que mostra a dependência temporal entre os dados sob outro ângulo.

Diante do exposto neste estudo empírico e o que há disponível na literatura, em trabalhos futuros é possível abordar, de forma aprofundada, os efeitos da dependência amostral (entre as observações e temporal) em modelos de aprendizado de máquina, bem como estudar e explorar formas de trabalhar com estes dados sem que haja deterioração da performance ao longo do tempo. Dado que é comum na indústria que os dados sejam modelados de forma granular, mas avaliados e projetados para planejamento dos meses seguintes. Tais achados forneceriam mais assertividade ao negócio e reduziria a necessidade de ajuste de novos modelos em curto espaço de tempo, dada a deterioração da performance.

REFERÊNCIAS

ALPAYDıN, E. **Introduction to Machine Learning - Second Edition**. Londres: The MIT Press, 2010. Citado na página 28.

ANALYTICS, C. R. **Dataset Mortgage**. 2016. Disponível em: <http://www.creditriskanalytics.net>. Acesso em: 11/11/2020. Citado na página 39.

ANDRYCHOWICZ, M.; DENIL, M.; COLMENAREJO, S. G.; HOFFMAN, M. W.; PFAU, D.; SCHAUL, T.; SHILLINGFORD, B.; FREITAS, N. de. Learning to learn by gradient descent by gradient descent. **Advances in neural information processing systems**, p. 3981–3989, 2016. Citado na página 31.

BENZSCHAWEL, T. **Credit Risk Modelling - Facts, Theory and Applications**. Londres: Risk Books, a Division of Incisive Media Investments Ltd, 2012. Citado na página 39.

BüHLMANN, P. Bagging, boosting and ensemble methods. **Handbook of Computational Statistics**, 2012. Citado na página 34.

DUMITRESCU, E.; SULLIVANHUE; HURLIN, C.; TOKPAVI, S. **Machine Learning for Credit Scoring: Improving Logistic Regression with Non Linear Decision Tree Effects**. Dissertação (Mestrado) — University of Orléans - France, França, 2018. Citado na página 23.

FICO. **FICO Score**. 2020. Disponível em: <https://www.fico.com/br/products/fico-score>. Acesso em: 14/11/2020. Citado na página 43.

FRIEDMAN, J. H. Greedy function approximation: A gradient boosting machine. **The Annals of Statistics**, v. 29, n. 5, p. 1189–1232, 2001. Citado nas páginas 35 e 36.

GUIDORIZZI, H. **Um Curso de Cálculo - Vol. 3: Volume 3**. Rio de Janeiro: LTC, 2002. Citado na página 31.

HAYES, A. **Loan-to-Value (LTV) Ratio**. 2020. Disponível em: <https://www.investopedia.com/terms/l/loantovalue.asp>. Acesso em: 11/11/2020. Citado na página 42.

KE, G.; MENG, Q.; FINLEY, T.; WANG, T.; CHEN, W.; MA, W.; YE, Q.; LIU, T.-Y. Lightgbm: A highly efficient gradient boosting decision tree. **Advances in neural information processing system**, p. 3146–3154, 2017. Citado nas páginas 36 e 37.

LANTZ, B. **Machine Learning with R**. Birmingham: Packt Publishing, 2013. Citado nas páginas 27, 32 e 33.

MARSLAND, S. **Machine Learning - An Algorithmic Perspective**. Nova York: CRC Press Taylor and Francis Group, 2015. Citado nas páginas 24, 27, 32 e 33.

MELLO, R. F.; PONTI, M. A. **Machine Learning: A Practical Approach on the Statistical Learning Theory**. New York: Springer, 2018. Citado nas páginas 27 e 30.

MORETTIN, P. A.; BUSSAB, W. O. **Estatística básica**. São Paulo: Saraiva, 2017. Citado nas páginas 23 e 58.

