Universidade de Brasília
Departamento de Estatística

Modelagem preditiva de projetos de lei da Câmara dos Deputados

Matheus Erbisti Pontes

Projeto apresentado para o Departamento de Estatística da Universidade de Brasília como parte dos requisitos necessários para obtenção do grau de Bacharel em Estatística.

Brasília
2022

Matheus Erbisti Pontes

Modelagem preditiva de projetos de lei da Câmara dos Deputados

Orientador(a): Prof. Eduardo Monteiro

Projeto apresentado para o Departamento de Estatística da Universidade de Brasília como parte dos requisitos necessários para obtenção do grau de Bacharel em Estatística.

Brasília
2022

Resumo

Dentre as quase infinitas aplicações de modelagem estatística, a predição de resultados políticos certamente se destaca como um grande desafio. Dificuldades como o volume dos dados, a imprevisibilidade de acontecimentos que transformam o mundo e o jogo político, e a complexidade da tramitações são alguns dos fatores que rapidamente trazem complicações para a construção dos modelos. Nesse sentido, os autores do presente trabalho decidiram abordar o problema sobre uma particular ótica: a predição da aprovação de projetos de lei através do texto de sua respectiva redação. Para tal, obtiveram-se conjuntos de dados contendo todas as proposições que tramitaram na Câmara dos Deputados na última decada, e seus respectivos autores e temas. Após processos de limpeza e transformação, o conjunto de dados construído conteve 17.879 projetos de lei e 40 variáveis, e deu-se início da leitura e representação dos seus textos por meio de diferentes técnicas, resultando em 36 formas diferentes de matrizes de palavras. Na modelagem, aplicou-se modelos Logísticos, LASSO, Elastic Net, Naive Bayes e de Árvores em todas as matrizes, gerando-se 168 combinações distintas de modelos capazes de predizer a aprovação de projetos de lei na Câmara dos Deputados. Após análise dos métodos aplicados e cálculo de medidas de performance, obtivemos três modelos de maior destaque, um Naive Bayes e Árvore com a presença de covariáveis, e um Naive Bayes exclusivamente composto pelo texto dos projetos de lei.

Palavras-chaves: Processo Legislativo, Estatística e Ciência de Dados, Modelagem Preditiva.

Abstract

Among the near infinite applications of statistical modeling, the prediction of political results is most certainly a great challenge. Difficulties such as data volume, unpredictability of events that can transform the world and its politics, and complexity of the legislative process are just a few factors that can quickly cause issues to model construction. In this sense, the authors decided to address the problem on a specific point of view: approval prediction of law bills by their respective text. As such, several datasets related to all propositions processed on the Chamber of Deputies in the last decade were obtained, as well as their respectives authors and themes. After several cleansings and transformations, the constructed dataset ended up with 17.879 bills and 40 variables, starting then the text scraping and representation phases, which lead up to 36 different word matrices. On the modeling subject, the applied models on all matrices were Logistic, LASSO, Elastic Net, Naive Bayes and Trees, generating 168 distinct combinations of models capable of predicting the approval of law bills in the Chamber of Deputies. After the implemented techniques and performance metrics analysis, three featured models were selected, a Naive Bayes and Tree with covariables, and a Naive Bayes entirely made by the law bills text.

Keywords: Legislative Process, Statistics and Data Science, Predictive Modeling.

Sumário

1 Introdução 4

2 Metodologia 5

2.1 Conjuntos e harmonização de dados 5

2.2 Processamento de Linguagem Natural 9

2.2.1 Obtenção do texto 9

2.2.2 Limpeza Textual 10

2.2.3 Pré-Processamento 11

2.2.4 Representação de texto 13

2.3 Modelagem 19

2.3.1 Modelos de Regressão 19

2.3.2 Modelos Naive Bayes 23

2.3.3 Modelos de Árvore de Classificação 23

2.3.4 Métricas de diagnóstico e performance 25

3 Resultados 31

3.1 Análise Exploratória Inicial 31

3.2 Matrizes de Palavras 35

3.3 Modelos 40

3.3.1 Produção primária geral 40

3.3.2 Estudo das técnicas de matrizes 43

3.3.3 Análise dos melhores modelos 48

4 Conclusão 59

4.1 Sugestões de trabalhos futuros e outras abordagens 61

1 Introdução

De acordo com Montesquieu (2010), toda formação de Estado possui três formas de poder: o que cria as leis (legisla), o que as executa e aplica, e o que julga o exercício das mesmas. Nos regimes políticos ocidentais contemporâneos, é frequentemente observada a tripartição desses poderes em entidades do poder Legislativo, Executivo e Judiciário, que juntos formam a união dos poderes no Estado. Dessa forma, é importante compreender os processos políticos que regimentam nossas vidas, pois definem nossos direitos e deveres como cidadãos; afinal, segundo frase (ou variações da mesma) comumente associada à Platão, "o castigo dos bons que não fazem política, é serem governados pelos maus que a fazem".

No Brasil, a Constituição Federal de 1988 (República Federativa do Brasil, 1988) define no artigo 44 que o poder Legislativo é composto pelo Congresso Nacional, composto pela Câmara dos Deputados e Senador Federal. Porém, conforme (EIRÃO, 2018) explica, existem circustâncias especiais em que os poderes Executivo e Judiciário também podem propor leis. Ainda assim, o segundo artigo da Constituição (República Federativa do Brasil, 1988) dita a união harmônica dos poderes, isso é, o Legislativo possui capacidade de interferência nessas leis.

Dentre os agentes propositores de lei, destaca-se a Câmara dos Deputados, que é naturalmente um dos mais influentes no que se refere a atos legislativos. Segundo (OLIVEIRA, 2019), o número de novos projetos de lei ordinária (PL) propostos pela Câmara é de cerca de 1,6 mil por ano, porém o curioso é que apenas aproximadamente 10% dos mesmos são aprovados. Em um estudo parecido, (EIRÃO, 2016) argumenta que um dos principais motivos para a rejeição da maioria dos projetos são problemas recorrentes sobre questões da redação e incoerências nos comandos dos PLs.

Tendo em mente tudo o que foi exposto, o presente trabalho e seus autores compreendem a importância do entendimento sobre o processo legislativo e buscam, através de técnicas estatísticas e computacionais, produzir um modelo preditivo que aponte de maneira mais precisa e acurada possível a aprovação ou rejeição de um projeto de lei, a fim de contribuir para uma maior eficiência na produção de leis.

2 Metodologia

Naturalmente, o primeiro passo de uma análise estatística é obtenção dos bancos de dados. No contexto do presente estudo, a aquisição foi realizada através da Interface de Programação de Aplicações (em inglês, API) da Câmara dos Deputados. Seguindo as orientações de utilização da API, optou-se por extrair arquivos relativos a todas as proposições apresentadas na Câmara entre 2010 e 2020 (ambos inclusive), junto com seus respectivos temas e autores.

O desafio inaugural do projeto é que todos os conjuntos de dados se encontram divididos por ano, além da divisão já comentada entre proposições, temas e autores. Isso significa que precisou-se obter no total 33 bancos de dados, que ao final seriam todos unidos em apenas um. Todos os arquivos originais são de formato "valores separados por vírgula" (CSV), então a estratégia aplicada foi de conseguir e unir imediatamente os bancos de anos diferentes e de mesma estrutura. Isso é, conseguiu-se três arquivos CSV, cada um contendo as proposições, temas e autores de todas as proposições que tramitaram na Câmara dos Deputados do ínicio de 2010 até o fim de 2020.

2.1 Conjuntos e harmonização de dados

Abaixo, encontram-se tabelas dos três bancos de dados iniciais, com suas respectivas variáveis e descrição:

Tabela 1: Proposições 1

Proposições	
Variável	Descrição
id	Número de identificação das observações.
uri	Contém o link das informações do respectivo projeto em xml.
siglaTipo	Sigla identificadora do tipo de proposição
numero	Número do projeto associado.
ano	Ano no qual a proposta foi apresentada.
codTipo	Código númerico relativo ao tipo da proposição.
descricaoTipo	Tipo da proposição escrito por extenso.
ementa	Resumo descritivo da proposta.
ementaDetalhada	Texto ligeiramente mais elaborado da ementa
keywords	Palavras-chave do projeto.
dataApresentacao	Data e hora de apresentação inicial da proposição.
uriOrgaoNumerador	Link para informações sobre o órgão numerador do projeto em xml.
uriPropAnterior	Link com informações em xml sobre a proposta anterior do projeto.
uriPropPrincipal	Link com informações em xml sobre a proposta principal do projeto.

Tabela 2: Proposições 2

Proposições	
Variável	Descrição
uriPropPosterior	Link com informações da proposição no Senado.
urlInteiroTeor	Link direcionado ao arquivo pdf com todo o conteúdo da proposta.
urnFinal	Variável antiga sem propósito nos bancos utilizados.
ultimoStatus_dataHora	Data e hora da última atualização de status
ultimoStatus_sequencia	Variável ordinal que representa a sequência de tramitação
ultimoStatus_uriRelator	Link que contém informações em xml sobre o relator do projeto
ultimoStatus_idOrgao	Código numérico do órgão no qual a proposta se encontra
ultimoStatus_siglaOrgao	Sigla do nome do órgão no qual a proposta se encontra
ultimoStatus_uriOrgao	Link com informações em xml sobre o órgão que a proposta se encontra
ultimoStatus_regime	Regime sobre o qual a proposta tramita
ultimoStatus_descricaoTramitacao	Breve descrição sobre o última tramitação da proposição
ultimoStatus_idTipoTramitacao	Código númerico correspondente à última tramitação
ultimoStatus_descricaoSituacao	Atual situação do projeto
ultimoStatus_idSituacao	Código númerico correspondente à atual situação da proposição
ultimoStatus_despacho	Última ação despachada pelo presidente da Câmara sobre a proposta
ultimoStatus_url	Link do arquivo pdf do diário da Câmara na página referente ao projeto

Tabela 3: Autores

Autores	
Variável	Descrição
id	Número de identificação da observação
uriProposicao	Link com informações em xml sobre o projeto
idDeputadoAutor	Código númerico relativo ao deputador autor do projeto
uriAutor	Link que contém informações em xml sobre o autor da proposta
codTipoAutor	Código numérico do tipo do autor
tipoAutor	Classificação de tipo de autor
nomeAutor	Nome completo do autor
siglaPartidoAutor	Sigla do partido do qual o autor é afiliado
siglaUFAutor	Sigla do estado no qual o deputado foi eleito
ordemAssinatura	Ordem de posição da assinatura do deputado no projeto
proponente	Variável indicadora caso haja um agente proponente da proposta

Tabela 4: Temas

Temas	
Variável	Descrição
id	Número de identificação da observação
uriProposicao	Link com informações em xml sobre o projeto
siglaTipo	Sigla referente ao tipo de proposição
numero	Número associado à proposta
ano	Ano de apresentação da proposta
codTema	Código numérico associado ao tema do projeto
tema	Classificação temática do projeto

Imediatamente, seguiu-se uma análise dos três bancos, revelando os seguintes problemas:

- Muitas variáveis tiveram problema de codificação. Uma simples e repetitiva conversão solucionou esse item;
- Variáveis de data e hora não foram reconhecidas adequadamente. Uma simples conversão também resolveu essa questão;
- "uriPropAnterior" e "urnFinal" são variáveis defasadas e contém apenas valores faltantes. Logo, resolveu-se por remover ambas as variáveis;
- O "id" discrimina, por exemplo, observações com mesma proposição, mas com temas diferentes. Isso significa que seu funcionamento é como uma mera variável de contagem, e não apresenta nenhuma informação útil. Portanto, ela foi retirada de todos os bancos;
- O número dos projetos zera a cada ano, o que significa que todos os anos existe um novo PL 0001, por exemplo. Para facilitar a identificação, criou-se uma nova variável "numProjeto", combinando o número da proposição, uma barra "/" e os dois últimos dígitos do ano de sua apresentação;
- Existem muitos partidos políticos antigos que trocaram de nome ou se fundiram com outros. Através de pesquisa e histórico individual de cada partido, transformou-se todas as observações para as siglas mais atuais;
- Caso um projeto tenha mais de um tema ou autor, ele terá todas as suas informações duplicadas nos respectivos bancos, alterando apenas a respectiva variável. Como queremos que o modelo receba proposições únicas, precisou-se unir as observações repetidas em uma mesma linha, separadas por vírgulas.

Dado o processo de harmonização inicial dos dados, uniu-se os três bancos em apenas um só. Entretanto, essa união foi feita de maneira a produzir dois conjuntos de dados com todas as variáveis descritas e suas respectivas observações. São eles:

- **banco_pl** - Corresponde a um conjunto de 40 variáveis e 17.884 observações, no qual cada valor representa um único Projeto de Lei com seus autores, partidos e demais valores múltiplos agrupados.
- **banco_dup_pl** - Corresponde a um conjunto de 40 variáveis e 68.792 observações. Caso um PL possua vários autores ou classificações temáticas, ele aparecerá mais de uma vez, alterando apenas os respectivos valores múltiplos.

Ainda que pareça contraintuitivo manter um banco de dados com valores replicados, já que o modelo precisa de valores únicos de proposições, fez-se necessário manter um conjunto dessa forma, cujo motivo é para as análises exploratórias. Por exemplo, considere as observações de autor "Deputado A, Deputado B" e "Deputado B, Deputado A"; em uma construção de gráficos, seria contado ambos os casos de autores como única, enquanto que o correto seria contabilizar os dois individualmente. Além disso, tal problema também prejudicaria e impossibilitaria a visualização apropriada dos dados, pois seriam muitas combinações distintas para observar em um gráfico, em que muitas são virtualmente a mesma.

Por fim, deve-se notificar que ambos os bancos são analogamente o mesmo, pois possuem as mesmas observações, diferindo apenas na organização das mesmas.

Dentre as variáveis mais relevantes, temos:

- urlInteiroTeor - Responsável pela obtenção da redação completa do Projeto de Lei, que será utilizada para processamento de linguagem natural;
- ultimoStatus_descricaoSituacao - Como ela representa a situação atual do PL, será importantíssima para delimitar qual será o ponto da tramitação que o modelo irá predizer;
- ano - Variável representativa do ano em que o projeto de lei foi apresentado. Existe a possibilidade de anos específicos terem influência na aprovação, como em anos de eleição;
- siglaPartidoAutor - Dado que existem partidos mais influenciadores do que outros na Câmara, espera-se que essa variável seja significativa no modelo;
- tema - É possível que a Câmara favoreça alguns temas mais do que outros, logo, essa variável pode ser importante para o modelo.

A partir deste ponto, fez-se necessário mais uma etapa de exploração dos dados e pesquisa sobre o funcionamento e terminologia dos projetos de lei, a fim de definir exatamente a variável predita na etapa de modelagem. Dentre todas as alternativas, descobriu-se então que a distribuição dos dados favorecia a escolha de projetos que tenham sido aprovados na Câmara e, dessa forma, apresentando-se como a configuração ideal para a predição dos modelos.

Portanto, foi criado uma nova variável dicotômica e indicativa, que recebe valores "1" se o projeto tenha sido aprovado na Câmara (ignorando a maneira de sua aprovação), e "0" caso contrário. Equivalentemente, podemos escrever:

$$I_{\text{PL aprovado na Câmara}} = \begin{cases} 1, \text{Se o projeto de Lei foi aprovado na Câmara dos Deputados} \\ 0, \text{Caso contrário} \end{cases}$$

Note que a aprovação de um projeto na Câmara não significa de fato que o PL se tornou norma jurídica de fato, podendo posteriormente ser ou ter sido rejeitado pelo Senado Federal, ou também vetado pelo Presidente da República. Dessa maneira, o modelo final desenvolvido predirá apenas a aprovação ou arquivamentos de projetos de lei na Câmara dos Deputados.

2.2 Processamento de Linguagem Natural

O Processamento de Linguagem Natural (PLN ou NLP em inglês) é uma área da ciência da computação que consiste em métodos de compreensão, modelagem e análise de textos escritos em línguas humanas através do uso de máquinas. Ela faz parte do campo de Inteligência Artificial, juntamente com o Aprendizado de Máquina e *Deep Learning*.

Embora o PLN possua diversas aplicações, desde as mais simples como corretores de texto até as complicadas assistentes virtuais e "chatbots", a aplicação de interesse é para a predição do modelo de Projetos de Lei. Portanto, temos aqui apenas os subtópicos convenientes e que já foram estudados e implementados ao presente trabalho.

2.2.1 Obtenção do texto

O primeiro passo de todo PLN é justamente conseguir o texto a ser lido. Nessa etapa constituiu-se de:

1. Ler todas as observações da variável "urlInteiroTeor" e transformá-las do formato pdf para texto através do pacote *pdftools* 3.0.1 do R 4.1.0;
2. Para os PLs que possuem o documento com várias páginas, juntá-las em apenas uma única linha textual;
3. Calcular uma nova variável contendo o número original de páginas do documento;

Então, construiu-se o banco de dados "prop_texto" de 17.879 observações e 2 variáveis, cada uma contendo individualmente a redação do PL e o respectivo número de páginas original. Perceba que esse mesmo banco possui 5 observações a menos que o banco original "banco_pl", pois os respectivos projetos possuíam um documento totalmente em

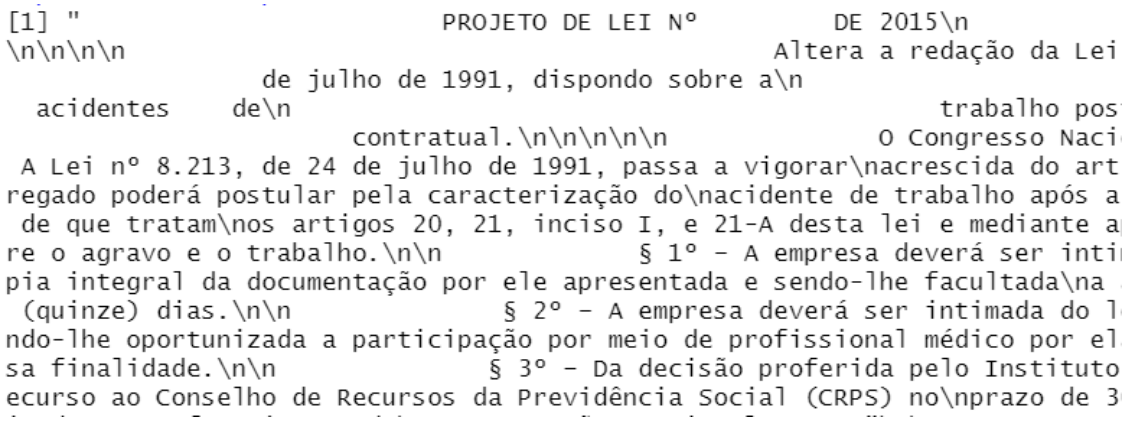
branco como seu inteiro teor, ocasionando erros no algoritmo. Sendo assim, essas 5 observações problemáticas foram então retiradas de todos os bancos.

