Francisco Almeida Barroso

Modelos de previsão de dificuldades financeiras de empresas com *Machine Learning*

Brasil

2022

Francisco Almeida Barroso

Modelos de previsão de dificuldades financeiras de empresas com *Machine Learning*

Projeto de Pesquisa apresentado ao Curso de Mestrado Profissional em Economia, Universidade de Brasília, como requisito parcial para a obtenção do título de Mestre em Economia

Universidade de Brasília - UnB

Faculdade de Administração Contabilidade e Economia - FACE

Departamento de Economia - ECO

Programa de Pós-Graduação

Orientador: Prof. Daniel Oliveira Cajueiro PhD

Brasil

2022

Francisco Almeida Barroso
Modelos de previsão de dificuldades financeiras de empresas com *Machine Learning* / Francisco Almeida Barroso. – Brasil, 2022-
56p. : il. (algumas color.) ; 30 cm.

Orientador: Prof. Daniel Oliveira Cajueiro PhD

Dissertação (Mestrado) – Universidade de Brasília - UnB
Faculdade de Administração Contabilidade e Economia - FACE
Departamento de Economia - ECO
Programa de Pós-Graduação, 2022.

1. Dificuldades financeiras. 2. *Machine Learning*. 3. Indicadores financeiros. I. Universidade de Brasília. II. Faculdade de Administração, Contabilidade e Economia - FACE. III. Departamento de Economia IV. Modelos de previsão de dificuldades financeiras de empresas com *Machine Learning*

Francisco Almeida Barroso

Modelos de previsão de dificuldades financeiras de empresas com *Machine Learning*

Projeto de Pesquisa apresentado ao Curso de Mestrado Profissional em Economia, Universidade de Brasília, como requisito parcial para a obtenção do título de Mestre em Economia

Trabalho aprovado. Brasil, 24 de junho de 2022:

Prof. Daniel Oliveira Cajueiro PhD
Orientador

Prof. Herbert Kimura PhD
Convidado 1

Prof. Regis Augusto Ely PhD
Convidado 2

Brasil
2022

Agradecimentos

Dedico este trabalho aos meus pais Osmar Barroso Cordeiro (*in memorian*) e Maria Almeida Barroso e, em especial, à minha esposa Tatiana e ao meu filho Artur, que tiveram a compreensão de tolerar a minha ausência em alguns momentos.

Agradeço à Deus, Pai, Todo Poderoso e razão primeira da minha existência.

Agradeço aos professores do curso pela excelência da qualidade técnica de cada um e, em especial, ao meu orientador Daniel Olveira Cajueiro, pela atenção, disponiblidade e gentileza nas orientações.

Agradeço, ainda, aos colegas de estudos, dos quais destaco Jueline, Lemonier, Camila e Patrick, pelo apoio e incentivo durante o curso. Por fim, agradeço ao Pedro, bolsista da UnB, e ao meu amigo Nícolas pelas orientações na execução dos modelos no *Python*.

Resumo

Identificar antecipadamente se uma empresa tem propensão a enfrentar dificuldades financeiras é de grande relevância para diversos agentes da economia, em especial para credores e investidores. Neste estudo, com a utilização de técnicas de *Machine Learning*, comparamos alguns modelos que podem contribuir para diagnosticar antecipadamente possíveis dificuldades financeiras de empresas no futuro próximo. Para realizar nossa análise capturamos dados históricos de empresas reais listadas na Bolsa de Valores do Brasil (Brasil Bolsa Balcão - B3) do período de 2002 a 2021. Nosso objetivo é prever antecipadamente o estágio de dificuldade financeira de uma empresa que possa levá-la a oficializar um pedido de Recuperação Judicial (RJ) ou um pedido de Recuperação Extrajudicial (RE) no trimestre seguinte, representando um risco para credores e investidores. Temos um problema de classificação binária, em que nossa variável dependente é a possibilidade de pedir RJ ou RE no trimestre seguinte, ou não. Para compor as variáveis explicativas extraímos indicadores financeiros das demonstrações contábeis das empresas. Com esse objetivo, testamos diferentes técnicas de classificação considerando vários algoritmos de *Machine Learning*, tais como *Random Forest (RF)*, *Gradient Boosting (GB)*, *Logistic Regression (LR)*, *Naive Bayes (NB)*, *Suport Vector Machine (SVM)* e *Artificial Neural Networks (ANN)*. Os resultados obtidos demonstram que a maioria dos modelos estudados apresentam bom desempenho no cenário testado, sendo que o *RF* e o GB, considerando a métrica *F1-score*, superam os demais em todas as simulações. Após a execução do processo de *Cross Validation K-fold*, o *RF* supera o *GB*. Adicionalmente, uma importante contribuição do nosso trabalho foi demonstrar que os indicadores financeiros de liquidez, rentabilidade e endividamento desempenham um papel importante na previsão de dificuldades financeiras das empresas e que esses indicadores são, predominantemente, referentes aos três trimestres anteriores ao pedido de RJ ou RE oficializado pelas empresas.

Palavras-chave: Dificuldades financeiras, *Machine Learning*, Indicadores financeiros.

Abstract

Identifying in advance whether a company is likely to face financial difficulties is of great importance for various agents of the economy, especially for creditors and investors. In this study, using Machine Learning techniques, we compare some models that can help to diagnose in advance possible financial difficulties of companies in the near future. To carry out our analysis, we captured historical data from real companies listed on the Brazilian Stock Exchange (Brasil Bolsa Balcão - B3) from 2002 to 2021. Our objective is to predict in advance the stage of financial difficulty of a company that could lead it to formalize a request for Judicial Recovery (RJ) or a request for Extrajudicial Recovery (RE) in the following quarter, representing a risk for creditors and investors. we have a problem with binary classification, in which our dependent variable is the possibility of ordering RJ or RE in the following quarter, or not. To compose the explanatory variables, we extracted financial indicators from the companies' financial statements. With that objective, we tested different classification techniques considering several Machine Learning algorithms, such as Random Forest (RF), Gradient Boosting (GB), Logistic Regression (LR), Naive Bayes (NB), Support Vector Machine (SVM) and Artificial Neural Networks (ANN). The results obtained demonstrate that most of the models studied present good performance in the tested scenario, and the RF and GB, considering the F1-score metric, outperform the others in all simulations. After performing the K-fold Cross Validation process, the RF exceeds the GB. Additionally, an important contribution of our work was to demonstrate that the financial indicators of liquidity, profitability and indebtedness play an important role in predicting the financial difficulties of companies and that these indicators are predominantly referring to the three quarters prior to the RJ or RE request made official by the companies.

Keywords: Financial difficulties, Machine Learning, Financial indicators.

Sumário

Lista de tabelas

Lista de abreviaturas e siglas

ANN	*Artificial Neural Networks*
ANOVA	*Analysis Of Variance*
CF	Ciclo Financeiro
CO	Ciclo Operacional
CV	*Cross Validation*
DBA	Dívida Bruta sobre o Ativo
DBPL	Dívida Bruta sobre o Patrimônio Líquido
DCP	Dívida de Curto Prazo
DLPL	Dívida Líquida sobre o Patrimônio Líquido
EBIT	*Earnings Before Interest and Taxes* (Lucro Antes dos Juros e Imposto de Renda - LAJIR)
EG	Endividamento Geral
FN	*False Negative*
FP	*False Positive*
GA	Giro do Ativo
GAF	Grau de Alavancagem Financeira
GAO	Grau de Alavancagem Operacional
GB	*Gradient Boosting*
LC	Liquidez Corrente
LG	Liquidez Geral
LR	*Logistic Regression*
LREF	Lei de Recuperação de Empresas e Falências
LS	Liquidez Seca
MB	Margem Bruta

ML	Margem Líquida
MLP	*Multi-layer Perceptron*
NB	*Naive Bayes*
NOPAT	*Net Operating Profit After Tax* (Lucro Operacional Líquido após os Impostos)
PCT	Participação Capital Terceiros
PL	Patrimônio Líquido
PMPF	Prazo Médio de Pagamento aos Fornecedores
PMRE	Prazo Médio de Rotação dos Estoques
PMRV	Prazo Médio de Recebimento das Vendas
RF	*Random Forest*
RE	Recuperação Extrajudicial
RJ	Recuperação Judicial
ROA	*Return on Assets*
ROC	*Receiver Operating Character*
ROE	*Return on Equit*
ROIC	*Return on Invested Capital*
SVM	*Support Vector Machine*
TN	*True Negative*

Lista de ilustrações

1 Introdução

A disponibilidade de crédito para as empresas e a disposição de investimento pelos agentes econômicos são condições necessárias para alavancar o crescimento das economias. Contudo, implementar tais condições não é tarefa fácil, haja vista, a existência de riscos que devem ser mensurados, tal como o risco de crédito e o risco do negócio. É necessário, portanto, que os credores e os investidores disponham de mecanismos ágeis e eficientes para avaliação de risco antecipado que possam identificar se uma empresa tem propensão a enfrentar dificuldades financeiras que possa leva-la a pedir Recuperação Judicial (RJ) ou Recuperação Extrajudicial (RE) ou, até mesmo decretar falência.

Encontrar o modelo ideal para prever dificuldades financeiras das empresas tem sido uma das principais preocupações dos credores, investidos, analistas e também de estudiosos do assunto. Realizar um diagnóstico assertivo da situação econômica e financeira da empresa, antes da tomada de decisão, seja de conceder um crédito ou de fazer um investimento, pode evitar perdas financeiras pesadas. Zhang, Zhao e Yao (2021) afirmam que avisos de risco antecipados precisos podem não apenas reduzir as perdas dos investidores, mas também ajudar a manter a estabilidade do mercado financeiro. Nesse contexto, há um debate importante em busca de um modelo que atenda esses anseios.

O objetivo do nosso estudo é testar modelos de *Machine Learning* para previsão de dificuldades financeiras de empresas, que possam identificar antecipadamente se a empresa apresenta características que podem levá-la a pedir RJ ou RE em um período próximo, como o trimestre seguinte, por exemplo, ou seja, identificar a situação de dificuldades financeiras anterior à insolvência. Conforme Geng, Bose e Chen (2015), a dificuldade financeira de uma empresa geralmente se refere à situação em que seu fluxo de caixa operacional não pode substituir os ativos líquidos negativos. Tal situação pode trazer severas perdas para os proprietários da empresa e para os credores em geral, portanto é importante termos modelos eficientes de previsão. Como esse objetivo aplicamos e comparamos técnicas de classificação com algoritmos de *Machine Learning* tais como *Random Forest (RF)*, *Gradient Boosting (GB)*, *Logistic Regression (LR)*, *Naive Bayes (NB)*, *Suport Vector Machine (SVM)* e *Artificial Neural Networks (ANN)*.

A literatura mostra que muitos modelos de previsão de falências ou de previsão de inadimplência já foram desenvolvidos ao longo do tempo, desde modelos utilizando técnicas tradicionais como análise discriminante e regressão logística a modelos mais sofisticados com a utilização de inteligência artificial e algoritmos de *Machine Learning* como *ANN*, *SVM*, *RF*, entre outros. A maior parte desses estudos utilizam indicadores de balanços como insumos para construção das variáveis explicativas. Um dos primeiros

modelos de previsão de falência, considerado tradicional, foi realizado por Edward et al. (1968), que aplicou análise multivariada com base em indicadores financeiros extraídos das demonstrações financeiras das empresas.

Com o desenvolvimento da inteligência artificial houve um aprimoramento dos modelos de previsão de falências. Para Leo, Sharma e Maddulety (2019), as técnicas de aprendizado de máquina demonstram ter um desempenho melhor do que as técnicas estatísticas tradicionais, tanto na classificação quanto na precisão preditiva. Barboza, Kimura e Altman (2017), constataram que os modelos de *Machine Learnig* apresentam, em média, aproximadamente 10% mais precisão em relação aos modelos tradicionais.

Outros trabalhos recentes, nessa mesma linha de pesquisa, utilizaram indicadores de balanços para prever dificuldades financeiras de empresas e compararam diferentes modelos de *Machine Learning*, como os estudos de Geng, Bose e Chen (2015), Teles et al. (2021) e BoneLLo, BrÉdart e VeLLa (2018). Os resultados mostram a força preditiva dos modelos de *Machine Learning*, porém não ha um modelo que apresente melhor desempenho em todas as situações. Portanto, novos estudos explorando diferentes modelos, contextos e conjuntos de dados, são relevantes, uma vez que os resultados sobre a superioridade dos modelos ainda são inconclusivos (BARBOZA; KIMURA; ALTMAN, 2017).

Para realização do nosso estudo, acessamos um banco de dados de companhias de capital aberto listadas na Bolsa de Valores do Brasil (Brasil, Bolsa, Balcão - B3), contendo índices financeiros de balanços de 426 empresas, das quais 46 declararam dificuldades financeiras e pediram RJ ou RE no período de 2005 a 2021. O período de 2005 a 2021 foi definido considerando o início da vigência da Lei de Recuperação de Empresas e Falências (LREF) do Brasil, nº 11.101, de 9 de fevereiro de 2005 (BRASIL, 2005), que dispõe sobre recuperação judicial, recuperação extrajudicial e falência dos empresários e das sociedades empresárias.

De acordo com a LREF, a petição inicial de recuperação judicial deve ser instruída com a exposição das causas concretas da situação patrimonial do devedor e das razões da crise econômico-financeira, anexando as demonstrações contábeis dos último três exercícios. Conforme Art. 47 da referida Lei, a RJ tem por objetivo viabilizar a superação da situação de crise econômico-financeira do devedor, a fim de permitir a manutenção da fonte produtora, do emprego dos trabalhadores e dos interesses dos credores, promovendo, assim, a preservação da empresa, sua função social e o estímulo à atividade econômica. Analisar as demonstrações contábeis das empresas referentes aos últimos três exercícios contábeis é uma prática do mercado, quando das avaliações de risco de crédito, realizadas pelos bancos, investidores e agências de *ratings*.

