Ciência de Dados aplicada ao Portfólio de Produtos Financeiros

Resumo

O Portfólio de Produtos Financeiros (PPF) é uma ferramenta utilizada pelo Ministério de Ciência, Tecnologia e Inovações para organizar e disponibilizar informações sobre fontes de financiamento de projetos de CT&I e oportunidades para os projetos brasileiros. O objetivo desse trabalho é coletar as informações de entrada para o PPF e classificar a elegibilidade de forma automatizada. A metodologia consiste na modelagem com técnicas de ciências de dados, tais como *Web Scraping*, *Machine Learning* e Processamento de Linguagem Natural. Apresentam-se neste documento os resultados obtidos e suas respectivas discussões, bem como as considerações finais e os trabalhos futuros para a continuidade do projeto. **Palavras-chave**: aprendizado de máquina, ciência de dados, financiamento de projetos de CT&I, processamento de linguagem natural, raspagem da web.

Abstract

The Financial Products Porfolio (FPP) is a tool utilised by the Ministry os Science, Technology and Inovations to organize and make available information about ST&I project finance sources and opportunities for Brazilian projects. The objective of this work is to collect input information for the FPP and to classify the eligibility in an automated manner. The methodology consists in data science modeling with Web Scrapping, Machine Learning and Natural Language Processing techniques. In this document, the results obtained and their discussions are presented, as well as final considerations and future work for the continuity of the project. **Keywords**: *machine learning*, *data science*, *ST&I project finance*, *natural language processing*, *web scraping*.

1 Introdução

Alinhado às estratégias de Governo Digital e de Transformação Digital, a fim de implementar suas competências com efetividade, a Secretaria de Estruturas Financeiras e de Projetos (SEFIP) organizou internamente um Grupo de Trabalho (GT) com o fito de desenvolver um projeto de Ciência de Dados (CD) e Inteligência Artificial (IA) para automatizar os processos de coleta, classificação e recomendação de fontes de financiamento para o Portfólio de Produtos Financeiros - PPF (BRASIL, 2019).

O PPF é um instrumento criado pelo Departamento de Estruturas para Viabilização Financeira de Projetos (DECFI/SEFIP) para auxiliar o Ministério de Ciência, Tecnologia e Inovações (MCTI) a elaborar ações de captação de recursos não orçamentários para o fomento a projetos

de pesquisa e dar mais visibilidade às oportunidades existentes de financiamento ao desenvolvimento científico e tecnológico do país.

Dessa forma, o PPF foi criado como um instrumento para organizar e sistematizar a captação de recursos não orçamentários, permitindo estruturar e listar fontes nacionais e internacionais que podem fomentar projetos de pesquisa, mapear oportunidades abertas de fomento à pesquisa, de bolsas de fomento e posições de pesquisa oferecidas em instituições nacionais e estrangeiras. Como a busca por recursos financeiros é uma das funções da *Delivery Unit* do Departamento de Estruturas de Projetos em Ciência, Tecnologia e Inovação (DEPRO/SEFIP), o sistema tem grande importância para alavancar os projetos prioritários selecionados pelas unidades vinculadas ao MCTI.

A questão do projeto de pesquisa é como aplicar a ciência de dados ao Portfólio de Produtos Financeiros, com uso de técnicas de Varredura de Dados, Aprendizado de Máquina, Inteligência Artificial, Processamento de Linguagem Natural, com possíveis intersecções entre essas áreas. Na sua construção, são utilizadas linguagens de programação como R e Python. O objetivo desse trabalho é coletar as informações de entrada para o PPF e classificar a elegibilidade de forma automatizada.

2 Referencial Teórico

2.1 Aprendizado de Máquina e Processamento de Linguagem Natural

O Aprendizado de Máquina (*Machine Learning*) é o termo usado para definir um sistema que modifica seu comportamento com base em sua própria experiência de aprendizado. Neste contexto, o processamento de linguagem natural (PLN) relaciona a linguística humana com a linguagem computacional. Dessa forma o PLN possibilita a leitura e interpretação de textos, identificando contextos importantes. Nessa seção, serão abordadas as principais metodologias de *Machine Learning* e PLN utilizadas.

2.1.1 *Bag of Words* (BoW)

Bag of Words (BoW) é um método utilizado para extrair recursos de textos de importância para ser utilizado para treinar algoritmos de aprendizado de máquina (ZHANG; JIN; ZHOU, 2010). Ele cria uma espécie de dicionário de todas as palavras exclusivas que ocorrem em todos os textos do conjunto de dados.

O BoW nada mais é que uma coleção de palavras representadas por uma contagem que desconsidera a ordem em que as palavras aparecem em cada texto. Amplamente utilizado em Processamento de Linguagem Natural (PLN) e classificação de documentos, um dos problemas desse método é a presença de palavras similares em cada texto sendo classificadas isoladamente, mesmo possuindo o mesmo sentido semântico. Além disso, algumas palavras de conexão e elementos de pontuação também precisam receber um tratamento.

Esse tratamento é o processo de Tokenização das sentenças e eliminação das chamadas *stopwords*, que são palavras com alta frequência e baixo conteúdo léxico, como adjetivos, artigos e preposições. Na PLN tradicional ((RUSSELL; NORVIG, 2010), cap. 22), as palavras são tokens (LIU; LIN; SUN, 2020), usados para estimar probabilidades condicionais das próximas palavras (n-gram).

