Universidade Federal do ABC
Centro de Engenharia, Modelagem e Ciências Sociais Aplicadas
Trabalho de Graduação em Engenharia de Informação

Estudo sobre Técnicas de Classificação de Clientes Propensos ao Churn

Leonardo Alves Ferretti

Santo André

Novembro de 2022

Leonardo Alves Ferretti

Estudo sobre Técnicas de Classificação de Clientes Propensos ao Churn

Trabalho de Graduação apresentado ao curso de Graduação em Engenharia de Informação, como parte dos requisitos necessários para a obtenção do Título Bacharel em Engenharia de Informação.

Universidade Federal do ABC

Orientador: Murilo Bellezoni Loiola

Santo André

Novembro de 2022

Ferretti, Leonardo Alves
Estudo sobre Técnicas de Classificação de Clientes Propensos ao Churn. / Leonardo Alves Ferretti. – Santo André.- 2022.
25 p. : il.

Monografia (Graduação - Engenharia de Informação) – Universidade Federal do ABC, 2022.
Orientador: Murilo Bellezoni Loiola

Bibliografia

1. Aprendizado de Máquina. 2. Churn. I. Estudo sobre Técnicas de Classificação de Clientes Propensos ao Churn. II. Universidade Federal do ABC

Leonardo Alves Ferretti

Estudo sobre Técnicas de Classificação de Clientes Propensos ao Churn

Trabalho de Graduação apresentado ao curso de Graduação em Engenharia de Informação, como parte dos requisitos necessários para a obtenção do Título Bacharel em Engenharia de Informação.

Santo André, 28 de Novembro de 2022:

Murilo Bellezoni Loiola
Orientador

Luneque Del Rio de Souza e Silva Junior
Avaliador 1

Karla Vittori
Avaliador 2

Santo André
Novembro de 2022

Resumo

O sistema bancário é um dos setores mais competitivos da economia brasileira com o aparecimento das *Fintechs* nos últimos anos também aumentou o interesse das instituições para capturar e reter clientes. Este trabalho propõe analisar o *churn*, que seria o ato do cliente abandonar um produto ou serviço, neste estudo em específico será aplicado este conceito para classificar os clientes propensos a abandonar um serviço financeiro. Para esta classificação foram utilizados os seguintes algoritmos de *machine learning*: *Random Forest, Naive Bayes, K-Nearest Neighbors* e Regressão Logística. Para esta análise, foi utilizado um conjunto de dados que estão em domínio público. A análise dos resultados foi feita em função do *recall*, para priorizar a captura dos clientes propensos ao *churn*. Segundo os resultados obtidos, o modelo *Random Forest* obteve a maior taxa de *recall*, cerca de 52% e além disso, apesar de existirem variáveis com uma maior significância para a predição do resultado, este número foi obtido utilizando todo o conjunto de variáveis.

Palavras-chaves: Aprendizado de Máquina. Algoritmo. Classificação. *Churn*.

Abstract

The banking system is one of the most competitive sectors of the Brazilian economy. The increasing number of Fintechs in last few years, has also increased the interest of institutions to capture and retain customers. This work proposes to analyze churn, which would be the act of the customer abandoning a product or service, in this specific study we will apply this concept to classify tolerant customers to abandon a financial service. For this classification, the following machine learning algorithms were used: Random Forest, Naive Bayes, K-Nearest Neighbors and Logistic Regression. For this analysis, a set of data that are in the public domain was used. The analysis of the results was made according to the recall metric, to prioritize the capture of customers subject to churn. According to the results obtained, the Random Forest model obtained the highest recall rate, around 52% and in addition, although there are variables with a greater significance for predicting the result, this number was obtained using the entire set of variables.

Keywords: Machine Learning. Algorithm. Classification. Churn.

Sumário

1 Introdução

O sistema bancário brasileiro é um dos setores mais competitivos e lucrativos do país tendo um lucro líquido em 2021 de R$ 132 bilhões de reais. No entanto, cerca de 78% desse lucro ficou concentrado entre os cinco maiores bancos do país sendo eles: Itaú Unibanco, Bradesco, Santander, Caixa Econômica e Banco do Brasil [1].

De acordo com Conceição de Graça [2], o setor financeiro é um dos que mais investe em novas tecnologias para melhorar o desempenho de produtos e serviços. Assim, foi possível ganhar flexibilidade, reduzir custos e erros humanos. Vale ressaltar que apesar de todo o investimento em eficiência operacional, inicialmente do ponto de vista do cliente, houve um certo atrito entre cliente e instituição financeira, pois durante esse processo perdeu-se o foco no cliente.

E foi justamente em cima dessa vulnerabilidade dos bancos tradicionais que as *fintechs* surgiram. *Fintechs* são empresas que combinam tecnologia e serviços financeiros para trazer inovações na forma de operar dentro do sistema financeiro.

Essas inovações tem como objetivo melhorar a experiência do cliente com serviços financeiros, utilizar novas tecnologias e uma grande quantidade de massa de dados para oferecer serviços melhores e que façam sentido para os clientes, além de otimizar tomadas de decisões e reduzir o custos operacionais.

O estudo *Fintech Mining Report* (2021) [3] mostra que atualmente existem no Brasil cerca 1.158 fintechs e que 59,1% dessas empresas foram criadas a patir de 2016. Assim, a entrada de novos concorrentes no mercado financeiro brasileiro fez com que os clientes que antes precisavam se contentar com as opções disponíveis, agora consigam encontrar oportunidades mais vantajosas de atendimento, produtos e serviços financeiros, consequentemente aumentando o interesse das *fintechs* e bancos tradicionais em criar formas de reter e fidelizar os clientes.

Dessa forma, predizer quais são os potenciais clientes que podem deixar de consumir um produto ou serviço é de extrema importância não só para mercado financeiro, mas para outros setores da economia. Irfan Ullah [4] utilizou o algoritmo *Random Forest* para classificar clientes propesos ao *churn* (a definição desse termo será explicada na página 3) no setor de telecomunicações. Já Essam Shaaban [5] comparou o desempenho de árvores de decisão, SVM *(Suppot Vector Machines)* e redes neurais para classificar os clientes propensos ao *churn*, enquanto Dzulijana Popovic [6] utilizou a lógica fuzzy para predizer os clientes propensos ao *churn* em um banco de varejo.

1.1 Objetivo

Este trabalho tem como principal objetivo comparar o desempenho de alguns algoritmos de aprendizado de máquina aplicados a classificação de clientes propensos ao churn de um serviço financeiro e compreender qual ou quais variáveis são mais importantes para a classificação dos clientes, sendo eles: *Random Forest*, *K-Nearest Neighbor*, *Naive Bayes* e Regressão Logística. E a comparação desses algoritmos será realizada através de métricas de resultado comumente utilizadas para modelos classificação.

2 Referencial Teórico

Neste capítulo, será discutido o conceito do *churn* aplicado ao contexto do mercado financeiro. Sendo uma situação que resulta em um problema de classifiação binária. Além disso, este capítulo também contém os algortimos que serão utilizados para classificar os clientes propensos ao *churn*, bem como as métricas de avaliação dos modelos de classificação.

2.1 Conceitos Básicos

2.1.1 Fidelidade

Segundo [7], define-se o conceito de fidelidade como:

> Fidelidade em marketing significa um sentimento de afinidade em relação a produtos e marcas de uma empresa, que vai além da simples repetição de compra, embora este seja um indicador comumente utilizado como forma de se auferir a satisfação dos clientes - o qual, todavia, desconsidera fatores como conveniência, inércia e o grau de competitividade ou de concentração de um determinado mercado (Day, 1999, p. 146-147).