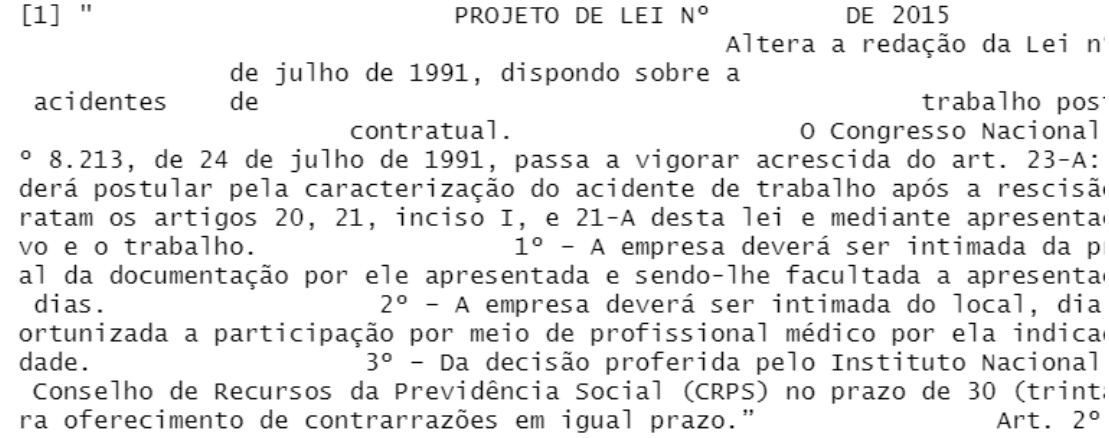
2.2.2 Limpeza Textual

A seguinte etapa do processamento de linguagem natural é a chamada "limpeza textual". Sendo assim, essa parte consiste da remoção dos seguintes:

- Remoção de caracteres especiais;
- Espaços em branco;
- Erros de digitação.

Para efeito ilustrativo, compare as imagens diretamente abaixo e perceba a remoção de "\n" e "§":





Por fim, como a base de dados compõe-se de documentos formulados e revisados pela alta classe política brasileira, espera-se que existam poucos erros de escrita.

Dessa maneira, não se faz necessária uma correção dos mesmos pois, devido à sua baixa frequência, serão eventualmente ignorados pelo modelo final.

2.2.3 Pré-Processamento

Em uma primeira vista, parece contra-intuitivo realizar uma etapa de pré-processamento dado que uma limpeza do texto acabou de ser realizada. Contudo, a parte descrita na subseção anterior é realizada justamente para permitir que o pré-processamento ocorra perfeitamente. Sendo assim, existem os seguintes pontos dessa fase:

1. **Segmentação de frases**
   O primeiro passo do pré-processamento é dividir o texto em frases, normalmente detectado pela presença de pontuação final, exclamativa ou de interrogação. Para tal, bastou empregar a função *corpus()* do pacote *quanteda* nos dados de texto.

2. **"Tokenização" de palavras**
   Após a segmentação do texto em frases, é então realizada a separação das frases em palavras, numeração, pontuação, etc. Por exemplo, a frase
   "PROJETO DE LEI Nº DE 2015 (Do Sr. Jorge Côrte Real) Altera a redação da Lei nº 8.213, de 24 de julho de 1991, dispondo sobre a caracterização dos acidentes de trabalho posteriormente à rescisão contratual."
   se torna:

"PROJETO"	"DE"	"LEI"
"Nº"	"DE"	"2015"
"("	"Do"	"Sr"
"."	"Jorge"	"Côrte"
"Real"	")"	"Altera"
"a"	"redação"	"da"
"Lei"	"nº"	"8.213"
","	"de"	"24"
"de"	"julho"	"de"
"1991"	","	"dispondo"
"sobre"	"a"	"caracterização"
"dos"	"acidentes"	"de"
"trabalho"	"posteriormente"	"à"
"rescisão"	"contratual"	"."

   Tal resultado foi obtido combinando o resultado da função *corpus()* com *tokens()*.

3. **Remoção de pontuações e "palavras vazias"**
   Dada a "tokenização" do texto, podemos então remover as partes que não interessam para modelagem e análise de PLN. Além de símbolos pontuais, excluímos também as "palavras vazias" (*stop words*, em inglês), que são aquelas que não carregam significado ou informação por si própria. Alguns exemplos de palavras vazias são: "são", "era", "eles", "terão", etc.

Entretanto, vale notar que não existe um único conjunto universal de palavras vazias, pois cada texto possui diferentes cenários. Para ilustração, a palavra "Brasília" será uma palavra vazia para o presente trabalho, dado que é o local de produção do PL e aparecerá em todas as redações. Entretanto, caso tivéssemos utilizando PLN em um estudo em que é importante descobrir o local de moradia de um indivíduo, "Brasília" não seria considerada uma palavra vazia.

No caso, a função *tokens()* possui a opção para remover automaticamente a pontuação e números, com as restantes palavras vazias removidas através de *tokens_select()*, utilizando o parâmetro "remove" no argumento *pattern*. Abaixo, encontra-se a coleção de todas as 255 palavras e caracteres vazios retirados de forma manual via *tokens_select()*:

de, a, o, que, e, do, da, em, um, para, com, não, uma, os, no, se, na, por, mais, as, dos, como, mas, ao, ele, das, à, seu, sua, ou, quando, muito, nos, já, eu, também, só, pelo, pela, até, isso, ela, entre, depois, sem, mesmo, aos, seus, quem, nas, me, esse, eles, você, essa, num, nem, suas, meu, às, minha, numa, pelos, elas, qual, nós, lhe, deles, essas, esses, pelas, este, dele, tu, te, vocês, vos, lhes, meus, minhas, teu, tua, teus, tuas, nosso, nossa, nossos, nossas, dela, delas, esta, estes, estas, aquele, aquela, aqueles, aquelas, isto, aquilo, estou, está, estamos, estão, estive, esteve, estivemos, estiveram, estava, estávamos, estavam, estivera, estivéramos, esteja, estejamos, estejam, estivesse, estivéssemos, estivessem, estiver, estivermos, estiverem, hei, há, havemos, hão, houve, houvemos, houveram, houvera, houvéramos, haja, hajamos, hajam, houvesse, houvéssemos, houvessem, houver, houvermos, houverem, houverei, houverá, houveremos, houverão, houveria, houveríamos, houveriam, sou, somos, são, era, éramos, eram, fui, foi, fomos, foram, fora, fôramos, seja, sejamos, sejam, fosse, fôssemos, fossem, for, formos, forem, serei, será, seremos, serão, seria, seríamos, seriam, tenho, tem, temos, tém, tinha, tínhamos, tinham, tive, teve, tivemos, tiveram, tivera, tivéramos, tenha, tenhamos, tenham, tivesse, tivéssemos, tivessem, tiver, tivermos, tiverem, terei, terá, teremos, terão, teria, teríamos, teriam, +, $, ¡, ¿, é, u, n, r, b, lei, art, ser, sobre, congresso nacional, i, ii, iii, iv, v, vi, vii, ix, x, janeiro, fevereiro, março, abril, maio, junho, julho, agosto, setembro, outubro, novembro, dezembro, deputado, senhor, federal, excelentíssimo, excelentíssima, excelência, sala, sessões, vossa, parágrafo, congresso, artigo, projeto, deputados, câmara, brasília, sr.

4. ***Lowercasing, Lemmatization*** **e Stemização**
   Por fim, o último passo do pré-processamento resume-se em métodos de agrupar

palavras parecidas, com o intuito de facilitar a modelagem de PLN. São esses:

- *Lowercasing* - Colocar todas as palavras restantes em letras minúsculas, a fim de permitir o computador reconhecer a equivalência das palavras de maneira mais fácil.
- Stemização - Do inglês *"Stemming"*, é o processo de transformar as palavras no seu significado mais básico possível. Por exemplo, a palavra "computadores" pode ser reduzida a "computador", "rodando" para "rodar", etc.
- *Lemmatization* - Corresponde a técnica de transformar todas as palavras de mesmo "lema" em uma em comum. Exemplificando, uma palavra como "ave" seria substituída para "pássaro", "pior" para "ruim", etc.

Ainda assim, optou-se por não implementar a stemização e *Lemmatization*, devido ao fato de ambas prejudicarem a interpretação das palavras do modelo, e não terem apresentado desempenho superior frente as suas respectivas ausências. Dessa maneira, apenas o *lowercasing* foi utilizado, sendo empregado automaticamente no *corpus()*.

2.2.4 Representação de texto

Sequencialmente ao pré-processamento, a etapa de representação de texto se vê necessário devido a dificuldade de compreensão que computadores possuem com linguagens naturais. Nesse contexto, produziram-se diversas maneiras de representar texto natural em formato numérico, permitindo o uso simplificado de máquinas para PLN; porém, para essa subseção, o foco principal são as técnicas utilizadas no desenvolvimento do presente projeto. Imediatamente abaixo, considere os três textos que irão exemplificar os métodos descritos:

- **Texto 1 -** Processamento de linguagem natural é complexo, mas muito legal!
- **Texto 2 -** A linguagem de algumas pessoas é naturalmente complexa.
- **Texto 3 -** O processamento é muito complexo, e requer muitas pessoas.

Para todos os exemplos, assuma o pré-processamento dos textos conforme explicado na subseção 3.2.3.

1. **Sacola de Palavras**
   Comumente conhecida como *Bag of Words*, a "Sacola de Palavras" é um método muito utilizado em problemas de classificação de texto. Metodologicamente, a

técnica consiste em construir um vetor com todas as palavras de todos os textos e, após isso, construir vetores numéricos com a contagem de frequência de cada palavra em cada texto. Ao fim, todos os vetores são empilhados horizontalmente, formando uma matriz. Abaixo, encontra-se o exemplo visual dessa matriz:

Tabela 5: Sacola de Palavras

Textos	processamento	linguagem	natural	complexo	muito	legal	algum	pessoa
Texto 1	1	1	1	1	1	1	0	0
Texto 2	0	1	1	1	0	0	1	0
Texto 3	1	0	0	1	2	0	0	1

Superficialmente explicando, a ideia por trás da Sacola de Palavras é que a classificação de um texto pode ser caracterizado devido a um conjunto específico de palavras; dessa forma, é possível analisar as palavras presentes no texto, e reconhecer sua classe. É de simples aplicação, sendo necessário apenas suprir a função *dfm()* com o resultado de *tokens()*.

Por fim, note que a Sacola de Palavras ignora qualquer tipo de ordem ou frases do texto original. Embora isso não afete o algoritmo de classificação de interesse desse projeto, esse motivo torna o método contra-indicado em outras circunstâncias, por exemplo para análise de sentimento do texto.

2. **Sacola de *n-grams***

Muito similar à Sacola de Palavras, uma *Bag of n-grams* é uma tentativa de preservar a ordem de palavras do texto, enquanto mantém a estrutura básica do primeiro método. Dessa forma, ao invés de separar o texto por palavras, a matriz é construída contabilizando os chamados "*n-grams*", que por sua vez são agrupamentos de tamanho arbitrário "n" das palavras. Por exemplo, se formos considerar uma matriz de 2-grams, também chamada de n-grams 2 e bigrama, obtemos:

Tabela 6: Sacola de *n-grams* 2

Textos	processamento_linguagem	linguagem_natural	...	muito_pessoa
Texto 1	1	1	...	0
Texto 2	0	0	...	0
Texto 3	0	0	...	1

Também é possível construir uma matriz em que "n" é um vetor, ao invés de apenas um número. Então, considere um (1,2)-grams, ou n-grams 1-2 abaixo:

Tabela 7: Sacola de *n-grams* 1 e 2

Textos	processamento	linguagem	...	muito_complexo	muito_pessoa
Texto 1	1	1	...	0	0
Texto 2	0	1	...	0	0
Texto 3	1	0	...	1	1

Dessa forma, perceba que uma Sacola de Palavras nada mais é do que um caso particular da Sacola de *n-grams*, em que n é igual a 1. Portanto, sua execução é análoga, requerindo apenas combinar os *tokens* obtidos com *tokens_ngrams()* com o "n" desejado antes de chamar *dfm()*.

3. ***TF-IDF***

Nos métodos descritos anteriormente, a matriz final apresenta apenas a contagem do termo, sem considerar sua relevância geral ao texto individual, e no conjunto de dados. Dessa maneira, o *Term Frequency-Inverse Document Frequency* ou TF-IDF (em tradução livre, "Frequência do Termo-Inverso da Frequência de Documento") se diferencia dos demais ao apresentar essas informações na matriz.

Conforme o nome sugere, o TF-IDF combina duas métricas matemáticas para atingir seu objetivo final. A primeira parte consiste da frequência do termo, comumente calculada pela fórmula:

$$TF(i,d) = \frac{\text{Quantidade de vezes em que a palavra i ocorre no documento d}}{\text{Quantidade palavras no documento d}}$$

Enquanto que o Inverso da Frequência de Documento é dado por:

$$IDF(i) = \log\left(\frac{\text{Quantidade de documentos no banco de dados}}{\text{Número de documentos que possuem a palavra i}}\right)$$

É importante ressaltar que ambos os índices podem ser calculados de outras formas; ainda assim, as maneiras aqui descritas são as mais comuns e as utilizadas no desenvolvimento do presente estudo. Obtido ambos os índices, o TF-IDF se dá por:

$$TF\text{-}IDF = TF \times IDF$$

Assim sendo, considere abaixo os exemplos de cálculo do TF-IDF de algumas das palavras do Texto 1:

$$TF_{\text{processamento}} = \frac{1}{6} = 0.167$$

$$IDF_{\text{processamento}} = \log\left(\frac{3}{2}\right) = 0.176$$
$$TF\text{-}IDF_{\text{processamento}} = 0.167 \times 0.176 = 0.0293$$

$$TF_{\text{complexo}} = \frac{1}{6} = 0.167$$
$$IDF_{\text{complexo}} = \log\left(\frac{3}{3}\right) = 0$$
$$TF\text{-}IDF_{\text{complexo}} = 0.167 \times 0 = 0$$

Note que devido à sua presença em todos os textos, o TF-IDF da palavra "complexo" foi zerado, em contraste com o da palavra "processamento". Dessa forma, o TF-IDF de uma palavra em um texto só será zero se ela não aparecer no mesmo, ou caso apareça em todos os textos.

Abaixo encontra-se a matriz TF-IDF de exemplo:

Tabela 8: TF-IDF

Textos	processamento	linguagem	natural	complexo	muito	legal	algum	pessoa
Texto 1	0.0293	0.0293	0.0293	0	0.0293	0.0795	0	0
Texto 2	0	0.0352	0.0352	0	0	0	0.0954	0.0352
Texto 3	0.0352	0	0	0	0.0704	0	0	0.0352

Perceba que a matriz da Tabela 8 foi construída usando um n-grams 1, porém o uso de n-grams com outros valores também é facilmente implementado. No contexto do presente estudo, tais resultados são obtidos suprindo a matriz formada por *dfm()* para a função *dfm_tfidf()* com o argumento *scheme_tf* igual a "prop".

4. **Problemas gerais dos métodos descritos**

Na construção das matrizes descritas na seção 3.2.4, existem dois problemas muito comuns: esparsidade elevada e uso excessivo de memória do computador. Felizmente, o primeiro costuma ser a causa do segundo, e existem soluções para ambos.

Esparsidade é uma taxa percentual representativa da quantidade de observações iguais a zero em relação ao número total de observações de uma matriz. Abaixo, localiza-se uma tabela as esparsidades das matrizes das Tabelas 5, 6, 7 e 8:

Tabela 9: Esparsidade das matrizes

Tabela	Método	Esparsidade
Tabela 5	Sacola de Palavras	37.5%
Tabela 5	Sacola de *2-grams*	60.61%
Tabela 6	Sacola de *(1-2)-grams*	50.88%
Tabela 7	TF-IDF*	37.5%*

O cálculo da esparsidade de uma matriz TF-IDF é feito com a frequência original das palavras*

Observando a tabela 9, observa-se que é natural que uma matriz dos métodos descritos apresenta boa parte de suas observações iguais a zero, principalmente caso os textos sejam maiores, e tenham palavras bem distintas entre si.

Isso é especialmente agravante no método dos *n-grams*, pois embora resolva a perda de ordem e fraseamento da Sacola de Palavras, o preço a se pagar ocorre no aumento da esparsidade. Conforme "n" cresce, o número de colunas da matriz aumenta muito, dificultando as chances de observar-se aquele conjunto específico de palavras nos outros textos, o que se traduz em uma matriz maior e com muito mais zeros.

Por sua vez, uma matriz grande demais causa um volume gigantesco de cálculos para o computador, o que pode ocasionar uso excessivo e insuficiência de memória.

Portanto, sabemos que uma matriz muito grande e que seja muito esparsa causa problemas computacionais e possui muitas informações desnecessárias. Então, a solução escolhida foi a filtragem das matrizes, com intuito de diminuir seu tamanho. Mas antes, vale mencionar também que uma alternativa viável e não implementada para isso seria a utilização de outras formas de representação de texto, como *Word2Vec*.

Retornando à filtragem, existem duas maneiras de aplicá-la:

- Removendo palavras com frequência em todo os textos abaixo ou acima de um número arbitrário.
- Removendo palavras que aparecem em menos ou mais do que um número arbitrário de documentos.

Note que é possível realizar qualquer combinação possível dos parâmetros dos filtros, inclusive a utilização de ambas técnicas juntas, e filtrando conjuntamente acima e abaixo de uma frequência arbitrária escolhida. Relativo ao presente trabalho, escolheu-se gerar matrizes com frequência mínima das palavras de 10% e 20% e frequência máxima de 90% e 80%, respectivamente. Em ambos os casos, também removeu-se palavras que apareceram menos do que 100 vezes em todos os 17.884 projetos analisados, sendo necessário apenas especificar os argumentos *min_docfreq*,

max_docfreq, *doc_freqtype* e *min_termfreq* para a função *dfm_trim()* ao supri-la com uma matriz de palavras de *dfm()*.

5. **Implementação das covariáveis**

   Como a estrutura das matrizes providas aos modelos é completamente diferente do visto na seção 2.1, não seria possível simplesmente adicionar todas as covariáveis no modelo. Com o intuito de resolver esse problema, fez-se crucial uma transformação dos dados das covariáveis, acomodando-as para o formato de uma matriz de palavras.

   Sendo assim, a estratégia adotada foi contabilizar cada valor possível das covariáveis como sendo uma palavra presente no texto. A fim de ilustração, considere os fictícios projetos de lei:

   - PL 0001, de autor "João Ninguém", oriundo de SP e temas "Saúde" e "Comunicações";
   - PL 0002, de autores "João Ninguém" e "Maria Qualquer", oriundos de SP, com tema de "Saúde".

   Então, as observações das variáveis de ambos os projetos ficariam no formato de matriz de palavras da seguinte forma:

Tabela 10: Covariáveis

Textos	JoãoNinguém	MariaQualquer	SP	Saúde	Comunicações
PL 0001	1	0	1	1	1
PL 0002	1	1	2	1	0

   Dessa forma, foram incluídas as variáveis dos autores, suas respectivas UFs e partidos, junto com o tema, ano e regime de vigência dos projetos. Nessa mesma etapa, também foi criada a contagem das quatro primeiros variáveis, totalizando em 10 covariáveis acomodadas no formato de matriz de palavras.