Tokenização é o ato de quebrar uma sequência de *strings* em pedaços, como palavras, palavras-chave, frases, símbolos e outros elementos chamados tokens. Os tokens podem ser palavras individuais, frases ou até frases inteiras. No processo de tokenização, alguns caracteres,

como sinais de pontuação e caracteres especiais, são eliminados junto com as *stopwords* para limpar a base de textos. O pacote Natural Language Toolkit - NLTK (NLTK, 2021; LOPER; BIRD, 2002; BIRD, 2006) possui uma implementação de stopwords prontas para ser utilizada.

Uma das principais limitações do uso do BoW é a não consideração do contexto e da ordem das palavras, logo, o sentido semântico é totalmente ignorado na abordagem. A abordagem computacional também pode ser um problema para textos muito grandes, nos quais o cálculo para cada vetor pode tomar muito tempo de execução.

2.1.2 TF-IDF

O TF-IDF (*term frequency - inverse document frequency*) é uma ferramenta estatística utilizada para ponderação de pesos de termos em documentos (AIZAWA, 2003). Em suma, a ferramenta atribui pesos aos termos em uma coleção de documentos dados dois parâmetros: a quantidade de vezes que um termo aparece em um documento específico e quantas vezes aparece na coleção como um todo.

O TF-IDF é bastante empregado em análise textual, na codificação ou vetorização da entrada, para aprendizado de máquina e em algoritmos de busca. O TF-IDF é composto por duas partes:

- TF (*term frequency*): o termo ganha peso proporcional a quantas vezes aparece no documento.
- IDF (*inverse document frequency*): o termo perde peso proporcional a quantas vezes aparece na coleção de documentos.

Essa dinâmica possibilita a atribuição de pesos maiores a palavras realmente relevantes para a categorização dos textos. O TF é responsável por atribuir sentido a palavras de destaque, enquanto, por outro lado, o IDF é responsável por diminuir o peso de palavras muito repetidas, como artigos e conectivos.

Existem muitos métodos descritos na literatura para cálculo dos pesos do modelo TF-IDF. A formula de cálculo padrão dos pesos do termo i do documento j para o número total de documentos N é representada por $TFIDF(i,j,N) = TF(i,j) \times IDF(i,N)$.

Essa função do TF-IDF é feita pela multiplicação da componente local (TF) com a componente global (IDF). Além disso, normaliza-se os documentos resultantes para um comprimento unitário por uma função logarítmica. No caso do pacote scikit-learn é o logaritmo natural, ou neperiano. Assim, essa função dos pesos é representada por (SKLEARN, 2020a):

$$peso_{i,j} = frequencia\_local_{i,j} \times \left[log\left(\frac{D}{frequencia\_global_i}\right) + 1 \right],$$

em que $frequencia\_local_{i,j}$ representa a frequência do termo i no documento j, $frequencia\_global_i$ representa a frequência do termo i em todos os documentos, e D é um inteiro que indica a quantidade de documentos. Esta fórmula vale para o caso da flag smooth_idf ser falsa, ou seja, $frequencia\_global_i \neq 0$. Para evitar divisão por zero, o sistema utiliza uma fórmula alternativa (SKLEARN, 2020a):

$$IDF(i) = \left[log\left(\frac{D+1}{frequencia\_global_i + 1}\right) + 1 \right],$$

O efeito de adicionar "1" fora do logaritmo na equação do IDF acima tem a função de lidar com os termos com IDF zero, ou seja, os termos que ocorrem em todos os documentos em um conjunto de treinamento. Assim, os termos com $log(1) = 0$ não serão totalmente ignorados.

Nota-se que esta fórmula difere da definição usual da componente global (IDF), que é geralmente definida como (SKLEARN, 2020a):

$$IDF(i) = log\left(\frac{D}{frequencia\_global_i + 1}\right),$$

No entanto, esta diferença não altera a representação vetorial do conjunto global de termos nos documentos considerados. O resultado prático é que os valores dos pesos serão menores em decorrência da adição de "1" ao denominador.

2.1.3 *Naive Bayes*

O *Naive Bayes* é um modelo classificador probabilístico, simples e surpreendentemente eficaz em classificação binária (TING et al., 2011). Pode ser usado para decidir se uma mensagem é ou não *spam*, por exemplo. Além disso, é importante ressaltar que o *Naive Bayes* possui uma hipótese simplificadora que é a suposição de independência entre as características (*features*) do modelo. Assim, no caso do seu uso em classificadores textuais, a potencial dependência estatística entre os tokens é desconsiderada.

Sejam duas variáveis que se relacionam X e Y, o algoritmo de *Naive Bayes* consiste em encontrar uma probabilidade a posteriori dado à ocorrência do evento à priori através do teorema de Bayes, de forma que o classificador escolherá a variável Y em que a probabilidade é maior.

O método de *Naive Bayes* multinomial, como diz o nome, é utilizado para dados multinomialmente distribuídos. Sua aplicação segundo a literatura é bem realizada para dados que são normalmente representados como contagens de vetores de palavras, mas que também funciona bem para os vetores TF-IDF na prática (SKLEARN, 2021).

A distribuição é parametrizada pelo vetor $\theta_y = (\theta_{y1}, \ldots, \theta_{yn})$ para cada rótulo (*label*) y, em que n representa o número de características (*feature*) e θ_{yi} a probabilidade condicional $P(x_i \mid y)$ da característica i do vetor x, dado o rótulo y.

O parâmetro θ_{yi} tem sua estimação dada por (SKLEARN, 2020b):

$$\hat{\theta}_{yi} = \frac{N_{yi} + \alpha}{N_y + \alpha n}$$

Em que $N_{yi} = \sum_{x \in T} x_i$ representa o número de vezes que a característica i aparece em uma amostra do rótulo y no conjunto de treinamento T, e $N_y = \sum_{i=1}^{n} N_{yi}$ é a contagem total de todos os elementos da amostra para classificar o rótulo y. $\alpha \geq 0$ é uma espécie de parâmetro de correção e evita que uma probabilidade seja igual a zero nos cálculos.