2.1.2 Churn

Churn é uma métrica que indica a taxa de cancelamento de clientes em um determinado período. Desta forma, o churn mostra-se uma importantíssima métrica para entender a saúde do serviço que está sendo oferecido. Embora ainda não tenha ganhado uma tradução explicita para o português, o *churn* compreende-se como a ação do cliente de trocar um produto ou serviço de uma empresa por outra concorrente, encerrando assim sua relação com a empresa antiga [8].

Uma vez definido o conceito do *churn*, com a crescente competição entre instituições financeiras, os clientes possuem maiores oportunidades para escolher produtos e serviços que lhe são mais vantajosos, como visto no capítulo 1, assim surge a necessidade de classificar os clientes com maior risco de apresentar o *churn*. Esta classificação pode ser feita através de um algoritmo de aprendizado de máquina que utiliza um conjunto de dados com variáveis que buscam explicar o comportamento do cliente.

2.2 Algoritmos Estudados

2.2.1 Árvore de Decisão

Árvores de decisão são algoritmos não paramétricos, cujo objetivo é maximizar o ganho de informação a cada ramificação, um exemplo de construção de árvore de decisão pode ser visto na figura 1. Possuem alta variância, ou seja, uma mudança mínima nos dados de entrada pode resultar em uma grande mudança na classificação final e baixo viés, porém é possível controlar a variância através de técnicas como a poda, ou seleção de parâmetros como a profundidade máxima da árvore [9].

Além disso, as árvores podem se basear em dois critérios para tomada de decisão. Um desses critérios é conhecido como entropia e o outro como Gini e ambos buscam quantificar a impureza em ramos de árvores de decisão [9].

A entropia pode ser calculada como:

$$Entropia = \sum_{j} -p_j \times \log_2(p_j) \tag{2.1}$$

onde p é a probabilidade para cada classe do conjunto de dados.

Já o critério de Gini é dado por:

$$Gini = 1 - \sum_{j} p_j^2 \tag{2.2}$$

onde p é a probabilidade para cada classe do conjunto de dados. Os detalhes sobre cada um dos critérios podem ser encontrados em [10].

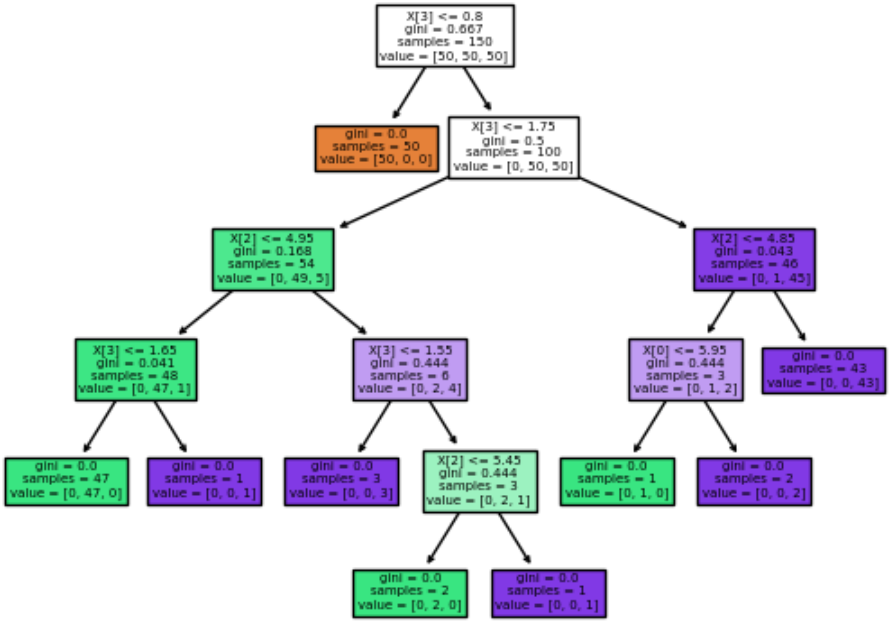
Segundo James Gareth [9] pode-se citar como vantagens desse algoritmo:

- É um algoritmo fácil de entender e o resultado pode ser visualizado;
- Os dados necessitam de pouco pré-processamento para serem introduzidos no modelo.

Porém, como todo algoritmo também possui desvantagens [9]:

- Árvores de decisão podem ser instáveis devido a sua alta variação, pois uma pequena modificação no conjunto de dados de entrada pode significar uma grande alteração na classificação;
- Algumas árvores de decisão podem ser tornar muito complexas e por conta disso existindo grande chance de acontecer o sobreajuste.

Figura 1 – Visualição de um exemplo de árvore de decisão.



Fonte: scikit-learn [11]

2.2.2 Bagging

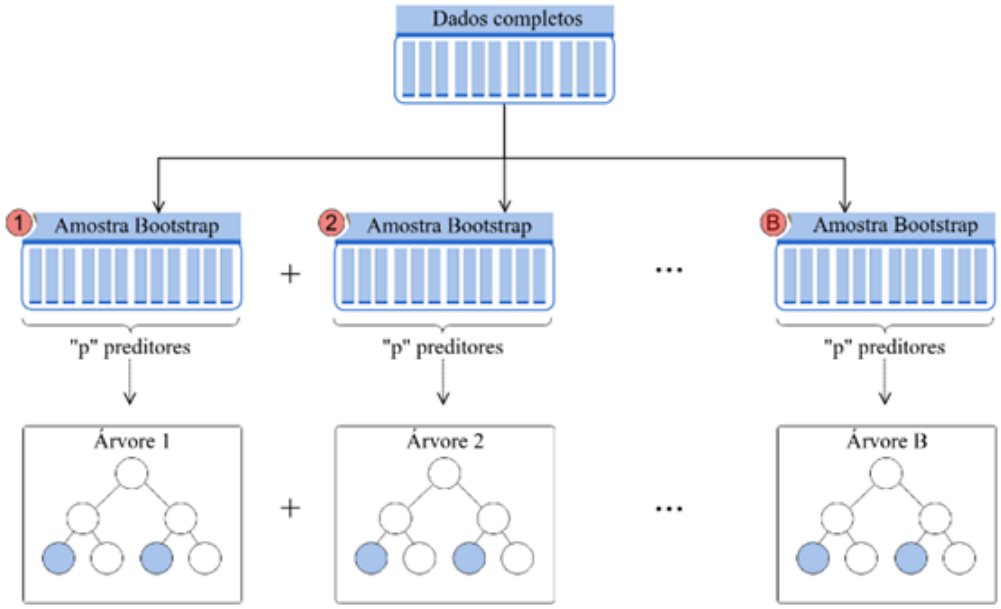
Bagging, é uma combinação de modelos ou conhecido em inglês como *ensemble*. Um classificador *ensemble*, consiste em um conjunto de classificadores treinados individualmente (classificadores de base), cujas decisões são de alguma forma combinadas, tipicamente ponderadas pelo voto. Verificou-se que na maioria dos casos os classificadores *ensemble* produzem previsões mais precisas do que os classificadores básicos que os compõem [12].

Dessa forma, o algoritmo consegue reduzir a variância e aumentar a acurácia de uma árvore de decisão criando B conjuntos de treino, em que cada conjunto de treino é uma amostragem aleatória com reposição do conjunto inicial (*Boostrap*) [9]. Em seguida, para cada conjunto é obtido um classificador que utilizará todos os preditores disponíveis. Então a predição final será baseada no voto majoritario de cada um desses classificadores, como pode-se observar na figura 2.

James Gareth [9] demostrou que, em cada amostra *Boostrap*, apenas 2/3 do conjunto total de observações é utilizado, ou seja, o 1/3 restante contempla o chamado out-of-bag (OOB) e por não participar do conjunto de treino, pode ser utilizado para testar o algoritmo.