   Ao fim, o processo gerou uma matriz de covariáveis com todos os 17879 projetos e 1660 colunas, que encontra-se representada por uma pequena amostra diretamente abaixo:

Tabela 11: Amostra das covariáveis

Textos	JorgeCôrteReal	AugustoCoutinho	...	Especial (Arts. 142 e 143, RCCN)	RegimeUrgência
PL 0001	1	1	...	0	0
PL 0002	1	1	...	0	0
PL 0003	0	0	...	0	0

2.3 Modelagem

No cerne de sua motivação, o presente estudo possui como objetivo final a predição da aprovação ou arquivamento de projetos de lei na Câmara dos Deputados. Para tal, o caminho natural é através de modelos estatísticos.

Por sua vez, um modelo estatístico é um modelo matemático que assume um certo conjunto de propriedades estatísticas em relação aos dados analisados. Dessa forma, o modelo estatístico representa a relação matemática entre uma ou mais variáveis aleatórias do banco de dados, podendo ser utilizado para a predição ou previsão para dados futuros e, com cautela, definir relações causais entre as variáveis.

Dentre as formas de implementação na modelagem estatística, uma prática comum é a separação do conjunto de dados completo em "dados de treinamento" maior e "dados de teste" menor; isso é, treinamos o modelo de interesse com o primeiro subconjunto, e então o utilizamos para predizer os resultados do segundo. Tal procedimento é empregado a fim de evitar e identificar modelos muitos viciados em suas bases iniciais, e será empregado em toda a modelagem desse estudo. Apesar disso, também temos que levar em consideração que a proporção de projetos aprovados é muito inferior que a dos arquivados, o que muito provavelmente causará vieses a favor do arquivamento nos modelos, mesmo com a separação do banco de dados inicial.

Normalmente, a separação dos subconjuntos de treinamento e teste são feitos de maneira a valorizar mais o de treino, contendo 95% ou 90% do total através de amostragem aleatória sem reposição. No entanto, realizou-se a divisão no vigente estudo com os números arbitrários de 70% (12516 observações) dos dados para treinamento, e o restantes 30% (5364 observações) para teste.

No caso desse projeto, utilizamos diversos tipos de modelos estatísticos apropriados; combinados com todos os tratamentos diferentes descritos na seção 2.2 e algumas das variáveis dos bancos de dados na seção 2.1, a fim de obter o modelo que melhor prediz a aprovação de projetos de lei. Além disso, produziu-se também um modelo "aleatório", cuja métricas derivam da média de 5 modelos de classificação totalmente aleatória, a fim de comparação do desempenho dos demais.

2.3.1 Modelos de Regressão

A modelagem regressiva relaciona uma variável resposta (ou predita) com uma ou mais variáveis explicativas (ou preditoras). Sua aplicação mais ampla e comum é a criação e construção de modelos preditivos, servindo ao propósito de elucidar e explicar o comportamento de um ou mais objetos de interesse.

Existem inúmeros tipos de modelos de Regressão, contudo, apenas alguns são apropriados para o escopo deste projeto. Sendo assim, encontram-se imediatamente abaixo apenas os tipos de modelos empregados na realização do mesmo.

1. **Regressão Logística**

   A regressão logística faz parte do conjunto de modelos lineares generalizados, e seu uso é amplamente difundido. Sua principal característica é o fato de sua variável predita ser categórica, podendo ser binária, multinomial ou ordinal, ainda que a aplicação mais frequente seja relativo à primeira opção.

   Devido a estrutura de sua variável resposta, a regressão logística estuda dados de distribuição binomial ou bernoulli. Isso é, temos:

$$Y_i \sim Bernoulli(p_i), i = 1, 2, ..., m$$

   em que as probabilidades de sucesso p_i são desconhecidas. Então, o modelo logístico modela a relação entre as variáveis preditoras lineares X_i e p_i através do logaritmo das chances (*odds*), também chamado de logito. Logo, temos a seguinte estrutura:

$$\text{logito}(p_i) = \ln\left(\frac{p_i}{1-p_i}\right) = \beta_0 + \beta_1 X_{1,i} + ... + \beta_j X_{j,i}$$

   Equivalentemente, podemos predizer a probabilidade da seguinte forma:

$$p_i = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X_{1,i} + ... + \beta_j X_{j,i})}} = \left(1 + e^{-(\beta_0 + \beta_1 X_{1,i} + ... + \beta_j X_{j,i})}\right)^{-1}$$

   Sendo assim, também podemos utilizar a regressão logística para predizer a probabilidade de eventos previamente especificados acontecerem.

   No contexto do presente trabalho, é natural a utilização da regressão logística binária, cuja variável predita é a indicadora de aprovação na Câmara, descrita previamente ao final da seção 2.1. Sua implementação será feita pela linguagem de programação R através do pacote glmnet, que possui a função *cv.glmnet()* capaz de construir diversos modelos lineares generalizados, inclusive de regressão logística.

   É importante destacar que a preferência pela utilização do glmnet para esta aplicação específica é, além de sua velocidade superior aos demais pacotes, a sua utilização de Validação Cruzada no treinamento dos modelos. De maneira simplificada, a Validação Cruzada significa que a própria função irá separar o banco fornecido à ela e separá-lo em um número k de grupos (também chamado de *k-folds*). Então, o modelo seleciona aleatoriamente um dos grupos para teste, e realiza o treinamento de modelos em cada um dos restantes k-1 grupos. Realizado o teste, a função avalia

qual dos modelos possuiu o melhor desempenho, e o aplica ao conjunto completo fornecido.

Embora pareça redundante a aplicação de Validação Cruzada, já que também foi realizado uma separação prévia do conjunto de dados, ela representa mais uma camada de qualidade da qual o modelo deve ser aprovado, elevando a eficiência geral dos modelos. Abaixo, encontra-se a tabela 12 com breves explicações dos argumentos utilizados:

Tabela 12: Função cv.glmnet - Argumentos utilizados

Argumento	Explicação
x	Matriz fornecida à função como banco de dados.
y	Matriz de dados que a função deverá predizer após a escolha do melhor modelo.
alpha	Argumento opcional. 0 empregará Regressão Ridge, 0.5 Elastic Net, 1 LASSO, e sua ausência não aplicará nenhuma das três.
nfold	Número k de grupos da validação cruzada.
family	Indica o tipo de modelo linear generalizado escolhido.

Assim sendo, para produzir os modelos logísticos especificamente, foi ignorado o argumento alpha, selecionado um número de 5 grupos (conforme recomendado pela função para bancos de dados grandes) e especificado a família "binomial".

2. **Regularização LASSO**

Antes de explicar do que se trata LASSO, Ridge e Elastic Net, devemos primeiro contextualizar o que é regularização, e do que se trata. Resumidamente, a regularização é uma técnica que diminui o ajuste excessivo dos dados, por meio do favorecimento de alguns componentes do modelo em relação à outros. De certa forma, esse método insere um viés controlado no modelo, a fim de melhorar seu desempenho geral.

Especificamente, LASSO é uma forma de regularização via penalidade, que possui o objetivo de minimizar a seguinte equação:

$$\min_{\beta_0,\boldsymbol{\beta}} \left(\sum_{i=1}^{N} (y_i - \beta_0 - \boldsymbol{\beta}.\boldsymbol{x_i})^2 \right) \text{ sujeito à } \sum_{j=1}^{k} |\beta_j| \leq t$$

em que N é a quantidade de observações, y_i é a variável predita, β_0 e $\boldsymbol{\beta}$ são respectivamente o coeficiente constante e o vetor de coeficientes, $x_i = (x_1, ..., x_k)$ é o vetor das covariáveis para a i-ésima observação e t é um parâmetro pré-especificado que define o nível da regularização. Alternativamente, podemos reescrever a equação em sua forma Lagrangiana:

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^k}\left(\frac{1}{N}||y-\boldsymbol{X\beta}||_2^2+\lambda||\beta||_1\right)$$

onde **X** é a matriz de covariáveis e a relação entre t e λ é dependente dos dados. É exatamente nesse formato que a função *cv.glmnet()* opera, e $\lambda||\beta||_1$ é o que chamamos de L1, ou coeficiente LASSO.

A característica definitiva da penalidade LASSO é que ela opera zerando covariáveis pouco importantes no modelo, como no caso de múltiplas variáveis correlacionadas que transmitem informações muito próximas. Dessa forma, o LASSO ajuda na parcimônia do modelo, além de facilitar a sua interpretação. Como o coeficiente L1 é aplicável em todos os modelos lineares generalizados, utilizou-se a função *cv.glmnet()* para gerar modelos logísticos com LASSO, divergindo da aplicação padrão apenas pela especificação do parâmetro alpha igual a 1.

3. **Regularização Ridge**

   Bastante similar à LASSO, temos a regularização Ridge ou coeficiente L2. No caso, temos a equação do seguinte formato:

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^k}\left(\frac{1}{N}||y-\boldsymbol{X\beta}||_2^2+\lambda||\beta||_2^2\right)$$

   Embora o objetivo do L2 também seja penalizar coeficientes próximos, note que a regularização Ridge opera com os quadrados dos coeficientes, o que resulta em uma penalização que nunca zera os β_i parecidos, ao contrário da LASSO.

   Outra consequência da penalidade quadrática é que isso torna coeficientes de alto valor desproporcionalmente grandes, tornando o cálculo do modelo computacionalmente muito custoso. Por este motivo, a regressão Ridge não foi implementada para o presente trabalho; ainda assim, fez-se necessário abordá-la, já que os modelos Elastic Net foram utilizados com sucesso.

4. **Regularização Elastic Net**

   Por sua vez, a regularização Elastic Net trata-se simplesmente de combinar ambos os coeficientes L1 e L2 previamente descritos. Portanto, a equação apresenta-se da seguinte forma:

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^k}\left(\frac{1}{N}||y-\boldsymbol{X\beta}||_2^2+\lambda_1||\beta||_1+\lambda_2||\beta||_2^2\right)$$

   Sua principal vantagem é combinar o melhor dos dois casos, pois enquanto a LASSO zera coeficientes desnecessários, a Ridge penaliza os restantes que julgar necessário.

Na conjuntura do presente projeto, a presença do coeficiente L1 permitiu a aplicação do Elastic Net, pois sua simplificação do modelo resolveu o gargalo computacional que o L2 causava quando utilizado sozinho. Analogamente à Regularização LASSO, a implementação via *cv.glmnet()* é trivial, bastando sinalizar a utilização do parâmetro alpha igual a 0.5 na construção de um modelo logístico.

2.3.2 Modelos Naive Bayes

Naive Bayes é um classificador probabilístico definido por duas características que o nomeiam: a ingenuidade e o Teorema de Bayes. A primeira parte é razoavelmente simples, e se deve ao fato do modelo ignorar qualquer correlação que exista entre suas variáveis, como se todas fossem independentes entre si.

Agora, considere um determinado documento B, uma determinada classificação A. Então, o que o modelo tenta descobrir é a probabilidade do documento B pertencer à classe A. O Teorema de Bayes especifica que essa relação se apresenta pela forma:

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

Analogamente, o mesmo vale para qualquer outra classe possível apresentada ao modelo. Para a implementação dos modelos Naive Bayes, utilizou-se a função *textmodel_nb()* do *quanteda*, sendo necessário apenas especificar os dados de treinamento e suas classificações verdadeiras. Como estamos interessados em saber apenas se um projeto foi aprovado na Câmara ou não, existem apenas duas categorias para o modelo classificar cada PL. A fim de considerar as palavras (ou *features*) fornecidas no cálculo da probabilidade, a função *textmodel_nb()* utiliza o Teorema de Bayes com algumas transformações algébricas, dada pela seguinte fórmula:

$$\ln\left(\frac{P(A|B)}{P(\overline{A}|B)}\right) = \ln\left(\frac{P(A)}{P(\overline{A})}\right) + \sum_i \ln\left(\frac{P(k_i|A)}{P(k_i|\overline{A})}\right)$$

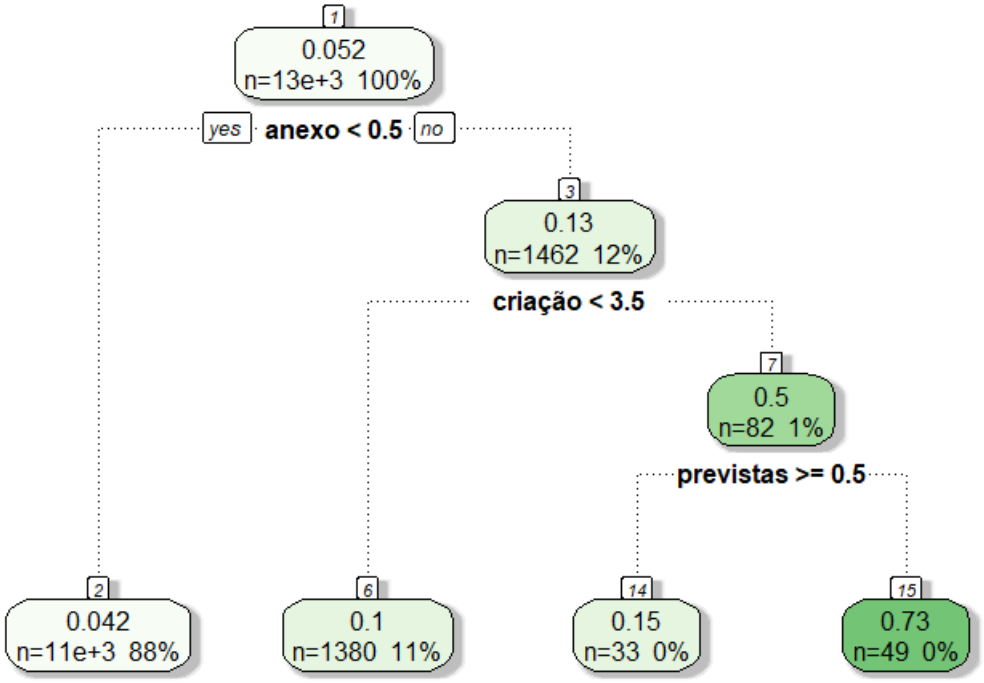
em que A é a classe dos projetos aprovados, $\overline{A}$ é a classe de projetos arquivados, e k_i é a i-ésima palavra da matriz de dados. Sendo assim, caso o valor calculado seja superior a 0, o PL é considerado aprovado, e arquivado caso contrário.

2.3.3 Modelos de Árvore de Classificação

Modelos de árvore de decisão são um assunto bastante extenso na área da Estatística e Ciência de Dados, abrangendo diversas possibilidades de construção e modela-

gem. Relativo ao escopo do projeto, estamos especificamente interessados nas chamadas "Árvores de Classificação", cuja variável predita é categórica, isso é, se um projeto de lei foi aprovado ou arquivado pela Câmara dos Deputados. A terminologia "árvore" se deve ao formato produzido pelo modelo, conforme exemplo abaixo:

Figura 1: Exemplo de modelo de árvore



Todo modelo de árvore começa pelo nó raiz, representado pelo nó "1" da figura. O algoritmo, então, utiliza uma regra de decisão pré-definida para separar o conjunto de dados em dois, o que no exemplo diretamente acima resultou no nó terminal (ou folha) "2" e em um sub-nó de decisão "3". O modelo continua sucessivamente até que não existam mais separações vantajosas de acordo com sua(s) regra(s) de decisão e outros critérios, restando ao fim apenas os nós terminais.

A função utilizada para construir todas as árvores apresentadas no modelo é a *rpart()*, do pacote homônimo. Existem diversas regras de decisão a serem escolhidas em um algoritmo de árvore de classificação; entretanto, o *rpart()* foca apenas na "redução de impureza". Considerando C como a quantidade de classes da variável predita, p_{iA} a probabilidade de um nó A qualquer da árvore e f uma função de impureza qualquer, temos:

$$I(A) = \sum_{i=1}^{C} f(p_{iA})$$

Mais especificamente, existem duas possibilidades que o *rpart()* implementa para f: índice de informação $f(p_{iA}) = -p_{iA}\log(p_{iA})$, ou o índice de Gini $f(p_{iA}) = p_{iA}(1-p_{iA})$. Em problemas de classificação binária como o de projetos aprovados ou arquivados, ambos possuem desempenho parecido, não sendo necessário distinção entre os dois. Finalmente, apresenta-se a regra de decisão de maior redução de impureza:

$$\Delta I = p(A)I(A) - p(A_L)I(A_L) - p(A_R)I(A_R)$$

Em que A_L e A_R são dois sub-nós em que A seria dividido. Dessa forma, um nó é apenas dividido se isso representar uma redução de impureza geral. Por exemplo, na figura superior o modelo reconheceu que a divisão dos projetos que possuem ao menos um anexo em relação aos que não possuem resultou em uma redução de impureza, gerando dois novos nós.

Por fim, é importante mencionar que a probabilidade discriminante de aprovação adotado especificamente nas árvores foi definido através de tentativas manuais, chegando no valor de 0,15. Portanto, caso o nó do projeto tenha probabilidade empírica superior ao valor de corte, ele será considerado aprovado, ou arquivado caso contrário.

2.3.4 Métricas de diagnóstico e performance

Naturalmente, a construção de um modelo por si só não é suficiente. É necessário, por exemplo, saber o quão bem eles se adequam aos dados e, é claro, conseguir comparar diferentes abordagens de modelagem.

Com esse intuito, existem incontáveis métricas de diagnóstico e performance adequadas à cada tipo de modelo. Em relação a modelos de classificação binários, uma das abordagens mais utilizadas e efetivas é através de uma tabela 2x2, em que cruzamos os valores verdadeiros dos dados de teste com os valores preditos pelo modelo. Visualmente, estrutura-se:

	Valores Verdadeiros		
Valores Preditos	Projeto Aprovado	Projeto Arquivado	Total
Projeto Aprovado	A	B	A+B
Projeto Arquivado	C	D	C+D
Total	A+C	B+D	A+B+C+D

Partindo dessa estrutura, é possível calcular diversos indicadores de qualificação dos modelos gerados. Dessa forma, foi empregado o uso da função *confusionMatrix()* do pacote *caret*, que retorna algumas métricas agrupadas no que se chama de "matriz de

confusão". A seguir, encontra-se a descrição de todas as medidas usadas:

- **Verdadeiros e falsos positivos ou negativos**

  É uma nomenclatura que se refere aos erros e acertos do modelo. Considere um projeto X qualquer, que não foi aprovado; se o modelo considerá-lo como aprovado, isso configuraria o mesmo como um "falso positivo", enquanto que caso ele fosse corretamente classificado, seria um "verdadeiro negativo". Analogamente, se um projeto Y aprovado for dito como aprovado pelo modelo, ele seria um "verdadeiro positivo", ou "falso negativo" caso contrário.

  Sendo assim, temos que as letras A, B, C e D representam o número total de casos em cada uma das possibilidades. Isso é,

$$\textit{Verdadeiros Positivos} = VP = \sum_{i=1}^{n} I_{VP} = A$$

$$\textit{Falsos Positivos} = FP = \sum_{i=1}^{n} I_{FP} = B$$

$$\textit{Falsos Negativos} = FN = \sum_{i=1}^{n} I_{FN} = C$$

$$\textit{Verdadeiros Negativos} = VN = \sum_{i=1}^{n} I_{VN} = D$$

  Logicamente, um modelo perfeito teria sempre o valor de B e C iguais a zero, e os respectivos números corretos de A e D; entretanto, isso também é impossível de ocorrer de forma recorrente. Ainda assim, espera-se que um bom modelo chegue o mais perto possível da perfeição, ou seja, tenha valores de B e C muito menores do que A e D.