PAGLIOSA, L. de C.; MELLO, R. F. de. Applying a kernel function on time-dependent data to provide supervised-learning guarantees. **Expert Systems with Applications**, v. 71, p. 216–229, 2017. Citado na página 24.

QUINLAN, J. Induction of decision trees. **Machine learning**, v. 1, n. 1, p. 81–106, 1986. Citado na página 32.

REPOSITORY, U. M. L. **Bank Marketing Data Set**. 2020. Disponível em: <https://archive.ics.uci.edu/ml/datasets/Bank+Marketing>. Acesso em: 14/11/2020. Citado na página 44.

SEGAL, T. **Understanding the Mortgage Payment Structure**. 2019. Disponível em: <https://www.investopedia.com/mortgage/mortgage-rates/payment-structure/>. Acesso em: 11/11/2020. Citado na página 41.

STEINWART, I.; HUSH, D.; SCOVEL, C. Learning from dependent observations. **Journal of Multivariate Analysis**, v. 100, n. 1, p. 175–194, 2009. Citado na página 24.

TAKENS, F. Detecting strange attractors in turbulence. **Dynamical Systems and Turbulence**, v. 898, p. 366–381, 1981. Citado na página 24.

TAN, P.; STEINBACH, M.; A., K.; KUMAR, V. **Introduction to Data Mining (Second Edition)**. Londres: Pearson, 2019. Citado na página 33.

VAPNIK, V. N. An overview of statistical learning theory. **IEEE transactions on neural networks**, v. 10, n. 5, p. 988–999, 1999. Citado nas páginas 23, 29 e 58.

VENCO, S. Do 'call center' ao laudo a distância. **Jornal da Unicamp**, I, n. 460, p. 4, 2010. Citado na página 44.

WALDRON, D. **Derisking machine learning in banking**. 2019. Disponível em: <https://www.mckinsey.com/industries/financial-services/our-insights/banking-matters/derisking-machine-learning-in-banking>. Acesso em: 14/09/2020. Citado na página 23.

APÊNDICE

A

CÓDIGO DESENVOLVIMENTO MODELO PARA HIPOTECA RESIDENCIAIS

Código-fonte 1 – Código Python do desenvolvimento do modelo para os dados de hipoteca residenciais

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import math

from lightgbm import LGBMClassifier
from sklearn.feature_selection import SelectFromModel
from sklearn import metrics
from sklearn.model_selection import train_test_split
from statsmodels.tsa.stattools import acf
from statsmodels.tsa.stattools import pacf

def metricas(y_treino, y_pred_treino, y_proba_treino,
             y_teste, y_pred_teste, y_proba_teste,
             y_oot, y_pred_oot, y_proba_oot):

    tab_metric = {}

    tab_metric['TREINO'] = {}

    vn, fp, fn, vp = metrics.confusion_matrix(y_treino,
    y_pred_treino).ravel()
```