Ainda pode-se citar como ponto de vantagem o fato do algoritmo combinar estimadores fracos que juntos produzem resultados melhores que um único estimador forte, além de reduzir a variância e mitigar a change de acontecer o sobreajuste, porém essa combinação de estimadores também introduz perda de interpretabilidade do modelo [14].

Figura 2 – Seleção de atributos através do algoritmo *Bagging*.



Fonte: [13]

2.2.3 Random Forest

Random Forest, ou Florestas Aleatórias traduzindo para o português, utilizam-se da técnica do *bagging* para gerar um conjunto de subamostras aleatórias e, assim, reduzir a variabilidade das árvores de decisão, o modelo *random forest* promove uma melhoria em relação ao *bagging*, pois além de produzir subamotras aleatórias elas também são descorrelacionadas umas das outras [9], pois cada amostra pode conter apenas "p" atributos de um conjunto "m", onde tipicamente é escolhido $p = \sqrt{m}$. Esta é a principal diferença entre o *bagging* e o *random forest* e caso $m = p$, o *random forest* torna-se novamente apenas um *bagging* [9].

Esta restrição é necessária, pois dado um conjunto de atributos, caso nesse conjunto exista uma variável que seja muito significativa para predizer a variável resposta, as árvores construídas utilizando o *bagging* colocarão essa variável como sendo a variável raiz. Assim, como afirma James Gareth [9], calcular a média de um conjunto que possui muitas variáveis correlacionadas não reduz tanto o problema da variabilidade, quanto calcular a média de um conjunto que possui variáveis descorrelacionadas e isso significa que, neste caso, o *bagging* não irá reduzir substancialmente a variabilidade, como a *random forest*.

2.2.4 K-Nearest Neighbors

O *K-Nearest Neighbors*, ou traduzindo para português como K vizinhos mais próximos, é uma algoritmo não paramétrico do tipo supervisionado, cuja premissa assume que objetos similares estão em um espaço de *features* [15].

Dado um conjunto de treino Z e um objeto de teste x, o algoritmo computa a distância entre x e todos os objetos de treino para determinar os K vizinhos mais próximos, uma vez feito isso, o algoritmo atribui x a uma classe utilizando a categoria majoritária. Assim, tipicamente K deve assumir um valor inteiro positivo, maior que zero e ímpar [9].

A escolha de K pode produzir grandes mudanças no resultado final da classificação. Caso $K = 1$ cada elemento de teste x será atribuído ao elemento de treino mais próximo, o que acarretará em um modelo com alta variância e baixo viés. Por outro lado, caso K assuma um valor muito grande, o modelo se torna menos flexível e, por sua vez apresenta baixa variância, porém alto viés [9].

2.2.5 Naive Bayes

O modelo *Naive Bayes* é um classificador probabilístico baseado no Teorema de Bayes e como Harry Zhang [16] discute no artigo *The Optimality of Naive Bayes*, a premissa central do algoritmo considera que todos os atributos de um conjunto de dados são independentes entre si. Esta premissa, em muitos cenários da realidade não é verdadeira, porém apesar disso o classificador consegue obter bons níveis de acurácia, quando comparados a modelos mais robustos.

Desse modo, seja A e B duas variáveis, o Teorema de Bayes fornece uma relação entre as probabilidades de A ($P(A)$) e B ($P(B)$) e a probabilidade condicional de A dado B ($P(A|B)$) e B dado A ($P(B|A)$) [17]. Assim, o teorema pode ser escrito como:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{2.3}$$

onde,

- $P(A)$ é a probabilidade a priori;
- $P(B|A)$ é probabilidade condicional;
- $P(B)$ é a probabilidade total dos eventos ocorrerem;
- $P(A|B)$ é a probabilidade a posteriori.

O desenvolvimento do processo de classificação do algoritmo *Naive Bayes* é feito a partir do Teorema de Bayes (2.3). Assim, seja, $X = \{X_1, X_2...X_n\}$ um vetor de variáveis explicativas que precisam ser classificadas e seja $Y = \{Y_1, Y_2, ..., Y_k\}$ um conjunto finito de classes que o vetor de variáveis pode pertencer. Dado que o modelo assume a independência entre os atributos explicativos é possível predizer a classe $Y^{pred} \in \{Y_1, Y_2, ..., Y_k\}$ para um dado X [18], como mostra a equação (2.4).

$$Y^{pred} = \arg\max_{y} P(Y = y) \prod_{n=1}^{m} P(X_j = x_j | Y = y) \tag{2.4}$$

A dedução da equação (2.4) pode ser encontrada em [18], além disso um aprofundamento sobre a formulação do Teorema de Bayes pode ser visto em [19].

O modelo *Naive Bayes*, pode ser aplicado em diversos problemas de classificação, por conseguir lidar com informações discretas é recorrentemente utilizado para classificação de textos ou filtros de spam [20].

Como todo algoritmo de aprendizado de máquina o *naive bayes* possui vantagens e desvantagens. Como vantagens pode-se citar a simplicidade do algoritmo, sendo muito rápido de ser treinado [21]. Por outro lado, esse ao algoritmo é suscetível ao "problema da probabilidade zero", este problema acontece quando o algoritmo tenta classificar um novo conjunto de dados que não esteve presente no conjunto de treinamento, mas uma solução para isso é usar a chama suavização Laplaciana [21] que adiciona uma constante que contabiliza os recursos que não estão presentes no conjunto de treinamento e impede que haja probabilidade igual a zero. [21].

2.2.6 Regressão Logística

O algoritmo de regressão logística, tornou-se popular por ser um algoritmo de fácil interpretação e flexível [22].

A regressão logística descreve a relação entre a variável resposta e as variáveis explicativas através da combinação linear de um conjunto de preditores, ou seja, busca-se investigar como cada atributo explicativo contribui para classificação da variável resposta, diferentemente de uma regressão linear, a regressão logística retorna a probabilidade de uma observação pertencer a uma determinada classe.

Assim, seja um conjunto de variáveis explicativas, cuja combinação linear é dada por:

$$\hat{Y} = \beta_0 + \beta_1 X_1 + \beta_2 X_2 \ldots + \beta_i X_i \tag{2.5}$$

onde,

- $\hat{Y}$ é a predição do modelo;
- β_0 é uma constante;
- $\beta_1 X_1 \ldots \beta_i X_i$ é um conjunto de atributos explicativos (X_i), ponderados pelos respectivos pesos (β_i).

No entanto, para conseguir restringir o resultado entre [0, 1] precisa-se aplicar uma transformação logarítmica em ambos os lados da equação. Assim obtém-se:

$$prob(\hat{Y}) = \frac{1}{1 + e^{\beta_0 + \beta_1 X_1 + ... + \beta_i X_i}} \tag{2.6}$$

onde,

- $prob(\hat{Y})$ representa a probabilidade de uma observação pertencer a uma classe;
- $e^{\beta_0 + \beta_1 X_1 + ... + \beta_i X_i}$ representa a combinação linear para as variáveis independentes expressas em escala logarítmica.

Os detalhes desta dedução matemática podem ser encontrados em [22].