- **Relação de Não-Informação**

  A relação de não-informação é simplesmente a proporção da maior classe verdadeira em relação ao total. É útil principalmente para bancos de dados muito desbalanceados, pois é comum que o modelo seja tendencioso para a maior classe. Através dessa medida, conseguimos ver então a proporção dos dados que está sendo direcionada à classe de menor interesse, e um possível indicador de viés no modelo.

  Relativo ao projeto, sabemos que existem muito mais projetos arquivados do que aprovados, o que significa que essa medida sempre se resumirá a:

$$\text{RNI} = \frac{B + D}{A + B + C + D}$$

- **Acurácia**

  No caso de modelos de classificação binária, a acurácia define-se pelo número de acertos do modelo em relação ao total de casos do banco de dados. Portanto, calcula-se:

$$\text{Acurácia} = Acc = \frac{A + D}{A + B + C + D}$$

  Espera-se que um modelo adequado possua boa acurácia, embora esta métrica não seja útil por si só. No caso de uma base desbalanceada como a de projetos de lei, um modelo pode obter uma boa acurácia simplesmente por classificar todos os modelos como arquivados, obtendo um valor muito alto de D e um pequeno de C, o que resultaria em uma proporção alta de *Acc*. Dessa maneira, é necessário sempre observar a acurácia com cautela, comparando-a com outras medidas calculadas.

- **Teste Binomial Exato**

  De acordo com item anterior, o teste binomial exato apresenta-se como uma forma de comparar a acurácia de um modelo com a sua relação de não-informação. O intuito por trás desta comparação é verificar se a acurácia está sendo "carregada" pela classe de maior tamanho, isso é, se o modelo não está com um bom poder discriminativo entre as classes. Caso a acurácia seja inferior ou igual à RNI, isso significa que o modelo possui um certo viés para a maior classe, enquanto que um valor superior de acurácia indica um modelo de melhor poder preditivo.

  Portanto, o teste estrutura-se com as seguintes hipóteses:

$$H_0 : Acc \leq \text{RNI}$$

$$H_1 : Acc > \text{RNI}$$

  Podemos então empregar o uso do teste binomial exato para a comparação das duas medidas. No caso do presente trabalho, a métrica utilizada para definição do resultado do teste é o "p-valor", em que calculamos a probabilidade de obtermos um valor de acurácia maior ou igual ao calculado, dado que H_0 seja verdadeiro. Isso se traduz da seguinte maneira:

$$p = \sum_{i=k}^{N} \binom{N}{i} \text{RNI}^i (1 - \text{RNI})^{(N-i)}$$

  em que $k = A + D$ é o número de acertos do modelo e $N = A + B + C + D$ é o total de projetos. Considerando um número de significância igual a 10%, rejeita-se H_0 para p-valores inferiores a α.

- **Sensitividade e Especificidade**

  Também chamado de *recall* e relação de verdadeiros positivos, a sensitividade refere-se à quantidade de verdadeiros positivos do modelo sobre o total de positivos reais. É, portanto, calculada por:

  $$Se = \frac{A}{A + C}$$

  Por sua vez, a especificidade é a relação entre verdadeiros negativos e o total de negativos reais, obtida da maneira:

  $$Es = \frac{D}{B + D}$$

  Ambas as métricas são importantes para verificar o quanto o modelo está acertando dentro de cada classe, ainda que o grau da importância dependa do contexto aplicado. No caso do vigente estudo, a sensitividade é a mais relevante, pois estamos mais interessados em um modelo que consiga acertar mais os projetos aprovados, do que um modelo assertivo em PLs não-aprovados.

- **Acurácia Balanceada**

  No que lhe diz respeito, a acurácia balanceada é uma acessível medida que busca um pouco mais de robustez comparada à acurácia tradicional. Enquanto que a acurácia leva em consideração apenas o total de acertos relativos ao total, a forma balanceada pesa os acertos dentro de sua própria classe real. Em outras palavras, a acurácia balanceada é apenas a média aritmética da sensitividade e especificidade. Sendo assim:

  $$Acc_B = \frac{Se + Es}{2}$$

  Por consequência, a acurácia balanceada não é tão fortemente influenciada por um viés do modelo para a maior classe, pois tal modelo viesado apresentaria baixa sensitividade, reduzindo pela metade o valor de Acc_B. Mesmo assim, a referida métrica não é necessariamente a mais informativa de todas, pois Se e Es possuem o mesmo peso, sendo que para a presente monografia, a sensitividade é mais relevante ao modelo e deveria ter maior ponderação.

- **Teste de McNemar**

  O teste de McNemar é empregado para verificar a concordância do modelo binário com os valores reais, ou seja, descobrir se a predição do modelo está próxima da realidade. A maneira da qual o teste funciona é averiguando se a probabilidade do

modelo predizer que os projetos serão aprovados é a mesma probabilidade de um projeto de fato ser aprovado. Matematicamente, isso se traduziria:

$$P(A) + P(B) = P(A) + P(C)$$

Na prática, isso significa que estamos comparando se:

$$P(B) = P(C)$$

Analogamente, o mesmo ocorre se estivermos observando a probabilidade de projetos arquivados preditos e reais. Logo, temos como hipóteses:

$$H_0 : P(B) = P(C)$$

$$H_1 : P(B) \neq P(C)$$

E a estatística do teste:

$$\chi^2_{Mc} = \frac{(B-C)^2}{B+C} \sim \chi^2_1 \text{ g.l.}$$

Novamente, calcula-se o p-valor como a probabilidade de obter o respectivo valor de χ^2_{Mc} sob H_0. Rejeita-se a hipótese nula se o p-valor calculado for inferior à 0,1.

- **Prevalência**

  Medida básica que quantifica o número total de projetos aprovados reais sobre o total de PLs. Ou seja:

$$\text{Prev} = \frac{A+C}{A+B+C+D}$$

  Embora indique o quão desbalanceado os dados positivos estejam em relação ao total, não é uma métrica muito útil sozinha. Entretanto, a prevalência é necessária para calcular outras medidas mais informativas.

- **Prevalência de Detecção**

  A prevalência de detecção é um indicador de quanto o modelo está predizendo os projetos aprovados em relação ao total. Portanto:

$$\text{Prev}_D = \frac{A+B}{A+B+C+D}$$

  Como existe um forte desbalanceio em relação à projetos aprovados e arquivados, espera-se sempre que a prevalência de detecção assuma valor baixos e próximos de

zero. Ainda assim, essa métrica será informativa caso seus números sejam muito próximos de zero, apontando um modelo com viés para arquivamento de projetos.

- **Precisão**

  Alternativamente chamada de "Valor Positivo Preditivo" (em inglês, PPV), a precisão pode ser definida como o total de acertos do modelo em todas as suas predições positivas. Em outras palavras, a precisão representará o percentual de acertos do modelo relativo à todas as vezes que ele prediz que um projeto será aprovado. Matematicamente, existem duas formas de calcular esse resultado:

$$Pr = \frac{A}{A+B} = PPV = \frac{Se \times \text{Prev}}{(Se \times \text{Prev}) + [(1 - Es) \times (1 - \text{Prev})]}$$

  Sendo assim, a precisão é útil para descobrir se o modelo está de fato predizendo a aprovação dos projetos, ou se está apenas realizando muitas apostas aleatorizadas.

  Contudo, é também uma medida que requer cautela ao ser analisada. Considere um modelo que tenha predito apenas a aprovação de um projeto, e que ele seja um verdadeiro positivo. Dessa forma, o modelo possui uma precisão de 100% e uma alta acurácia, ainda que claramente não seja um modelo útil e bem ajustado.

- **Pontuação F1**

  Buscando robustez superior em dados desbalanceados, a pontuação F1 apresenta-se como uma alternativa à acurácia, baseando-se em uma média harmônica da sensitividade e precisão de um modelo. Tradicionalmente, adota-se:

$$F1 = 2 \times \frac{Se \times Pr}{Se + Pr}$$

  A principal vantagem do F1 é o fato de ele levar em consideração ambas métricas relacionadas ao projetos aprovados. Isso é, a sensitividade mede o percentual acertado de todos os projetos realmente aprovados, enquanto a precisão mede quantos acertos foram feitos de todas as predições de "aprovado". Portanto, a pontuação F1 será de primordial utilização na escolha dos modelos mais adequados.

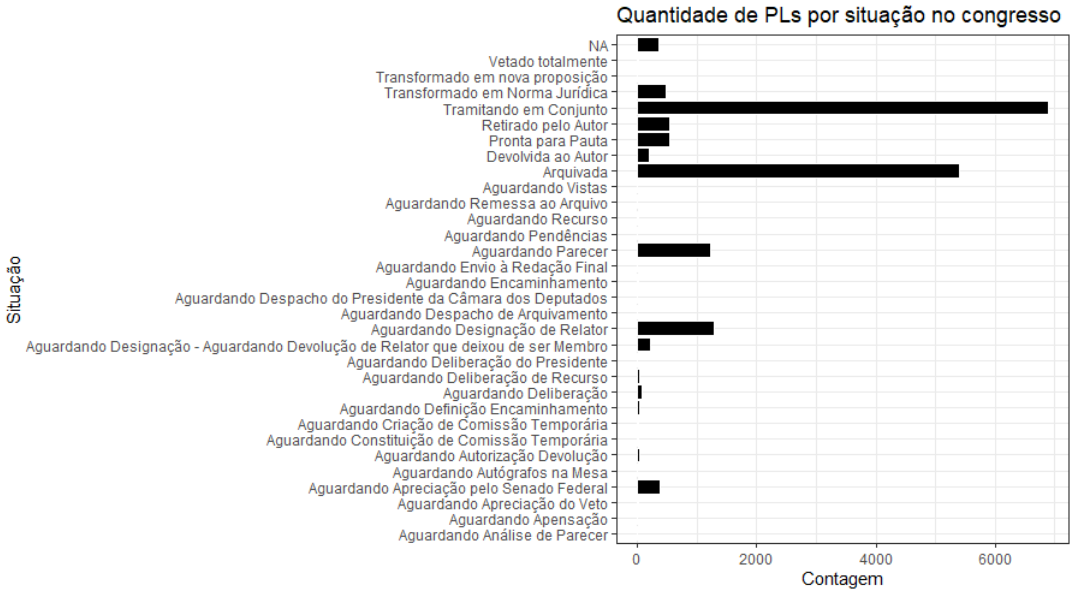
3 Resultados

Esta é a seção em que será mostrado o desfecho das aplicações técnicas descritas até então, e que fundamentará as conclusões do presente trabalho.

3.1 Análise Exploratória Inicial

Naturalmente, a primeira etapa após a obtenção dos dados em um formato adequado é a exploração introdutória. Nesse sentido, buscou-se primeiramente uma forma de definir a variável predita do modelo, o que foi feito por meio da variável "ultimoStatus_descricaoSituacao". Portanto, elaborou-se o seguinte gráfico presente na figura 2:

Figura 2: Gráfico de frequência da variável "ultimoStatus_descricaoSituacao"



Dentre as informações relevantes, destacam-se:

- Observa-se que maioria dos PLs estão tramitando em conjunto com outro projeto. Isso é relativamente intuitivo, dado que a observação de "Tramitação em conjunto"é no mínimo duplicada;
- A segunda categoria mais comum é a dos projetos arquivados, de acordo com o estudo e afirmações dos autores de referência;
- "Retirado pelo Autor", "Aguardando Parecer"e "Aguardando Designação de Relator"são categorias com frequência considerável, mas são condições desinteressantes para construir um modelo preditivo sobre. Logo, são descartadas para comporem a variável predita;

- "Pronta para Pauta"aparece em um primeiro instante como uma opção viável, isso é, poderíamos construir um modelo preditivo de projetos que fossem ser pautados na Câmara. Entretanto, uma análise mais a fundo revelou que projetos que aguardam pauta em comissões e em plenário são igualmente classificados nessa categoria, embora sejam situações bem distintas. Portanto, desconsiderou-se essa possibilidade;
- Por último, "Transformada em Norma Jurídica"e "Aguardado apreciação pelo Senado Federal"são as duas classes realçadas, possuindo uma contagem considerável dos casos e pertinentes para serem preditas por um modelo.

Nesse sentido, a variável predita do modelo resumiu-se a duas alternativas: a aprovação total do projeto, ou a aprovação parcial do projeto na Câmara. Então, gerou-se uma tabela resumindo as categorias pertinentes com sua frequência absoluta:

Tabela 13: Classes notáveis da variável "ultimoStatus_descricaoSituacao"

Situação	Contagem
Vetado totalmente	18
Transformado em Norma Jurídica	489
Tramitando em Conjunto	6897
Pronta para Pauta	538
Arquivada	5407
Aguardando Apreciação pelo Senado Federal	375
Aguardando Apreciação do Veto	25

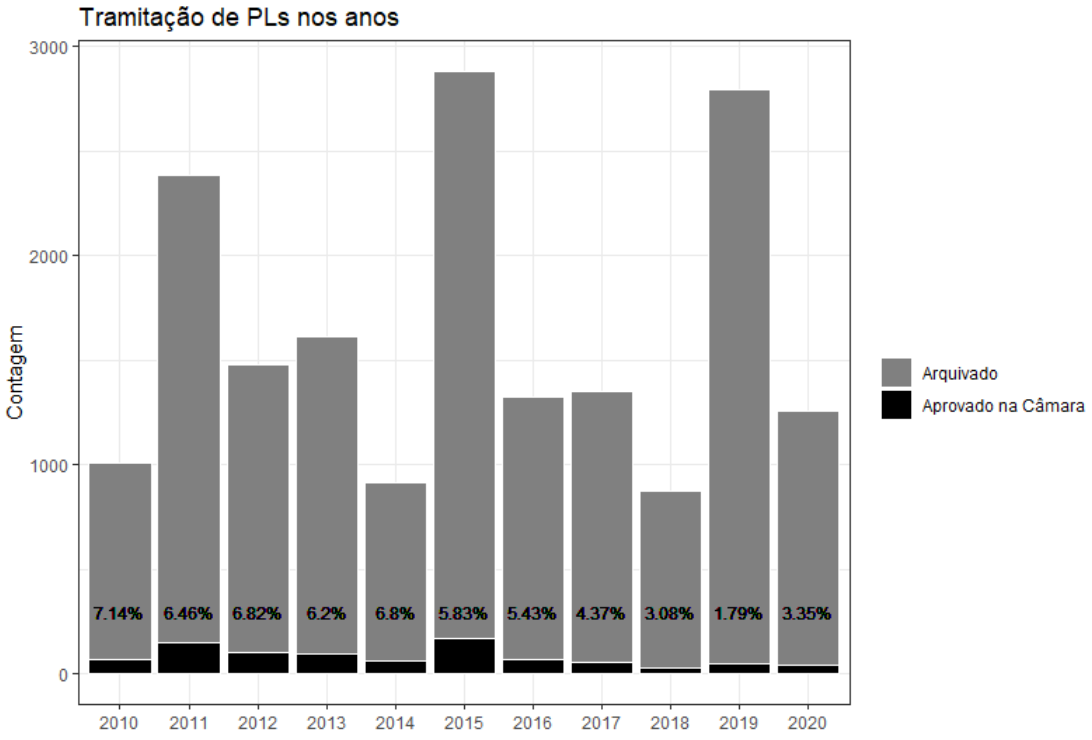
Através da Tabela 13, vemos que:

- Optando pela aprovação total do projeto, a variável indicadora consistiria de 489 projetos apenas, já que não podemos contabilizar nenhum projeto vetado, que esteja no aguardo ou tramitando no Senado Federal.
- Para a aprovação do projeto apenas na Câmara, teríamos um total de 907 projetos somando as aprovações completas, vetos e tramitação no Senado. Tal número é quase o dobro da opção alternativa, o que facilitaria o desempenho do modelo dado o grande desbalanceio dos dados.

Logo, a escolha da variável predita do modelo é a de projetos aprovados apenas na Câmara dos Deputados.

A seguir, realizou-se um pequeno estudo das variáveis julgadas como relevantes a serem incluídas na matriz de palavras, a começar pela variável "ano". Nesse caso, produziu-se o gráfico apresentado na figura 3:

Figura 3: Variável "ano"



Curiosamente, enxergamos que anos eleitorais são os que menos possuem tramitação de projetos, enquanto que as maiores quantidades de PLs são os anos de posse do mandato. Isso é, 2011, 2015 e 2019 foram disparados os anos que mais tiveram tramitação de PLs, em contraste com os respectivos anos antecedentes que têm a menor frequência.

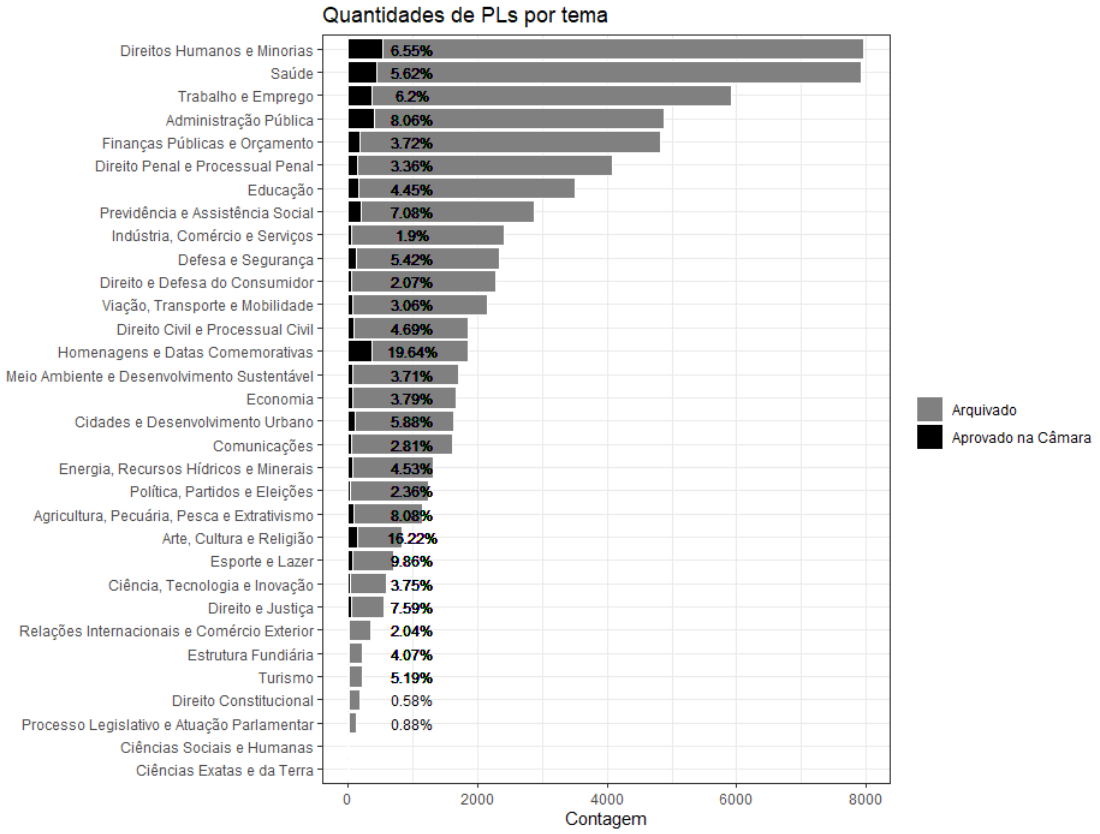
Existem algumas possíveis explicações para isso:

- Anos eleitorais são muito corridos para políticos, com os deputados precisando utilizar boa parte do seu tempo para firmar alianças, discutir filiações a partidos, etc;
- Com medo de uma possível rejeição atrapalhar seu desempenho eleitoral, os deputados estão menos inclinados a produzirem novos projetos, especialmente de temas polêmicos ou controversos;
- No que se refere aos anos de posse de mandato, a maior quantidade de PLs tramitados pode ser devido à uma "renovação"dos deputados da casa, pois alguns novos deputados assumem, enquanto outros não conseguem a reeleição;
- O ano de posse é o que o presidente mais possui força e popularidade, o que leva ele e sua base a apresentarem mais projetos.