Temos, portanto, um problema de classificação, onde queremos prever se uma empresa vai pedir RJ ou RE no trimestre seguinte. Nossa variável dependente é uma *dummye*, com $y = 1$ se a empresa vai pedir RJ ou RE e $y = 0$ se a empresas não vai pedir

RJ ou RE. As variáveis explicativas são indicadores financeiros extraídos das demonstrações contábeis das empresas, correspondentes a três anos, pertencentes aos seguintes grupos: liquidez, estrutura de capital e endividamento, rentabilidade e desempenho, alavancagem e atividade e eficiência. Para empresas que pediram RJ ou RE colhemos os 12 trimestres anteriores ao pedido de RJ ou RE, entre os anos de 2002 a 2021 e, para as empresas que não pediram recuperação judicial, tomamos os trimestres de dezembro de 2018 a setembro de 2021. Além dos indicadores financeiros de balanços, utilizamos um indicador de governança (segmento de listagem) e indicador referente ao setor econômico de atuação da empresa.

Diante do exposto, além desta introdução, nosso estudo contém mais 5 capítulos. No capítulo 2 descrevemos, de forma resumida, os modelos preditivos utilizados. No capítulo 3 apresentamos os dados e variáveis, como estão estruturados, a forma de tratamento, como são calculados os indicadores de balanços e apresentamos as variáveis categóricas utilizadas; No capítulo 4 trazemos os aspectos metodológicos adotados no estudo, onde descrevemos as técnicas de *feature selection (Feature importance)*, balanceamento das amostras e métricas de validação e otimização dos modelos. No capitulo 5 fazemos a análise dos resultados obtidos, com comparações e avaliações com as métricas de *Accuracy*, Curva *ROC*, *Precision*, *Recall*, *F1-score*, Matriz de Confusão, *Cross validation* e otimização com hiperparâmetros, conforme propomos no capítulo 4. Por fim, uma breve conclusão, com recomendações para estudos futuros.

2 Modelos preditivos

Nosso objetivo neste capítulo é fazer uma breve explanação dos modelos preditivos utilizados no estudo. Aplicamos técnicas de classificação com algoritmos de *Machine Learning* para prever dificuldades financeiras de empresas. *Machine Learning* é uma inteligência artificial que permite que computadores tomem decisões com a ajuda de algoritmos. Esses algoritmos reconhecem padrões e se tornam capazes de fazer predições. A predição usa as informações que você tem, geralmente chamadas de "dados", para gerar as que não tem. Conforme Agrawal, Gans e Goldfarb (2020) predição é o processo de preencher as informações ausentes. Leo, Sharma e Maddulety (2019) afirmam que a adoção do aprendizado de máquina tem sido motivada pelas oportunidades potenciais de redução de custos, melhoria da produtividade e melhor gestão de riscos.

Os problemas de *Machine Learning* são divididos, basicamente, em aprendizagem supervisionada (*Supervised Learning*) e aprendizagem não supervisionada (*Unsupervised Learning*). Os modelos de aprendizagem supervisionada abrangem as subáreas da Classificação e Regressão. Utilizam dados já observados que possuem respostas ou classificações rotuladas, possibilitando comparar as previsões das respostas/rótulos com os rótulos reais. Os algoritmos de classificação são algoritmos de aprendizagem supervisionada, cujo objetivo é prever uma classe ou rótulo associado com uma variável de entrada, contendo determinados atributos. De acoredo com Hastie et al. (2009), quando se utilizam conjuntos de valores de entrada e de saída afim de tentar prever novos valores de saída, este tipo de aprendizado se denomina supervisionado. Inicialmente, o algoritmo é treinado com um conjunto de dados com classes conhecidas, podendo estes dados estarem divididos em somente duas classes (classificação binária) ou em várias classes (classificação multiclasse). Após a aplicação dos modelos, escolhemos o(s) algoritmo(s) que apresenta(m) o melhor desempenho para os dados em questão. No nosso estudo testamos os seguintes algoritmos de classificação.

2.1 *Random Forest*

Random Forests (RF) é um modelo bastante utilizado, portanto bem popular, desenvolvido por Breiman (2001). É formado por uma combinação de preditores de árvores de modo que cada árvore depende dos valores de um vetor aleatório amostrado independentemente e com a mesma distribuição para todas as árvores da floresta. Para Teles et al. (2021), a popularidade das florestas aleatórias pode ser atribuída ao fato de que além de serem rápidas e fáceis de implementar, elas também produzem previsões precisas e podem lidar com inúmeras variáveis de entrada sem *overfitting*. Esses autores

afirmam, ainda, que elas são consideradas entre as técnicas de aprendizado de máquina mais precisas no mercado.

De acordo com a literatura, os algoritmos *RF* são criados por várias *decision tree*, geralmente treinados com o método de *bagging*, cuja ideia principal é que a combinação de modelos aumenta o resultado final, ou seja, um classificador de *RF* tem todos os hiperparâmetros de uma *decision tree* e também todos os hiperparâmetros de um classificador de *bagging*, para controlar a combinação de árvores. Ao invés de construir um classificador *bagging* e passá-lo para um classificador de *decision tree*, é mais conveniente utilizar a classe do *RF*. Petropoulos et al. (2020) afirmam que a filosofia básica do *RF* se baseia na combinação de três conceitos: (i) *decision tree* de classificação ou regressão, (ii) agregação ou ensacamento *bootstrap* e (iii) subespaços aleatórios.

No estudo de Barboza, Kimura e Altman (2017) o *RF* obeve o melhor desempenho em comparação com modelos tradicionais de previsão de falência (análise discriminante e *Logistic Regression*) e os modelos de *Machine Learning* (*SVM*, aprendizagem por reforço e ensacamento).

2.2 *Logistic Regression*

A *Logistic Regression (LR)* é uma técnica estatística muito utilizada para construir, com base em um conjunto de dados, um modelo que possibilita a predição de valores a partir de uma variável categórica, frequentemente binária, em função de uma ou mais variáveis independente contínuas e/ou binárias. Ou seja, é um algoritmo que lida com questões e problemas de classificação, analisando diferentes aspectos ou variáveis de um objeto para depois determinar uma classe na qual ele se encaixa melhor.

De acordo com Veganzones, Séverin e Chlibi (2021), o modelo de *LR* é um dos principais métodos de previsão usados em modelos de falhas corporativas por causa de sua capacidade de interpretação e análise de dados e sua robustez. Esses mesmos autores afirmam, ainda, que o modelo de *LR* é mais adequado do que a análise discriminante porque é menos exigente sobre suposições estatísticas de probabilidades anteriores e distribuições de preditores. Para Bruce e Bruce (2019), sua popularidade se deve ao fato da sua grande velocidade computacional para estimação dos parâmetros e sua facilidade de implementação para pontuação de novos dados.

Há três modelos principais de *LR*: binomial, ordinal e multinomial. Considerando objetivo do nosso estudo, usamos o modelo de regressão logística binomial, cujos objetos são classificados em dois grupos ou categorias. Por exemplo, identificar se uma empresa tem propensão a pedir recuperação judicial no próximo trimestre ou não.

2.3 *Gradient Boosting*

O *Gradient Boosting (GB)* é uma técnica *boosting* que segundo Friedman (2002), constrói modelos de regressão aditiva ajustando sequencialmente uma função parametrizada simples (aprendiz de base) para os "pseudo-residuais atuais por mínimos quadrados em cada iteração. Trata-se de uma generalização do método de Adaboost proposto por Freund, Schapire et al. (1996), utilizado para a resolução de problemas de classificação e regressão.

2.4 *Support Vector Machines*

O algoritmo *Support Vector Machines (SVM)* é um algoritmo bastante utilizado para problemas de classificação, embora possa ser usado também para problemas de regressão (LEO; SHARMA; MADDULETY, 2019). Apesar de o treinamento do *SVM* geralmente ser lento, esses modelos exigem poucos ajustes e tendem a apresentar boa acurácia, conseguindo modelar fronteiras de decisão complexas e não lineares (ESCOVEDO; KOSHIYAMA, 2020).

De forma resumida, o *SVM* realiza um mapeamento não linear (utilizando funções *kernel*) para transformar os dados de treino originais em uma dimensão maior, buscando nesta nova dimensão um hiperplano que separe os dados linearmente de forma ótima. Com um mapeamento apropriado para uma dimensão suficientemente alta, dados de duas classes poderão ser sempre separados por um hiperplano. O *SVM* encontra este hiperplano usando vetores de suporte (exemplos essenciais para o treinamento) e margens, definidas pelos vetores de suporte. Para previsão de risco de crédito, Teles et al. (2021) encontrou no modelo *SVM* uma maior precisão preditiva em comparação com o *RF*.

2.5 *Naive Bayes*

Naive Bayes (NB) é um teorema desenvolvido no século XVIII pelo matemático britânico Thomas Bayes, que consiste em uma fórmula de categorização para determinar a probabilidade condicional (Teoria de Bayes). Segundo Soria et al. (2011), é uma técnica de classificação supervisionada rápida que é adequada para tarefas de previsão e classificação em conjunto de dados complexos e incompletos. O teorema fornece uma maneira de revisar as previsões ou teorias existentes com base em evidências novas ou adicionais, com a suposição adicional de independência entre os preditores (TELES et al., 2021).

2.6 *Artificial Neural Networks*

Artificial Neural Networks (ANN) são frequentemente usadas para prever dificuldades financeiras de empresas. São sistemas de computação com nós interconectados que

funcionam como os neurônios do cérebro humano. São compostas por camadas de um nó, com uma camada de entrada, uma ou mais camadas ocultas e uma camada de saída. Cada nó, ou neurônio artificial, conecta-se a outro e tem um peso e um limite associados. Se a saída de qualquer nó individual estiver acima do valor do limite especificado, esse nó será ativado, enviando dados para a próxima camada da rede. Caso contrário, nenhum dado será transmitido para a próxima camada da rede. Segundo Tavana et al. (2018), a *ANN* define uma função das variáveis de entrada (conjunto de dados) e tenta encontrar os melhores pesos (coeficientes) para as variáveis. Uma vez que esta função tenha "aprendido" o suficiente, está pronta para aproximar os valores alvo e fazer as previsões.

Geng, Bose e Chen (2015), ao comparar modelos de previsão de dificuldades financeiras de empresas, identificaram as *NNA* como o modelo com melhor precisão em relação a outros classificadores, como *Decision tree* e *SVM*. Já BoneLLo, BrÉdart e VeLLa (2018) constataram a força e capacidade dos modelos de *Machine Learning* para prever negócios eminentes de fracasso, mas os resultados indicaram que a *Decision tree* obteve maior desemprenho preditivo em comparação aos classificadores *NB* e *ANN*.

3 Dados e Variáveis

Este capítulo tem como objetivo apresentar de forma mais detalhada os dados usados para realização do estudo e como foram tratados, bem como demonstrar quais variáveis são utilizadas.

3.1 Dados

Os dados do estudo foram extraídos da plataforma economática [1], considerada a maior empresa de informações financeiras sobre o mercado latino-americano. Utilizamos dados das empresas brasileiras listadas na bolsa de valores do Brasil (Brasil Bolsa Balcão – B3). Foram obtidos dados de 426 empresas, considerando todas as empresas brasileiras listadas, incluindo 46 empresas que oficializaram pedido de RJ ou RE no período de 2005 a 2021. Contudo, considerando a existências de empresas com dados ausentes, para garantir um modelo de previsão mais estável, foi necessária a exclusão dessas empresas. Assim, após o tratamento da base, trabalhamos com 279 empresas, incluindo 44 empresas que pediram RJ ou RE no período considerado.

Na fase de tratamento dos dados foram excluídas da base as empresas com dados ausentes em 5 ou mais períodos em pelo menos uma variável, ou seja, considerando que cada variável contém 12 trimestres (equivalente a três anos), foram mantidas na base apenas as empresas que possuíam dados, no mínimo, em 8 períodos em todas as variáveis (2/3 da série de cada variável). Para as empresas com ausência de dados em até 4 trimestres, equivalente a 1/3 da série em alguma variável, esses dados foram preenchidos da seguinte forma: ausência de dados de 2 a 4 períodos em alguma das variáveis – adotamos a média dos demais dados da série para preenchimento dos dados ausentes; ausência de dados em apenas um trimestre nos extremos da série – repetimos o dado vizinho; ausência de dados em apenas um trimestre no meio da série – repetimos o dado vizinho subsequente.

Considerando a base após o tratamento dos dados ausentes, para as 235 empresas que não entraram com o pedido de RJ ou RE no período considerado, colhemos os dados dos últimos 12 trimestres (dezembro/2018 a setembro/2021) e para as 44 empresas que entraram com pedido de RJ ou RE, utilizamos dados dos 12 trimestres anteriores ao trimestre em que foi oficializado o pedido de RJ ou RE pela empresa. A Tabela 1 demonstra as empresas que pediram recuperação.

[1] https://economatica.com/

Tabela 1 – **Empresas que pediram RJ ou RE de 2005 a 2021**

2005	2008	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	TOTAL
4	2	1	2	4	6	5	3	4	3	6	3	1	44

Fonte: Autoria Própria.

Quando uma empresa oficializa um pedido de RJ ou RE, está declarando que pode entrar em um estágio de instabilidade financeira em um período futuro próximo. De acordo com Kiseleva et al. (2019), o estágio de instabilidade financeira é definido pelo aparecimento de problemas no cumprimento das obrigações de curto prazo.

3.2 Variáveis

As variáveis estudadas compreendem um conjunto de indicadores financeiros extraídos das demonstrações contábeis das empresas e duas variáveis categorias como o setor econômico de atuação da empresa e o segmento de listagem na bolsa, que tem relação com o nível de governança adotado pela empresa. O negócio de uma empresa pode ser conhecido, visitando fábricas e entrevistando os administradores. Mas também podemos observar o negócio por meio de demonstrações financeiras, uma vez que estas são as lentes que aplicamos ao negócio (PENMAN, 2013).

Em nossa busca por estudos que tratam de previsão de insolvência corporativa, observamos que a maioria utiliza indicadores de balanço como variáveis explicativas. De acordo com Serrano-Cinca, Gutiérrez-Nieto e Bernate-Valbuena (2019) esses estudos pressupõem que as contas fornecem uma avaliação justa e realista da situação financeira de uma empresa. No nosso estudo acrescentamos, ainda, uma variável setorial (setor econômico de atuação das empresas) e uma variável de governança (segmento de listagem na Bolsa de Valores). A qualificação da nossa base está respaldada no fato de que as empresas de capital aberto passam por rigorosas exigências legais e por auditorias periódicas quanto à fidedignidade de seus números e indicadores.

3.2.1 Índices financeiros

Os indicadores financeiros usados como variáveis explicativas para a implementação dos modelos preditivos estão divididos em 5 grupos, conforme demonstra a Tabela 2.