2.1.4 SVM

Caso todas as amostras de um conjunto de dados consigam ser plotadas no espaço e divididas por uma linha reta, o conjunto de dados é designado linearmente separável. Para esse tipo de conjunto de dados, uma abordagem interessante é o *Support Vector Machine* - SVM (MOE et al., 2018).

Assim, utiliza-se o SVM como um classificador linear binário, que utiliza um algoritmo de aprendizado supervisionado. No SVM, as amostras pertencentes à um conjunto de dados com n características são plotados em um espaço com n dimensões. O objetivo do SVM é encontrar uma fronteira de decisão, também chamada de hiperplano ou linha, que melhor segregue e classifique os dados de acordo com as classes impostas. Para a decisão do melhor hiperplano,

leva-se em conta, em primeira instância, a assertividade do hiperplano e, em segundo lugar, qual hiperplano possui maior distância entre os dados próximos.

Para classificação de amostras que não são linearmente separáveis, o SVM conta com funções matemáticas chamadas de *Kernel*, ou núcleos, que aumentam a dimensão do problema e aplicam transformações no conjunto de dados de modo a facilitar o traçado do hiperplano.

Dentre as principais vantagens de se utilizar o SVM, destacam-se boa assertividade em espaços multidimensionais, diversidade de funções núcleo que podem ser abordadas e ainda mantém-se eficaz em casos que o número de características é maior que o número de amostras.

2.1.5 Random Forest

Um dos maiores problemas em se utilizar árvores de decisão para modelos previsores é a grande chance de *overfitting*, ou sobreajuste. Por definição, o sobreajuste ocorre quando um modelo se corresponde firmemente a uma base de dados de treinamento e possui dificuldades de se ajustar a outras amostras não previstas, o que piora a generalização do modelo.

Como tentativa de contornar esse problema, o último modelo classificador abordado é o *Random Forest* (AUNG; MYANMAR; HLA, 2009). Por ser composto por um grande número de árvores de decisão, esse método possibilita menor propensão à sofrer *overfitting*. Justifica-se esse efeito por dois fatores principais.

O primeiro diz respeito ao fato de que o modelo é um *ensemble* de árvores de decisão. A decisão final do modelo é formado pela soma das decisões individuais de cada árvore do conjunto. A resposta mais frequente é a resposta do modelo como um todo. Sendo assim, árvores enviesadas têm pouca influência na resposta final.

O segundo decorre da maneira como as árvores são formadas: o *bagging* ou *bootstrap agregation*. O processo consiste em criar árvores a partir de subamostras com reposição do conjunto de dados. Tal técnica de subamostragem permite que as árvores se divirjam quanto à relevância e à quantidade de características levadas em consideração, o que implica em maior generalização do modelo e, consequentemente, menor tendência ao sobreajuste.

3 Metodologia

3.1 Tipo da Pesquisa

A pesquisa adota um enfoque epistemológico indutivo, com aprendizado de máquina tradicional supervisionado. Esta é uma pesquisa de modelagem, inspirada no design science (DRESCH; LACERDA; JÚNIOR, 2015). Ao mesmo tempo, é uma pesquisa-ação (THIOLLENT, 2009), onde a pesquisa é conduzida em um espaço de interlocução, onde os atores envolvidos participam da identificação e resolução de problemas.

A coleta, análise e comunicação de dados foram realizadas por múltiplos métodos e modelos quantitativos de ciência de dados.

A pesquisa foi gerenciada com um enfoque de gestão de projetos ágil em ciência de dados. A abordagem utilizada (BAIJENS; HELMS; IREN, 2020) integra o enfoque de Scrum com o CRISP-DM (*Cross Industry Standard Process for Data Mining*), integrando, assim, os eventos, artefatos e papéis do Scrum.

Para se fazer a gestão macro, por meio de sistema, utilizou-se o SIGE3P (Sistema de Gestão de Projetos do MCTI - adaptado do MS-Project) e para a gestão técnica do projeto (dia a dia - as sprints e respectivas tarefas) o Azure Devops (ROSSBERG, 2019). Nesta última, a abordagem mais apropriada para fazer a evolução das entregas e o devido refinamento baseou-se no TSDP

- *Team Data Science Process* (MICROSOFT, 2020), com contínuas iterações entre a Equipe de Desenvolvimento e a *Product Owner* (PO).

3.2 Métodos

A necessidade de uma metodologia mais automatizada surgiu com o volume de dados necessários para abastecer o Portfólio de Produtos Financeiros (PPF) diariamente. As principais áreas de busca, em sites selecionados de fontes de financiamento à ciência, tecnologia e inovação (CT&I), que este trabalho busca acessar, foram divididas em quatro campos correspondendentes:

- Oportunidades Abertas: Campo que corresponde à oportunidades elegíveis para o Brasil de modo geral como bolsa de estudos, financiamento de projetos entre outros relacionados.
- Notícias: Busca por notícias dessas instituições que estão relacionadas com produtos financeiros.
- Políticas: Informações sobre a descrição da instituição e a forma de atuação da instituição relacionadas à produtos financeiros.
- Projetos: Projetos já financiados e aprovados pela instituição que se relacionam com o Brasil de alguma forma.