Em relação ao percentual de projetos aprovados na Câmara, não se percebe nenhum padrão ou tendência.

Subsequentemente, abordou-se a variável referente ao tema de cada projeto de lei, visando elucidar algum tipo de viés presente em alguma das categorias. Portanto:

Figura 4: Variável "tema"

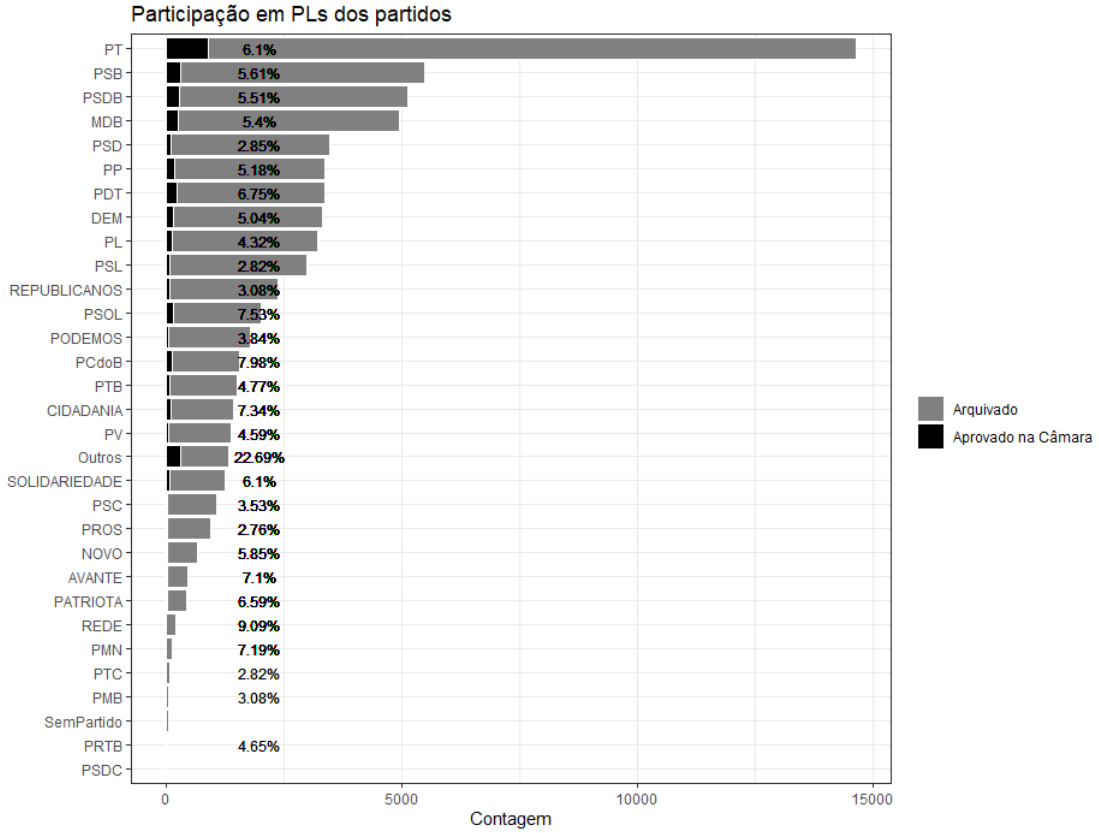


Nota-se, então, que os temas de "Direitos Humanos e Minorias"e "Saúde"são os que mais tramitam pela Câmara, embora o último muito provavelmente tenha seus números impulsionados pela pandemia do COVID-19 de 2020.

Em relação à aprovação, constata-se dois temas que possuem uma proporção maior do que aos demais, sendo esses "Homenagens e Datas Comemorativas"e "Arte, Cultura e Religião", com respectivos 19.64% e 16.22%. Ainda que seja um triste fato, tal resultado era esperado, já que projetos desses cunhos costumam ser mais fáceis de se aprovar, dado a sua simplicidade e baixa resistência por parte da população e demais deputados. Presume-se, portanto, que essas categorias sejam identificadas pelo modelo como significativas para a aprovação de um PL.

No mesmo raciocínio que a variável de temas, aplicou-se o mesmo conceito para os partidos políticos, culminando na figura 5 abaixo:

Figura 5: Variável "siglaPartidoAutor"



Aqui, observa-se que o Partido dos Trabalhadores (PT) é de longe o partido que mais tem participação nos projetos, com quase 15 mil observações. Embora seja um número muito acima dos demais, isso se explica ao fato de que o PT foi o maior partido do Brasil durante todo o período de estudo, sempre tendo uma bancada muito grande na Câmara, além de ser o partido do Presidente da República durante quase 7 dos 11 anos estudados, o que facilita a tramitação dos PLs.

Entretanto, a maior proporção de aprovação de 22.69% dos PLs fica por parte de "Outros", que é a classe que engloba os órgãos do poder Executivo e Judiciário, como o Presidente da República, Ministérios, Supremo Tribunal Federal, etc. Logo, é possível que os modelos identifiquem esse tipo de categoria como significante.

3.2 Matrizes de Palavras

Em prosseguimento à análise exploratória inicial, construiu-se as matrizes de palavras que alimentarão os dados aos modelos. Nessa etapa, utilizou-se muito de gráficos de nuvem de palavras e tabelas de frequência, a fim de identificar palavras vazias que fossem muito frequentes ou que pudessem atrapalhar o funcionamento do modelo. Ainda assim, é importante destacar que mesmo que muitas palavras comprometedoras tenham

sido retiradas, é quase impossível retirar todas elas, dado a quantidade gigantesca de palavras ao longo de todos os quase 18 mil projetos de lei.

Dessa forma, a nuvem de palavras final das três matrizes n-grams encontra-se na figura 6 abaixo:

Figura 6: Matrizes de Palavras

(a) Matriz 1-gram

(b) Matriz 2-grams

(c) Matriz (1,2)-grams

Com o auxílio gráfico, constata-se alguns pontos:

- A palavra "nacional" aparece como a mais frequente em todas as três matrizes, com a subfigura (b) indicando um pouco melhor o contexto em que ela é mais usada. Embora isso talvez indique que ela deveria ser removida, tal palavra é caracterizada por muita ambiguidade. Seu uso pode variar desde "feriado nacional", "Instituto Nacional" até uma frase de criação de um programa de "âmbito nacional", o que caracteriza informações muito distintas e que não podem ser consideradas como vazias;

- Naturalmente, a nuvem da matriz com 1-gram é mais concentrada do que as demais, já que é mais provável ocorrer uma frequência maior de uma dada palavra, do que observar muitas vezes uma combinação de duas palavras;

- Em todas as nuvens, nota-se a presença de muitas palavras que remetem e concordam com o visto na análise exploratória inicial, como "saúde", "direito", "administração_pública"e "poder_executivo". Esse fato indica que as matrizes estão realmente representativas dos dados e com pouca interferência de palavras vazias.

Ainda que as referidas matrizes aparentem ter uma boa estrutura, também devemos ter em mente que elas são muito extensas e esparsas, dada pela tabela 14 abaixo.

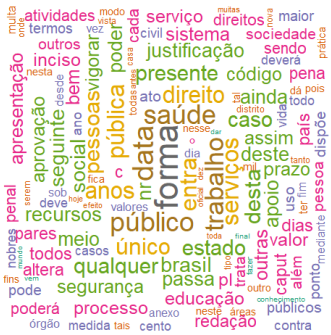
Tabela 14: Características das matrizes de palavras

Matriz	Projetos de Lei (Linhas)	Palavras (Colunas)	Esparsidade
1-gram	17.879	127.492	99.81%
2-grams	17.879	2.859.649	99.99%
(1,2)-grams	17.879	2.987.098	99.98%

Nesse sentido, criou-se novos conjuntos de dados a partir da aplicação de dois filtros simétricos nas matrizes originais, removendo 10% e 20% das palavras mais comuns e incomuns, culminando nas seguintes nuvens de palavras:

Figura 7: Matrizes de Palavras com filtro de 10%

(a) Matriz 1-gram



(b) Matriz 2-grams

(c) Matriz (1,2)-grams

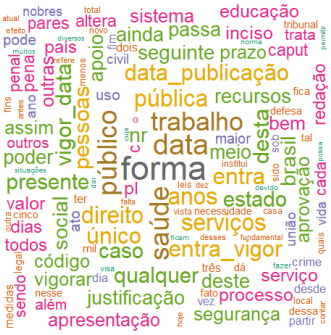
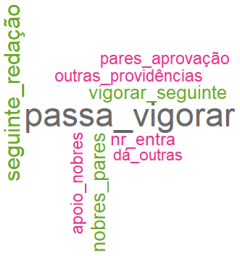


Figura 8: Matrizes de Palavras com filtro de 20%

(a) Matriz 1-gram

(b) Matriz 2-grams



(c) Matriz (1,2)-grams

Em ambas as figuras, vemos uma perceptível diferença nas suas respectivas densidades e na ausência de algumas palavras específicas, como "nacional". O destaque, entretanto, é para as matrizes de 2-grams, que sofreram um grande corte de palavras devido a sua concentração menor de palavras idênticas.

É claro, deve-se manter mente que apesar de os filtros aplicados ajudarem na redução de esparsidade e ganho computacional, é possível que eles cortem demais as palavras das matrizes e forneçam poucos dados para os modelos, prejudicando o poder preditivo. Por exemplo, isso é especialmente plausível de ocorrer na matriz 2-grams de filtro 20%, em que temos poucas palavras sem nenhuma indicação de concordância com a análise exploratória. Ademais, montou-se também uma tabela descritiva das 6 novas matrizes produzidas, localizada diretamente abaixo:

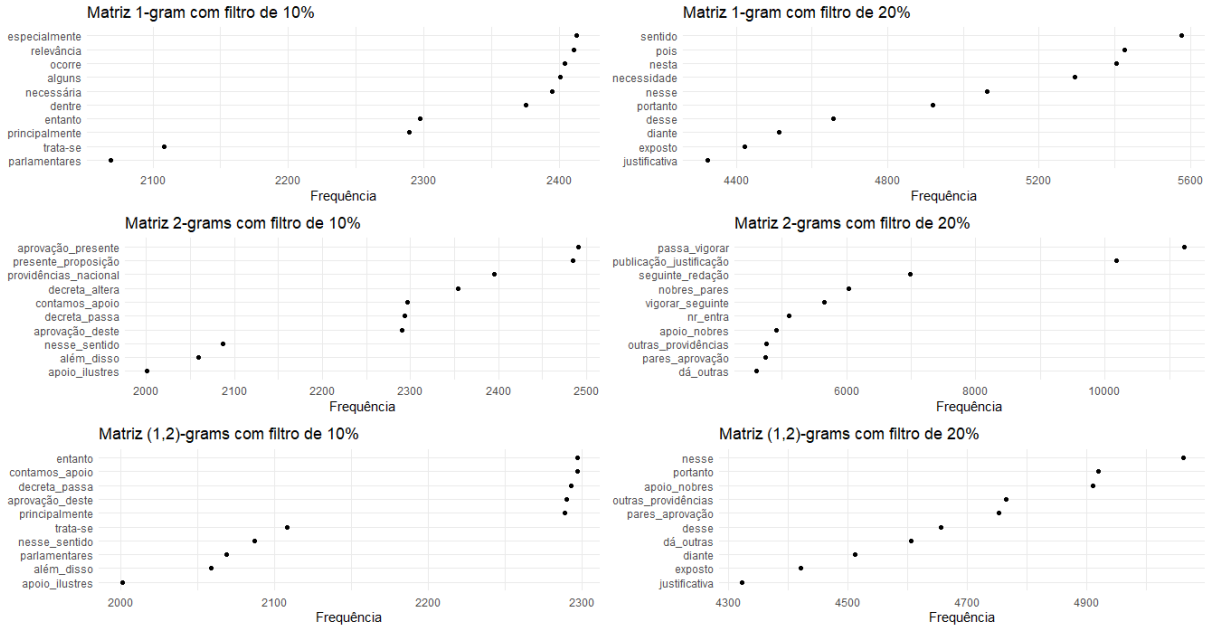
Tabela 15: Características das matrizes filtradas

Matriz	Filtro	Projetos de Lei (Linhas)	Palavras (Colunas)	Esparsidade
1-gram	10%	17.879	370	81.02%
2-grams	10%	17.879	30	73.54%
(1,2)-grams	10%	17.879	400	80.46%
1-gram	20%	17.879	113	70.45%
2-grams	20%	17.879	10	67.84%
(1,2)-grams	20%	17.879	123	70.24%

Em conformidade com o descoberto nas nuvens gráficas, observa-se uma redução massiva da quantidade de palavras, com um corte superior a 90% relativo do total das matrizes originais. No mesmo contexto, vemos também que o corte diminuiu significamente a esparsidade, ainda que os valores associados continuem altos e acima de 2/3 de todas as matrizes. Por fim, elaborou-se também 6 gráficos com as 10 palavras menos frequentes das matrizes filtradas. Portanto:

Figura 9: Matrizes de Palavras com filtro de 20%

(a) Matrizes de filtro 10% (b) Matrizes de filtro 20%



Em relação ao filtro de 10%, percebe-se que as palavras ficaram em intervalos de frequência próximos em todas as matrizes, o que indica que o corte ocasionou uma relativa padronização. Na diminuição de 20%, vemos a matriz de 2-grams muito distinta das demais, já que suas 10 palavras menos frequentes também são as mais frequentes, pois são as únicas 10 em toda a matriz.

3.3 Modelos

No momento em que os dados estavam organizados de maneira adequada, finalmente deu-se início a modelagem. Uma questão, entretanto, seria a forma de implementação dos modelos, pois existem muitas combinações possíveis de todas as técnicas descritas na metodologia, o que geraria um grande volume de modelos e métricas para estudar.

A solução adotada, portanto, foi dividir a modelagem em três etapas consecutivamente reduzidas, embora internamente mais detalhadas entre si. Considerando todas as combinações iniciais praticáveis, gerou-se resultados de verdadeiros e falsos positivos ou negativos (VP, VN, FP e FN), acurácia balanceada (Acc_B), sensitividade (Se), precisão (Pr) e pontuação F1 de todos os 168 modelos.

3.3.1 Produção primária geral

Conforme explicado no final da subseção 2.3.4, a medida F1 é a principal discriminação entre os modelos produzidos e, sendo assim, as tabelas 16 e 17 encontram-se ordenadas de forma decrescente pelo mesmo, evidenciando aqueles de melhor desempenho. Logo, encontram-se abaixo as tabelas 16, 17 e 18 com os resultados obtidos:

Tabela 16: Resultados Gerais 1 - Melhores Modelos

Modelo	Cov.	TFIDF	Trim	N-gram	FN	FP	VP	VN	Acc_B	Se	Pr	F1
Árvore	Sim	Não	10%	1	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Árvore	Sim	Não	20%	1	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Árvore	Sim	Sim	10%	1	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Árvore	Sim	Sim	20%	1	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Árvore	Sim	Não	10%	2	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Árvore	Sim	Não	20%	2	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Árvore	Sim	Sim	10%	2	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Árvore	Sim	Sim	20%	2	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Árvore	Sim	Não	10%	1 e 2	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Árvore	Sim	Não	20%	1 e 2	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Árvore	Sim	Sim	10%	1 e 2	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Árvore	Sim	Sim	20%	1 e 2	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Naive Bayes	Sim	Sim	10%	1 e 2	155	318	103	4787	0.66847	0.39922	0.24466	0.30339
Naive Bayes	Sim	Sim	20%	1	149	361	109	4744	0.67588	0.42248	0.23191	0.29945
Naive Bayes	Sim	Sim	10%	1	154	336	104	4769	0.66864	0.40310	0.23636	0.29799
Naive Bayes	Sim	Sim	20%	1 e 2	150	360	108	4745	0.67404	0.41860	0.23077	0.29752
Naive Bayes	Sim	Sim	20%	2	148	372	110	4733	0.67674	0.42636	0.22822	0.29730
Naive Bayes	Sim	Sim	10%	2	148	376	110	4729	0.67635	0.42636	0.22634	0.29570
Naive Bayes	Sim	Não	20%	2	139	459	119	4646	0.68566	0.46124	0.20588	0.28469
Naive Bayes	Não	Não	-	1	178	258	80	4847	0.62977	0.31008	0.23669	0.26846
Naive Bayes	Sim	Não	10%	2	134	557	124	4548	0.68576	0.48062	0.18209	0.26411
Árvore	Não	Não	20%	1	178	362	80	4743	0.61958	0.31008	0.18100	0.22857
Árvore	Não	Não	20%	1 e 2	178	362	80	4743	0.61958	0.31008	0.18100	0.22857
Naive Bayes	Não	Não	10%	1 e 2	157	693	101	4412	0.62786	0.39147	0.12720	0.19202
Naive Bayes	Não	Não	10%	1	152	760	106	4345	0.63099	0.41085	0.12240	0.18861
Naive Bayes	Não	Não	-	2	196	339	62	4766	0.58695	0.24031	0.15461	0.18816
Naive Bayes	Não	Sim	10%	1 e 2	131	1092	127	4013	0.63917	0.49225	0.10418	0.17197
Logístico	Sim	Não	10%	1 e 2	233	10	25	5095	0.54747	0.09690	0.71429	0.17065
Elastic Net	Sim	Sim	10%	1 e 2	233	10	25	5095	0.54747	0.09690	0.71429	0.17065
Logístico	Sim	Sim	20%	1 e 2	233	10	25	5095	0.54747	0.09690	0.71429	0.17065
Naive Bayes	Não	Não	20%	1 e 2	153	881	105	4224	0.61720	0.40698	0.10649	0.16881
Naive Bayes	Não	Sim	10%	1	127	1181	131	3924	0.63821	0.50775	0.09985	0.16688
LASSO	Sim	Sim	20%	1	234	9	24	5096	0.54563	0.09302	0.72727	0.16495
Elastic Net	Sim	Sim	20%	1	234	10	24	5095	0.54553	0.09302	0.70588	0.16438