Por fim, vale lembrar que a regressão logística é um algoritmo paramétrico que possui como premissas:

- Relação linear entre as variáveis explicativas (X) e a variável resposta (Y);
- Homocedasticidade [1];
- valor esperado dos resíduos igual a zero;
- Ausência de multicolinearidade. [2]

2.3 Métricas de Avaliação de Modelos de Classificação

2.3.1 Matriz de Confusão

A matriz de confusão é amplamente utilizada para avaliar os mais diversos modelos de classificação. A matriz exibe a frequência de acertos e erros na etapa de testes de um classificador. Os acertos encontram-se na diagonal, enquanto que as demais posições indicam detecções em classes erradas. A partir dela, será deduzido quatro métricas de avaliação, sendo elas: acurácia, precisão, revocação, mais conhecido como *recall*, e a métrica F1 também conhecida como *F1 Score*. Com exceção da métrica *F1 Score*, estas são as métricas que serão utilizadas para avaliar os modelos em questão neste estudo.

A matriz de confusão para um problema de classificação binário pode ser visto na tabela 1.

[1] Dado um conjunto de erros e_i e um conjunto da variáveis explicativas $E_{1_i}, E_{2_i}, ...E_{k_i}$. A variância é dada por: $Var(e_i/E_{1_i}, E_{2_i}, ...E_{k_i}) = \sigma^2$, ou seja, a variância dos erros condicionada aos valores das variáveis explicativas, será constante [23].

[2] De acordo com Karina Gernhardt [24], a multicolinearidade existe quando acontece uma associação linear exata ou aproximada entre as variáveis explicativas do modelo. Seu estudo mostra que dependendo do grau de associação entre as variáveis explicativas, os estimadores podem ficar imprecisos ou afetar a inferência dos parâmetros.

Tabela 1 – Matriz de confusão.

	Predito	
Real	Negativo	Postivo
Negativo	VN	FP
Positivo	FN	VP

Onde,

- VP (Verdadeiro Positivo) são as observações classificadas como positivas, que realmente eram positivas;
- VN (Verdadeiro Negativo) são as observações classificadas como negativas, que realmente eram negativas;
- FP (Falso Positivo) são as observações classificadas como positivas, mas que na verdade eram negativas;
- FN (Falso Negativo) são as observações classificadas como negativas, mas que na verdade eram positivas.

2.3.2 Acurácia

Acurácia, estabelecida em (2.7), é uma métrica utilizada para avaliação de modelos de classificação que pode ser extraída da matriz de confusão e que representa o percentual de acerto do modelo. A acurácia pode ser definida como:

$$\text{acurácia} = \frac{VP + VN}{VP + FN + VN + FP} \tag{2.7}$$

Apesar de indicar o percentual de acerto do modelo, nos casos em que as classes a serem classificadas estão desbalanceadas, esta métrica pode causar uma falsa impressão de que o modelo está desempenhando bem, pois a quantidade de VN será muito alta.

2.3.3 Precisão

A precisão, definida em (2.8), indica do que foi classificado como positivo, quanto realmente estava correto, ou seja, ao utilizar a precisão como métrica principal de avaliação de um modelo, quer-se que o algoritmo identifique a classe positiva o mais correto possível [25].

$$\text{precisão} = \frac{VP}{VP + FP} \tag{2.8}$$

2.3.4 Revocação

A revocação (do inglês, *recall*), indicado em (2.9), também conhecida como captura. É uma métrica que avalia a capacidade do modelo de detectar com sucesso os resultados classificados como positivos. E além disso, foi a métrica principal escolhida para avaliar os modelos de classificação neste estudo, (no capítulo 4 será explicado o porquê). A revocação é definida como:

$$\text{revocação} = \frac{VP}{VP + FN} \tag{2.9}$$

2.3.5 Métrica F1

A métrica F1 é a média harmônica da revocação e da precisão, como definida em (2.10), e portanto, terá seu valor ótimo quando existir um balaço entre essas duas métricas. A métrica F1 utiliza a média harmônica, pois penaliza valores extremos. Por exemplo, caso a precisão ou a revocação forem zeros, a métrica, também será zero. Assim, a métrica F1 é dada por:

$$F1 = \frac{2 \times \text{precisão} \times \text{revocação}}{\text{precisão} + \text{revocação}} \tag{2.10}$$

3 Descrição Experimental

3.1 Descrição da Base de Dados

Os dados para este problema de classificação foram retirados de uma base de dados livre disponibilizada na plataforma Kaggle chamada: *Bank Customer Churn Prediction* [26].

A base contém 14 atributos, como mostra a tabela 2 e um total de 10.000 linhas, onde mais adiante será feita uma análise exploratória para identificar quais atributos de fato fazem sentido continuar no estudo.

Tabela 2 – Descrição da fonte de dados.

Atributo	**Formato do Dado**	**Descrição**
RowNumber	Númerico	Número da linha
CustomerId	Númerico	Identificador único do cliente
Surname	String	Primeiro nome do cliente
CreditScore	Númerico	Grau de confiança na capacidade de pagamento do cliente
Geography	String	País ao qual cada cliente pertence
Gender	String	Sexo dos Clientes
Age	Númerico	Idade
Tenure	Númerico	Posses do cliente
Balance	Númerico	Saldo do cliente
NumOfProducts	Númerico	Quantidade de produtos que o cliente Possui
HasCrCard	Númerico	Marcação binaria que identifica se o cliente possui ou não cartão de crédito
IsActiveMember	Númerico	Identifica se o cliente é um cliente ativo
EstimatedSalary	Númerico	Salário estimado do cliente
Exited	Númerico	Variável *target* que identifica se o cliente deixou ou não de usar o produto (churn).

3.2 Pré-Processamento

O primeiro pré-processamento realizado foi é eliminar as colunas: *RowNumber*, *Surname* e *CustomerId*, pois elas são, respectivamente, a identificação da linha, o nome do cliente e o código do cliente, ou seja, não irão contribuir para o problema da classificação.

Já segundo passo foi transformar as variáveis: *Geography* e *Gender* que são categóricas, em numéricas para isso foi necessário realizar um processo de transformação das categorias de cada variável categórica em colunas. Para cada categoria, foi criada uma

coluna onde foi atribuído o valor 1 na linha, caso a característica exista e o valor zero, caso contrário. Esta transformação foi necessária afim de que esses atributos pudessem ser interpretados pelos algoritmos dos modelos.

O terceiro passo consistiu em checar se existem atributos com uma grande quantidade de registros nulos. Assim, foi observado que todos os atributos não possuem informações nulas, como mostra a tabela 3, caso algum atributo possuísse um alto percentual de observações nulas seria necessário trata-las, pois a maioria dos algoritmos não funcionará se houver dados ausentes. Sendo assim, existem algumas formas de lidar com esse tipo de situação, por exemplo: remover qualquer linha com dados ausentes ou remover qualquer coluna com dados ausentes ou imputar dados aos valores ausentes ou por fim, criar uma coluna que indica que os dados estavam ausentes [20].

Por fim, foi realizado o redimensionamento da escala dos dados, pois a grande maioria dos algoritmos de *machine learning* apresenta um desempenho melhor quando não existe uma diferença grande na escala dos atributos numéricos [27]. Para este estudo, o redimensionamento da escala foi feito utilizando a padronização dos dados que consiste em deixar cada coluna com um valor de média igual zero e um desvio padrão igual a 1 [27].

Tabela 3 – Descrição da fonte de dados.

Atributo	**Qtd. Valores Nulos**
RowNumber	0
CustomerId	0
Surname	0
CreditScore	0
Geography	0
Gender	0
Age	0
Tenure	0
Balance	0
NumOfProducts	0
HasCrCard	0
IsActiveMember	0
EstimatedSalary	0
Exited	0

3.3 Implementação

Os dados foram divididos em um grupo de teste e em um grupo de treino, sendo que 25% dos dados (2.500 linhas) foram destinados para teste e 75% das observações foram destinadas para treino (7.500 linhas).