Tabela 2 – **Lista de indicadores usados para prever eventos de dificuldades financeiras**

Classe	**Índice**	**Fórmula de cálculo**
1.Liquidez	Liquidez Geral (LG)	$Ativos/Passivos$
	Liquidez Corrente (LC)	$Ativo\ Circulante(AC)/Passivo\ Circulante(PC)$
	Liquidez Seca (LS)	$(Ativo\ Circulante(AC)$ - $Estoques)/Passivo\ Circulante$
2.Estrutura de capital	Dívida Bruta sobre o Ativo (DBA)	$(Divida\ Financeira\ CP + Dívida\ Financeira\ LP)/Ativo$
e endividamento	Dívida Bruta sobre o PL (DBPL)	$(Dívida\ Financeira\ CP + Dívida\ Financeira\ LP)/PL$
	Dívida Líquida sobre o PL (DLPL)	$(Dívida\ Financeira\ CP + Dívida\ Financeira\ LP - Caixa)/PL$
	Dívida de CP (DCP)	$PC/(PC + Passivo\ Não\ Circulante)$
	Endividamento Geral (EG)	$Exigível\ Total/Ativo$
	Participação Capital Terceiros (PCT)	$Exigível\ Total/PL$
	Imobilização do PL (IPL)	$Ativo\ Fixo/PL$
3.Rentabilidade	Giro do Ativo (GA)	$Receita\ Líquida/Ativo$
e desempenho	Margem Bruta (MB)	$Lucro\ Bruto/Receita\ Bruta$
	Margem EBIT	$EBIT/Receita\ Líquida$
	Margem Líquida (ML)	$Lucro\ Líquido/Receita\ Líquida$
	ROA (Return on Assets)	$Lucro\ Líquido/Ativo$
	ROE (Return on Equit)	$Lucro\ Líquido/PL$
	ROIC (Return on Invested Capital)	$NOPAT/capital\ investido$ (*próprio e de terceiros*)
4.Alavancagem	Grau de Alavancagem Financeira (GAF)	$Variação\ \%\ no\ LL/Variação\ \%\ no\ EBIT$
	Grau de Alavancagem Operacional (GAO)	$Variação\ \%\ no\ EBIT/Variação\ \%\ nas\ Vendas$
5.Atividade	Prazo Médio de Rotação dos Estoques (PMRE)	$(Estoque/Custo\ das\ Mercadorias\ Vendidas)x360$
e eficiência	Prazo Médio de Pagamento aos Fornecedores (PMPF)	$Fornecedores/Compras$ x 360
	Prazo Médio de Recebimento das Vendas (PMRV)	$(Contas\ a\ receber$ x 360$)/Receita\ líquida$
	Ciclo Operacional (CO)	$PMRE + PMRV$

Fonte: Autoria Própria.

Outros indicadores financeiros como *Dívida Bruta/Ebitda*, *Dívida Líquida/Ebitda*, *Margem Ebitda*, *Investimento/PL* não foram considerados no estudo, por problemas de ausência de dados. Os indicadores de atividade, com *Prazo Médio de Rotação de Estoques* (PME), *Prazo Médio de Pagamento a Fornecedores* (PMF), *Prazo Médio de Recebimento da Vendas* (PMR), *Ciclo Operacional* (CO) e *Ciclo Financeiro* (CF), pelo fato de serem medidos em prazo (dias), foram transformados em índices, da seguinte forma: foram todos divididos pelo *Ciclo Financeiro* (CF) e descartamos o CF no estudo. Desta forma a análise contou 23 indicadores financeiros, cada um com 12 trimestres, totalizando 276 variáveis financeiras.

3.2.2 Variáveis categóricas

Uma das variáveis categóricas utilizadas no estudo é referente aos setores de atividade econômica de atuação das empresas, os quais demonstramos na Tabela 3, distribuídos por classes de empresas que pediram RJ ou RE e que não pediram.

Tabela 3 – **Setores econômicos**

Setores	**Não pediram recuperação**	**Pediram recuperação**
Bens Industriais	42	9
Comunicações	4	1
Consumo cíclico	60	13
Consumo não cíclico	19	3
Financeiro	19	1
Materiais Básicos	19	8
Petróleo, gás e biocombustíveis	6	4
Saúde	19	1
Tecnolgia da informação	8	0
Utilidade pública	39	4

Fonte: Autoria Própria.

Uma outra variável categórica utilizada no estudo está relacionada ao nível de governança das empresas, representada pelos segmentos de listagem das empresas na bolsa, que estabelecem regras que vão além do que é definido por lei, de forma que, quanto mais elevado o nível de governança de uma empresa, teoricamente, maior confiança oferece para os investidores. Os níveis de governança foram ordenados de 1 a 5, conforme *menor nível* = 1 e o *maior nível* = 5 para facilitar o processamento dos modelos, os quis estão demonstrados na Tabela 4, por classes de empresas:

Tabela 4 – **Segmentos de listagem das empresas na Bolsa**

Segmento de listagem	**Não pediram recuperação**	**Pediram recuperação**	**Nível de governança**
Bovespa mais	14	1	1
Tradicional	62	27	2
Nível 1	17	3	3
Nível 2	12	2	4
Novo Mercado	130	11	5
TOTAL	**235**	**44**	

Fonte: Autoria Própria.

Diante do exposto, nossa estrutura de dados tem como variável dependente uma *dummy*, em que $Y = 1$ para as empresas que pediram RJ ou RE no trimestre seguinte ($Trimestre\ T$) e $Y = 0$ para empresas que não pediram recuperação no trimestre seguinte ($Trimestre\ T$). No caso de $Y = 1$, o $Trimestre\ T$ pode ser qualquer trimestre entre 2005 e 2021, dependendo da data que a empresa pediu recuperação e, quando $Y = 0$, o $Trimestre\ T$ é o último trimestre de 2021. Como variáveis independentes ou explicativas, temos 23 indicadores de balanços, cada indicador com 12 trimestres defasados. Considerando que o momento T é a situação que queremos prever, as minhas varáveis explicativas referem-se as 12 trimestres anteriores, de $T-1$ a $T-12$. Temos, ainda, como variáveis independentes,

o segmento de listagem e o setor de atuação das empresas. A Equação 3.1 a seguir ilustra essa estrutura.

$$\mathrm{Y} = [\mathrm{X}_{1(T-12)}, X_{1(T-11)}, ..., X_{1(T-1)}]; [X_{2(T-12)}, X_{2(T-11)}, ..., X_{2(T-1)}]; \textbf{.....}; [X_{23(T-12)}, X_{23(T-11)}, ..., X_{23(T-1)}]; [\mathrm{SegmentodeListagem}]; [\mathrm{SetorEconômico}] \quad (3.1)$$

Onde:

$Y = 1$, se a empresa pediu RJ ou RE no trimestre T;

$Y = 0$, se a empresa não pediu RJ ou RE no trimestre T.

X_i = indicadores Financeiros de balanços nos trimestres $T-1$ a $T-12$;

$i = 1, 2, 3, 4, ..., 23$.

4 Metodologia

Nesta seção descrevemos de forma mais detalhada a metodologia adotada para implementação dos 6 modelos descritos na seção 2, desde o pré-processamento dos dados, seleção das variáveis, técnicas utilizadas para lidar com o viés por desbalanceamento das amostras, processamento dos modelos, métodos da avaliação, comparação e otimização.

4.1 Pré-processamento dos dados

No processamento dos modelos utilizamos a linguagem de programação *python*, onde carregamos os nossos dados. Trabalhamos com as 24 variáveis numéricas, formadas por 23 índices financeiros de balanços e uma variável referente ao segmento de listagem na Bolsa, que foi ordenada para facilitar a execução dos modelos, conforme demonstramos no Capítulo 3, Tabela 4. A variável categórica referente aos 10 setores de atuação das empresas foi convertida em vetores *one-hot* para que o computador a entenda corretamente, ou seja, foi transformada em *dummies*. Temos, portanto, 287 variáveis explicativas.

4.1.1 *Feature selection*

Diversas técnicas de *feature selection* foram propostas na literatura em estudos envolvendo modelos preditivos com *Machine Learning*. A ideia principal da *feature selection* é escolher um subconjunto de variáveis de entrada eliminando recursos com pouca ou nenhuma informação preditiva (WANG; MA; YANG, 2014). No que diz respeito às diferentes estratégias de *feature selection*, os métodos podem ser amplamente categrorizados como *filter* (filtro), *wrapper* (embrulho) e *embedded* (embutido) (LI et al., 2017).

Os métodos de seleção de recursos baseados em filtros usam medidas estatísticas - como chi-quadrado, correlação, *Analysis Of Variance (ANOVA)* - para pontuar a correlação ou dependência entre as variáveis de entrada que podem ser filtradas para escolher os recursos mais relevantes, tal como o *SelectKBest* da *Sklearn*. Já os métodos baseados em *wrapper* (embrulho) são criados vários modelos com diversas variações diferentes e são escolhidos os que resultam na melhor performance para o modelo, de acordo com as medidas de avaliação, tal como *Recursive Feature Elimination (RFE)*. O método embutido utiliza algoritmos que realizam a seleção automática de recursos durante o treinamento, tal como *Decision Tree*, *RF*, *GB*, entres outros.

Neste estudo, aplicamos o método filtro, ainda na fase de pré porcessamento, com a utlização do *SelectKBest* com filtro *ANOVA*, considerando que temos um modelo de classificação, com entradas preponderantemente numéricas e saída categórica, conforme

resultados demonstrados no Capítulo 5, Figura 2. Na prática, o *SelectKBest* recebe dois parâmetros: *Score func* e k. Ao definir k, estamos simplesmente dizendo ao método para selecionar apenas o melhor número k de recursos e devolvê-los. Se k é definido com n recursos, por exemplo, vai retornar n recursos. O *Score func* é o parâmetro que seleciona o método estatístico,

ANOVA é um método estatístico para verificar se existe diferenças significativas entre as médias de grupos de dados, sendo possível inferir se as variáveis são dependentes uma sobre a outra (BRUCE; BRUCE, 2019). Essa metodologia mede o *F-Value* que é a relação entre a variância entre grupos dividido pela variância dentro dos grupos. [1] Ou seja, quanto maior a variância entre os grupos, mais diferentes as duas variáveis serão. Então, quanto maior o valor de F maior é a evidência de que as variáveis são diferentes entre si e que exercem influência uma sobre a outra, sendo um bom indicativo para selecionar as variáveis mais importantes do modelo.

$$F = \frac{Variânciaentreosgrupos}{Variânciadentrodosgrupos} \tag{4.1}$$

Após o processamento, validação e otimização dos modelos, aplicamos o método embutido denominado *Feature Importance*, A *Feature Importance* é uma classe embutida que vem com classificadores baseados em árvore. O resultado desse processo está demonstrado no Capítulo 5, Figura 8.

4.1.2 Técnicas utilizadas para balanceamento das amostras

A base de dados usada neste estudo contém 279 empresas, dividida em dois conjuntos de dados, um conjunto com 235 empresas que não declararam dificuldades financeira, e um outro conjunto com 44 empresas que declararam dificuldades financeiras ao oficializarem o pedido de RJ ou RE. Este último conjunto de dados representa 15,77% do total. Isto ocorre porque no mundo real, a quantidade de empresas que apresentam dificuldades financeiras é bem menor do que a quantidade de empresas que não passam por dificuldades. A questão do desbalanceamento das amostras é uma das limitações para previsão de dificuldades financeiras de empresas comuns a todos os autores.

Existem, basicamente, duas técnicas que são mais utilizadas para lidar com o problema de viés por desbalanceamento da amostra: *Undersampling* e *Oversampling*. A técnica de *Undersampling* (subamostragem) visa remover as amostras de classe majoritária, até que a amostra majoritária se iguale, em número, às amostras da classe minoritária. Os dois tipos principais de *Undersampling* são: *random Undersampling*, em que a escolha dos dados da classe majoritária é feita de forma aleatória e; *NearMiss*, onde são eliminadas as amostras "mais próximas"(ou que menos diferem) entre si, de modo a maximizar a

[1] https://medium.com/data

variabilidade entre as classes e, também, entre os dados que compõem a própria classe (majoritária). A técnica de *Oversampling* (sobreamostragem), de forma resumida, visa adicionar novas amostras (sintéticas) na classe minoritária, até que se iguale ao número de amostras da classe majoritária. Os dois tipos principais de oversampling são: *Random Oversampling*, em que os dados da classe minoritária são, aleatoriamente, replicados e; *SMOTE (Synthetic Minority Over-sampling Technique)*, onde novas amostras sintéticas são geradas a partir de amostras vizinhas.

Choi, Son e Kim (2018) se deparam com dados desbalanceados ao propor modelo para prever dificuldades financeiras na indústria de construção antes da contratação e utilizam técnica de sobreamostragem de minoria sintética *(SMOTE)* para resolver o problema de desequilíbrio entre o número de contratos normais e o número de empreiteiros em dificuldades financeiras. Contudo, neste estudo, optamos por testar as duas métricas *(SMOTE e NearMiss)* para possibilitar comparação.

4.2 Processamento dos modelos

Inicialmente, processamos os nossos modelos com os dados originais (279 empresas) divididos aleatoriamente em duas partes: amostra de treino com 70% (193 empresas) e amostra de teste com 30% (83 empresas). Os dados de treinamento são usados para construir os modelos de aprendizagem, enquanto os dados de testes são usados para testar a capacidade preditiva dos modelos. Porém, segundo Kim, Jo e Shin (2016), a realização de tarefas de classificação usando dados desequilibrados deteriora o desempenho da classificação. Por isso, em seguida, processamos os modelos novamente com as amostras balanceadas, para comparar os resultados. Depois, com o objetivo de melhorar o desempenho dos modelos, aplicamos validação cruzada em todos os modelos e a otimização com hiperparâmetros para os modelos que obtiveram melhor performance na validação cruzada e, por fim, avaliamos e comparamos os resultados.

4.3 Métricas de avaliação

Considerando a proposta do nosso estudo, selecionamos várias métricas de avaliação e seleção do melhor modelo, as quais são frequentemente utilizadas nos estudos precedentes, tais como *Accuracy*, Curva *ROC*, *Precision*, *Recall*, *F1-score* e Matriz de confusão. Cada métrica tem suas peculiaridades que devem ser levadas em consideração na escolha de como o modelo de classificação será avaliado. Não se deve pensar em uma métrica como melhor ou pior que a outra de maneira geral, e sim deve-se analisar o problema e escolher a(s) que melhor se adapta(m). De acordo com BoneLLo, BrÉdart e VeLLa (2018), as técnicas de *Machine Learning* precisam ser avaliadas empiricamente porque seu desempenho depende muito do conjunto de dados de treinamento.