```
23:     tab_metric['TREINO']['ACURACIA'] = metrics.accuracy_score(
        y_treino, y_pred_treino, normalize=True)
24:     tab_metric['TREINO']['PRECISAO'] = metrics.precision_score(
        y_treino, y_pred_treino)
25:     tab_metric['TREINO']['SENSIBILIDADE'] = metrics.
        recall_score(y_treino, y_pred_treino)
26:     tab_metric['TREINO']['ESPECIFICIDADE'] = vn / (fp + vn)
27:     tab_metric['TREINO']['F1_SCORE'] =  metrics.f1_score(
        y_treino, y_pred_treino)
28:     tab_metric['TREINO']['AUC'] =  metrics.roc_auc_score(
        y_treino, y_proba_treino)
29:     tab_metric['TREINO']['GINI'] =  2*metrics.roc_auc_score(
        y_treino, y_proba_treino) - 1
30:
31:
32:     tab_metric['OOS'] = {}
33:
34:     vn, fp, fn, vp = metrics.confusion_matrix(y_teste,
        y_pred_teste).ravel()
35:     tab_metric['OOS']['ACURACIA'] = metrics.accuracy_score(
        y_teste, y_pred_teste, normalize=True)
36:     tab_metric['OOS']['PRECISAO'] = metrics.precision_score(
        y_teste, y_pred_teste)
37:     tab_metric['OOS']['SENSIBILIDADE'] = metrics.recall_score(
        y_teste, y_pred_teste)
38:     tab_metric['OOS']['ESPECIFICIDADE'] = vn / (fp + vn)
39:     tab_metric['OOS']['F1_SCORE'] =  metrics.f1_score(y_teste,
        y_pred_teste)
40:     tab_metric['OOS']['AUC'] =  metrics.roc_auc_score(y_teste,
        y_proba_teste)
41:     tab_metric['OOS']['GINI'] =  2*metrics.roc_auc_score(
        y_teste, y_proba_teste) - 1
42:
43:
44:     tab_metric['OOT'] = {}
45:
46:     vn, fp, fn, vp = metrics.confusion_matrix(y_oot, y_pred_oot
        ).ravel()
47:     tab_metric['OOT']['ACURACIA'] = metrics.accuracy_score(
        y_oot, y_pred_oot, normalize=True)
48:     tab_metric['OOT']['PRECISAO'] = metrics.precision_score(
        y_oot, y_pred_oot)
```

```
49:     tab_metric['OOT']['SENSIBILIDADE'] = metrics.recall_score(
      y_oot, y_pred_oot)
50:     tab_metric['OOT']['ESPECIFICIDADE'] = vn / (fp + vn)
51:     tab_metric['OOT']['F1_SCORE'] =  metrics.f1_score(y_oot,
      y_pred_oot)
52:     tab_metric['OOT']['AUC'] =  metrics.roc_auc_score(y_oot,
      y_proba_oot)
53:     tab_metric['OOT']['GINI'] =  2*metrics.roc_auc_score(y_oot,
       y_proba_oot) - 1
54:
55:     return tab_metric
56:
57: #leitura da base
58: base2 = pd.read_csv('mortgage.csv')
59:
60: #dados agragados por periodo
61: rate_time = pd.DataFrame(base2[base2.time<=30].groupby('time')[
      'default_time'].mean()).reset_index()
62:
63: #limite intervalo confianca da autocorrelacao para dados
      desagregados
64: LI = 1.96*len(base2[base2.time<=30].default_time)**(-0.5)
65: #limite intervalo confianca da autocorrelacao para dados
      agregados
66: LI2 = 1.96*len(rate_time)**(-0.5)
67:
68: #grafico autocorrelacao
69: plt.figure(figsize=(16, 7))
70:
71: plt.subplot(1,2,1)
72: plt.stem(range(0,21),acf(base2[base2.time<=30].default_time)
      [0:21] )
73: plt.axhline(y=LI, color='r', linestyle='--')
74: plt.axhline(y=-LI, color='r', linestyle='--')
75: plt.title("ACF")
76:
77: plt.subplot(1,2,2)
78: plt.stem(range(0,21),acf(rate_time.default_time)[0:21] )
79: plt.axhline(y=LI2, color='r', linestyle='--')
80: plt.axhline(y=-LI2, color='r', linestyle='--')
81: plt.title("ACF AGREGADO")
82:
```