Para obter o conjunto de dados dos campos de interesse do PPF, as informações de cada site foram obtidas através de técnicas de raspagem de dados (*Web Scraping*) com o software Python (MITCHELL, 2018), utilizando o pacote *Beautiful Soup* (RICHARDSON, 2007).

3.3 População e Amostra

No presente projeto, existem mais de 100 sites de instituições e órgãos que são utilizados para a busca de diversas áreas do PPF. O total da população de fontes de financiamento à CT&I, em múltiplos idiomas, ainda é desconhecido.

Assim, aplica-se a busca com um conjunto de palavras-chave, em uma lista de sites pré-definidos. Além disso, aplica-se uma adaptação de uma metodologia de bola de neve *snowball sampling* (DOBROVOLSKYI; KEBERLE, 2018), acrescentando novos sites por amostragem de conveniência.

3.4 Coleta de Dados

O principal módulo utilizado para obter os dados para este trabalho é o módulo *Beautiful Soup*, que é uma biblioteca Python de extração de dados de arquivos HTML e XML (RICHARDSON, 2007). Ela permite a utilização do interpretador (*parser*) favorito, a fim de navegar, buscar e modificar uma árvore de análise (*parse tree*).

Os sites utilizados para este trabalho são provenientes de várias fontes diferentes, portanto a construção interna de cada HTML é única de cada fonte. Dessa forma, não é possível aplicar uma metologia única para todos os sites. Assim, a definição de uma padronização das saídas necessárias para o trabalho é de extrema importância.

Cada um dos campos de extração possui informações obrigatórias na busca pelos dados, as quais estão descritas na Tabela 1.

Tabela 1: Informações essenciais para cada campo

Oportunidades	**Notícias**
Título, link, descrição, deadline, critério de elegibilidade e data da atualização	Título, link, descrição da notícia e data da atualização
Políticas	**Projeto**
Nome da instituição, nome da política, descrição da política, link e data da atualização	Titulo, link, nome da instituição, critério de elegibilidade e data da atualização

Para cada informação em cada campo de busca existe uma <tag> dentro do HTML correspondente. Em princípio, assume-se que para o mesmo site, a <tag> será a mesma para a informação do campo.

Tendo em vista a grande quantidade de sites à serem raspados, optou-se por criar um script em Python para cada site, dentro dos quais são incluídas as funções para extração de cada campo de oportunidade. Assim, a arquitetura de raspagem inclui um código customizado para cada site e um arquivo central ppfcentral.py, no qual todos os sites considerados são diariamente raspados e as informações armazenadas. Vários outros módulos do Python são utilizados no processo (*numpy, pandas, re, ...*).

Cada um dos quatro campos de busca de interesse para o PPF possuem características e informações diferentes. Dessa forma, torna-se necessária a separação desses campos em diferentes arquivos. A forma de armazenamento escolhida foram arquivos CSV, resultando assim em até quatro arquivos CSV para cada site correspondentes aos quatro campos. Posteriormente, os arquivos CSV são agrupados em formato XLSX, formando uma base de dados com uma planilha diferente para cada campo, facilitando o abastecimento das áreas do PPF.

3.4.1 Critério de elegibilidade

Além da obtenção dos dados, um dos maiores obstáculos deste trabalho é classificar corretamente a elegibilidade das oportunidades. O critério de elegibilidade classifica se aquela oportunidade é elegível para os pesquisadores brasileiros. Esta é uma variável dicotômica de valores **Y**: Elegível para brasileiros e **N**: Não elegível para brasileiros.

Essa classificação baseia-se em palavras-chave (*keywords*) pré-definidas no texto. As principais *keywords* utilizadas para classificar a elegibilidade foram: Brasil, América Latina, América do Sul e Países de baixa ou média renda.

3.5 Análise de dados

A análise de dados envolveu processamento de linguagem natural (PLN). Para esta análise, apenas um dos campos do PPF foi utilizado, o campo de Oportunidades Abertas que é considerado o de maior importância para este trabalho e seus textos correspondem ao mesmo tipo de documento, de descrição de oportunidade e elegebilidade. O código para análise dos dados foi desenvolvido no *Jupyter Notebook* (PERKEL, 2018). Os principais módulos de Python utilizados foram: *collections, gensim, nltk, numpy, pandas, re, sklearn*.

O primeiro passo em PLN envolve o pré-processamento dos dados para sua posterior interpretação, com uso dos pacotes *pandas* e re (*regular expressions*) do Python. O código implementa uma classe com uma série de funções de pré-processamento, as quais permitem remover caracteres não-alfanuméricos, espaços múltiplos, palavras coladas, caracteres indesejados, etc.

Na sequência, utiliza-se o pacote Natural Language Toolkit - NLTK (NLTK, 2021; LOPER; BIRD, 2002; BIRD, 2006). A utilização do pacote NLTK com Python é bem documentada

(BIRD; KLEIN; LOPER, 2009). Com NLTK, implementa-se a remoção de *stopwords*, a tokenização, a lematização, *bag of words*, e o mapeamento do corpus em dicionário.

Daí, vem a codificação ou vetorização da entrada, quando se obtém o *bag of words* com dicionário e a frequência relativa das palavras (TF-IDF).

Finalmente, o aprendizado de máquina tradicional é aplicado, com mensuração da acurácia e da sensibilidade na classificação automatizada da elegibilidade de oportunidades. Para isso, aplica-se o pacote scikit-learn (SKLEARN, 2021; PEDREGOSA et al., 2011).