4.3.1 *Accuracy*

A *Accuracy* é uma boa indicação geral de como o modelo performou, ou seja, indica dentre todas as classificações, quantas o modelo classificou corretamente, embora não seja considerada uma boa métrica para problemas de classificação com dados desbalanceados. Há situações em que ela é enganosa, por exemplo, se a categoria alvo representa apenas 10% do total da amostra, um modelo simplório mesmo errando todos os alvos, ainda assim, acerta 90%. No caso do nosso estudo, por exemplo, considerando as amostras originais, em que a classe minoritária, de empresas que se declararam em dificuldades financeiras, representa 15,77% do total, então o modelo mesmo considerando todas as empresas como boas, já atinge uma acurácia de 84,33%. Porém, preferimos manter essa métrica para fazer comparações com o modelo após o balanceamento das amostras e com os resultados obtidos após a aplicação das técnicas de otimização.

4.3.2 *Precision*, *Recall* e *F1-Score*

A *Precision* calcula dentre todas as classificações de classe positiva que o modelo fez, quantas estão corretas; o *Recall* indica, dentre todas as situações de classe positiva como valor esperado, quantas estão corretas e; o *F1-Score* representa a média harmônica entre *Precision* e *Recall*. A *Precision* é preferencialmente usada em uma situação em que os Falsos Positivos são considerados mais prejudiciais que os Falsos Negativos, já *Recall* é mais usado em situações em que os Falsos Negativos são considerados mais prejudiciais que os Falsos Positivos, que é o nosso caso. Por exemplo, ao classificar uma empresa como boa, como um bom investimento ou como boa pagadora, é necessário que o modelo esteja correto, mesmo que acabe classificando boas empresas como empresas ruins (situação de Falso Positivo). Ou seja, o modelo deve ser preciso em suas classificações, pois a partir do momento que consideramos um investimento bom quando na verdade ele não é, uma grande perda de dinheiro pode acontecer.

O *F1-Score* é uma maneira de observar somente uma métrica ao invés de duas (*Precision* e *F1-Score*). É uma média harmônica entre as duas, que está muito mais próxima dos menores valores do que uma média aritmética simples. Ou seja, quando se tem *F1-Score* baixo, é um indicativo de que ou a *Precision* ou o *Recall* está baixo. O *F1-Score* é bastante utilizado quando se tem classes desbalanceadas.

4.3.3 Curva *ROC*

A Curva *ROC* (*Receiver Operating Character*) ou curvatura das características operacionais do receptor, é uma ferramenta bastante comum para avaliar modelos de classificação. A curva *ROC* mostra a relação entre a taxa de verdadeiros positivos (*true positive rate - TPR*) e a taxa de falsos positivos (*false positive rate - FPR*). É, portanto,

representado por um gráfico que resume o desempenho de um modelo de classificação binário em relação à classe positiva (previsão, pelo modelo, de ocorrência do evento de interesse). O eixo x indica a taxa de falsos positivos (modelo indicou que o evento ocorreria, mas não ocorreu) e o eixo y indica a taxa de verdadeiros-positivos (modelo indicou que o evento ocorreria, e ocorreu).

O desempenho geral de um classificador, resumido em todos os possíveis limiares, é dado pela área sob a curva *ROC* (*Area Under the Curve - AUC*). Portanto, uma curva *ROC* ideal se aproxima do canto superior esquerdo e quanto maior a *AUC*, melhor o classificador, ou seja, um classificador perfeito tem $AUC = 1$ e um classificador aleatório tem $AUC = 0,5$.

4.3.4 Matriz de confusão

A Matriz de Confusão é uma medida de desempenho para problemas de classificação, representada por uma tabela que indica os erros e acertos do modelo, comparando com o resultado esperado. A Curva *ROC* e sua área (*ROC AUC*), assim como a *Accuracy*, a *Precision* o *Recall* e o *F1-Score*, são uma métrica baseada em conceitos da matriz de confusão binária. A Figura 1 demonstra um exemplo de uma matriz de confusão.

Figura 1 – **Demonstração da Matriz de confusão**

Real	**Previsto**	
	Negativo (0)	**Positivo (1)**
Negativo (0)	Verdadeiro Negativo - VN (classificação correta da classe Negativo)	Falso Positivo - FP (Erro Tipo I - erro em que o modelo previu a classe Positivo quando o valor real era classe Negativo)
Positivo (1)	Falso Negativo - FN (Erro Tipo II - erro em que o modelo previu a classe Negativo quando o valor real era classe Positivo)	Verdadeiro Positivo - VP (classificação correta da classe Positivo)

Fonte: Autoria Própria.

Ao contarmos todos esses termos e obtermos a matriz de confusão, calculamos as métricas de avaliação para a classificação, conforme segue.

$$Accuracy = \frac{VP + VN}{VP + VN + FP + FN} \tag{4.2}$$

$$Precision = \frac{VP}{VP + FP} \tag{4.3}$$

$$Recall = \frac{VP}{VP + FN} \tag{4.4}$$

$$F1 - score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{4.5}$$

4.4 Otimização dos modelos

No processamento dos modelos, há a possibilidade de *overfitting* no treinamento, que ocorre quando o modelo treinado fica super ajustado aos dados de treinamento, porém com capacidade preditiva significantemente inferior quando aplicado em novos conjuntos de dados. Com o processo de validação cruzada (*cross validation*) e com a utilização das técnicas de otimização de hiperparâmetros é possível reduzir esse risco.

4.4.1 Validação Cruzada

A validação cruzada tem a capacidade de generalização de um modelo, a partir de um conjunto de dados. É amplamente empregada em problemas cujo objetivo da modelagem é a predição. Busca-se, então, estimar o quão preciso é o modelo na prática, ou seja, o seu desempenho para um novo conjunto de dados consiste no particionamento do conjunto de dados em subconjuntos mutuamente exclusivos, e posteriormente, o uso de alguns destes subconjuntos para a estimação dos parâmetros do modelo (dados de treinamento), sendo os subconjuntos restantes (dados de validação ou de teste) empregados na validação do modelo. Os métodos mais utilizados para realizar o particionamento dos dados são os seguintes: *holdout*, o *k-fold* e o *leave-one-out*. Neste estudo utilizamos o método *k-fold*.

Na validação cruzada utilizando *k-fold* divide-se aleatoriamente os dados em k subconjuntos, treina o modelo com $k - 1$ e usa a parte restante dos dados como conjunto de testes para validar o modelo, momento em que a precisão ou o erro do teste do modelo é medido. Esse processo é repetido k vezes. No nosso estudo utilizamos a base balanceada com a técnica *SMOTE* e aplicamos o *cross validation* com $k = 5$.

4.4.2 Otimização de hiperparâmetros

A utilização do método de validação cruzada nos permite testar várias configurações de hiperparâmetros para estimar a melhor generalização para conjuntos de dados independentes. Esses parâmetros foram ajustados com o objetivo de otimizar o desempenho dos modelos. As técnicas de otimozação de hiperparâmetros são, basicamente, *Grid Search* (busca em grade) e *Random Search* (busca aleatória). O *Grid Search* é um algoritmo de busca que recebe um conjunto de valores de um ou mais hiperparâmetros e testa todas as combinações dentro dessa vizinhança. O algoritmo tabela qual foi o desempenho de cada configuração e ao final de todos os testes, fala qual é a melhor escolha. O *Random Search*, embora semelhante, ao *Grid Search*, ao invés de testar todas as combinações na

vizinhança, faz uma busca aleatória e testa combinações aleatórias de hiperparâmetros. Optamos pelo *Random Search* por ser considerado superior ao *Grid Search*.

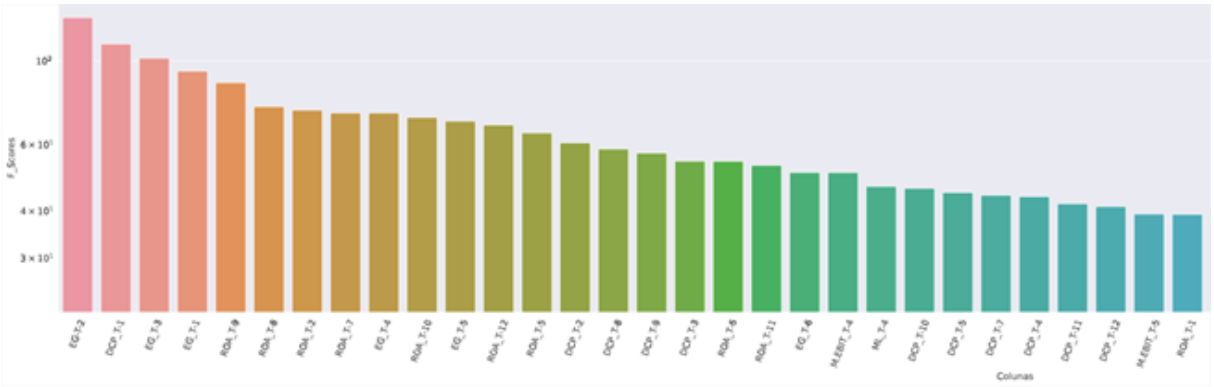
5 Análise dos resultados

Após aplicarmos modelos de *Machine Learning* com o objetivo de prever antecipadamente situação de dificuldade financeira de uma empresa, que possa levá-la a pedir recuperação judicial ou extrajudicial em um período futuro próximo, torna-se necessário fazermos uma avaliação dos resultados obtidos. Para isso, três etapas são essenciais: comparar, avaliar e selecionar o melhor modelo. Descrevemos nesta seção os achados encontrados, considerando as diferentes abordagens utilizadas, bem como a nossa percepção sobre a utilidade prática dos resultados.

Na fase de pré-processamento dos modelos, fizemos uma análise da importância das variáveis, 223 indicadores financeiros, uma variável de governança e 10 variáveis *dummies* referentes ao setor econômico das 279 empresas, aplicamos o método de filtro, conforme descrito na subseção 4.1.1 e utilizamos a classe *SelectKBest* da biblioteca *feature selection* combinada com o método de máximo coeficiente de informações para selecionar recursos.

o *SelectKBest* é um método de seleção de *features* através de *univariate statistical test*, Trata-se um método bem simples, no qual podemos selecionar apenas as K maiores *features* do nosso *dataset* com base em um teste estatístico. A Figura 2 demonstra as 30 variáveis mais relevantes, de acordo com essa técnica.

Figura 2 – **As 30 variáveis mais importantes - *SelectKBest***



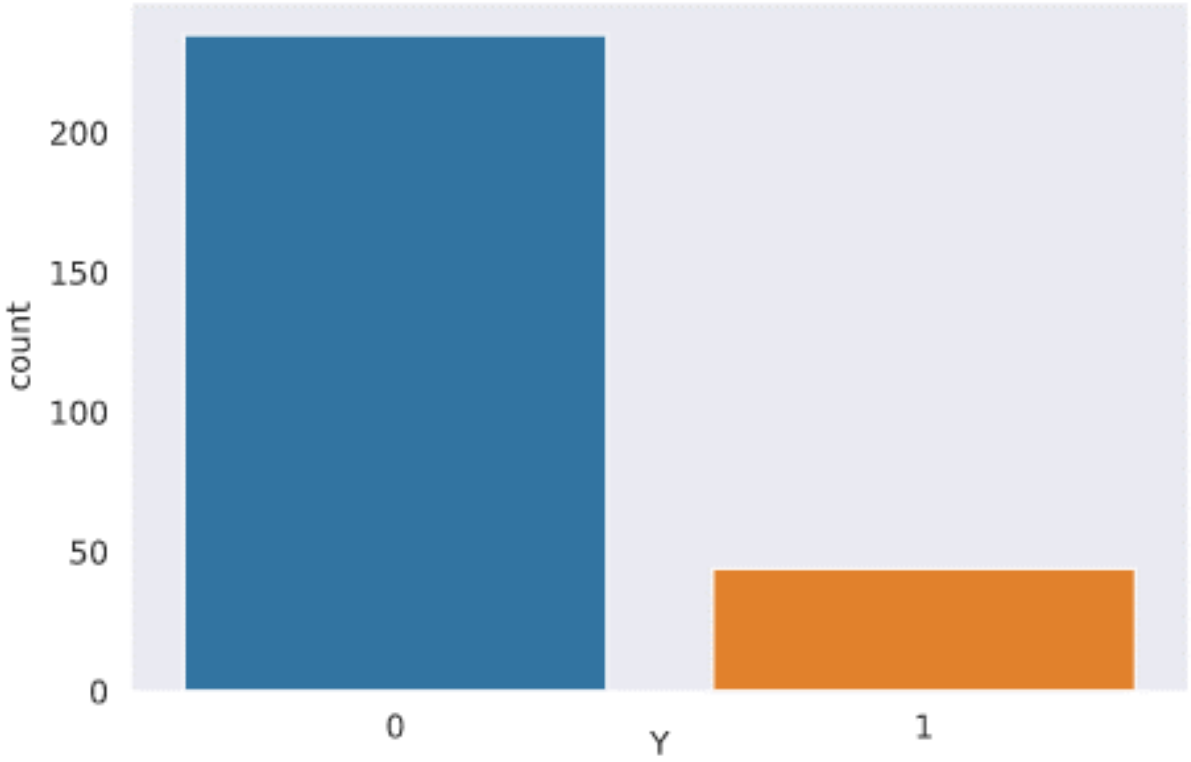
Fonte: Autoria própria.

Considerando essa metodologia, entre as 30 variáveis mais importantes para determinar se uma empresa está ou não em dificuldades financeiras, destacam-se os indicadores de endividamento e de rentabilidade, em especial, os demonstrados no gráfico acima, tais como Endividamento Geral (EG), Dívida de Curto Prazo (DCP), Retorno sobre o Ativo (ROA), Margem EBIT e Margem Líquida (ML). Ao apliarmos a quantidade variáveis para construção do gráfico, observamos que a variável nível de governança é exibida na posição 83º, e as variáveis *dummies*, referentes aos setores de atuação das empresas, começam a

ser selecionadas a partir da 144º posição. Optamos por excluir as variáveis dos setores econômicos na implementação dos modelos, dada a pouca relevância dessas variáveis.