```
83:
84: #divisao da base em treino, validacao e teste
85: treino, teste = train_test_split(base2[base2.time<=30],
       test_size=0.3, random_state=240)
86: oot = base2[(base2.time>30) & (base2.time<=36)]
87:
88: #variaveis testadas
89: lista = ['balance_time','interest_rate_time','hpi_time',
90:           'gdp_time','uer_time','REtype_CO_orig_time',
91:           'REtype_PU_orig_time','REtype_SF_orig_time',
92:           'investor_orig_time','balance_orig_time',
93:           'FICO_orig_time','LTV_orig_time',
94:           'Interest_Rate_orig_time','hpi_orig_time',
95:           'LTV_time']
96:
97: #selecao por modelo
98: lbm_m = LGBMClassifier(random_state=0)
99:
100: model_selec = SelectFromModel(lbm_m, threshold=300)
101:
102: model_selec.fit(treino[lista], treino.default_time)
103:
104: V_select = treino[lista].columns[model_selec.get_support()].
       to_list()
105:
106:
107: #Ajuste do modelo
108: modelo_rf = LGBMClassifier(random_state=0, learning_rate=0.001,
       max_depth=4,n_estimators=600,
109:                            class_weight={0: 1, 1:64},
110:                            subsample=0.7)
111:
112:
113: modelo_rf.fit(treino[V_select], treino.default_time)
114:
115: #Avaliacao de performance
116: y_treino_cat  = modelo_rf.predict(treino[V_select])
117: y_treino_prob = modelo_rf.predict_proba(treino[V_select])[:,1]
118:
119: y_teste_cat  = modelo_rf.predict(teste[V_select])
120: y_teste_prob = modelo_rf.predict_proba(teste[V_select])[:,1]
121:
```