3.5.1 Métricas

O pacote scikit-learn disponibiliza diversas métricas. Entre essas, foram consideradas:

- ***Confusion Matrix***: por meio da matriz de confusão, calcula-se a quantidade de falsos positivos e falsos negativos, bem como de verdadeiros positivos e verdadeiros negativos, propiciando calcular acurácia e sensibilidade;
- ***Classification Report***: relatório com principais métricas de classificação.

Os relatórios de classificação incluem as seguintes métricas (SKLEARN, 2021):

- ***precision***: acurácia, que é proporção de acertos de uma determinada classe;
- ***recall***: sensibilidade, que indica a habilidade do modelo de encontrar todas as amostras "Y"ou "N";
- ***f1-score***: média harmônica entre as métricas precision e recall;
- ***support***: quantidade total de amostras pertencentes a cada classe.

Além disso, também são reportados valores de médias específicas, que são:

- ***macro avg***: média por rótulo não-ponderada;
- ***weighted avg***: média por rótulo ponderada pelo "support".

3.5.2 Divisão dos Dados

A partir da base de dados com os textos das oportunidades coletadas, após a eliminação das *stopwords* e da lematização, foi extraída uma base de treinamento correspondendo a 75% do total, restando 25% dos dados na base de teste. Além disso, por meio do parâmetro *stratify*, foi assegurada uma proporção similar de dados com rótulos *Y* e *N*, entre as bases de treinamento e teste.

3.5.3 Classificação

Para avaliar a classificação, cada uma das três metodologias consideradas (*Naive Bayes, SVM, Random Forest*) foram avaliadas considerando o pré-processamento baseado no BoW e no TF-IDF. Para cada uma das seis combinações, foram selecionados e apresentados os dois melhores resultados de acurácia.

Para obter as bases de treinamento e teste a partir do BoW, foi utilizado o módulo *CountVectorizer* do pacote *scikit-learn*, com posterior aplicação dos submódulos *fit_transform(X_train)*

e *transform(X_test)*. Para as bases a partir do TF-IDF, foi usado o módulo *TfidfVectorizer*, com posterior aplicação de sub-módulos análogos.

A classificação Naive Bayes aplica o módulo *MultinomialNB*, com parâmetros $alpha = (0,01; 0,21; 0,41; 0,61 e 0,81)$. A classificação SVM aplica o módulo SVC, avaliando diferentes *kernel (linear, rbf, sigmoid)* e diferentes valores do parâmetro de regularização $C = (0,1; 0,3; 0,6; 0,8)$. A regularização é inversamente proporcional a C (SKLEARN, 2021). A classificação Random Forest aplica o módulo *RandomForestClassifier*, com diferentes números de estimadores (5, 10, 100, 500, 1000).

3.5.4 Validação Cruzada (*Cross Validation*)

Para além dos 12 resultados analisados, foi realizada uma validação cruzada de todas as combinações possíveis (dois métodos de pré-processamento, três metodologias de classificação, diferentes opções dos algortimos) para avaliar a variabilidade e permitir a comparação entre esses. A técnica de validação cruzada utilizada é a k-fold, onde temos um número de k (folds), também chamado de número de dobras, com 50 dobras realizadas para cada método.

4 Resultados

Os métodos de *Machine Learning* foram aplicados apenas para o campo de Oportunidades. O campo em questão consiste de 200 observações referentes a oportunidades abertas. A variável de entrada é o texto que descreve a oportunidade e a variável binária de saída é a elegibilidade. O código central dos métodos de *Machine Learning* assim como a parte de *Web Scrapping* dos campos do PPF pode ser visualizado através do repositório *Github*[1].

A seguir são apresentados os resultados para cada uma das combinações, incluindo variação nos parâmetros de cada modelo.

4.1 Naive Bayes

Para o Naive Bayes, variou-se o seguinte parâmetro:

- α: parâmetro de suavização, como visto em 2.1.3. Os valores testados foram 0,01; 0,21; 0,41; 0,61 e 0,81.

Tabela 2: Melhor modelo utilizando Naive Bayes e BoW

Naive Bayes - Bag of Words - $\alpha = 0,01$ Acurácia: 82%				
Parâmetros	precision	recall	f1-score	support
N	0,71	0,83	0,77	18
Y	0,90	0,81	0,85	32
macro avg	0,81	0,82	0,81	50
weighted avg	0,83	0,82	0,82	50

[1] https://github.com/bzimons/MCTI-PPF

Tabela 3: Matriz de Confusão para Naive Bayes e BoW

		Real	
		N	Y
Previsão	N	15	3
	Y	6	26

O melhor modelo para o método Naive Bayes em *Bag of Words* foi com um parâmetro de $\alpha = 0,01$, resultando em 82% de acertos totais e 90% de verdadeiros positivos.

Tabela 4: Melhor modelo utilizando Naive Bayes e TF-IDF

Naive Bayes - TF-IDF - $\alpha = 0,01$ Acurácia: 86%				
Parâmetros	precision	recall	f1-score	support
N	0,79	0,83	0,81	18
Y	0,90	0,88	0,89	32
macro avg	0,85	0,85	0,85	50
weighted avg	0,86	0,86	0,86	50

Tabela 5: Matriz de confusão para Naive Bayes e e TF-IDF

		Real	
		N	Y
Previsão	N	15	3
	Y	4	28

O melhor modelo para o método Naive Bayes com TF-IDF também foi utilizando o parâmetro de $\alpha = 0,01$, resultando em 86% de acertos totais e 90% de verdadeiros positivos.

4.2 SVM

Para o SVM, variou-se os seguintes parâmetros:

- C: parâmetro de regularização. Os valores testados foram 0,1; 0,3; 0,6; 0,8.
- Kernel: função de transformação das amostras, como visto em 2.1.4. As funções testadas foram *sigmoid*, *rbf* e *linear*.