Inicialmente foi verificado o desempenho de cada um dos modelos utilizando todos os atributos contido na tabela 2. Dessa forma, conseguiu-se ter uma visão geral de como

cada um dos modelos desempenhou ao fazer a classificação. Segue abaixo a tabela 4 contendo os resultados.

Tabela 4 – Desempenho dos modelos.

	Acurácia	Precisão	Recall
Random Forest	87%	79%	52%
K Nearest Neighbors	83%	62%	41%
Regressão Logística	81%	56%	23%
Naive Bayes	79%	39%	9%

Colhido os resultados utilizando todas as variáveis, foi analisado se é possível reduzir a quantidade de variáveis utilizada pelos modelos de forma a manter os índices parecidos de acurácia, precisão e *recall* e também identificar quais variáveis são mais significantes para realizar a classificação.

3.3.1 Seleção das Variáveis

3.3.1.1 Estratégia 1: Verificar Correlação entre as Variáveis

Esse tipo de verificação faz-se necessária, pois variáveis que apresentam forte correlação entre si podem ser retiradas da tabela, pois como estão fortemente correlacionadas, elas não ajudam o modelo a classificar melhor, além de não agregarem valor para a solução final [20].

Para fazer essa verificação foi utilizado o método `corr` contido na biblioteca pandas, que computa a correlação entre as colunas em pares, excluindo valores nulos, e o método de correlação selecionado foi o *pearson* [28].

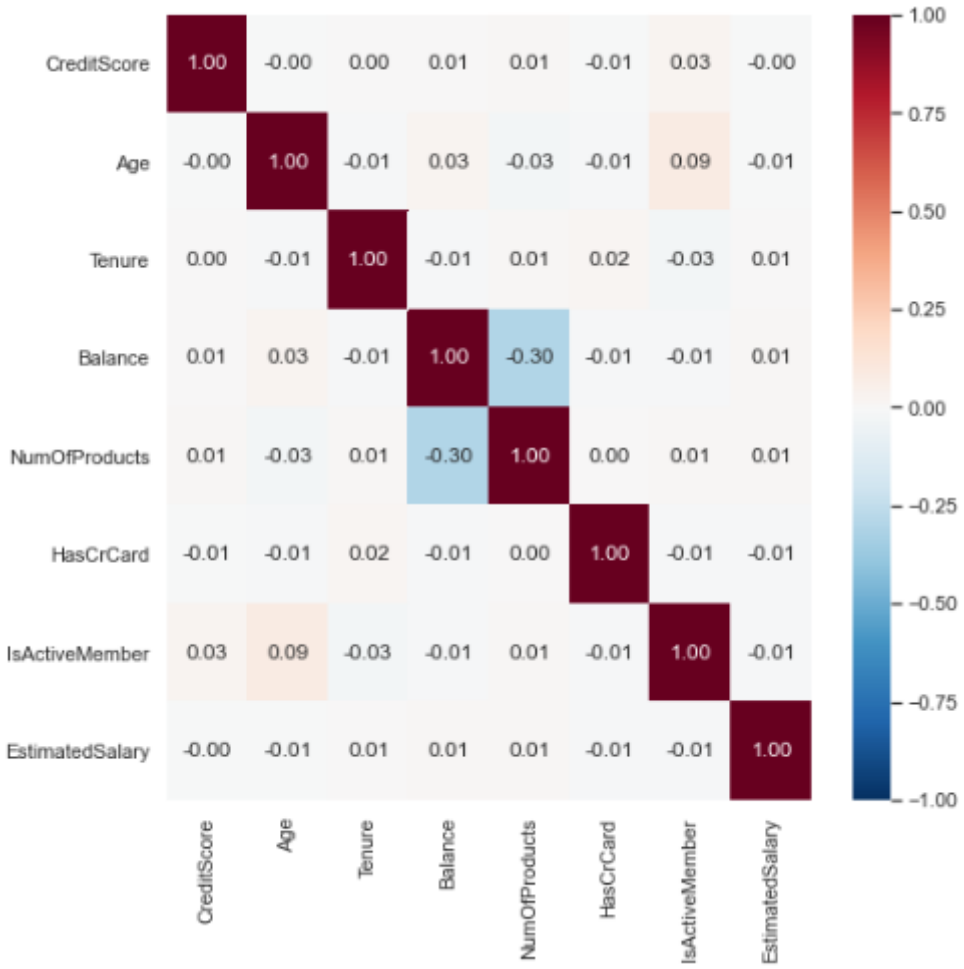
Após a realização dos cálculos de correlações, através da figura 3 é possível ver que todos os atributos possuem um baixo grau de correlação e portanto nenhuma variável será retirada do conjunto de dados.

3.3.1.2 Estratégia 2: Utilizar a Importância das Variáveis

Uma das vantagens de se utilizar o algoritmo *Random Forest* é o fato desse modelo retornar a importância relativa que cada varável teve para a classificação do problema. Esse método pode ser útil quando existe uma grande quantidade de atributos e é necessário entender quais contribuirão melhor para a resolução do problema.

Existem algumas formas de determinar o grau de importância de cada atributo. Uma dessas formas seria usar o ganho de informação em cada nível da árvore para determinar o ganho de informação relativo entre os atributos. Uma vez que o ganho de informação foi computado para cada árvore da floresta, pode-se tirar a média dos valores encontrados e, então, descobrir a importância de cada variável para a floresta aleatória.

Figura 3 – Correlação das variáveis



Os detalhes da implementação do algoritmo para o cálculo da importância das variáveis em uma floresta aleatória podem ser encontrados em [10].

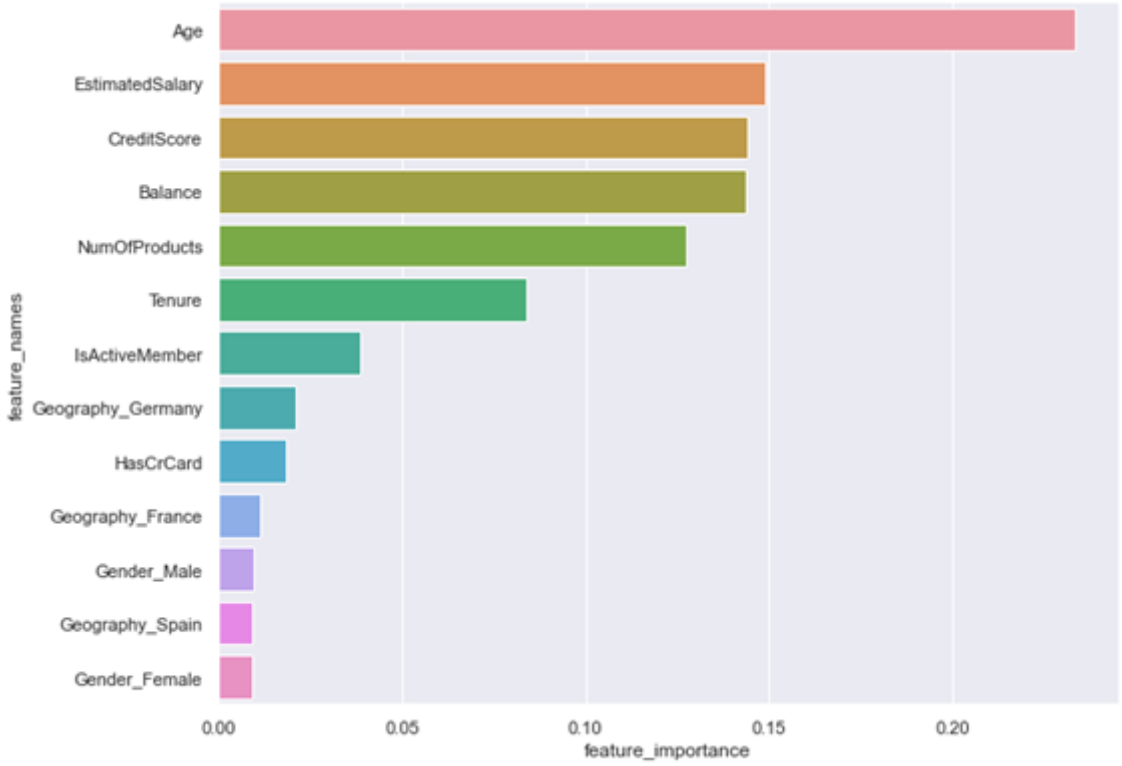
Pela figura 4, é possível observar que os atributos de maior importância são: *Age*, seguido por *EstimatedSalary*, *CreditScore*, *Balance* e *NumOfProducts*.