Implementamos os modelos, inicialmente, com a base original, ou seja, em uma base desbalanceadas, contendo 279 empresas, das quais 235 não declararam dificuldades financeiras, e 44 pediram RJ ou RE, conforme demonstra a Figura 3.

Figura 3 – **Base original - antes do balanceamento**



Fonte: Autoria própria.

A base foi dividida em duas amostras: treino (70% = 195 empresas) e teste (30% = 84 empresas). Os resultados obtidos estão demonstrados na Tabela 5.

Tabela 5 – Resultado dos modelos com a base original

Models	**Accuracy**	**AUC**	**F1**	**Recall**	**Precision**	**Error**
GB	96.43	94.75	88.89	92.31	85.71	0.04
RF	95.24	90.90	84.62	84.62	84.62	0.05
NB	91.67	85.64	74.07	76.92	71.43	0.08
LR	90.48	75.51	63.64	53.85	77.78	0.10
SVM	90.48	75.51	63.64	53.85	77.78	0.10
ANN	84.52	50.00	0	0	0	0.15

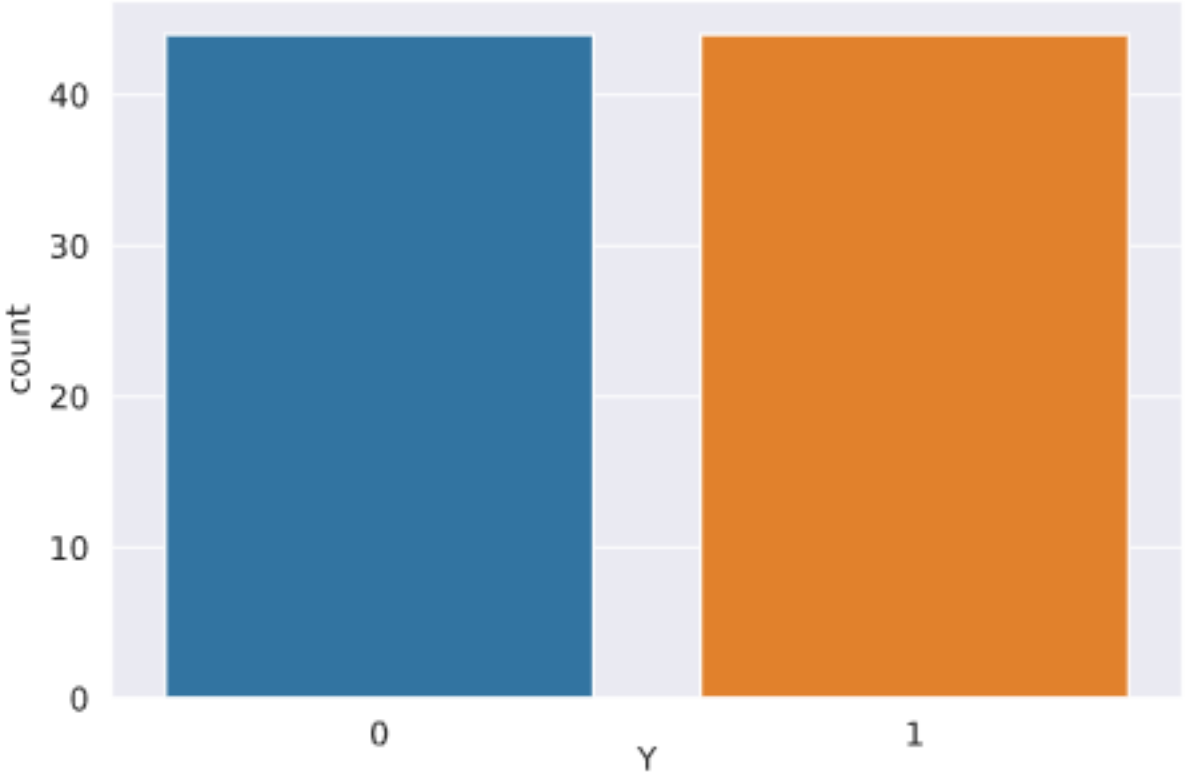
Fonte: Autoria Própria.

Com a utilização da base original, os resultados mostram que os modelos aplicados, ranqueados pela métrica *F1 score*, obtiveram performances na seguinte ordem: 1. *GB*, 2.

RF, 3. *NB*, 4. *LR* e *SVM*, 5. *ANN*. Contudo, Kim, Jo e Shin (2016) afirmam que, se a diferença do tamanho dos dados entre as duas classes for maior, a maioria dos dados será classificada predominantemente como a classe majoritária. Como as nossas amostras se enquadram nessa situação, é possível que os modelos sobrecarreguem os casos de empresas que não pediram recuperação e ignore os casos de empresa que pediram recuperação judicial. Por isso, além de implementarmos os modelos com a base original, utilizamos duas técnicas de balanceamento: *NearMiss* e *SMOTE* para possibilitar a comparação.

Aplicamos a técnica *NearMiss* para diminuição da majoritária, até se igualar com a base minoritária, de forma que ficaram ambas com 44 empresas normais e 44 empresas que pediram RJ ou RE, conforme demonstra a Figura 4, cujos resultados dos modelos estão na Tabela 6.

Figura 4 – **Base balanceada com *NearMiss***



Fonte: Autoria própria.

Tabela 6 – Resultado dos modelos com a base balanceada - *NearMiss*

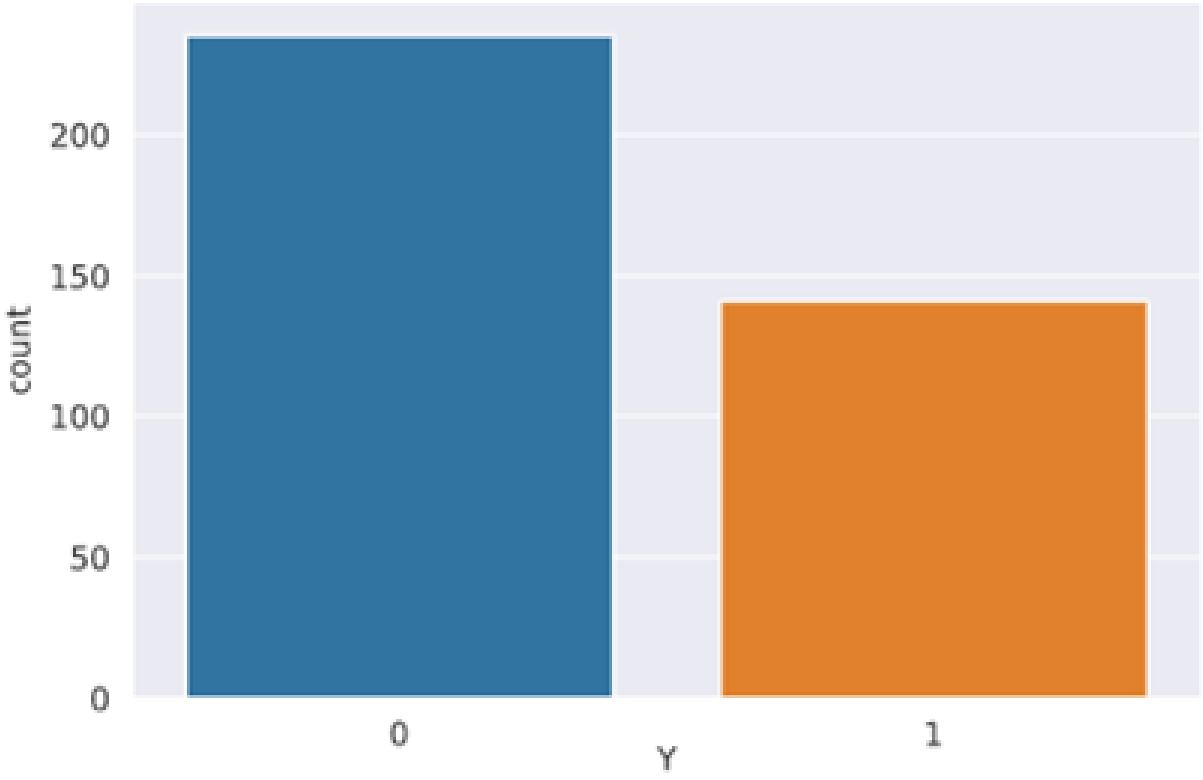
Models	***Accuracy***	***AUC***	***F1***	***Recall***	***Precision***	***Error***
RF	96.30	96.15	96.00	92.31	100.00	0.04
GB	88.89	88.74	88.00	84.62	91.67	0.11
SVM	85,19	84.62	81.82	69.23	100.00	0.15
NB	81,48	81.32	80.00	76.92	83.33	0.19
LR	81,48	80.77	76.19	61.54	100.00	0.19
ANN	62.96	61.54	37.50	23.08	100.0	0.37

Fonte: Autoria Própria.

Com a utilização da base balanceada com *NearMess*, os resultados mostram que os modelos aplicados obtiveram performances na seguinte ordem, considerando o *F1-score*: 1. *RF*, 2. *GB*, 3. *SVM*, 4. *NB*, 5. *LR*, 6. *NNA*.

Aplicamos a técnica *SMOTE* na base minoritária, com a ampliação da base minoritária em até 60% da base majoritária, considerado que no mundo real essa proporção é sempre menor para essas bases, de forma que ficamos com 235 empresas normais (amostra original) e 141 empresas que pediram recuperação (amostra balanceada), conforme demonstra a Figura 5.

Figura 5 – **Base balanceada com *SMOTE***



Fonte: Autoria própria.

Após o balanceamento da base, ficamos com um total de 376 empresas. Dividimos

novamente em base em treino (70% = 263) e teste (30% = 113). Os resultados, com a aplicação dos modelos na base balanceada com *SMOTE*, constam na Tabela 7.

Tabela 7 – Resultado dos modelos com a base balanceada - *SMOTE*

Models	***Accuracy***	***AUC***	***F1***	***Recall***	***Precision***	***Error***
GB	97.35	97.42	96.70	97.78	95.65	0.03
RF	95.58	95.95	94.62	97.78	91.67	0.04
LR	94.69	94.46	93.33	93.33	93.33	0.05
SVM	94.69	94.46	93.33	93.33	93.33	0.05
NB	93.81	93.73	92.31	93.33	91.30	0.06
ANN	65.49	57.42	29.09	17.78	80.00	0.35

Fonte: Autoria Própria.

A Tabela 7 mostra que os resultados dos modelos obtiveram performances na seguinte ordem, considerando a métrica *F1 Score*: 1. *GB*; 2. *RF*; 3. *LR* e *SVM*; 4. *NB* e; 5. *ANN*. Destacamos que o *GB* e o *RF* obtiveram o mesmo *reccal*.

Considerando os resultados obtidos até então, observamos que os modelos *GB* e *RF* apresentaram melhores performances em todas as simulações. Contudo, para conseguirmos melhor ajuste, aplicamos a seguir a técnica de *Cross validation* em todos os modelos. Utilizamos par esse processo a base balanceada pela técnica *SMOTE*, contendo 376 empresas, dividimos novamente em base em treino (70% = 263) e teste (30% = 113), aplicamos a validação cruzada com *n-folds* = 5 e os resultados estão demonstrados na Tabela 8.

Tabela 8 – Resultado dos modelos - base balanceada *SMOTE* com *cross-validation*

Models	***Accuracy***	***AUC***	***F1***	***Recall***	***Precision***
RF	97.32	98.89	96.46	98.95	94.14
GB	94.27	96.10	92.70	96.95	89.24
LR	92.92	92.24	90.91	89.00	93.00
NB	85.93	88,91	82.37	89.63	76.25
ANN	72,24	63,77	44,20	30.26	84,64
SVM	70.69	85.20	41.20	29.26	76.60

Fonte: Autoria Própria.

Conforme demonstra a Tabela 8, os resultados dos modelos, ranqueados pela métrica *F1 score*, são os seguintes: 1. *RF*, 2. *GB*, 3. *LR*, 4. *NB*, 5. *ANN* e 6. *SVM*. Após essa etapa, temos que modelo *Random Forest* obteve a melhor performance. Esse resultado corrobora com o estudo de (BARBOZA; KIMURA; ALTMAN, 2017), mesmo utilizando estrutura de dados diferente, observaram que o modelo *RF* apresenta melhor desempenho de previsão. Uma síntese dos resultados dos modelos nas quatro simulações, considerando somente a métrica *F1 score* está demonstrada na Tabela 9.

Tabela 9 – Resultado comparativo dos modelos em cada simulação (*F1 score*)

Models	**Base original**	***Near Miss***	***SMOTE***	***Cross Validation***
RF	84.62	96.00	94.62	96.46
GB	88.89	88.00	96.70	92.70
LR	63.64	76.19	93.33	90.91
NB	74.07	80.00	92.31	82.37
ANN	0	37.50	29.09	44.20
SVM	63.64	81.82	93.33	41.20

Fonte: Autoria Própria.

Já conseguimos um bom resultado com o *RF*, após o processo de *cross-validation*. Porém, o *RF*, como acontece com muitos algoritmos de aprendizado de máquina estatístico, pode ser considerada um algoritmo de caixa preta com botões para ajustar o funcionamento da caixa (BRUCE; BRUCE, 2019). Esses botões são chamados de hiperparâmetros, que são parâmetros que precisamos definir antes de ajustar um modelo, pois esses parâmetros não são otimizados como parte do processo de treinamento. Para tratar essa questão aplicamos a técnica *Random Search* para determinar os melhores hiperparâmetros de forma a maximizar os resultados e conseguir um melhor desempenho, cujos resultados estão demonstrados na Tabela 10.