Tabela 6: Melhor modelo utilizando SVM e Bag of Words

SVM - Bag of Words - Kernel = linear, C = 0,1 Acurácia: 78%				
Parâmetros	precision	recall	f1-score	support
N	0,71	0,67	0,69	18
Y	0,82	0,84	0,83	32
macro avg	0,76	0,76	0,76	50
weighted avg	0,78	0,78	0,78	50

Tabela 7: Matriz de confusão para SVM e Bag of Words

		Real	
		N	Y
Previsão	N	12	6
	Y	5	27

O método SVM de kernel linear apresentou os mesmos resultados independente da variação do parâmetro C. Portanto para os valores entre $C = 0,1$ e $C = 0,8$ os resultados foram os mesmos. Esse modelo apresenta de 78% de acerto total e de 82% para verdadeiros positivos.

Tabela 8: Melhor modelo utilizando SVM e TF-IDF

SVM - TF-IDF - Kernel = sigmoid, C = 0,8 Acurácia: 82%				
Parâmetros	precision	recall	f1-score	support
N	0,80	0,67	0,73	18
Y	0,83	0,91	0,87	32
macro avg	0,81	0,79	0,80	50
weighted avg	0,82	0,82	0,82	50

Tabela 9: Matriz de confusão para SVM e TF-IDF

		Real	
		N	Y
Previsão	N	10	8
	Y	1	31

Para SVM em TF-IDF, o melhor resultado foi com o Kernel = sigmoid e C = 0,8, a classificação total foi de 82% dos resultados e de 83% de verdadeiros positivos.

4.3 Random Forest

Para o Random Forest, variou-se o seguinte parâmetro:

- n_est: número total de estimadores do *ensemble*. Os valores testados foram 5, 10, 100, 500, 1000.

Tabela 10: Melhor modelo utilizando Random Forest e TF-IDF

Random Forest - Bag of Words - n_est = 500 Acurácia: 82%				
Parâmetros	precision	recall	f1-score	support
N	0,80	0,67	0,73	18
Y	0,83	0,91	0,87	32
macro avg	0,81	0,79	0,80	50
weighted avg	0,82	0,82	0,82	50

Tabela 11: Matriz de confusão para Random Forest e Bag of Words

		Real	
		N	Y
Previsão	N	12	3
	Y	6	29

Para o modelo Random Forest em Bag of Words, o melhor resultado foi para o número de estimadores = 500, com 82% de acurácia total e verdadeiros positivos de 83%.

Tabela 12: Melhor modelo utilizando Random Forest e Bag of Words

Random Forest - TFIDF - n_est = 100 Acurácia: 84%				
Parâmetros	precision	recall	f1-score	support
N	0,86	0,67	0,65	18
Y	0,83	0,94	0,88	32
macro avg	0,85	0,80	0,82	50
weighted avg	0,84	0,84	0,83	50

Tabela 13: Matriz de confusão para Random Forest e TF-IDF

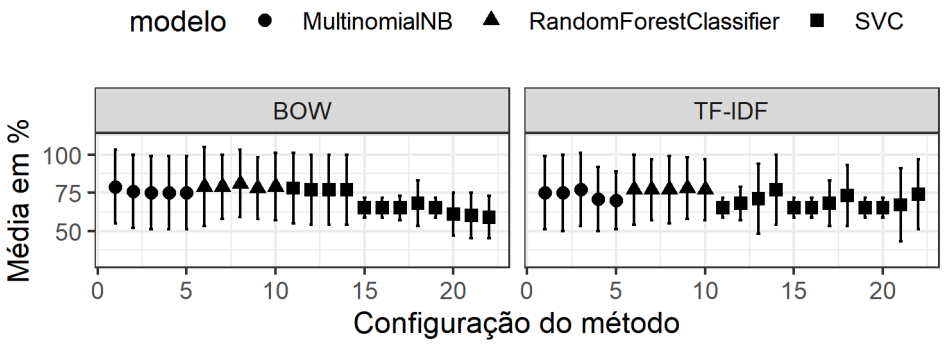
		Real	
		N	Y
Previsão	N	12	6
	Y	2	30

Para o TF-IDF, o modelo Random Forest foi o melhor com número de estimadores=100, o default do método computacional. A sua acurácia total foi de 84% e para os verdadeiros positivos, de 83%.

4.4 Validação Cruzada

A validação cruzada, apresentada na Figura 1 e na Tabela 14, destaca a variabilidade dos resultados e permite a comparação entre esses.

Figura 1: Variação das medidas



A Figura 1 representa a variação das medidas dos métodos avaliados na validação cruzada. O eixo X representa as combinações de cada método para cada tipo de processamento textual. Por exemplo, a primeira observação representa o método Naive Bayes de configuração $\alpha = 0,01$ no BoW. O eixo Y representa a média das validações feitas em porcentagem. As linhas em cada medida representa o desvio padrão.