```
122: y_oot_cat  = modelo_rf.predict(oot[V_select])
123: y_oot_prob = modelo_rf.predict_proba(oot[V_select])[:,1]
124:
125: metricas_rf = metricas(treino.default_time,
126:                        y_treino_cat,
127:                        y_treino_prob,
128:                        teste.default_time,
129:                        y_teste_cat,
130:                        y_teste_prob,
131:                        oot.default_time,
132:                        y_oot_cat,
133:                        y_oot_prob  )
134:
135: pd.DataFrame(metricas_rf).T
```

APÊNDICE

B

CÓDIGO DESENVOLVIMENTO MODELO PARA DEPÓSITOS A LONGO PRAZO

Código-fonte 2 – Código Python do desenvolvimento do modelo para os dados de deposito a longo prazo

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import math

from lightgbm import LGBMClassifier
from sklearn.feature_selection import SelectFromModel
from sklearn import metrics
from sklearn.model_selection import train_test_split
from statsmodels.tsa.stattools import acf
from statsmodels.tsa.stattools import pacf

def metricas(y_treino, y_pred_treino, y_proba_treino,
             y_teste, y_pred_teste, y_proba_teste,
             y_oot, y_pred_oot, y_proba_oot):

    tab_metric = {}

    tab_metric['TREINO'] = {}

    vn, fp, fn, vp = metrics.confusion_matrix(y_treino,
    y_pred_treino).ravel()
```

```
23:     tab_metric['TREINO']['ACURACIA'] = metrics.accuracy_score(
        y_treino, y_pred_treino, normalize=True)
24:     tab_metric['TREINO']['PRECISAO'] = metrics.precision_score(
        y_treino, y_pred_treino)
25:     tab_metric['TREINO']['SENSIBILIDADE'] = metrics.
        recall_score(y_treino, y_pred_treino)
26:     tab_metric['TREINO']['ESPECIFICIDADE'] = vn / (fp + vn)
27:     tab_metric['TREINO']['F1_SCORE'] =  metrics.f1_score(
        y_treino, y_pred_treino)
28:     tab_metric['TREINO']['AUC'] =  metrics.roc_auc_score(
        y_treino, y_proba_treino)
29:     tab_metric['TREINO']['GINI'] =  2*metrics.roc_auc_score(
        y_treino, y_proba_treino) - 1
30:
31:
32:     tab_metric['OOS'] = {}
33:
34:     vn, fp, fn, vp = metrics.confusion_matrix(y_teste,
        y_pred_teste).ravel()
35:     tab_metric['OOS']['ACURACIA'] = metrics.accuracy_score(
        y_teste, y_pred_teste, normalize=True)
36:     tab_metric['OOS']['PRECISAO'] = metrics.precision_score(
        y_teste, y_pred_teste)
37:     tab_metric['OOS']['SENSIBILIDADE'] = metrics.recall_score(
        y_teste, y_pred_teste)
38:     tab_metric['OOS']['ESPECIFICIDADE'] = vn / (fp + vn)
39:     tab_metric['OOS']['F1_SCORE'] =  metrics.f1_score(y_teste,
        y_pred_teste)
40:     tab_metric['OOS']['AUC'] =  metrics.roc_auc_score(y_teste,
        y_proba_teste)
41:     tab_metric['OOS']['GINI'] =  2*metrics.roc_auc_score(
        y_teste, y_proba_teste) - 1
42:
43:
44:     tab_metric['OOT'] = {}
45:
46:     vn, fp, fn, vp = metrics.confusion_matrix(y_oot, y_pred_oot
        ).ravel()
47:     tab_metric['OOT']['ACURACIA'] = metrics.accuracy_score(
        y_oot, y_pred_oot, normalize=True)
48:     tab_metric['OOT']['PRECISAO'] = metrics.precision_score(
        y_oot, y_pred_oot)
```

```
49:     tab_metric['OOT']['SENSIBILIDADE'] = metrics.recall_score(
      y_oot, y_pred_oot)
50:     tab_metric['OOT']['ESPECIFICIDADE'] = vn / (fp + vn)
51:     tab_metric['OOT']['F1_SCORE'] =  metrics.f1_score(y_oot,
      y_pred_oot)
52:     tab_metric['OOT']['AUC'] =  metrics.roc_auc_score(y_oot,
      y_proba_oot)
53:     tab_metric['OOT']['GINI'] =  2*metrics.roc_auc_score(y_oot,
       y_proba_oot) - 1
54:
55:     return tab_metric
56:
57: #leitura da base
58: base = pd.read_csv('bank-additional/base_banco_trat.csv')
59:
60: #dados agragados por periodo
61: taxa_mes = pd.DataFrame(base.groupby('anomes')['y_binario'].
      mean())
62:
63: #limite intervalo confianca da autocorrelacao para dados
      desagregados
64: LI = 1.96*len(base.y_binario)**(-0.5)
65: #limite intervalo confianca da autocorrelacao para dados
      agregados