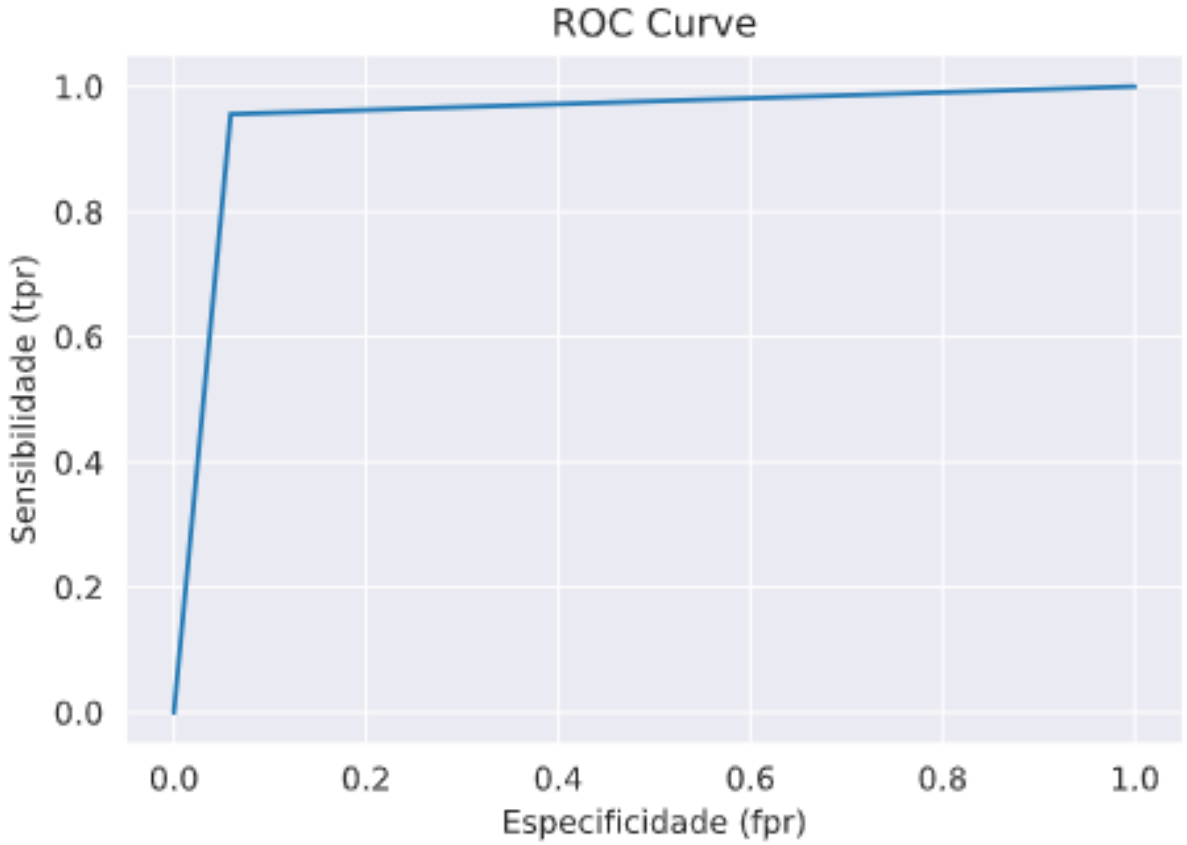
Tabela 10 – Resultado do *Random Forest* com *SMOTE*, *Cross-Validation* e hiperparâmetros

Models	***Accuracy***	***AUC***	***F1***	***Recall***	***Precision***	***Error***
RF	94.69	94.84	93,48	0.96	0.91	0.05

Fonte: Autoria Própria.

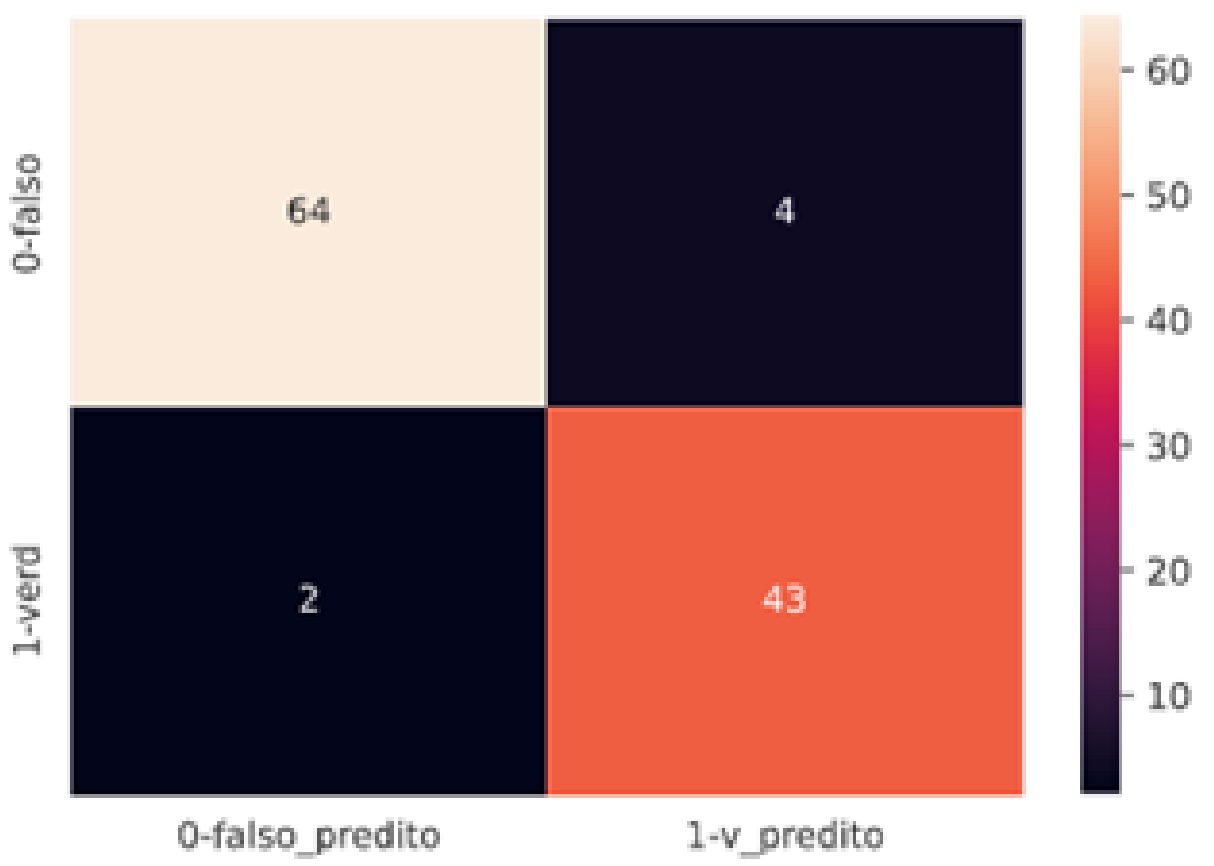
Entendemos que este é o melhor resultado para a nossa pesquisa, para o qual plotamos a Curva *ROC* (Figura 6) e a Matriz de Confusão (Figura 7), conforme figuras a seguir.

Figura 6 – **Curva *ROC RF***



Fonte: Autoria própria.

Figura 7 – **Matriz de confusão *RF***

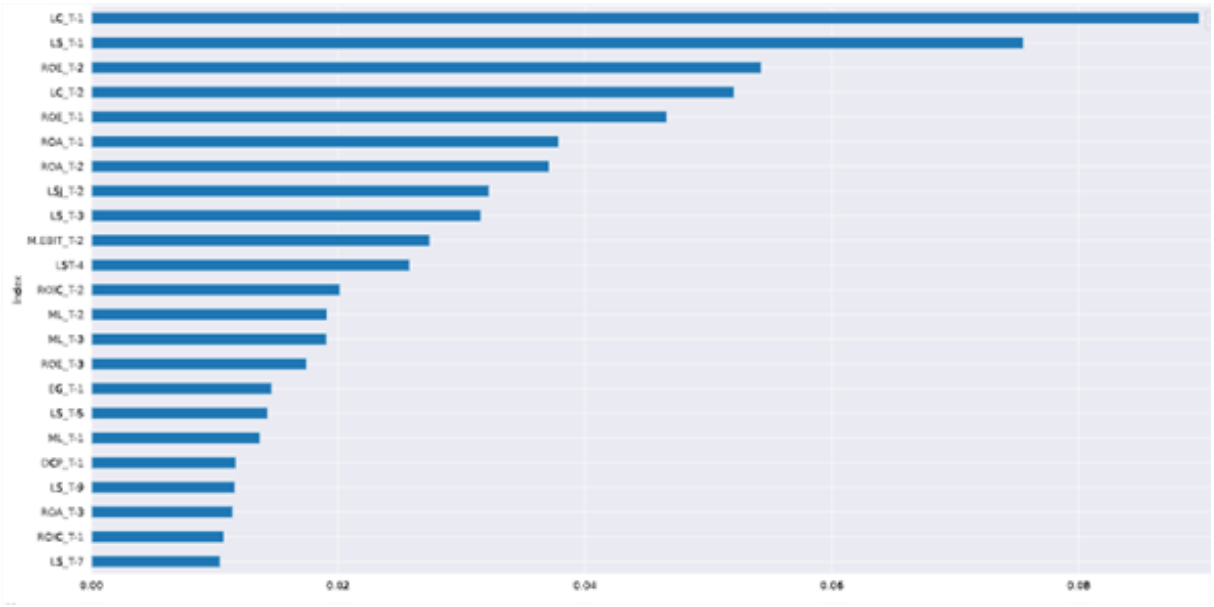


Fonte: Autoria própria.

A matriz de confusão demonstra que o modelo errou 6 empresas, duas empresas que deveriam ser classificadas como se estivesse em dificuldades financeiras, foram classificadas como normais (*False Negative*) e 4 empresas que deveriam ser classificadas como normais, foram classificadas como se estivessem com dificuldades financeiras (*False Positive*). As empresas com dificuldades financeiras podem vir a pedir RJ ou RE no trimestre seguinte.

Com base neste resultado, utilizamos o atributo *feature importance*, conforme explicado no Capítulo 4, Subseção 4.1.1, para identificar as variáveis mais importantes para a previsão, com base no modelo *RF*, cujas variáveis estão demonstradas no gráfico da Figura 8.

Figura 8 – **Variáveis mais importantes para prever dificuldade financeira das empresas (*RF*)**



Fonte: Autoria própria.

Podemos observar pela Tabela 11, onde estão transcritas as variáveis indicadas no gráfico da Figura 8, que as variáveis mais importantes para prever dificuldades financeiras das empresas estão nos grupos de liquidez, rentabilidade e endividamento. As variáveis relevantes identificadas no nosso estudo são, predominantemente, referentes aos três trimestres anteriores ao pedido de RJ ou RE das empresas ($T-1$, $T-2$ e $T-3$). Mas chama a nossa atenção também o fato da Liquidez Seca (LS) ter sido selecionada nos 5 trimestres antecedentes, no sétimo e no nono trimestres, o que pode possibilitar, com base nessa variável, a identificação da deterioração das condições financeiras das empresas com maior antecedência.

Tabela 11 – Resumo das variáveis mais importantes para oo modelo *RF*

Liquidez	**Rentabilidade e Retorno**	**Endividamento**
LS	ML	EG
LC	ROA	DCP
	ROE	
	ROIC	
	Margem EBIT	

Fonte: Autoria Própria.

Os nossos achados, quanto à importância das variáveis, corroboram com estudos precedentes, por exemplo Geng, Bose e Chen (2015), descobrem que indicadores financeiros, como Margem Líquida (ML), Retorno sobre o ativo (ROA), Fluxo de Caixa (FC), desempenham um papel importante na previsão de deterioração da lucratividade. Scalzer et al. (2019) verificam que o Retorno sobre o Ativo (ROA), a liquidez Imediata (LI) e a Liquidez corrente (LC) se destacam em seu poder de prever dificuldades financeiras das empresas de energia no Brasil. Crespí-Cladera, Martín-Oliver e Pascual-Fuster (2021), em estudo sobre o setor de hospitalidade durante o desastre do Covid-19, constatam que as empresas que tinham níveis mais baixos de ativos líquidos e maior alavancagem eram mais propensas a falir durante a Grande Recessão.

Ressaltamos que a Liquidez Seca (LS) está diretamente relacionada com o fluxo de caixa das empresas e que, embora não tenhamos utilizado a liquidez Imediata (LI) em nosso estudo, destacamos que esse indicador tem relação direta com a Liquidez Seca (LS), com pequenas alterações na sua fórmula de cálculo.

Considerando as variáveis identificadas como mais importantes para o nosso modelo e a quantidade de erros demonstrados pela matriz de confusão, num total de 6 empresas que foram classificadas incorretamente, podemos fazer uma análise dos indicadores dessas empresas, demonstrados nas seguintes tabelas:

Tabela 12 – Índices de liquidez seca das empresas classificadas incorretamente pelo modelo *RF*

Empresa	LS_{T-9}	LS_{T-7}	LS_{T-5}	LS_{T-4}	LS_{T-3}	LS_{T-2}	LS_{T-1}	Y	YPred.
357	0,85	0,87	0,79	0,92	0,90	1,01	1,00	1	0
76	0,37	0,57	0,59	0,57	0,50	0,72	0,72	0	1
113	0,10	0,13	0,18	0,22	0,39	0,37	0,34	0	1
16	2,34	0,69	0,41	0,40	0,80	0,32	0,33	0	1
93	1,13	0,81	0,69	0,76	0,61	0,53	0,57	1	0
163	0,55	0,44	0,62	0,56	064	0,61	0,52	0	1

Fonte: Autoria Própria.

Tabela 13 – Índices de liquidez corrente e de endividamento das empresas classificadas incorretamente pelo modelo *RF*

Empresa	LC_{T-2}	LC_{T-1}	DCP_{T-1}	EG_{T-1}	Y	YPred.
357	1,24	1,21	49,14	87,53	1	0
76	0,72	0,72	26,08	91,72	0	1
113	0,51	0,59	20,80	61,11	0	1
16	0,32	0,33	64,43	72,09	0	1
93	1,02	0,97	47,95	103,19	1	0
163	0,68	0,58	64,60	143,83	0	1

Fonte: Autoria Própria.

Tabela 14 – Índices de lucratividade das empresas classificadas incorretamente pelo modelo *RF*

Empresa	ML_{T-3}	ML_{T-2}	ML_{T-1}	$M.EBIT_{T-2}$	Y	YPred.
357	-2,10	-1,31	-4,08	7,14	1	0
76	-11,79	-45,27	-40,90	-36,67	0	1
113	12,46	38,92	84,26	44,50	0	1
16	-18,53	-19,48	-16,96	-5,47	0	1
93	-15,45	-2,58	-0,42	7,65	1	0
163	-8,51	-10,22	-21,37	5,69	0	1

Fonte: Autoria Própria.