Tabela 17: Resultados Gerais 2 - Modelos Medianos

Modelo	Cov.	TFIDF	Trim	N-gram	FN	FP	VP	VN	Acc_B	Se	Pr	F1
Logístico	Sim	Sim	10%	2	234	10	24	5095	0.54553	0.09302	0.70588	0.16438
Naive Bayes	Sim	Não	-	1	219	186	39	4919	0.55736	0.15116	0.17333	0.16149
LASSO	Sim	Não	20%	1	235	7	23	5098	0.54389	0.08915	0.76667	0.15972
Naive Bayes	Não	Não	20%	1	156	921	102	4184	0.60747	0.39535	0.09971	0.15925
Logístico	Sim	Não	10%	2	235	8	23	5097	0.54379	0.08915	0.74194	0.15917
Logístico	Sim	Não	20%	2	235	8	23	5097	0.54379	0.08915	0.74194	0.15917
Logístico	Sim	Sim	10%	1	235	10	23	5095	0.54359	0.08915	0.69697	0.15808
Elastic Net	Sim	Sim	10%	1	235	10	23	5095	0.54359	0.08915	0.69697	0.15808
LASSO	Sim	Sim	10%	1	235	10	23	5095	0.54359	0.08915	0.69697	0.15808
Elastic Net	Sim	Não	10%	2	235	10	23	5095	0.54359	0.08915	0.69697	0.15808
Naive Bayes	Sim	Não	20%	1 e 2	187	573	71	4532	0.58148	0.27519	0.11025	0.15743
Naive Bayes	Não	Sim	20%	1 e 2	125	1390	133	3715	0.62161	0.51550	0.08733	0.14935
Naive Bayes	Não	Sim	20%	1	124	1413	134	3692	0.62130	0.51938	0.08662	0.14848
Logístico	Sim	Não	-	1	237	5	21	5100	0.54021	0.08140	0.80769	0.14789
Logístico	Sim	Sim	10%	1 e 2	237	6	21	5099	0.54011	0.08140	0.77778	0.14737
Naive Bayes	Sim	Não	20%	1	190	600	68	4505	0.57302	0.26357	0.10180	0.14687
Elastic Net	Sim	Sim	10%	2	238	5	20	5100	0.53827	0.07752	0.80000	0.14134
Elastic Net	Sim	Não	10%	1 e 2	238	5	20	5100	0.53827	0.07752	0.80000	0.14134
Logístico	Sim	Não	10%	1	238	6	20	5099	0.53817	0.07752	0.76923	0.14085
Naive Bayes	Sim	Não	10%	1 e 2	192	616	66	4489	0.56757	0.25581	0.09677	0.14043
LASSO	Sim	Não	10%	1 e 2	238	7	20	5098	0.53807	0.07752	0.74074	0.14035
Naive Bayes	Sim	Não	10%	1	192	625	66	4480	0.56669	0.25581	0.09551	0.13909
Logístico	Sim	Sim	-	1	239	4	19	5101	0.53643	0.07364	0.82609	0.13523
Elastic Net	Sim	Sim	-	1	239	4	19	5101	0.53643	0.07364	0.82609	0.13523
LASSO	Sim	Sim	-	1	239	4	19	5101	0.53643	0.07364	0.82609	0.13523
LASSO	Sim	Sim	-	2	239	4	19	5101	0.53643	0.07364	0.82609	0.13523
Logístico	Sim	Sim	-	1 e 2	239	4	19	5101	0.53643	0.07364	0.82609	0.13523
LASSO	Sim	Sim	-	1 e 2	239	4	19	5101	0.53643	0.07364	0.82609	0.13523
Elastic Net	Sim	Não	10%	1	239	4	19	5101	0.53643	0.07364	0.82609	0.13523
LASSO	Sim	Não	10%	1	239	4	19	5101	0.53643	0.07364	0.82609	0.13523
Logístico	Sim	Não	20%	1	239	4	19	5101	0.53643	0.07364	0.82609	0.13523
Elastic Net	Sim	Não	20%	1	239	4	19	5101	0.53643	0.07364	0.82609	0.13523
LASSO	Sim	Não	10%	2	239	4	19	5101	0.53643	0.07364	0.82609	0.13523
Elastic Net	Sim	Não	20%	2	239	4	19	5101	0.53643	0.07364	0.82609	0.13523
LASSO	Sim	Não	20%	2	239	4	19	5101	0.53643	0.07364	0.82609	0.13523
LASSO	Sim	Sim	10%	2	239	4	19	5101	0.53643	0.07364	0.82609	0.13523
Logístico	Sim	Sim	20%	2	239	4	19	5101	0.53643	0.07364	0.82609	0.13523
Elastic Net	Sim	Sim	20%	2	239	4	19	5101	0.53643	0.07364	0.82609	0.13523
Logístico	Sim	Não	20%	1 e 2	239	4	19	5101	0.53643	0.07364	0.82609	0.13523
LASSO	Sim	Não	20%	1 e 2	239	4	19	5101	0.53643	0.07364	0.82609	0.13523
LASSO	Sim	Sim	10%	1 e 2	239	4	19	5101	0.53643	0.07364	0.82609	0.13523
Elastic Net	Sim	Sim	20%	1 e 2	239	4	19	5101	0.53643	0.07364	0.82609	0.13523
LASSO	Sim	Sim	20%	1 e 2	239	4	19	5101	0.53643	0.07364	0.82609	0.13523
LASSO	Sim	Sim	20%	2	239	5	19	5100	0.53633	0.07364	0.79167	0.13475
Elastic Net	Sim	Não	20%	1 e 2	239	5	19	5100	0.53633	0.07364	0.79167	0.13475
Logístico	Sim	Sim	20%	1	239	7	19	5098	0.53614	0.07364	0.73077	0.13380
Elastic Net	Sim	Não	-	2	241	1	17	5104	0.53285	0.06589	0.94444	0.12319
LASSO	Sim	Não	-	1 e 2	241	1	17	5104	0.53285	0.06589	0.94444	0.12319
Elastic Net	Sim	Não	-	1 e 2	241	4	17	5101	0.53255	0.06589	0.80952	0.12186
Naive Bayes	Não	Não	10%	2	117	1939	141	3166	0.58334	0.54651	0.06779	0.12062
Naive Bayes	Não	Sim	10%	2	106	2289	152	2816	0.57038	0.58915	0.06227	0.11263
Logístico	Sim	Não	-	2	243	1	15	5104	0.52897	0.05814	0.93750	0.10949
LASSO	Sim	Não	-	2	243	1	15	5104	0.52897	0.05814	0.93750	0.10949
Logístico	Sim	Não	-	1 e 2	243	1	15	5104	0.52897	0.05814	0.93750	0.10949
Naive Bayes	Não	Não	-	1 e 2	243	10	15	5095	0.52809	0.05814	0.60000	0.10601
Elastic Net	Sim	Não	-	1	245	2	13	5103	0.52500	0.05039	0.86667	0.09524
Naive Bayes	Não	Sim	20%	2	158	1855	100	3250	0.51211	0.38760	0.05115	0.09038
Elastic Net	Sim	Sim	-	2	246	1	12	5104	0.52316	0.04651	0.92308	0.08856
Aleatório	-	-	-	-	130.4	2554.2	127.6	2550.8	0.49712	0.49457	0.04758	0.08681
Árvore	Não	Não	10%	1	246	13	12	5092	0.52198	0.04651	0.48000	0.08481
Árvore	Não	Não	10%	1 e 2	246	13	12	5092	0.52198	0.04651	0.48000	0.08481
Naive Bayes	Sim	Não	-	2	244	60	14	5045	0.52126	0.05426	0.18919	0.08434
Árvore	Não	Sim	10%	1 e 2	246	17	12	5088	0.52159	0.04651	0.41379	0.08362
Árvore	Não	Sim	10%	1	246	20	12	5085	0.52130	0.04651	0.37500	0.08276
LASSO	Sim	Não	-	1	247	1	11	5104	0.52122	0.04264	0.91667	0.08148
Logístico	Sim	Sim	-	2	247	1	11	5104	0.52122	0.04264	0.91667	0.08148
Elastic Net	Sim	Sim	-	1 e 2	247	1	11	5104	0.52122	0.04264	0.91667	0.08148
Logístico	Não	Não	-	2	247	5	11	5100	0.52083	0.04264	0.68750	0.08029
Naive Bayes	Não	Não	20%	2	184	1620	74	3485	0.48474	0.28682	0.04368	0.07582
LASSO	Não	Não	-	1 e 2	248	5	10	5100	0.51889	0.03876	0.66667	0.07326
Elastic Net	Não	Não	-	2	249	5	9	5100	0.51695	0.03488	0.64286	0.06618
LASSO	Não	Não	-	2	249	5	9	5100	0.51695	0.03488	0.64286	0.06618
Logístico	Não	Não	-	1 e 2	249	5	9	5100	0.51695	0.03488	0.64286	0.06618
Elastic Net	Não	Não	-	1 e 2	249	5	9	5100	0.51695	0.03488	0.64286	0.06618
Naive Bayes	Sim	Não	-	1 e 2	249	21	9	5084	0.51539	0.03488	0.30000	0.06250
Logístico	Não	Não	-	1	254	1	4	5104	0.50765	0.01550	0.80000	0.03042
LASSO	Não	Não	-	1	254	1	4	5104	0.50765	0.01550	0.80000	0.03042
Elastic Net	Não	Sim	10%	1	254	2	4	5103	0.50756	0.01550	0.66667	0.03030
LASSO	Não	Sim	10%	1	254	2	4	5103	0.50756	0.01550	0.66667	0.03030
Logístico	Não	Sim	10%	1 e 2	254	2	4	5103	0.50756	0.01550	0.66667	0.03030
Elastic Net	Não	Sim	10%	1 e 2	254	2	4	5103	0.50756	0.01550	0.66667	0.03030
LASSO	Não	Sim	10%	1 e 2	254	2	4	5103	0.50756	0.01550	0.66667	0.03030
Logístico	Não	Sim	10%	1	254	3	4	5102	0.50746	0.01550	0.57143	0.03019

Tabela 18: Resultados Gerais 3 - Piores Modelos

Modelo	Cov.	TFIDF	Trim	N-gram	FN	FP	VP	VN	Acc_B	Se	Pr	F1
Elastic Net	Não	Não	10%	1 e 2	255	3	3	5102	0.50552	0.01163	0.50000	0.02273
Elastic Net	Não	Não	10%	1	255	4	3	5101	0.50542	0.01163	0.42857	0.02264
Logístico	Não	Não	10%	1	255	5	3	5100	0.50532	0.01163	0.37500	0.02256
LASSO	Não	Não	10%	1	255	5	3	5100	0.50532	0.01163	0.37500	0.02256
Elastic Net	Não	Não	-	1	256	1	2	5104	0.50378	0.00775	0.66667	0.01533
Logístico	Não	Não	10%	1 e 2	256	2	2	5103	0.50368	0.00775	0.50000	0.01527
LASSO	Não	Não	10%	1 e 2	256	2	2	5103	0.50368	0.00775	0.50000	0.01527
Naive Bayes	Não	Sim	-	2	256	23	2	5082	0.50162	0.00775	0.08000	0.01413
Logístico	Não	Sim	-	1	257	1	1	5104	0.50184	0.00388	0.50000	0.00769
Naive Bayes	Não	Sim	-	1 e 2	258	1	0	5104	0.49990	0.00000	0.00000	0.00000
Demais	-	-	-	-	258	0	0	5105	0.50000	0.00000	-	-

Nas três tabelas, conseguimos extrair e sumarizar diversas informações:

- De ponto de vista estatístico geral, todos os modelos possuem rendimento fraco, com baixos valores de sensitividade, precisão e F1. Mesmo assim, vemos que diversos modelos conseguiram um resultado substancialmente superior ao modelo aleatório;
- Os modelos de árvore e naive bayes foram os que tiveram o melhor desempenho via F1 no geral, sendo consideravelmente superiores aos de regressão;
- Não se percebe uma grande diferença dentre os modelos de regressão, com os três sempre ficando próximos em eficiência;
- Em primeiro momento, a presença das covariáveis e de filtragem parecem melhorar a predição, dada a sua concentração nos melhores modelos da tabela 16;
- Na contrapartida, o uso de TF-IDF e o tipo de n-gram não parecem influenciar muito na performance, com ambos encontrando-se bastante dispersos em toda as tabelas;
- Mesmo que os modelos de árvore consigam um balanço melhor entre sensitividade e precisão, sendo portanto melhores, os naive bayes conseguem atingir números maiores de verdadeiros positivos (VP), com o valor máximo de 152 projetos;
- Dos 168 modelos produzidos, 42 não conseguiram prever nem um projeto aprovado sequer, e estão agrupados na categoria "Demais"da coluna "Modelo"por motivos de simplificação. Todos os tipos de modelos, filtro e n-grams estão inclusos, bem como a ausência e presença de TF-IDF e covariáveis. Ainda assim, a informação mais notável é que 39 desses modelos não possuem a presença de covariáveis, sugerindo sua importância na modelagem.
- Certos modelos mais simples tiveram alguns dos mais satisfatórios resultados, figurando nos 20 melhores modelos. No caso, refere-se ao Naive Bayes sem nenhuma transformação na matriz 1-gram e ao modelo de árvore com o corte de 20%, alcançando respectivamente um F1 próximo de 0.27 e 0.23.

3.3.2 Estudo das técnicas de matrizes

Apesar das elucidações descritas na subseção anterior, ainda é necessário um aprofundamento nas questões referentes aos procedimentos aplicados nas matrizes, nominalmente as covariáveis, TF-IDF, filtragem e n-grams. Nesse caso, será buscado arbitrariamente modelos com apenas uma diferença nos esquemas aplicados, mas com performances vastamente discrepantes. Nesse intuito, espera-se explicar o impacto individual da técnica específica na modelagem.

1. **Filtragem**

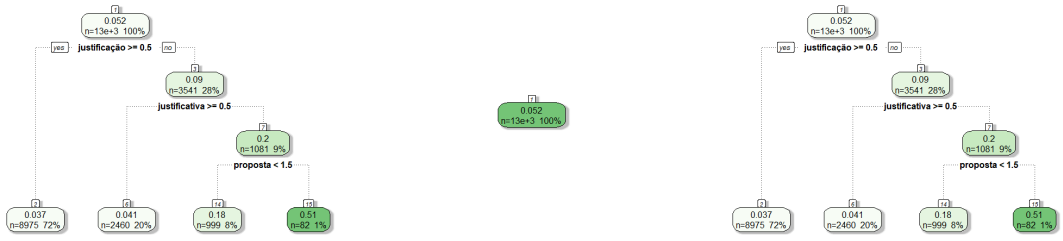
   O primeiro caso estudado é justamente sobre os efeitos da filtragem nas diferentes matrizes. Para tal, selecionamos os modelos de árvore sem covariáveis e TF-IDF, relacionando de início apenas os n-grams e mantendo o corte de 20%. Então, temos as seguintes informações:

Tabela 19: Comparação de corte de 20% entre as matrizes

Modelo	Cov.	TFIDF	Trim	N-gram	FN	FP	VP	VN	Acc_B	Se	Pr	F1
Árvore	Não	Não	20%	1	178	362	80	4743	0.61958	0.31008	0.18100	0.22857
Árvore	Não	Não	20%	2	258	0	0	5105	0.50000	0.00000	-	-
Árvore	Não	Não	20%	1 e 2	178	362	80	4743	0.61958	0.31008	0.18100	0.22857

Figura 10: Matrizes de Palavras com filtro de 10%

(a) Matriz 1-gram (b) Matriz 2-grams (c) Matriz (1,2)-grams



De acordo com a tabela 19, o corte de 20% prejudicou substancialmente a matriz com 2-grams, produzindo um modelo sem poder preditivo algum. Além disso, a figura 10 explicita ainda mais o grau do prejuízo, demonstrando que a matriz de 2-grams ficou sem nenhuma variável significativa, enquanto que o modelo com (1,2)-grams é exatamente o mesmo que o da matriz 1-gram.

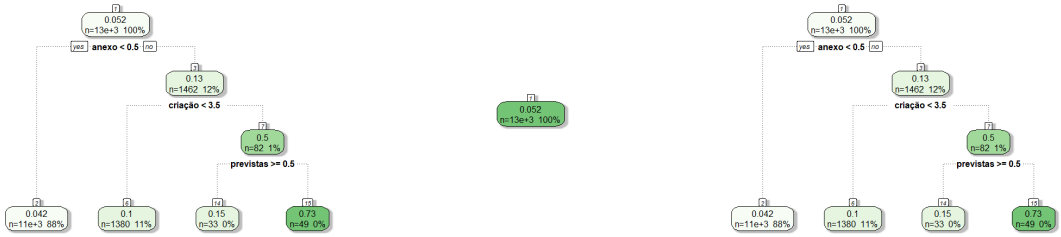
De fato, os modelos descritos exemplificam como o corte de 20% pode ser excessivo. Entretanto, isso também levanta a questão sobre as consequências do corte de 10% e, sendo assim:

Tabela 20: Comparação de corte de 10% entre as matrizes

Modelo	Cov.	TFIDF	Trim	N-gram	FN	FP	VP	VN	Acc_B	Se	Pr	F1
Árvore	Não	Não	10%	1	246	13	12	5092	0.52198	0.04651	0.48000	0.08481
Árvore	Não	Não	10%	2	258	0	0	5105	0.50000	0.00000	-	-
Árvore	Não	Não	10%	1 e 2	246	13	12	5092	0.52198	0.04651	0.48000	0.08481

Figura 11: Matrizes de Palavras com filtro de 10%

(a) Matriz 1-gram (b) Matriz 2-grams (c) Matriz (1,2)-grams



De maneira análoga, vemos aqui a mesma situação descrita no filtro de 20%. Entretanto, a comparação das figuras 10a e 10c com as respectivas 11a e 11c ilustra o cenário oposto, em que uma filtragem maior melhora consideravelmente o desempenho dos modelos, conforme as medidas das tabelas 19 e 20. Isso é, ambos os casos demonstram como a técnica de filtragem pode atrapalhar ou ajudar na modelagem, dependendo do seu uso e contexto.

Outro motivo que reforça a ideia de que o corte controlado dos dados gera resultados positivos é a concentração de matrizes filtradas na tabela 16, dos melhores modelos. Apenas 2 dos 34 modelos presentes não possuem algum tipo de filtragem, ao mesmo tempo que a proporção de matrizes sem corte nas tabelas 17 e 18 é razoavelmente superior.

2. **Covariáveis**

Na subseção 3.3.1, ficou evidente um certo favorecimento da presença das covariáveis nos modelos, com muitos dos melhores modelos contendo-as, e os piores sendo prejudicados em sua ausência.