Assim, uma vez identificados as variáveis que mais contribuem para a classificação dos clientes, segundo o *Random Forest*, foi feito um processo interativo aglutinando os cinco primeiros atributos, de forma a verificar se esses atributos individualmente conseguem produzir um resultado melhor do que utilizando todos os atributos da tabela 2. Desse modo, obtém-se os seguintes grupos para o teste.

1. *Age*;
2. *Age*, *EstimatedSalary*;
3. *Age*, *EstimatedSalary*, *CreditScore*;
4. *Age*, *EstimatedSalary*, *CreditScore*, *Balance*;

5. *Age*, *EstimatedSalary*, *CreditScore*, *Balance*, *NumOfProducts*;
6. Todas as variáveis da tabela 2.

Figura 4 – Importância das variáveis retornado pelo modelo *Random Forest*.



Então colocou-se cada um desses grupos como atributos de entrada para cada modelo selecionado para o estudo e comparou-se o desempenho de cada um dos grupos, com todos os atributos existentes na tabela 2 através das métricas: acurácia, precisão e *recall*.

Segue abaixo o gráfico de cada resultado.

Figura 5 – Resultado: *Random Forest*.

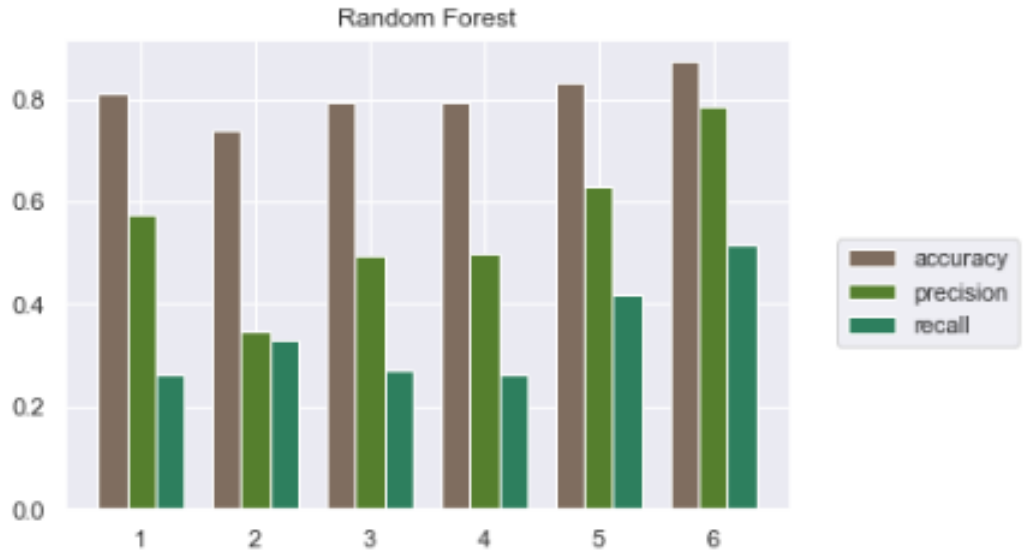


Figura 6 – Resultado: Regressão Logística.

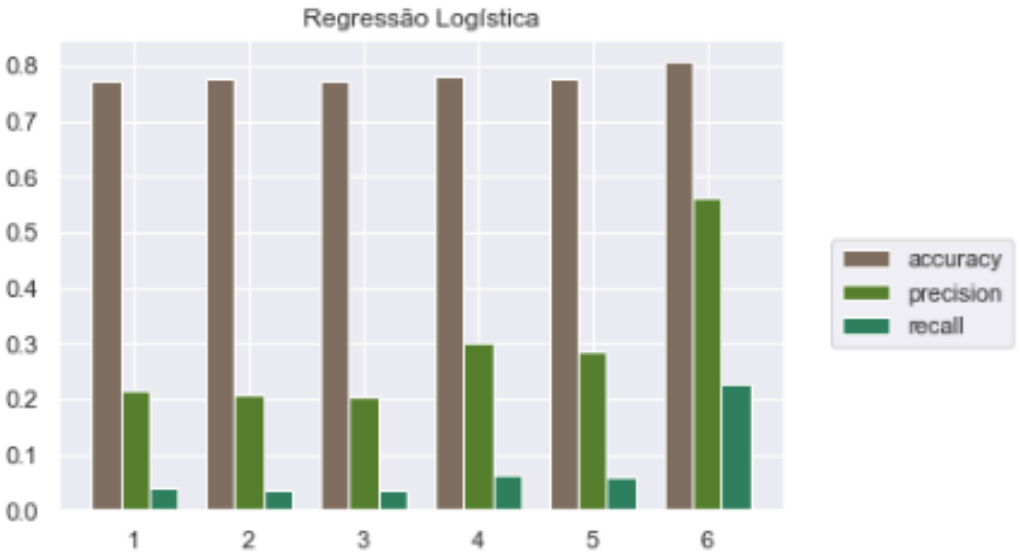


Figura 7 – Resultado: *Naive Bayes*.