Tabela 15 – Índices de rentabilidade das empresas classificadas incorretamente pelo modelo *RF*

Empresa	ROA_{T-3}	ROA_{T-2}	ROA_{T-1}	ROE_{T-3}	ROE_{T-2}	ROE_{T-1}	Y	YPred.
357	-0,38	-0,45	-3,98	-0,14	-1,43	-33,43	1	0
76	-6,48	-23,98	-23,55	-31,71	-186,19	-284,51	0	1
113	1,03	6,56	17,08	4,55	23,71	43,91	0	1
16	-6,81	-8,23	-7,60	-22,44	-28,80	-27,24	0	1
93	-13,97	-2,31	-0,37	-317,32	-317,32	-317,32	1	0
163	-8,33	-10,57	-18,60	13,61	13,61	13,61	0	1

Fonte: Autoria Própria.

Ao analisar os indicadores das empresas que foram classificadas incorretamente pelo modelo, percebemos que as quatro empresas normais que foram classificadas como se estivesse em dificuldades financeiros (76, 113, 16 e 163), de fato os indicadores de liquidez e endividamento não são bons e as margens líquidas para essas empresas também não são boas, com exceção da empresa 113, ou seja, três empresas podem, eventualmente, pedir RJ ou RE no trimestre seguinte, evidenciando um risco para os credores e investidores se tivessem sido classificadas como empresas normais.

As duas empresas que pediram RJ ou EJ, que foram classificadas como normais (357 e 93), embora os indicadores de liquides tenham apresentado melhora nos último três trimestres que antecedem ao pedido de recuperação, os demais indicadores ainda são ruins, o que ainda representa um risco para os credores e investidores.

No contexto da gestão de riscos, quando um modelo classifica uma empresa boa como ruim (*False Positive* ou erro tipo 1) é menos prejudicial do que classificar uma empresa ruim como boa (*False Negative* ou erro tipo 2). No primeiro caso, os investidores, certamente, não investem na empresa e os credores não concedem crédito para a postulante, enquanto no segundo caso o risco de perda é maior, ou seja, os investidores podem vir a investir em uma empresa ruim e os credores podem vir a conceder crédito para uma empresa ruim. O resultado do nosso estudo previu 4 empresas na primeira situação e 2 empresas na segunda situação, o que fortalece o bom resultado obtido pelo modelo *RF*, demonstrando uma boa capacidade de previsão no cenário estudado.

O resultado obtido atende ao nosso objetivo, que é identificar um modelo que possa prever dificuldades financeiras de empresas, para subsídio à tomada de decisão, seja de conceder um crédito ou fazer um investimento, evitando assim, perdas financeiras. O modelo pode ser aplicado, ainda, ao se analisar uma renegociação de dívidas pelos credores, considerando que, a partir do momento que uma empresa entra em RJ ou RE, as condições de negociação são prejudicadas. Aplicando esta técnica, os credores podem atuar com mais eficiência antes que a empresa oficialize o pedido de RJ ou RE.

6 Conclusão

Em nosso estudo, testamos e comparamos 6 modelos de *Machine Learning* de classificação para prever antecipadamente se uma empresa pode vir a pedir RJ ou RE em um período posterior próximo, considerando, basicamente, os índice financeiros extraídos das demonstrações contábeis das empresas e, ao final, buscamos encontrar o melhor modelo para este cenário, com a otimização por meio do processo de validação cruzada e ajuste com hiperparâmetros.

Os resultados obtidos com os modelos testados em nossa base demonstram que são capazes de prever dificuldades financeiras das empresas com antecedência em nível comparável com os estudos antecedentes, com destaque para os modelos *GB* e o *RF* que evidenciam desempenho superior aos demais modelos testados. Considerando as métricas de validação utilizadas, o *GB* obteve melhor desempenho com a base original e com a base balanceada com *SMOTE*, já o *RF* demonstra ser o melhor, considerando a base balanceada com *NearMiss* e também após o processo de *cross validation* e otimização com hiperparâmetros, Concluímos, portanto, que esses dois modelos são os mais recomendados para serem utilizados pelos analistas para subsidiar a tomada de decisão dos credores e dos investidores. Observamos, ainda, das variáveis utilizadas, que mais contribuem para determinar a dificuldade financeira das empresas são os indicadores de Liquidez, endividamento e de rentabilidade das empresas.

Os modelos de previsão de dificuldades financeiras de empresas aplicados neste artigo utilizaram amostra de empresas listadas no mercado de ações brasileiro, e obtivemos resultados satisfatórios. Em princípio, também é aplicável a outros mercados. No entanto, se as características das empresas, dos mercados e das demonstrações financeiras que subsidiarem os dados utilizados, forem bastante diferentes das do mercado do Brasil, serão necessários mais testes para o sistema de índice. Portanto, construir um sistema de índices que possa ser aplicável a vários mercados é um assunto digno de estudo aprofundado no futuro.

Além disso, podem ser feitos novos estudos por segmentos de empresas e empregando outras técnicas estatísticas e/ou de inteligência artificiais, inclusive com esses mesmos dados. Nós testamos e comparamos 6 modelos de *Machine Learning*, e consideramos indicadores de balanços defasados referentes aos 12 trimestres anteriores ao evento de pedido de RJ ou RE oficializado pela empresa ($T-1$, $T-2$, ..., $T-12$). Novos estudo podem ser realizados considerando uma maior defasagem, como por exemplo, tomando os indicadores a partir de 2 trimestres anteriores ao evento ($T-3$, $T-4$, ..., $T-12$) ou a partir de 4 trimestres anteriores ao evento ($T-5$, $T-6$, ..., $T-12$).

Há, ainda, a opção de estruturar os dados em forma de painel, bem como podem ser incluídas outras variáveis independentes, como o porte da empresa, tempo de constituição, indicadores macroeconômicos, etc. Conforme afirmam Barboza, Kimura e Altman (2017), o debate sobre os melhores modelos para prever falhas provavelmente continuará no curto e médio prazo.

Referências

AGRAWAL, A.; GANS, J.; GOLDFARB, A. *Máquinas Preditivas: a simples economia da inteligência artificial*. [S.l.]: Alta Books, 2020. Citado na página 16.

BARBOZA, F.; KIMURA, H.; ALTMAN, E. Machine learning models and bankruptcy prediction. *Expert Systems with Applications*, Elsevier, v. 83, p. 405–417, 2017. Citado 4 vezes nas páginas 14, 17, 36 e 45.

BONELLO, J.; BRÉDART, X.; VELLA, V. Machine learning models for predicting financial distress. *Journal of Research in Economics*, v. 2, n. 2, p. 174–185, 2018. Citado 3 vezes nas páginas 14, 19 e 27.

BRASIL. *Lei n. 11.101, de 9 de fevereiro de 2005. Regula a recuperação judicial, a extrajudicial ea falência do empresário e da sociedade empresária*. [S.l.]: DOU Brasília, 2005. Citado na página 14.

BREIMAN, L. Random forests. *Machine Learning*, v. 45, p. 5–32, 2001. Citado na página 16.

BRUCE, A.; BRUCE, P. *Estatística Prática para Cientistas de Dados*. [S.l.]: Alta Books, 2019. Citado 3 vezes nas páginas 17, 26 e 37.

CHOI, H.; SON, H.; KIM, C. Predicting financial distress of contractors in the construction industry using ensemble learning. *Expert Systems with Applications*, Elsevier, v. 110, p. 1–10, 2018. Citado na página 27.

CRESPÍ-CLADERA, R.; MARTÍN-OLIVER, A.; PASCUAL-FUSTER, B. Financial distress in the hospitality industry during the covid-19 disaster. *Tourism Management*, Elsevier, v. 85, p. 104301, 2021. Citado na página 41.

EDWARD, A. et al. Financial ratios, discriminant analysis and the prediction of corporate bankruptcy. *Journal of Finance*, v. 23, n. 4, p. 589–609, 1968. Citado na página 14.

ESCOVEDO, T.; KOSHIYAMA, A. *Introdução a Data Science: Algoritmos de Machine Learning e métodos de análise*. [S.l.]: Casa do Código, 2020. Citado na página 18.

FREUND, Y.; SCHAPIRE, R. E. et al. Experiments with a new boosting algorithm. In: CITESEER. *icml*. [S.l.], 1996. v. 96, p. 148–156. Citado na página 18.

FRIEDMAN, J. H. Stochastic gradient boosting. *Computational statistics & data analysis*, Elsevier, v. 38, n. 4, p. 367–378, 2002. Citado na página 18.

GENG, R.; BOSE, I.; CHEN, X. Prediction of financial distress: An empirical study of listed chinese companies using data mining. *European Journal of Operational Research*, Elsevier, v. 241, n. 1, p. 236–247, 2015. Citado 4 vezes nas páginas 13, 14, 19 e 41.

HASTIE, T. et al. *The elements of statistical learning: data mining, inference, and prediction*. [S.l.]: Springer, 2009. v. 2. Citado na página 16.

KIM, H.-J.; JO, N.-O.; SHIN, K.-S. Optimization of cluster-based evolutionary undersampling for the artificial neural networks in corporate bankruptcy prediction. *Expert systems with applications*, Elsevier, v. 59, p. 226–234, 2016. Citado 2 vezes nas páginas 27 e 34.

KISELEVA, I. et al. Models for assessing the bankruptcy probability for enterprises. *International Journal of Recent Technology and Engineering*, v. 8, n. 2, p. 6433–6439, 2019. Citado na página 21.

LEO, M.; SHARMA, S.; MADDULETY, K. Machine learning in banking risk management: A literature review. *Risks*, Multidisciplinary Digital Publishing Institute, v. 7, n. 1, p. 29, 2019. Citado 3 vezes nas páginas 14, 16 e 18.

LI, J. et al. Feature selection: A data perspective. *ACM computing surveys (CSUR)*, ACM New York, NY, USA, v. 50, n. 6, p. 1–45, 2017. Citado na página 25.

PENMAN, S. *Análise de demonstrações financeiras e security valuation*. [S.l.]: Elsevier Brasil, 2013. Citado na página 21.

PETROPOULOS, A. et al. Predicting bank insolvencies using machine learning techniques. *International Journal of Forecasting*, Elsevier, v. 36, n. 3, p. 1092–1113, 2020. Citado na página 17.

SCALZER, R. S. et al. Financial distress in electricity distributors from the perspective of brazilian regulation. *Energy Policy*, Elsevier, v. 125, p. 250–259, 2019. Citado na página 41.

SERRANO-CINCA, C.; GUTIÉRREZ-NIETO, B.; BERNATE-VALBUENA, M. The use of accounting anomalies indicators to predict business failure. *European Management Journal*, Elsevier, v. 37, n. 3, p. 353–375, 2019. Citado na página 21.

SORIA, D. et al. A 'non-parametric'version of the naive bayes classifier. *Knowledge-Based Systems*, Elsevier, v. 24, n. 6, p. 775–784, 2011. Citado na página 18.

TAVANA, M. et al. An artificial neural network and bayesian network model for liquidity risk assessment in banking. *Neurocomputing*, Elsevier, v. 275, p. 2525–2554, 2018. Citado na página 19.

TELES, G. et al. Comparative study of support vector machines and random forests machine learning algorithms on credit operation. *Software: Practice and Experience*, Wiley Online Library, v. 51, n. 12, p. 2492–2500, 2021. Citado 3 vezes nas páginas 14, 16 e 18.

VEGANZONES, D.; SÉVERIN, E.; CHLIBI, S. Influence of earnings management on forecasting corporate failure. *International Journal of Forecasting*, Elsevier, 2021. Citado na página 17.

WANG, G.; MA, J.; YANG, S. An improved boosting based on feature selection for corporate bankruptcy prediction. *Expert Systems with Applications*, Elsevier, v. 41, n. 5, p. 2353–2361, 2014. Citado na página 25.

ZHANG, X.; ZHAO, Y.; YAO, X. Forecasting corporate default risk in china. *International Journal of Forecasting*, Elsevier, 2021. Citado na página 13.

Apêndices

APÊNDICE A – Códigos *Phyton*

```
Bibliotecas:
## manipulação de dataframes
import matplotlib.pyplot as plt
import pandas as pd

## visualização de dados
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt

## analise dataframe
import pandas_profiling
import random from collections
import Counter

## ferramenta para avaliar relação entre variáveis e a target
from sklearn.feature_selection import mutual_info_classif
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import RFE

## algoritmos
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn import svm
from sklearn.naive_bayes import GaussianNB
from sklearn.neural_network import MLPClassifier

## seleção das amostras
from sklearn.model_selection import train_test_split

## Grid e Random Search com Cross Validation(CV)
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import RandomizedSearchCV

## balanceamento das amostras
from imblearn.over_sampling import SMOTE
from imblearn.under_sampling import NearMiss

## avaliação de desempenho dos classificadores
from sklearn import metrics
from sklearn.metrics import (accuracy_score, confusion_matrix,
precision_score,precision_recall_curve,auc,roc_curve, recall_score,
classification_report,
f1_score,precision_recall_fscore_support,
mean_absolute_error,roc_auc_score,mean_squared_error)

## graficos
%matplotlib inline
%config InlineBackend.figure_formats = ['svg']