Tabela 14: Validação Cruzada

Modelo	Bag of Words		TF-IDF	
	Média	Desvio	Média	Desvio
MultinomialNB(α=0,01)	0,79	0,24	0,75	0,24
MultinomialNB(α=0,21)	0,76	0,24	0,75	0,25
MultinomialNB(α=0,41)	0,75	0,24	0,77	0,24
MultinomialNB(α=0,61)	0,75	0,24	0,71	0,21
MultinomialNB(α=0,81)	0,75	0,24	0,7	0,19
SVC(C=0,1,k='linear')	0,78	0,23	0,65	0,065
SVC(C=0,3,k='linear')	0,77	0,23	0,68	0,11
SVC(C=0,6,k='linear')	0,77	0,23	0,71	0,23
SVC(C=0,8,k='linear')	0,77	0,23	0,77	0,23
SVC(C=0,1,k='rbf')	0,65	0,065	0,65	0,065
SVC(C=0,3,k='rbf')	0,65	0,065	0,65	0,065
SVC(C=0,6,k='rbf')	0,65	0,079	0,68	0,15
SVC(C=0,8,k='rbf')	0,68	0,15	0,73	0,2
SVC(C=0,1,k='sigmoid')	0,65	0,065	0,65	0,065
SVC(C=0,3,k='sigmoid')	0,61	0,14	0,65	0,065
SVC(C=0,6,k='sigmoid')	0,6	0,15	0,67	0,24
SVC(C=0,8,k='sigmoid')	0,59	0,14	0,74	0,23
RandomForestClassifier(n =5)	0,78	0,2	0,78	0,2
RandomForestClassifier(n=10)	0,79	0,26	0,77	0,23
RandomForestClassifier(n=100)	0,81	0,22	0,77	0,22
RandomForestClassifier(n=500)	0,79	0,22	0,77	0,2
RandomForestClassifier(n=10)	0,79	0,21	0,77	0,2

5 Discussão

Para exemplificar a classificação das palavras nos métodos, para o método *Naive Bayes* com $\alpha = 0,01$, os termos de maior peso para a classificação "N" foram números soltos como datas e valores. Já para a classificação de "Y" palavras como "*grant*", "*acceptance*', "*project*", "*fellowship*", "*research*" entre outras, apresentaram um peso maior. Esses mesmos tipos de termos são recorrentes com pequenas variações entre eles.

O melhor modelo do método *Naive Bayes* foi com o processamento TF-IDF e parâmetro $\alpha = 0,01$, que obteve 86% de acurácia. Porém, de acordo com a validação cruzada, a maior média para o pré processamento TF-IDF foi para o parâmetro $\alpha = 0,41$. Para o pré processamento *Bag of Words*, ocorreu para o $\alpha = 0,01$, o qual corresponde ao melhor método do BoW no treinamento inicial. Isso indica que, a validação do método *Naive Bayes* correspondeu ao

observado apenas para o BoW e $\alpha = 0,01$.

Para o método SVM, o modelo escolhido para o BoW de Kernel linear e C=0,1, teve uma acurácia de 78%. Enquanto que para o TF-IDF o modelo foi com o Kernel sigmoid e C=0,8, com um acurácia de 82%. Comparando os resultados com a validação cruzada na tabela 14, o método escolhido do BoW de fato apresentou a maior média entre os métodos. Para o método no TF-IDF, contudo, o modelo de maior média foi de Kernel linear e C=0,8. Portanto, o modelo selecionado para o TF-IDF de SVM não foi bem validado.

O modelo escolhido de *Random Forest* para o BoW também não foi o melhor modelo segundo a validação cruzada, na qual o modelo de maior média foi o modelo com 100 números de estimadores. Já para o modelo no TF-IDF, as médias ficaram muito próximas independente do número de estimadores, variando o desvio. Embora o modelo com 5 números de estimadores tenha uma média maior na validação cruzada, essa diferença não é necessariamente significativa, deixando claro que a classificação não está dependendo muito do número de estimadores.

6 Considerações Finais e Trabalhos Futuros

A aplicação de Ciência de Dados no projeto de Portfólio de Produtos Financeiros entregou bons resultados, os quais foram discutidos anteriormente.

Como a base utilizada tem um tamanho de 200 observações, o número de dados é relativamente pequeno para os métodos do SVM e *Random Forest*. Os pré-requisitos dos métodos não são considerados na realização dos modelos, pois o objetivo é tentar encontrar um modelo com uma acurácia que se destaque.

Pelo fato de o conjunto de dados ser considerado pequeno, os modelos estão muito suscetíveis ao *overfitting*, visto que alguns sites podem conter várias amostras com textos semelhantes, comprometendo a generalização e causando falha em amostrar não pertencentes a esses domínios. Dessa forma, nesse conjunto de dados o método de melhor resultado foi o *Naive Bayes*, pois seu ponto forte é a eficácia em tamanhos amostrais pequenos.

Das melhorias a serem feitas para trabalhos futuros, é evidente a necessidade de aumento da quantidade de dados na base. Além disso novos metodologias e abordagens podem ser estudadas com o objetivo de atingir uma meta de classificação total de 90% e de acima de 96% para os métodos de *deep learning*. Os métodos de *Machine Learning* também poderão ser aplicados nos outros campos obtidos com o *Web Scrapping* na Tabela 1, obtendo resultados como uma classificação de Notícias relevantes e projetos aprovados no Brasil.

Por fim, como objetivo principal, além dos métodos de *Machine Learning* estudados, algorítimos pré treinados para PLN poderão auxiliar na classificação dos dados, possibilitanto o desenvolvimento de uma interface interativa que envolveria um serviço de recomendação e *feedback*, assegurando o aprendizado por reforço.

Referências

AIZAWA, A. An information-theoretic perspective of tf–idf measures. *Information Processing & Management*, Elsevier, v. 39, n. 1, p. 45–65, 2003.

AUNG, W. T.; MYANMAR, Y.; HLA, K. H. M. S. Random forest classifier for multi-category classification of web pages. In: IEEE. *2009 IEEE Asia-Pacific Services Computing Conference (APSCC)*. [S.l.], 2009. p. 372–376.