66: LI2 = 1.96*len(taxa_mes)**(-0.5)
67: LI,LI2
68:
69: #grafico autocorrelacao
70: plt.subplot(1,2,1)
71: plt.stem(range(0,21),acf(base.y_binario)[0:21] )
72: plt.axhline(y=LI, color='r', linestyle='--')
73: plt.axhline(y=-LI, color='r', linestyle='--')
74: plt.title("ACF")
75:
76: plt.subplot(1,2,2)
77: plt.stem(range(0,21),acf(taxa_mes.y_binario)[0:21] )
78: plt.axhline(y=LI2, color='r', linestyle='--')
79: plt.axhline(y=-LI2, color='r', linestyle='--')
80: plt.title("ACF AGREGADO")
81:
82: #Tratamento das variaveis
83: def binaryType_(data):
```

```
84:
85:     #data.deposit.replace(('yes', 'no'), (1, 0), inplace=True)
86:     data.default.replace(('yes','no','unknown'),(1,0,-999),
      inplace=True)
87:     data.housing.replace(('yes','no','unknown'),(1,0,-999),
      inplace=True)
88:     data.loan.replace(('yes','no','unknown'),(1,0,-999),inplace
      =True)
89:     #data.marital.replace(('married','single','divorced')
      ,(1,2,3),inplace=True)
90:     data.month.replace(('Jan','Feb','Mar','Apr','May','Jun','
      Jul','Aug','Sep','Oct','Nov','Dec'),
91:                        (1,2,3,4,5,6,7,8,9,10,11,12),inplace=
      True)
92:     return data
93:
94: base = binaryType_(base)
95:
96: def cont_(data):
97:     data['contato']=0
98:     data.loc[data['contact'] == 'telephone','contact'] = 0
99:     data.loc[data['contact'] == 'cellular','contact'] = 1
100:    data.loc[data['contact'] == 'unknown','contact'] = -999
101:
102:    return data
103:
104: def age_(data):
105:
106:    data['Adult'] = 0
107:    data['Middle_Aged'] = 0
108:    data['old'] = 0
109:    data.loc[(data['age'] <= 35) & (data['age'] >= 18),'Adult']
      = 1
110:    data.loc[(data['age'] <= 60) & (data['age'] >= 36),'
      Middle_Aged'] = 1
111:    data.loc[data['age'] >=61,'old'] = 1
112:
113:    return data
114:
115: def campaign_(data):
116:
117:
```

```
118:     data.loc[data['campaign'] == 1,'campaign'] = 1
119:     data.loc[(data['campaign'] >= 2) & (data['campaign'] <= 3),
       'campaign'] = 2
120:     data.loc[data['campaign'] >= 4,'campaign'] = 3
121:
122:     return data
123:
124: def duration_(data):
125:
126:     data['t_min'] = 0
127:     data['t_e_min'] = 0
128:     data['e_min']=0
129:     data.loc[data['duration'] <= 5,'t_min'] = 1
130:     data.loc[(data['duration'] > 5) & (data['duration'] <= 10),
       't_e_min'] = 1
131:     data.loc[data['duration'] > 10,'e_min'] = 1
132:
133:     return data
134:
135: def pdays_(data):
136:     data['pdays_not_contacted'] = 0
137:     data['months_passed'] = 0
138:     data.loc[data['pdays'] == -1 ,'pdays_not_contacted'] = 1
139:     data['months_passed'] = data['pdays']/30
140:     data.loc[(data['months_passed'] >= 0) & (data['
       months_passed'] <=2) ,'months_passed'] = 1
141:     data.loc[(data['months_passed'] > 2) & (data['months_passed
       '] <=6),'months_passed'] = 2
142:     data.loc[data['months_passed'] > 6 ,'months_passed'] = 3
143:
144:     return data
145:
146: def previous_(data):
147:
148:     data['Not_Contacted'] = 0
149:     data['Contacted'] = 0
150:     data.loc[data['previous'] == 0 ,'Not_Contacted'] = 1
151:     data.loc[(data['previous'] >= 1) & (data['pdays'] <=99) ,'
       Contacted'] = 1
152:     data.loc[data['previous'] >= 100,'Contacted'] = 2
153:
154:     return data
```

```
155:
156: def balance_(data):
157:     data['Neg_Balance'] = 0
158:     data['No_Balance'] = 0
159:     data['Pos_Balance'] = 0
160:
161:     data.loc[~data['balance']<0,'Neg_Balance'] = 1
162:     data.loc[data['balance'] == 0,'No_Balance'] = 1
163:     data.loc[(data['balance'] >= 1) & (data['balance'] <= 100),
       'Pos_Balance'] = 1
164:     data.loc[(data['balance'] >= 101) & (data['balance'] <=
       500),'Pos_Balance'] = 2
165:     data.loc[(data['balance'] >= 501) & (data['balance'] <=
       2000),'Pos_Balance'] = 3
166:     data.loc[(data['balance'] >= 2001) & (data['balance'] <=
       10000),'Pos_Balance'] = 4