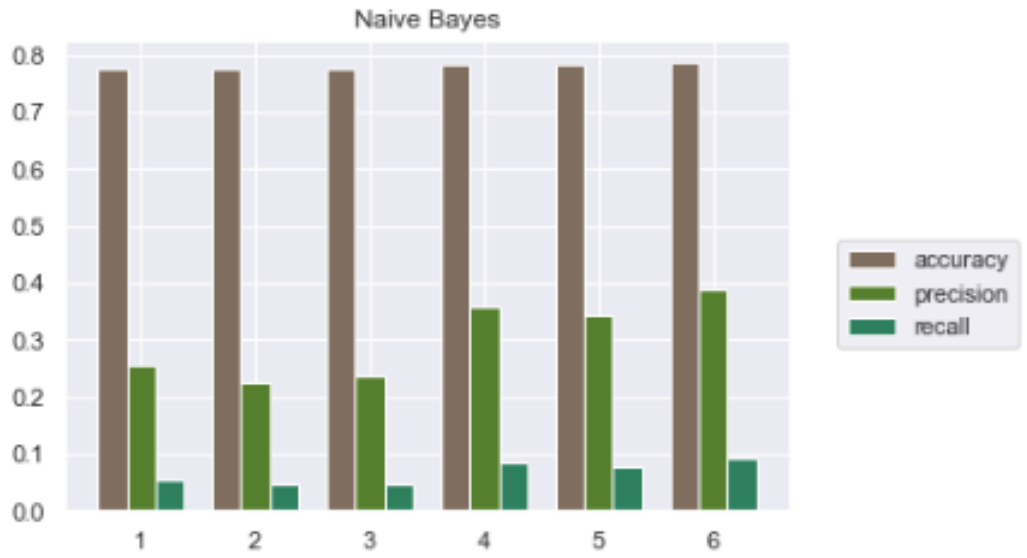
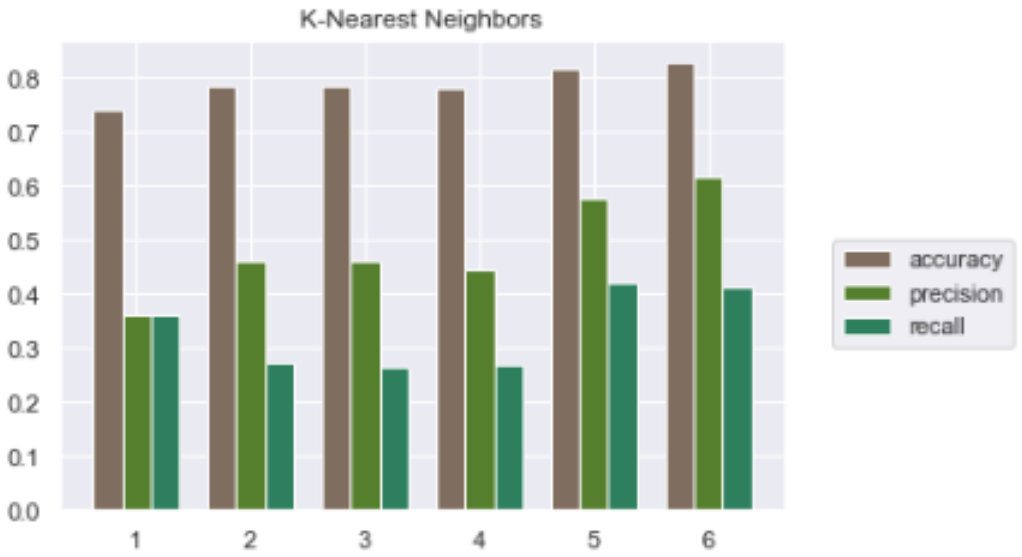


Figura 8 – Resultado: *K-Nearest Neighbors*.



4 Resultados e Discussão

Os modelos, de forma geral, apresentaram níveis de acurácia altos para todos os grupos. Isto possivelmente pode ser explicado pelo fato de o conjunto de dados ser desbalanceado. Existem 7.963 observações classificadas como não *churn* e 2.037 observações classificadas como churn. Desta forma, nota-se que a acurácia não é melhor indicador de desempenho desse estudo, pois ela indica apenas o percentual de acerto geral do modelo. Assim foi escolhido como métrica principal de avaliação dos modelos o *recall*, porque está métrica indica o percentual de captura dos clientes que estão propensos a realizar o *churn*.

4.1 Análise dos Resultados: *Random Forest*

O algoritmo *random forest* é um método de *ensemble*, ou seja, uma combinação de algoritmos, cujo objetivo é criar uma floresta de árvores descorrelacionadas. Para isso, o algoritmo utiliza-se das técnicas como *bootstrap* e *bagging*. No entanto, diferentemente do *bagging* apenas uma parcela das variáveis da amostra *bootstrap* compõem as árvores numa *random forest*. Assim, ao observar o gráfico de resultados, nota-se que o pior desempenho é quando o algoritmo é alimentado com apenas uma variável e o melhor desempenho é obtido ao deixar o algoritmo utilizar todas as variáveis, o que condiz com o funcionamento do modelo, pois ao utilizar uma única variável não é possível criar uma floresta de árvores descorrelacionadas, mas conforme aumenta a quantidade de variáveis, obtém-se árvores descorrelacionadas que acarretam na diminuição da variabilidade da função de predição, assim obtendo resultados melhores.

4.2 Análise dos Resultados: Regressão Logística

A regressão logística é um algoritmo de classificação que retorna a probabilidade de uma observação pertencer a uma certa classe. Assim, como a regressão linear, a regressão logística também necessita encontrar um coeficiente, beta (β_i) ideais que ponderem as respectivas variáveis independentes, para fazer isto, o algoritmo pode utilizar algum método de optimização, a máxima verossimilhança é um método comumente utilizado por ter melhores propriedades estatísticas [9].

Ao olhar para o gráfico da figura 6, verifica-se que no geral o desempenho foi pior quando comparado ao *random forest*. Além disso, para os grupos (1, 2 e 3) as métricas de desempenho foram muito parecidas, apenas apresentando uma melhora na precisão e no *recall* para os grupos (4 e 5). Por fim, as melhores métricas de desempenho foram obtidas quando todas as variáveis foram utilizadas. Acredita-se que este resultado deve-se

ao fato da regressão logística tratar-se de um algoritmo paramétrico, que assume certas premissas a respeito do conjunto de dados como, por exemplo: a relação linear entre o vetor das variáveis explicativas X e a variável dependente Y, valor esperado dos resíduos igual a zero, ausência de heterocedasticidade e ausência de multicolinearidade [22]. Como foi analisado anteriormente, o conjunto de dados não possui multicolinearidade, então acrescentar mais variáveis explicativas contribuiu para que o modelo conseguisse criar uma combinação linear das variáveis que melhor realizasse a classificação.

4.3 Análise dos Resultados: *Naive Bayes*

Como é explicado em [16] o algoritmo *Naive Bayes* assume uma premissa que é quase sempre violada nas aplicações no mundo real: dado um conjunto de atributos, todos os atributos são independentes entre si.

Além disso, intuitivamente, como a premissa que se baseia o algoritmo quase nunca é verdadeira é esperado que seu desempenho não seja muito bom, mas apesar disso, como mencionado anteriormente, o classificador consegue obter bons níveis de acurácia, quando comparados a modelos mais robustos [29].

Através do gráfico da figura 7, pode-se constatar esta afirmação, pois em todos os grupos a acurácia manteve níveis altos, em média 78%, porém não pode-se dizer o mesmo do *recall* que se manteve uma média de 6%, atingindo o ponto máximo (9%) ao utilizar todas as variáveis do conjunto de dados.

Dessa forma, apesar do algoritmo conseguir manter altas taxas de acurácia, mesmo assumindo que as variáveis explicativas são independentes entre si, para o conjunto utilizado e para a métrica avaliada não foi possível obter um desempenho aceitável desse algoritmo quando comparado com o *random forest*.

4.4 Análise dos Resultados: *K-Nearest Neighbors*

O *K-Nearest Neighbors* (KNN), em linhas gerais, assume que observações que estão próximas possuem classificações semelhantes, ou seja, o algoritmo faz a classificação com base nos K vizinhos mais próximos, realizando uma votação majoritária. Assim, neste trabalho, foi escolhido a opção padrão K igual 3.

Ao olhar o gráfico da figura 8, nota-se que o KNN obteve um *recall* de 36% no grupo 1, que utiliza apenas um atributo (*Age*), porém, para os grupos (2, 3 e 4) o desempenho da métrica *recall* diminuiu, acredita-se que isso foi motivado pelo fato das variáveis terem sido escolhidas com base na importância das variáveis da *random forest*. Além disso, os grupos (5 e 6) apresentam um *recall* respectivamente de 42% e 41%, vale ressaltar, que apesar de a métrica de desempenho ter melhorado comparando o grupo 5 em relação ao grupo 1,

ao comparar o grupo 6 com o grupo 5 o *recall* diminuiu, apesar de usar uma quantidade maior de atributos explicativos, também acredita-se que isso ocorreu devido a maldição da dimensionalidade, pois em conformidade com o aumento das variáveis explicativas torna-se mais custoso o cálculo da distância entre as variáveis e além disso a distância entre a observação mais próxima e mais distante tornam-se parecidas, assim não sendo possível discriminar as observações [30].