Ainda assim, desenvolveu-se um estudo com todos os tipos de modelo, alternando a existência das variáveis e controlando-se as demais opções. No caso, optou-se por fixar o filtro em 10%, utilizar (1,2)-grams e não optar pelo TF-IDF. Logo, temos a tabela 21 com as métricas:

Tabela 21: Efeitos das Covariáveis

Modelo	Cov.	TFIDF	Trim	N-gram	FN	FP	VP	VN	Acc_B	Se	Pr	F1
Logístico	Sim	Não	10%	1 e 2	233	10	25	5095	0.54747	0.09690	0.71429	0.17065
Logístico	Não	Não	10%	1 e 2	256	2	2	5103	0.50368	0.00775	0.50000	0.01527
Elastic Net	Sim	Não	10%	1 e 2	238	5	20	5100	0.53827	0.07752	0.80000	0.14134
Elastic Net	Não	Não	10%	1 e 2	255	3	3	5102	0.50552	0.01163	0.50000	0.02273
LASSO	Sim	Não	10%	1 e 2	238	7	20	5098	0.53807	0.07752	0.74074	0.14035
LASSO	Não	Não	10%	1 e 2	256	2	2	5103	0.50368	0.00775	0.50000	0.01527
Naive Bayes	Sim	Não	10%	1 e 2	192	616	66	4489	0.56757	0.25581	0.09677	0.14043
Naive Bayes	Não	Não	10%	1 e 2	157	693	101	4412	0.62786	0.39147	0.12720	0.19202
Árvore	Sim	Não	10%	1 e 2	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Árvore	Não	Não	10%	1 e 2	246	13	12	5092	0.52198	0.04651	0.48000	0.08481

Portanto, contemplamos que na maioria dos tipos de modelos as covariáveis tendem a aumentar consideravelmente a qualidade e poder preditivo dos mesmos, dado a comparação da pontuação F1 e número de verdadeiros positivos.

Contudo, o exemplar de Naive Bayes parece ser um ponto fora da curva, tendo uma boa performance sem covariáveis e piorando na sua presença, o que é bastante curioso. Entretanto, um rápido olhar na tabela 16 demonstra que os melhores modelos Naive Bayes tem, de fato, covariáveis em sua matriz; ainda assim, existe também uma razoável quantidade de Naive Bayes sem covariáveis e que possuem bom desempenho. Por consequência, sugere-se então que a melhoria ou piora da eficiência do Naive Bayes pelas covariáveis não é tão superficial, sendo dependente do contexto das demais configurações.

Como já demonstrado, o corte de 20% nas matrizes não interage bem com a matriz de 2-grams, mas apresenta melhoria na de 1-gram. Nesse raciocínio, aplicou-se as covariáveis nas matrizes de 1-gram e 2-grams, visando descobrir o efeito de interação em dados desfavorecidos e favorecidos pelo filtro.

Tabela 22: Efeitos das Covariáveis no Naive Bayes

Modelo	Cov.	TFIDF	Trim	N-gram	FN	FP	VP	VN	Acc_B	Se	Pr	F1
Naive Bayes	Sim	Não	20%	2	139	459	119	4646	0.68566	0.46124	0.20588	0.28469
Naive Bayes	Não	Não	20%	2	184	1620	74	3485	0.48474	0.28682	0.04368	0.07582
Naive Bayes	Sim	Não	20%	1	190	600	68	4505	0.57302	0.26357	0.10180	0.14687
Naive Bayes	Não	Não	20%	1	156	921	102	4184	0.60747	0.39535	0.09971	0.15925

Pela tabela 22, confirmamos que de fato, as covariáveis tem efeito distintos no Naive Bayes dependendo da matriz associada. No caso de dados de baixa qualidade, sua adição eleva o modelo à um dos melhores patamares alcançados, enquanto que no modelo com perfomance aceitável e com uma matriz de boa qualidade, adicioná-las só poluirá o Naive Bayes com mais informação e piorá-lo.

3. **TF-IDF**

Similarmente às demais técnicas já abordadas, iremos agora comparar o impacto do TF-IDF em todos os tipos de modelos aplicados, controlando os demais procedimentos. Na subseção 3.3.1, os recursos tabelares presentes indicaram que o TF-IDF

não tem grande impacto de maneira geral, trazendo poucas mudanças nas métricas finais dos modelos.

Sendo assim, escolheu-se matrizes sem covariáveis, com corte de 20% e de 1-gram, e observou-se o comportamento dos 5 tipos de modelos com e sem a implementação de TF-IDF.

Tabela 23: Efeitos do TF-IDF

Modelo	Cov.	TFIDF	Trim	N-gram	FN	FP	VP	VN	Acc_B	Se	Pr	F1
Árvore	Não	Não	20%	1	178	362	80	4743	0.61958	0.31008	0.18100	0.22857
Árvore	Não	Sim	20%	1	258	0	0	5105	0.50000	0.00000	-	-
Naive Bayes	Não	Não	20%	1	156	921	102	4184	0.60747	0.39535	0.09971	0.15925
Naive Bayes	Não	Sim	20%	1	124	1413	134	3692	0.62130	0.51938	0.08662	0.14848
Logístico	Não	Não	20%	1	258	0	0	5105	0.50000	0.00000	-	-
Logístico	Não	Sim	20%	1	258	0	0	5105	0.50000	0.00000	-	-
LASSO	Não	Não	20%	1	258	0	0	5105	0.50000	0.00000	-	-
LASSO	Não	Sim	20%	1	258	0	0	5105	0.50000	0.00000	-	-
Elastic Net	Não	Não	20%	1	258	0	0	5105	0.50000	0.00000	-	-
Elastic Net	Não	Sim	20%	1	258	0	0	5105	0.50000	0.00000	-	-

No caso, a informação extraída da tabela 23 é que em modelos ruins, o TF-IDF é simplesmente incapaz de melhorá-lo, produzindo o mesmo resultado. Entretanto, o mais preocupante é sua capacidade de piorar modelos aceitáveis, como foi o caso do Naive Bayes e especialmente o de árvore.

Vale ressaltar, é claro, que essa pequena análise não invalida completamente o TF-IDF. Conforme ilustrado na tabela 24 abaixo, ao incluir as covariáveis o TF-IDF tende a apresentar efeitos melhores, especialmente no Naive Bayes.

Tabela 24: Efeitos do TF-IDF

Modelo	Cov.	TFIDF	Trim	N-gram	FN	FP	VP	VN	Acc_B	Se	Pr	F1
Árvore	Sim	Não	20%	1	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Árvore	Sim	Sim	20%	1	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Naive Bayes	Sim	Não	20%	1	190	600	68	4505	0.57302	0.26357	0.10180	0.14687
Naive Bayes	Sim	Sim	20%	1	149	361	109	4744	0.67588	0.42248	0.23191	0.29945
Logístico	Sim	Não	20%	1	239	4	19	5101	0.53643	0.07364	0.82609	0.13523
Logístico	Sim	Sim	20%	1	239	7	19	5098	0.53614	0.07364	0.73077	0.13380
LASSO	Sim	Não	20%	1	235	7	23	5098	0.54389	0.08915	0.76667	0.15972
LASSO	Sim	Sim	20%	1	234	9	24	5096	0.54563	0.09302	0.72727	0.16495
Elastic Net	Sim	Não	20%	1	239	4	19	5101	0.53643	0.07364	0.82609	0.13523
Elastic Net	Sim	Sim	20%	1	234	10	24	5095	0.54553	0.09302	0.70588	0.16438

Ainda assim, vê-se também pequenas melhorias no LASSO e Elastic Net, uma quase insignificante degradação do Logístico e indiferença na árvore. Logo, pode-se considerar seguro afirmar que, em boa parcela das circunstâncias, o TF-IDF será indiferente ou piorará o modelo.

4. **N-grams**

Por fim, temos o estudo dedicado à influência dos *n-grams*. Semelhantemente ao TF-IDF, identificou-se em primeiro momento que os *n-grams* não aparentavam trazer muitas diferenças entre os modelos, resultando em desempenhos muito parecidos

entre 1-gram, 2-grams e (1,2)-grams. Além disso, também constatou-se na análise de filtragem que alguns modelos de 2-grams são muito prejudicados no processo.

Nesse sentido, montou-se duas abordagens para investigação dos *n-grams*: uma com a presença de corte 10% que inclui o modelo de árvore, e a outra sem nenhum tipo de filtro. Assim sendo, os resultados encontram-se organizados nas respectivas tabelas 25 e 26 abaixo:

Tabela 25: Efeitos dos n-grams com filtro

Modelo	Cov.	TFIDF	Trim	N-gram	FN	FP	VP	VN	Acc_B	Se	Pr	F1
Árvore	Sim	Não	10%	1	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Árvore	Sim	Não	10%	2	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Árvore	Sim	Não	10%	1 e 2	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Naive Bayes	Sim	Não	10%	1	192	625	66	4480	0.56669	0.25581	0.09551	0.13909
Naive Bayes	Sim	Não	10%	2	134	557	124	4548	0.68576	0.48062	0.18209	0.26411
Naive Bayes	Sim	Não	10%	1 e 2	192	616	66	4489	0.56757	0.25581	0.09677	0.14043
Logístico	Sim	Não	10%	1	238	6	20	5099	0.53817	0.07752	0.76923	0.14085
Logístico	Sim	Não	10%	2	235	8	23	5097	0.54379	0.08915	0.74194	0.15917
Logístico	Sim	Não	10%	1 e 2	233	10	25	5095	0.54747	0.09690	0.71429	0.17065
LASSO	Sim	Não	10%	1	239	4	19	5101	0.53643	0.07364	0.82609	0.13523
LASSO	Sim	Não	10%	2	239	4	19	5101	0.53643	0.07364	0.82609	0.13523
LASSO	Sim	Não	10%	1 e 2	238	7	20	5098	0.53807	0.07752	0.74074	0.14035
Elastic Net	Sim	Não	10%	1	239	4	19	5101	0.53643	0.07364	0.82609	0.13523
Elastic Net	Sim	Não	10%	2	235	10	23	5095	0.54359	0.08915	0.69697	0.15808
Elastic Net	Sim	Não	10%	1 e 2	238	5	20	5100	0.53827	0.07752	0.80000	0.14134

Tabela 26: Efeitos dos n-grams sem filtro

Modelo	Cov.	TFIDF	Trim	N-gram	FN	FP	VP	VN	Acc_B	Se	Pr	F1
Naive Bayes	Sim	Não	-	1	219	186	39	4919	0.55736	0.15116	0.17333	0.16149
Naive Bayes	Sim	Não	-	2	244	60	14	5045	0.52126	0.05426	0.18919	0.08434
Naive Bayes	Sim	Não	-	1 e 2	249	21	9	5084	0.51539	0.03488	0.30000	0.06250
Logístico	Sim	Não	-	1	237	5	21	5100	0.54021	0.08140	0.80769	0.14789
Logístico	Sim	Não	-	2	243	1	15	5104	0.52897	0.05814	0.93750	0.10949
Logístico	Sim	Não	-	1 e 2	243	1	15	5104	0.52897	0.05814	0.93750	0.10949
LASSO	Sim	Não	-	1	247	1	11	5104	0.52122	0.04264	0.91667	0.08148
LASSO	Sim	Não	-	2	243	1	15	5104	0.52897	0.05814	0.93750	0.10949
LASSO	Sim	Não	-	1 e 2	241	1	17	5104	0.53285	0.06589	0.94444	0.12319
Elastic Net	Sim	Não	-	1	245	2	13	5103	0.52500	0.05039	0.86667	0.09524
Elastic Net	Sim	Não	-	2	241	1	17	5104	0.53285	0.06589	0.94444	0.12319
Elastic Net	Sim	Não	-	1 e 2	241	4	17	5101	0.53255	0.06589	0.80952	0.12186

Observando primeiramente a tabela 25, nota-se que os três diferentes tipos de n-grams não causaram muito impacto positivo ou negativo nos modelos em geral, com poucas mudanças nas métricas, além de ser insignificante nos de árvore. A única exceção que se destaca é o Naives Bayes de 2-grams, que é completamente superior aos seus análogos em todos os sentidos. Tal *outlier* pode indicar uma interação mais complexa com o Naive Bayes, que deve ser observada na tabela sucessiva.

Em relação à tabela 26, também não se percebe muitas distinções entre as categorias de *n-grams*, principalmente no LASSO e Elastic Net. Com os modelos Naive Bayes e Logístico, a interação é um pouco mais curiosa, em que o 1-gram aparece como a opção superior; já na tabela anterior, os modelos análogos de 1-gram eram os piores.

Tendo em vista o que foi exposto, os indícios apontam para duas características que explicam os resultados apresentados: generalizando, os *n-grams* não possuem grande

impacto, mas eles podem ser capazes de aumentar a quantidade informação das matrizes. O referido potencial, embora improvável de ocorrer, pode ser benéfico ou prejudicial, dependendo das demais configurações escolhidas, como no caso extremo do Naive Bayes.

3.3.3 Análise dos melhores modelos

Na última etapa do processo de modelagem, buscamos eleger os melhores modelos produzidos na subseção 3.3.1, estudando-os com o conhecimento adquirido da subseção 3.3.2, além da apresentação de novas métricas e características dos modelos selecionados.

Conforme exposto na subseção 3.3.1, os modelos de regressão foram os de pior eficiência, constituindo apenas 5 dos 34 modelos na tabela 16, com todos próximo ao final da mesma. Mesmo dentre os dois melhores modelo regressivos, eles atingem uma pontuação F1 pouco acima da metade do valor máximo da tabela; além disso, também acertam apenas 25 verdadeiros positivos, em contraste com os 90 e mais de 100 alcançados pelas árvores e naive bayes, respectivamente. Portanto, todos os modelos de regressão estão excluídos das análises feitas na presente seção, pois as evidências já apresentadas desclassificam os mesmos de serem considerados como a melhor alternativa.

Sendo assim, os candidatos restantes são apenas os modelos de árvore e naive bayes. Em primeira vista superficial, parece lógico optar pelas árvores simplesmente, já que são os modelos que possuem a melhor pontuação F1; contudo, um olhar mais detalhado revela algumas vantagens do naive bayes. A pontuação F1 superior dos modelos de árvore aponta que, no geral, eles são mais "confiáveis"que o naive bayes, oferecendo um balanço melhor entre sensitividade e precisão.

Em contrapartida, o naive bayes se apresenta como uma alternativa mais "arriscada", sendo mais sensititivo e ao mesmo tempo menos preciso em relação aos projetos aprovados na Câmara. Por consequência, tal fato se traduz em um número razoavelmente maior de verdadeiros positivos, que é o fator mais importante para o presente trabalho.

Logo, existe um dilema a ser respondido na escolha do melhor tipo de modelo: "o quanto vale a pena sacrificar a confiança e precisão do modelo por um número maior de verdadeiros positivos?". As seguintes análises, portanto, irão aprofundar nos modelos candidatos, observando suas estruturas, resultados e métricas e, como produto final, definir a resposta do referido questionamento.

- **Modelos de Árvore**

  Dentre os modelos de árvore, existem 12 candidatos elegíveis como "melhor", apresentados na tabela 27 de resumo.

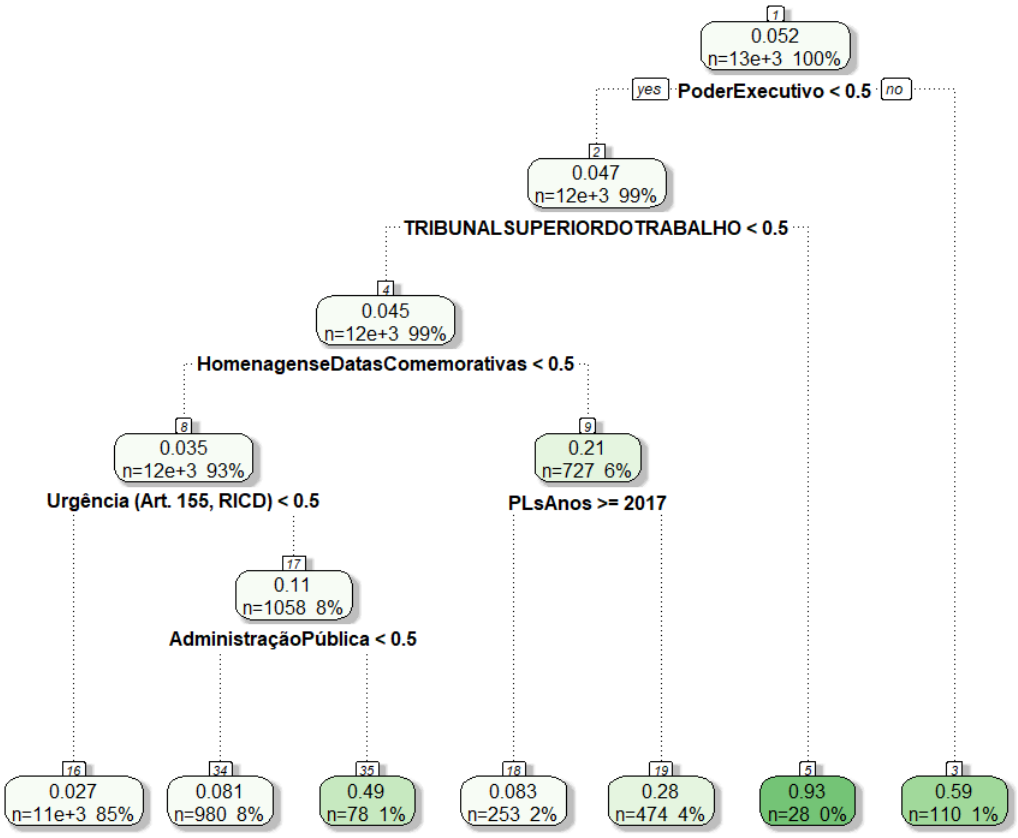
Tabela 27: Melhores Modelos de Árvore

Modelo	Cov.	TFIDF	Trim	N-gram	FN	FP	VP	VN	Acc_B	Se	Pr	F1
Árvore	Sim	Não	10%	1	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Árvore	Sim	Não	20%	1	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Árvore	Sim	Sim	10%	1	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Árvore	Sim	Sim	20%	1	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Árvore	Sim	Não	10%	2	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Árvore	Sim	Não	20%	2	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Árvore	Sim	Sim	10%	2	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Árvore	Sim	Sim	20%	2	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Árvore	Sim	Não	10%	1 e 2	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Árvore	Sim	Não	20%	1 e 2	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Árvore	Sim	Sim	10%	1 e 2	168	194	90	4911	0.65542	0.34884	0.31690	0.33210
Árvore	Sim	Sim	20%	1 e 2	168	194	90	4911	0.65542	0.34884	0.31690	0.33210

Conforme constatado, todos possuem exatamente a mesma performance, o que é curioso. Não apenas isso, percebe-se também que eles são compostos por todas as combinações possíveis de matrizes com covariáveis. Dado a descoberta da importâncias das covariáveis na subseção 3.3.2, tal fato levanta a suspeita desse componente estar influenciando totalmente o modelo e tornando os demais irrelevantes.

Com o intuito de averiguar as suspeitas iniciais, investigou-se a estrutura e fatores significativos dos 12 modelos de árvore. De fato, todos os modelos possuem o mesmo formato apresentado diretamente abaixo:

Figura 12: Melhor Modelo de Árvore



Além de descobrirmos que todos os modelos são equivalentes entre si, confirma-se também a suspeita de "dominação"das covariáveis, dado que todos os elementos significativos dos modelos são oriundos das mesmas. Mais detalhadamente, pode se interpretar a árvore construída nos seguintes pontos:

- Na raiz da árvore, identificada pelo nó "1", temos as descrições básicas do conjunto de treinamento, com aproximadamente 13 mil projetos analisados (12.516, de forma exata) e a proporção de 5% PLs aprovados.
- O primeiro fator significativo identificado pelo modelo é a presença da palavra da matriz "PoderExecutivo", que na verdade é uma categoria da variável "tipoAutor"dos dados originais.