167:     data.loc[data['balance'] >= 10001,'Pos_Balance'] = 5
168:
169:     return data
170:
171: def job_(data):
172:
173:     data.loc[data['job'] == "management",'job'] = 1
174:     data.loc[data['job'] == "technician",'job'] = 2
175:     data.loc[data['job'] == "entrepreneur",'job'] = 3
176:     data.loc[data['job'] == "blue-collar",'job'] = 4
177:     data.loc[data['job'] == "retired",'job'] = 5
178:     data.loc[data['job'] == "admin.",'job'] = 6
179:     data.loc[data['job'] == "services",'job'] = 7
180:     data.loc[data['job'] == "self-employed",'job'] = 8
181:     data.loc[data['job'] == "unemployed",'job'] = 9
182:     data.loc[data['job'] == "student",'job'] = 10
183:     data.loc[data['job'] == "housemaid",'job'] = 11
184:     data.loc[data['job'] == "unknown",'job'] = 12
185:
186:     return data
187:
188: def marital_(data):
189:
190:     data['married'] = 0
191:     data['singles'] = 0
192:     data['divorced'] = 0
```

```
193:     data.loc[data['marital'] == 'married','married'] = 1
194:     data.loc[data['marital'] == 'singles','singles'] = 1
195:     data.loc[data['marital'] == 'divorced','divorced'] = 1
196:
197:     return data
198:
199: def education_(data):
200:
201:     data['primary'] = 0
202:     data['secondary'] = 0
203:     data['tertiary'] = 0
204:     data['unknown'] = 0
205:     data.loc[data['education'] == 'primary','primary'] = 1
206:     data.loc[data['education'] == 'secondary','secondary'] = 1
207:     data.loc[data['education'] == 'tertiary','tertiary'] = 1
208:     data.loc[data['education'] == 'unknown','unknown'] = 1
209:
210:     return data
211:
212: base = campaign_(base)
213: base = age_(base)
214: base = education_(base)
215: base = job_(base)
216: base = previous_(base)
217: base = duration_(base)
218: base = pdays_(base)
219: base = marital_(base)
220: base = cont_(base)
221:
222:
223:
224: #divisao da base em treino, validacao e teste
225: treino, teste = train_test_split(base[(base.anomes<=201006) & (
       base.anomes>200906)], test_size=0.3, random_state=240)
226: oot = base[(base.anomes>201006)]
227:
228: #variaveis testadas
229: lista= ['job', 'default', 'housing', 'loan', 'contato', 'month'
       ,
230:         'campaign', 'Adult', 'Middle_Aged', 'old', 'primary',
231:         'secondary', 'tertiary','unknown', #'Neg_Balance',
```

```
232:         'No_Balance', 'Pos_Balance','Not_Contacted','Contacted'
      ,
233:         't_min', 't_e_min', 'e_min', 'pdays_not_contacted',
234:        'months_passed', 'married', 'singles', 'divorced']
235:
236:
237: #selecao por modelo
238: modelo_rf = LGBMClassifier(random_state=0, learning_rate=0.001,
      max_depth=4,n_estimators=500,
239:                           class_weight={0: 1, 1: 10},#'
      balanced',
240:                           subsample=0.7)
241:
242: model_selec = SelectFromModel(modelo_rf, threshold=20)
243:
244: model_selec.fit(treino[lista], treino.y_binario)
245:
246: V_select = treino[lista].columns[model_selec.get_support()].
      to_list()
247:
248:
249: #Ajuste do modelo
250: modelo_rf = LGBMClassifier(random_state=0,learning_rate=0.1,
251:                            max_depth=3,n_estimators=1000,
252:                            subsample=0.5)
253:
254:
255: modelo_rf.fit(treino[V_select], treino.y_binario)
256:
257: #Avaliacao do modelo
258: y_treino_cat  = modelo_rf.predict(treino[V_select])
259: y_treino_prob = modelo_rf.predict_proba(treino[V_select])[:,1]
260: /
261: y_teste_cat  = modelo_rf.predict(teste[V_select])
262: y_teste_prob = modelo_rf.predict_proba(teste[V_select])[:,1]
263:
264: y_oot_cat  = modelo_rf.predict(oot[V_select])
265: y_oot_prob = modelo_rf.predict_proba(oot[V_select])[:,1]
266:
267: metricas_rf = metricas(treino.y_binario, y_treino_cat,
      y_treino_prob,
```

```
268:                         teste.y_binario,  y_teste_cat,
      y_teste_prob,
269:                         oot.y_binario,  y_oot_cat,  y_oot_prob
      )
270:
271: pd.DataFrame(metricas_rf).T
```


ICMC
USP
SÃO CARLOS