```

```
## Converter variáveis categóricas em dummies
!pip install category_encoders

Importando Dataset
df = pd.read_excel("/content/base_economatica_acoes_tratado7.xlsx",
nrows=0, sheet_name= "Base_tratada")
Pré processamento
## Checando missing values
print("\nExiste algum missing value
(NULL)?:",df.isnull().sum()[df.isnull().sum()>0].sum())
print("\nExiste algum missing value(NA)?:",df.isna().sum()[df.isna().sum()>0].sum())

## Transformando as variáveis categóricas em dummies
import category_encoders as ce
from category_encoders.one_hot import OneHotEncoder
one_hot_encoder = OneHotEncoder(cols=['Setor_Economico_B3'])
dfcat = df.select_dtypes(include=['category'])
dummies = one_hot_encoder.fit_transform(dfcat)
merged = pd.concat([df, dummies], axis='columns')

## Feature Selection
# ANOVA Seleção de atributos para Classificação
from sklearn.feature_selection import SelectKBest, f_classif, chi2
fs = SelectKBest(score_func=f_classif, k=30)
res = resultados[:50]
plt.figure(figsize=(30,6))
g = sns.barplot(x='Colunas', y='F_Scores', data=res,order=res.sort_values('F_Scores',
ascending=False).Colunas);
g.set_yscale("log")
plt.xticks(rotation=70)
plt.tight_layout()
Preparando a amostra

## Separando em Treino e Teste (70% para treino e 30% para teste)
# Base original
df = df_semDummy
X = df.drop(["Y", "Nome", "Setor_Economico_B3"], axis =1)
y = df.Y
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3,
random_state = parametros['semente'])

# Balanceando a amostra com NearMiss
rm = NearMiss()

#aplicando nearmiss
X_nrm,y_nrm = nrm.fit_resample(X,y)
print("Distribuição:\n",y_nrm.value_counts(),"\n")
ax = sns.countplot(x=y_nrm)

#separando treino e teste
X_train_nrm, X_test_nrm, y_train_nrm, y_test_nrm = train_test_split(X_nrm, y_nrm,
test_size = 0.30,
random_state=parametros['semente'])

# Balanceando a amostra com SMOTE
smt = SMOTE(sampling_strategy=0.6)

#Balanceado a amostra com smote
```

```
X_smt,y_smt = smt.fit_resample(X,y)
print("Distribuição:\n",y_smt.value_counts(),"\n")
ax = sns.countplot(x=y_smt)

#separando treino e teste
X_train_smt, X_test_smt, y_train_smt, y_test_smt = train_test_split(X_smt, y_smt,
test_size = 0.30, random_state=parametros['semente'])

Aplicando os modelos
## Random Forest
#Carregando algoritmo
modeloRF      = RandomForestClassifier (n_estimators= 500,
random_state=parametros['semente'])
modeloRF_nrm  = RandomForestClassifier (n_estimators= 500,
random_state=parametros['semente'])
modeloRF_smt  = RandomForestClassifier (n_estimators= 500,
random_state=parametros['semente'])

#treinando modelos
modeloRF.fit(X_train,y_train)
modeloRF_nrm.fit(X_train_nrm,y_train_nrm)
modeloRF_smt.fit(X_train_smt,y_train_smt)

#classificando bases de teste
y_pred      = modeloRF.predict(X_test)
y_pred_nrm  = modeloRF_nrm.predict(X_test_nrm)
y_pred_smt  = modeloRF_smt.predict(X_test_smt)

## Gradient Boost
#carregando algoritmo
modeloGB = GradientBoostingClassifier(
    loss='deviance', learning_rate=20, n_estimators=500,
    subsample=1.0, criterion='friedman_mse', min_samples_split=3,
    min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_depth=10,
    min_impurity_decrease=0.0, init=None,
    random_state=parametros['semente'], max_features=None, verbose=0,
    max_leaf_nodes=None, warm_start=False,
    validation_fraction=0.1, n_iter_no_change=None, tol=0.0001)

modeloGB_nrm = GradientBoostingClassifier(
    loss='deviance', learning_rate=20, n_estimators=500,
    subsample=1.0, criterion='friedman_mse', min_samples_split=3,
    min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_depth=10,
    min_impurity_decrease=0.0, init=None,
    random_state=parametros['semente'], max_features=None, verbose=0,
    max_leaf_nodes=None, warm_start=False,
    validation_fraction=0.1, n_iter_no_change=None, tol=0.0001)

modeloGB_smt = GradientBoostingClassifier(
    loss='deviance', learning_rate=20, n_estimators=500,
    subsample=1.0, criterion='friedman_mse', min_samples_split=3,
    min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_depth=10,
    min_impurity_decrease=0.0, init=None,
    random_state=parametros['semente'], max_features=None, verbose=0,
    max_leaf_nodes=None, warm_start=False,
    validation_fraction=0.1, n_iter_no_change=None, tol=0.0001)

#treinando o classificador
```

```
168 modeloGB.fit(X_train,y_train)
169 modeloGB_nrm.fit(X_train_nrm,y_train_nrm)
170 modeloGB_smt.fit(X_train_smt,y_train_smt)
171
172 #classificando bases de teste
173 y_pred_gb = modeloGB.predict(X_test)
174 y_pred_gb_nrm = modeloGB_nrm.predict(X_test_nrm)
175 y_pred_gb_smt = modeloGB_smt.predict(X_test_smt)
176
177 ## Logistic Regression
178 #Carregando algoritmo
179 lr      = LogisticRegression(random_state = parametros['semente'],
180 solver = 'liblinear', max_iter=1000)
181 lr_nrm  = LogisticRegression(random_state = parametros['semente'],
182 solver = 'liblinear', max_iter=1000)
183 lr_smt  = LogisticRegression(random_state = parametros['semente'],
184 solver = 'liblinear', max_iter=1000)
185
186 #treinando o algoritmo
187 lr.fit(X_train,y_train)
188 lr_nrm.fit(X_train_nrm,y_train_nrm)
189 lr_smt.fit(X_train_smt,y_train_smt)
190
191 #classificando bases de teste
192 y_pred_lr       = lr.predict(X_test)
193 y_pred_lr_nrm   = lr_nrm.predict(X_test_nrm)
194 y_pred_lr_smt   = lr_smt.predict(X_test_smt)
195
196 ## SVM - Suport Vector Machine
197 #Carregando algoritmo
198 modeloSVM       = svm.LinearSVC(max_iter=50000, random_state=parametros['semente'],
199 C = 1)
200 modeloSVM_nrm   = svm.LinearSVC(max_iter=50000, random_state=parametros['semente'],
201 C = 1)
202 modeloSVM_smt   = svm.LinearSVC(max_iter=50000, random_state=parametros['semente'],
203 C = 1)
204
205 #treinando o algoritmo
206 modeloSVM.fit(X_train,y_train)
207 modeloSVM_nrm.fit(X_train_nrm,y_train_nrm)
208 modeloSVM_smt.fit(X_train_smt,y_train_smt)
209
210 #classificando bases de teste
211 y_pred_svm        = modeloSVM.predict(X_test)
212 y_pred_svm_nrm    = modeloSVM_nrm.predict(X_test_nrm)
213 y_pred_svm_smt    = modeloSVM_smt.predict(X_test_smt)
214
215 ## Naive Bayes
216 # Inicializando classificador
217 modeloGNB = GaussianNB()
218 modeloGNB_nrm = GaussianNB()
219 modeloGNB_smt = GaussianNB()
220
221 # Treinando o classificador
222 model = modeloGNB.fit(X_train,y_train)
223 model_nrm = modeloGNB_nrm.fit(X_train_nrm,y_train_nrm)
224 model_smt = modeloGNB_smt.fit(X_train_smt,y_train_smt)
225
```

```
#classificando bases de teste
y_pred_gnb = modeloGNB.predict(X_test)
y_pred_gnb_nrm = modeloGNB_nrm.predict(X_test_nrm)
y_pred_gnb_smt = modeloGNB_smt.predict(X_test_smt)

## Redes Neurais - MLPClassifier
# Inicializando classificador
modeloMLP = MLPClassifier(solver='lbfgs', alpha=1e-5, hidden_layer_sizes=(5, 2),
random_state=parametros['semente'])
modeloMLP_nrm = MLPClassifier(solver='lbfgs', alpha=1e-5, hidden_layer_sizes=(5, 2),
random_state=parametros['semente'])
modeloMLP_smt = MLPClassifier(solver='lbfgs', alpha=1e-5, hidden_layer_sizes=(5, 2),
random_state=parametros['semente'])

# Treinando o classificador
modeloMLP.fit(X_train,y_train)
modeloMLP_nrm.fit(X_train_nrm,y_train_nrm)
modeloMLP_smt.fit(X_train_smt,y_train_smt)

#classificando bases de teste
y_pred_mlp = modeloMLP.predict(X_test)
y_pred_mlp_nrm = modeloMLP_nrm.predict(X_test_nrm)
y_pred_mlp_smt = modeloMLP_smt.predict(X_test_smt)
Verificando desempenho dos modelos
## base normal (sem balancear)
modelos = showReportClassification('>Random Forest',y_test, y_pred, False)\
.join(showReportClassification('>Logit',y_test, y_pred_lr, False))\
.join(showReportClassification('>Gradient Boost',y_test, y_pred_gb, False))\
\
.join(showReportClassification('Suport Vector Machine',y_test, y_pred_svm, False))\
.join(showReportClassification('Gausian Naive Bayes',y_test, y_pred_gnb, False))\
.join(showReportClassification('Neural Network-MLP',y_test, y_pred_mlp, False))\

## base aplicada o nearmiss
modelos = showReportClassification('>Random Forest',y_test_nrm, y_pred_nrm, False)\
.join(showReportClassification('>Logit',y_test_nrm, y_pred_lr_nrm, False))\
.join(showReportClassification('>Gradient Boost',y_test_nrm, y_pred_gb_nrm, False))\
.join(showReportClassification('Suport Vector Machine',y_test_nrm, y_pred_svm_nrm,
False))\
.join(showReportClassification('Gausian Naive Bayes',y_test_nrm, y_pred_gnb_nrm,
False))\
.join(showReportClassification('Neural Network-MLP',y_test_nrm, y_pred_mlp_nrm,
False))\

## base aplicada o Smote
modelos = showReportClassification('>Random Forest',y_test_smt, y_pred_smt,
False)\
.join(showReportClassification('>Logit',y_test_smt, y_pred_lr_smt,
False))\
.join(showReportClassification('>Gradient Boost',y_test_smt, y_pred_gb_smt,
False))\
\
.join(showReportClassification('Suport Vector Machine',y_test_smt, y_pred_svm_smt,
False))\
.join(showReportClassification('Gausian Naive Bayes',y_test_smt, y_pred_gnb_smt,
False))\
.join(showReportClassification('Neural Network-MLP',y_test_smt, y_pred_mlp_smt,
False))\
```

```
284
285 Aplicando Cross validation
286 from sklearn.model_selection import cross_val_score
287 from sklearn.model_selection import cross_validate
288
289 # divide 70% para validação e 30% para teste
290 #train_x, test_x, train_y, test_y = train_test_split(X, y, test_size = 0.3,
291 random_state = parametros['semente'])
292
293 #amostra balanceada
294 train_x, test_x, train_y, test_y = train_test_split(X_smt, y_smt, test_size = 0.3,
295 random_state = parametros['semente'])
296 print('Treino :', train_x.shape)
297 print('Teste:', test_x.shape)
298
299 # conjunto de modelos
300 models = [#RandomForestClassifier(min_samples_leaf=2, min_samples_split=6,
301 n_estimators=25, random_state=42)
302           RandomForestClassifier(min_samples_leaf=2, n_estimators=200,
303           random_state=parametros['semente'])
304           #,LogisticRegression(random_state = parametros['semente']
305           , solver='liblinear')
306             ,GradientBoostingClassifier(
307             loss='deviance', learning_rate=20, n_estimators=500,
308             subsample=1.0, criterion='friedman_mse', min_samples_split=3,
309             min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_depth=10,
310             min_impurity_decrease=0.0, init=None,
311             random_state=parametros['semente'], max_features=None, verbose=0,
312             max_leaf_nodes=None, warm_start=False,
313             validation_fraction=0.1, n_iter_no_change=None, tol=0.0001)
314
315           ,SVC(random_state=parametros['semente'])
316
317           ,MLPClassifier(solver='lbfgs', alpha=1e-5, hidden_layer_sizes=(5, 2),
318           random_state=parametros['semente'])
319
320           ,GaussianNB()
321
322           ]
323
324 scoring = ['accuracy','roc_auc','f1','precision','recall']
325
326 resultado = pd.DataFrame(index = ['Acurácia','AUC', 'F1','Recall','Precision'])
327
328 for model in models:
329     val_scores = cross_validate(model, train_x, train_y, cv=5, scoring=scoring)
330     nome_modelo = type(model).__name__ # somente para exibição
331
332     acc = round(np.mean(val_scores['test_accuracy'])*100,2) #calculate accuracy (%)
333     auc = round(np.mean(val_scores['test_roc_auc'])*100,2) #calculate AUC of model
334     f1  = round(np.mean(val_scores['test_f1'])*100,2)
335     rec = round(np.mean(val_scores['test_recall'])*100,2)
336     pre = round(np.mean(val_scores['test_precision'])*100,2)
337
338
339     tmp = pd.DataFrame(
340         {nome_modelo: [str(acc) + '',str(auc)+ '', str(f1) +'', str(rec) +'',
341         str(pre) +'']},
```

```
        index = ['Acurácia','AUC', 'F1','Recall','Precision'])

    resultado = resultado.join(tmp)

resultado.transpose()

Otimização de hiperparâmetros
##  Tunning the models
# Random Search
# N Estimators
n_estimators = [25, 50, 100, 200, 500, 900, 1100, 1500, 2500]
# Min Samples Split
min_samples_split = [2, 4, 6, 10]
# Min Samples Leaf
min_samples_leaf = [1, 2, 4, 6, 8]
# Max Features
max_features = ['auto', 'sqrt','log2', None]
# Criando o Random Search
param = {
               'n_estimators': n_estimators,
               'min_samples_split': min_samples_split,
               'min_samples_leaf': min_samples_leaf,
               'max_features': max_features}

#scoring = ['accuracy','roc_auc','f1','precision','recall']

## Random Forest Tuning
tuning = RandomizedSearchCV(estimator=RandomForestClassifier
(random_state=parametros['semente']), ## Random Forest
                            param_distributions=param, ## Parâmetros
                            cv=5, ## Cross validation
                            n_iter=20, ## Número de Iterações
                            #scoring=scoring,
                            scoring = 'accuracy', ## Métrica
                            n_jobs = -1, ## Utilizando todos os processadores
                            verbose = 1,
                            random_state=parametros['semente'])

## Fazendo o fit do tuning
tuning.fit(train_x,train_y)

## Melhor configuração encontrada
tuning.best_estimator_
best_model = tuning.best_estimator_
best_model.fit(train_x, train_y)
y_pred = best_model.predict(test_x)
from sklearn.metrics import f1_score
print(f1_score(test_y, y_pred)*100)

## curva ROC

showROC(test_y, y_pred)

## Matriz de confusão
showConfusionMatrix(test_y, y_pred)

## Mostrando os erros
test=test_x.copy()
```

```
400 test['y']=test_y
401 test['ypred']=y_pred
402
403 #erros
404 errors = test[test['y']!=test['ypred']]
405
406 #salvando dados
407 errors.to_csv(index=False, path_or_buf='/content/erro-rf.csv', encoding='UTF-8')
408
409 errors
410
411 ## Aplicando Feature Importance
412 dfFI = pd.DataFrame(best_model.feature_importances_).join(test_x.columns.to_frame().
413 reset_index()['index'])
414 dfFI[dfFI[0]>0.01].sort_values(by=0).plot.barh(x='index', figsize=(20,10))
415 plt.show()
416 print(dfFI[dfFI[0]>0.01].count()[0])
```