BAIJENS, J.; HELMS, R.; IREN, D. Applying scrum in data science projects. In: IEEE. *2020 IEEE 22nd Conference on Business Informatics (CBI)*. [S.l.], 2020. v. 1, p. 30–38.

BIAU, G. Analysis of a random forests model. *The Journal of Machine Learning Research*, JMLR. org, v. 13, n. 1, p. 1063–1095, 2012.

BIRD, S. Nltk: the natural language toolkit. In: *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions*. [S.l.: s.n.], 2006. p. 69–72.

BIRD, S.; KLEIN, E.; LOPER, E. *Natural language processing with Python: analyzing text with the natural language toolkit*. [S.l.]: O'Reilly Media Inc, 2009.

BRASIL. *Ministério de Ciência, Tecnologia e Inovações. Portfólio de Produtos Financeiros*. 2019. Disponível em: <https://ppf.mctic.gov.br/>. Acesso em: 05 de mai. de 2021.

DOBROVOLSKYI, H.; KEBERLE, N. Collecting the seminal scientific abstracts with topic modelling, snowball sampling and citation analysis. In: *ICTERI*. [S.l.: s.n.], 2018. p. 179–192.

DREMEL, C. et al. Actualizing big data analytics affordances: A revelatory case study. *Information & Management*, Elsevier, v. 57, n. 1, p. 103121, 2020.

DRESCH, A.; LACERDA, D. P.; JÚNIOR, J. A. V. A. *Design science research: método de pesquisa para avanço da ciência e tecnologia*. [S.l.]: Bookman Editora, 2015.

FÉLIX, B. M.; TAVARES, E.; CAVALCANTE, N. W. F. Critical success factors for big data adoption in the virtual retail: Magazine luiza case study. *Revista Brasileira de Gestão de Negócios*, Centro Universitário FECAP, mantido pela Fundação Escola de Comericio ..., v. 20, n. 1, p. 112–126, 2018.

LIU, Z.; LIN, Y.; SUN, M. *Representation Learning for Natural Language Processing*. [S.l.]: Springer Nature, 2020.

LOPER, E.; BIRD, S. Nltk: The natural language toolkit. *arXiv preprint cs/0205028*, 2002.

MCKINNEY, W. et al. Data structures for statistical computing in python. In: AUSTIN, TX. *Proceedings of the 9th Python in Science Conference*. [S.l.], 2010. v. 445, p. 51–56.

MICROSOFT. *Team Data Science Process*. 2020. Disponível em: <https://docs.microsoft.com/en-us/azure/machine-learning/team-data-science-process/overview>. Acesso em: 05 de mai. de 2021.

MITCHELL, R. *Web scraping with Python: Collecting more data from the modern web*. [S.l.]: O'Reilly Media Inc, 2018.

MOE, Z. H. et al. Comparison of naive bayes and support vector machine classifiers on document classification. In: IEEE. *2018 IEEE 7th Global Conference on Consumer Electronics (GCCE)*. [S.l.], 2018. p. 466–467.

NLTK. *Natural Language Toolkit*. 2021. Disponível em: <http://www.nltk.org/>. Acesso em: 05 de mai. de 2021.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, JMLR. org, v. 12, p. 2825–2830, 2011.

PERKEL, J. M. Why jupyter is data scientists' computational notebook of choice. *Nature*, Nature Publishing Group, v. 563, n. 7732, p. 145–147, 2018.

ŘEHŮŘEK, R.; SOJKA, P. Software Framework for Topic Modelling with Large Corpora. In: *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Valletta, Malta: ELRA, 2010. p. 45–50. <http://is.muni.cz/publication/884893/en>.

RICHARDSON, L. *Beautiful soup documentation*. 2007. Disponível em: <https://www.crummy.com/software/BeautifulSoup/bs4/doc/>. Acesso em: 05 de mai. de 2021.

ROSSBERG, J. *Agile Project Management with Azure DevOps: Concepts, Templates, and Metrics*. [S.l.]: Apress, 2019.

RUSSELL, S. J.; NORVIG, P. *Artificial Intelligence (A Modern Approach)*. [S.l.]: Prentice Hall, 2010.

SANTOS, E. M. d. et al. Teoria e aplicação de support vector machines à aprendizagem e reconhecimento de objetos baseado na aparência. Universidade Federal de Campina Grande, 2002.

SCHWABER, K.; SUTHERLAND, J. Um guia definitivo para o scrum: As regras do jogo. *Processo de Desenvolvimento de Software*, 2013.

SKLEARN. *scikit-learn feature extraction text TF-IDF Transformer*. 2020. Disponível em: <https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfTransformer.html>. Acesso em: 05 de mai. de 2021.

SKLEARN. *scikit-learn Naive Bayes classification*. 2020. Disponível em: <https://scikit-learn.org/stable/modules/naive_bayes.html>. Acesso em: 05 de mai. de 2021.

SKLEARN. *Scikit-Learn: Machine Learning in Python*. 2021. Disponível em: <https://scikit-learn.org/stable/>. Acesso em: 05 de mai. de 2021.

THIOLLENT, M. *Pesquisa-ação nas organizações*. [S.l.]: Atlas, 2009.

TING, S. et al. Is naive bayes a good classifier for document classification. *International Journal of Software Engineering and Its Applications*, Citeseer, v. 5, n. 3, p. 37–46, 2011.

ZHANG, Y.; JIN, R.; ZHOU, Z.-H. Understanding bag-of-words model: a statistical framework. *International Journal of Machine Learning and Cybernetics*, Springer, v. 1, n. 1-4, p. 43–52, 2010.