5 Conclusão

Como exposto, o objetivo do estudo consiste em comparar o desempenho de alguns algoritmos de aprendizado de máquina e entender qual ou quais variáveis são mais importantes para a classificação dos clientes propensos ao *churn*.

Observando os resultados obtidos e analisados, pode-se concluir que o algoritmo *random forest* obteve o melhor desempenho com um *recall* igual a 52%, ou seja, o modelo é capaz de capturar mais da metade dos clientes propensos ao *churn*, ainda vale mencionar que esse resultado foi obtido utilizando as configurações padrões do modelo, assim existe a possibilidade de melhorar esse desempenho testando outras configurações. Além disso, o modelo obteve este resultado utilizando todos os atributos do conjunto de dados, ou seja, não foi possível afirmar que existe uma ou algumas variáveis em específico que permitem uma melhor classificação,

No entanto, vale ressaltar que apesar de todas as variáveis contribuírem para a classificação, algumas tem uma importância maior. Assim, no mundo real faz sentido destacar esses atributos para que essas variáveis sejam priorizadas em algum tipo de ação para mitigar o *churn*.

Referências

1 GARCIA, N. *Cinco maiores bancos concentram 78% dos lucros do sistema bancário em 2021*. 2021. <https://www1.folha.uol.com.br/mercado/2022/08/cinco-maiores-bancos-concentram-78-dos-lucros-do-sistema-bancario-em-2021.shtml>. [Acessado em 23 de Outubro de 2022]. Citado na página 1.

2 FEITOSA, M. C. de G. B. impactos na performance financeira do mercado bancário brasileiro. *REVISTA DEBATES EM ECONOMIA APLICADA*, n. 5, p. 6–10, November 2021. Citado na página 1.

3 DISTRITO. *Distrito Fintech Mining Report*. 2021. <https://f.hubspotusercontent30.net/hubfs/7735036/MR-FINTECH%202021-V13.pdf>. [Acessado em 2 de Novembro de 2022]. Citado na página 1.

4 ULLAH, I. et al. A churn prediction model using random forest: Analysis of machine learning techniques for churn prediction and factor identification in telecom sector. *IEEE*, p. 1–16, May 2019. Citado na página 1.

5 SHAABAN, E. et al. A proposed churn prediction model. *ACADEMIA*, p. 693–697, July 2012. Citado na página 1.

6 POPOVIC, D.; BASIC, B. D. Churn prediction model in retail banking using fuzzy c-means algorithm. p. 243–247, February 2009. Citado na página 1.

7 OLIVEIRA, B. A. C. D.; TOLEDO, G. L.; IKEDA, A. A. Fidelização e valor: uma interdependência inequívoca. 2004. Citado na página 3.

8 DARÉ, P. R. C.; CAMPOMAR, M. C. Retenção de clientes à luz do gerenciamento de churn: um estudo no setor de telecomunicações. 2007. Citado na página 3.

9 JAMES, G. et al. *An Introduction to Statistical Learning*: with application in r. [S.l.]: Springer, 2013. ISBN 978-1-4614-7137-0. Citado 5 vezes nas páginas 4, 5, 6, 7 e 19.

10 HARTSHORN, S. *Machine Learning with Randon Forest and Decision Trees*: A visual guide for beginners. [S.l.: s.n.], 2016. Citado 2 vezes nas páginas 4 e 15.

11 PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011. Citado na página 5.

12 MARQUÉS, A. I.; GARCÍA, V.; SÁNCHEZ, J. S. Exploring the behaviour of base classifiers in credit scoring ensembles. *Expert systems with Applications*, Elsevier, v. 39, n. 11, p. 10244–10250, 2012. Citado na página 5.

13 FERREIRA, E. V. *Métodos baseados em árvores*. 2019. <http://cursos.leg.ufpr.br/ML4all/apoio/Arvores.html>. [Acessado em 20 de Outubro de 2022]. Citado na página 6.

14 TEAM, C. *Bagging (Bootstrap Aggregation)*. 2022. <https://corporatefinanceinstitute.com/resources/data-science/bagging-bootstrap-aggregation/>. [Acessado em 10 de Novembro de 2022]. Citado na página 5.

15 THE k-NN algorithm. 2022. <https://www.cs.cornell.edu/courses/cs4780/2018fa/lectures/lecturenote02_kNN.html>. [Acessado em 08 de Novembro de 2022]. Citado na página 6.

16 ZHANG, H. The optimality of naive bayes. 2004. [Acessado em 25 de Outubro de 2022]. Disponível em: <http://www.cs.unb.ca/~hzhang/publications/FLAIRS04ZhangH.pdf>. Citado 2 vezes nas páginas 7 e 20.

17 KIRK, M. *Thoughtful machine learning*. [S.l.]: "O'Reilly Media, Inc.", 2017. Citado na página 7.

18 PARSIAN, M. *Data algorithms: Recipes for scaling up with hadoop and spark*. [S.l.]: "O'Reilly Media, Inc.", 2015. Citado 2 vezes nas páginas 7 e 8.

19 BOSLAUGH, S. *Statistics in a nutshell: A desktop quick reference*. [S.l.]: "O'Reilly Media, Inc.", 2012. Citado na página 8.

20 HARRISON, M. *Machine Learning Guia de Referências Rápidas*: Trabalhando com dados estruturados em python. [S.l.]: Novatec Editora Ltda, 2020. ISBN 978-85-7522-818-0. Citado 3 vezes nas páginas 8, 13 e 14.

21 BARBOSA, M. T. *Proposta de implementação paralela do classificador naive bayes em FPGA*. Dissertação (B.S. thesis) — Universidade Federal do Rio Grande do Norte, 2021. Citado na página 8.

22 FáVERO, L. P. et al. *análise de dados*: modelagem multivariada para tomada de decisões. [S.l.]: Elsevier, 2009. ISBN 9788535230468. Citado 3 vezes nas páginas 8, 9 e 20.

23 MAIA, A. G. *Econometria: conceitos e aplicações*. [S.l.]: Saint Paul Editora, 2019. Citado na página 9.

24 NAKAMURA, K. G. *Multicolinearidade em modelos de regressão logística*. Tese (Doutorado) — Universidade de São Paulo, 2013. Citado na página 9.

25 CLASSIFICAçãO: precisão e recall. 2022. <https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall>. [Acessado em 11 de Novembro de 2022]. Citado na página 10.

26 KMALIT. *Bank Customer Churn Prediction*. <https://www.kaggle.com/code/kmalit/bank-customer-churn-prediction/notebook>. [Acessado em 30 de Outubro de 2022]. Citado na página 12.

27 GéRON, A. *Hands on Machine Learning with Scikit-Learn, Keras & TensorFlow*: Concepts, tools, and techniques to build intelligent systems. [S.l.]: O'Reilly, 2019. ISBN 978-1-492-03264-9. Citado na página 13.

28 COEFICIENTE de correlação de Pearson. 2022. <http://www5.eesc.usp.br/saate/index.php/saate/Indicar-a-T%C3%A9cnica/Associar/2.-%C3%81rvore-de-decis%C3%A3o/Gloss%C3%A1rio/Coeficiente-de-correla%C3%A7%C3%A3o-de-Pearson>. [Acessado em 09 de Novembro de 2022]. Citado na página 14.

29 DOMINGOS, P.; PAZZANI, M. Beyond independence: Conditions for the optimality of the simple bayesian classifier. 1997. Disponível em: <https://www.ics.uci.edu/~pazzani/Publications/mlc96-pedro.pdf>. Citado na página 20.

30 KOUIROUKIDIS, N.; EVANGELIDIS, G. The effects of dimensionality curse in high dimensional knn search. IEEE, 2011. Citado na página 21.