  Como essa "palavra"da matriz é uma variável indicativa, o valor 1 representa um projeto do Poder Executivo, e 0 caso contrário. Portanto, a condição "PoderExecutivo < 0.5"traduz-se como "O projeto não é do Poder Executivo".

  Então, como a negação da condição está na direita da divisão, o modelo identificou que projetos de lei oriundos do Executivo podem ser classificados diferente dos demais, possuindo probabilidade empírica de 59% de aprovação dentre os 110 examinados.
- Após a divisão entre projetos do Poder Executivo e demais, os modelos distinguiram novamente entre os autores, dessa vez do Tribunal Superior do Trabalho (TST). Nesse caso, 26 dos 28 PLs estudados foram aprovados, o que significa uma incrível proporção de 93% de aprovação. À vista disso, o nó terminal "5"é de longe o que melhor apresenta a discriminação dos demais.
- Dado que os projetos não são assinados pelo Poder Executivo ou o TST, a próxima separação é dada pelos projetos de tema "Homenagens e Datas Comemorativas", com possibilidades de aprovação próximas a 20%.

  Além disso, caso o projeto tenha sido feito a partir de 2017, ele possui ainda mais chances de aprovação, alcançando uma proporção de 28% de aprovados, em contraste com os apenas 8.3% de PLs mais antigos.

  Por fim, nota-se também que a quantidade de projetos de "Homenagem e Datas Comemorativas"produzidos a partir de 2017 e até o fim de 2020 é quase o dobro dos anos anteriores, representando 4% de todos os projetos circulados na Câmara na última década. Além de curioso, tal fato também levanta questionamentos sobre as tendências nos anos mais recentes, e no que de fato nossos representantes têm utilizado de seu tempo e recursos financeiros.
- Prosseguindo na árvore, o penúltimo sub-nó é separado pelo regime de tramitação de urgência específico ao artigo nº 155 do Regime Interno da Câmara dos Deputados (RICD). Em seu texto, esse regime é empregado apenas para

projetos importantes e inadiáveis para o Brasil, sendo necessário pedido da maioria absoluta dos deputados.

Nessa divisão, encontra-se o nó terminal da maioria dos projetos estudados, representado pelo nódulo "16". Isso é, se um projeto não se encaixa em nenhuma das outras classificações, ele possui apenas 2.7% de probabilidade de ser aprovado. Embora esperado, esse número é alarmante, mostrando que 85% dos PLs tramitados na Câmara dos Deputados são infrutíferos.

Em contrapartida, projetos de lei tramitados pelo referido regime de urgência possuem chances maiores de serem aprovados, especialmente se forem de tema referente à administração pública. Por sua vez, define-se o tema de "administração pública"como qualquer projeto que tangencie as definições de poder do Estado e de seus agentes. No caso, quase metade dos 78 projetos do nó "35"foram aprovados, em contraste com os 8% que não se classificam nesse tema.

Dada a estrutura do modelo, aplicou-se o mesmo para o conjunto de teste, gerou as seguintes métricas de desempenho:

Tabela 28: Métricas dos melhores modelos de árvore

FN	FP	VP	VN	RNI	Acc	Se	Es	Acc_B	Prev	Prev_D	Pr	F1
168	194	90	4911	0.9519	0.9325	0.34884	0.962	0.65542	0.04811	0.05296	0.3169	0.3321

Por sua vez, a tabela 28 traz os novos pontos a serem ponderados:

- Conforme esperado, a Relação de Não-Informação (RNI) é alta, dado que a maior parte dos projetos são arquivados. Não apenas isso, seu valor também é superior ao da acurácia (Acc), sendo desnecessário o teste exato binomial de comparação;
- O valor aparente de Acc sugeriria um ótimo ajuste do modelo. Porém, como dito na metodologia, ela não é um bom indicativo no contexto do presente trabalho, pois o modelo possui um grande desbalanceio dos dados;
- Embora já tenha sido discutida anteriormente, a sensitividade (Se) ficou próxima de 0.35, o que significa que de todos os projetos verdadeiramente aprovados, o modelo conseguiu detectar corretamente 35% dos casos;
- A especificidade (Es) do modelo alcançou um grande valor de 0.962, ficando relativamente próximo da perfeição. Ainda assim, essa medida não é a mais interessante de se obter bons números, podendo apenas ser considerada como um "bônus"aos demais;

- A acurácia balanceada (Acc_B) fica próxima de 65%, o que parece aceitável. Entretanto, essa medida está sendo muito inflada pela alta especificidade, que no caso do estudo, é menos importante que a sensitividade;
- Por ser complementar à RNI, a Prevalência (Prev) tem resultado igual a 0.04811. Comparando com o valor de 0.052 no conjunto de treino (visto na raiz da árvore), percebe-se que os valores estão próximos, o que reforça a premissa de que ambos os bancos de dados foram divididos com proporções parecidas;
- A Prevalência de Detecção (Prev_D) calculada foi próxima de 0.053, e seria ideal que o mesmo se aproximasse o máximo possível de Prev, o que de fato ocorreu;
- A precisão (Pr) aponta que, dos 1.7% de projetos detectados, o modelo prediz corretamente apenas 31.69%;
- Por fim, os valores de verdadeiros e falsos positivos ou negativos, bem como a pontuação F1, já foram comentados e dispensam novas análises.

A última análise restante a se fazer é, portanto, o teste de McNemar para a concordância do modelo com a realidade. Logo, temos:

$$H_0 : P(B) = P(C)$$

$$H_1 : P(B) \neq P(C)$$

e a estatística do teste calculada χ^2_{Mc}:

$$\chi^2_{Mc} = \frac{(194 - 168)^2}{194 + 168} = 1.8674$$

Com 1 grau de liberdade, encontra-se o p-valor igual à 0.1718. Considerando um nível de significância $\alpha = 0.1$, não temos evidências para rejeitar H_0. Ou seja, os modelos de árvore selecionados estão concordantes com a classificação real dos PLs, o que é um ponto positivo.

Em vista de tudo que foi exposto, conseguimos descrever e apontar detalhadamente a performance de todos os 12 modelos, mas ainda não definiu-se qual deles é o melhor, pois são todos iguais. A resposta para o apontamento do modelo escolhido se dá na forma do conceito de parcimônia, que pode ser resumidamente descrito como "escolher aquele de melhor desempenho com a menor quantidade complicação e informação".

Sob os ideais de parcimônia, a escolha passa a ser quase trivial. Como todos os modelos tem eficiência idêntica e a presença de covariáveis, automaticamente desclassificam-se modelos com TF-IDF, 2-grams e (1,2)-grams; restam então, apenas os dois primeiros modelos dispostos na tabela 27. Por fim, exclui-se o modelo

de corte 10%, já que ele contém um conjunto de dados de tamanho maior do que aquele com filtragem 20%.

Sendo assim, o modelo elegido como melhor modelo de árvore é o que possui covariáveis, corte 20% e 1-gram.

- **Modelos Naive Bayes**

  No caso de modelos Naive Bayes, temos 9 candidatos ao melhor, dispostos na tabela abaixo:

Tabela 29: Melhores Modelos Naive Bayes

Nome	Modelo	Cov.	TFIDF	Trim	N-gram	FN	FP	VP	VN	Acc_B	Se	Pr	F1
1	Naive Bayes	Sim	Sim	10%	1 e 2	155	318	103	4787	0.66847	0.39922	0.24466	0.30339
2	Naive Bayes	Sim	Sim	20%	1	149	361	109	4744	0.67588	0.42248	0.23191	0.29945
3	Naive Bayes	Sim	Sim	10%	1	154	336	104	4769	0.66864	0.40310	0.23636	0.29799
4	Naive Bayes	Sim	Sim	20%	1 e 2	150	360	108	4745	0.67404	0.41860	0.23077	0.29752
5	Naive Bayes	Sim	Sim	20%	2	148	372	110	4733	0.67674	0.42636	0.22822	0.29730
6	Naive Bayes	Sim	Sim	10%	2	148	376	110	4729	0.67635	0.42636	0.22634	0.29570
7	Naive Bayes	Sim	Não	20%	2	139	459	119	4646	0.68566	0.46124	0.20588	0.28469
8	Naive Bayes	Não	Não	-	1	178	258	80	4847	0.62977	0.31008	0.23669	0.26846
9	Naive Bayes	Sim	Não	10%	2	134	557	124	4548	0.68576	0.48062	0.18209	0.26411

Diferentemente do visto nos modelos de árvore, vemos muitas distinções entre os 9 modelos. No mais, pontua-se:

  - Similar as árvores, percebe-se uma grande concentração de covariáveis, mais uma vez reforçando a influência das mesmas. O curioso, no entanto, é que nesse caso existe um modelo que não possui covariáveis e que conseguiu competir com os demais;
  - O TF-IDF se mostra importante no Naive Bayes, apesar de não tão dominante quanto as covariáveis. Na subseção 3.3.2, constatou-se um caso de interação positiva entre o Naive Bayes e TF-IDF, embora parecesse raro. Portanto, a tabela 29 resume todos os casos em que isso ocorre;
  - A filtragem é significativa dada sua concentração na tabela, embora o seu tamanho não pareça influenciar tanto, já que temos um número idêntico de filtros 10% e 20%; além de muitos modelos análogos de performances próximas. No entanto, vale notar o modelo 8, que não possui nenhum tipo de corte, destacando-se dos demais;
  - A quantidade de n-grams distintos está bem equilibrada, embora a concentração aponte que os 2-grams estejam na parte inferior da tabela.

Como temos diferenças de performance entre os modelos, o critério de seleção e análises terão que ser mais complexos do que nas árvores. Porém, é possível já aplicar os conceitos de parcimônia para reduzir a quantidade de modelos, dado que existem alguns análogos entre si e redundantes em desempenho. Nesse critério,

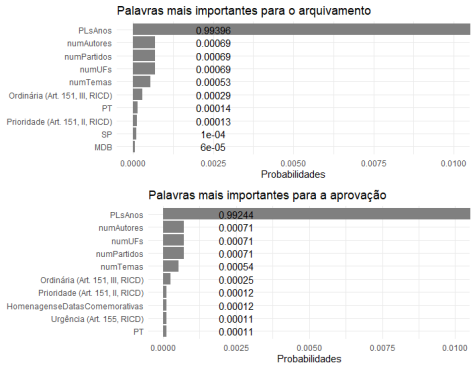
desclassificam-se os modelos 3, 6 e 9, optando apenas pelos respectivos modelos 2, 5, 7.

Dos 6 modelos restantes, elaborou-se pares de gráficos para analisar sua estrutura. Contudo, antes de estudá-los, é importante revisar o entendimento do Naive Bayes. O primeiro passo do modelo é calcular a probabilidade de cada palavra influenciar um projeto a ser arquivado ou aprovado, em que a soma de todas essas probabilidades resulta em 1. Após isso, calcula-se a probabilidade de um projeto ser aprovado ou arquivado, dado as probabilidades do seu conjunto de palavras; então, a probabilidade que for maior definirá a classificação do projeto pelo modelo.

Sendo assim, encontra-se abaixo os gráficos de todos os modelos, ilustrando as maiores probabilidades de arquivamento e aprovação dos 6 modelos ainda elegíveis:

Figura 13: Estrutura dos Melhores Modelos Naive Bayes

(a) Modelo 1



(b) Modelo 2

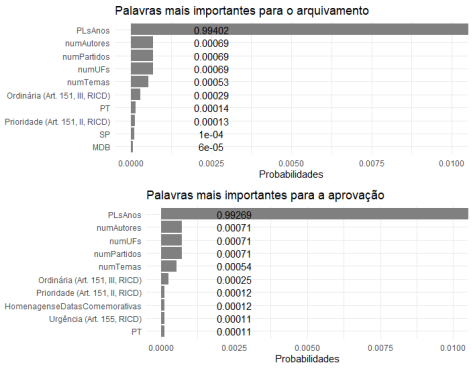
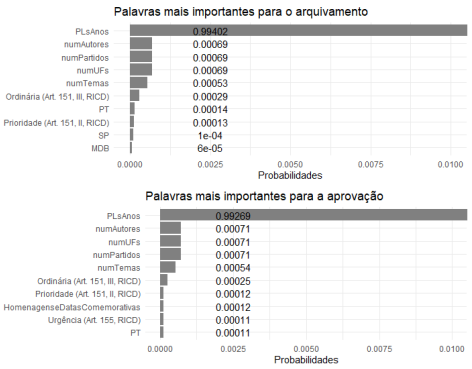
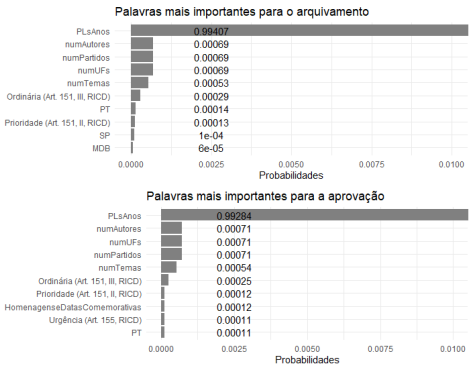


Figura 13: Estrutura dos Melhores Modelos Naive Bayes

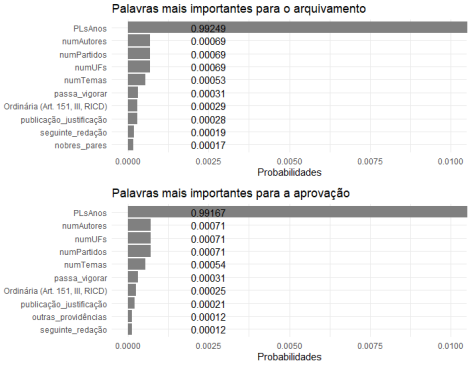
(c) Modelo 4



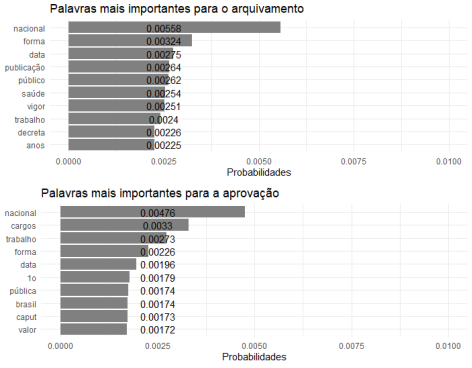
(d) Modelo 5



(e) Modelo 7



(f) Modelo 8



Existem dois pontos vistos rapidamente: a alta probabilidade que a variável "PLsAnos"assume em todos os modelos em que está presente, e as baixas probabilidades das demais palavras. Realmente, era esperado que as probabilidades se dividissem em pequenos valores, dado que existem muitas palavras nas matrizes, e a soma de todas as suas probabilidades obrigatoriamente resulta em 1. Já em relação o porquê do percentual da variável "PLsAnos"ser exponencialmente superior aos demais, é difícil argumentar e compreender, dado que o modelo não é feito para explicar as probabilidades das palavras, e sim dos projetos de lei.

Individualmente, o valor exato da probabilidade de cada palavra é irrelevante. A parte de fato informativa da análise da figura 13 é a comparação entre as palavras importantes para arquivamento com as de aprovação. É exatamente a diferença grande entre os valores das palavras que distingue a classificação dos projetos, e nos ajuda a entender o que é considerado importante para a aprovação dos modelos.

Por exemplo, perceba que 4 dos 5 modelos com covariáveis apresenta o tema "HomenagenseDatasComemorativas"como significativa na aprovação; comparando com as respectivas palavras de arquivamento, sequer vemos a variável no conjunto, o que reflete que sua probabilidade de arquivamento é inferior à 0.000006, em comparação

à 0.00012 de aprovação. Em outras palavras, o fato de um projeto de lei ser de tema "Homenagens e Datas Comemorativas"aumenta muito mais a sua probabilidade de aprovação do que a de arquivamento.

Ainda nos modelos 1, 2, 4 e 5, vemos que o regime de urgência de artigo 155 do RICD aparece em todos como variável importante para a aprovação, enquanto que o fato de os autores serem de São Paulo ou do partido MDB impacta mais no arquivamento. Outra característica dos quatro modelos é que eles sofrem muita interferência das covariáveis, já que todas as "palavras"que enxergamos nos gráficos são, na realidade, variáveis do conjunto de dados original. Por fim, percebe-se que todos esses quatro modelos possuem exatamente a mesma estrutura em seus gráficos, diferindo levemente em algumas das probabilidades. Esse é, portanto, um indicativo que tais modelos são redundantes entre si, e devemos então escolher apenas um, o que será feito após o estudo de eficiência dos mesmos.

No que lhe diz respeito, o modelo 7 se mostra relativamente diferente dos demais, em resultado da ausência de TF-IDF. Vemos enfim, um modelo que utiliza o texto dos projetos de lei como fator significativo para a classificação. No caso, vemos que projetos que contém "outras_providências"tem sua probabilidade de aprovação mais incrementada em relativo à de arquivamento, enquanto que PLs que tratam de "nobres_pares"são mais sugestionados para o arquivamento.

Enfim, temos o mais distinto de todos, o modelo 8. Este é um modelo especial, já que ele foge completamente da curva dos demais, não possuindo covariáveis, TF-IDF e nem sequer filtro. Se estivermos buscando um modelo que utiliza exclusivamente o texto dos projetos de lei, ele é indisputado o melhor de todos, e o fato do mesmo conseguir competir com outros modelos mais complexos é realmente impressionante. Além disso, ele é o modelo que mais possui diferença entre as palavras significativas de arquivamento e aprovação, apontando que PLs que contém "cargos", "1o", "brasil", "caput"e "valor"são mais direcionadas à aprovação. Em contrapartida, projetos com "publicação", "saúde", "vigor", "decreta"e "anos"são mais desviados para o arquivamento.

Agora que compreendemos as estruturas dos 6 modelos de estudo, podemos enfim aplicar os dados de teste e analisar todas as métricas obtidas.

Tabela 30: Métricas dos melhores modelos Naive Bayes

Nome	FN	FP	VP	VN	RNI	Acc	Se	Es	Acc_B	Prev	Prev_D	Pr	F1
1	155	318	103	4787	0.9519	0.9118	0.39922	0.93771	0.66847	0.04811	0.07850	0.24466	0.30339
2	149	361	109	4744	0.9519	0.9049	0.42248	0.92929	0.67588	0.04811	0.08764	0.23191	0.29945
4	150	360	108	4745	0.9519	0.9040	0.41860	0.92948	0.67404	0.04811	0.0876	0.23077	0.29752
5	148	372	110	4733	0.9519	0.903	0.42636	0.92713	0.67635	0.04811	0.08988	0.22822	0.29730
7	139	459	119	4646	0.9519	0.8885	0.46124	0.91009	0.68566	0.04811	0.10778	0.20588	0.28469
8	178	258	80	4847	0.9519	0.9187	0.31008	0.94946	0.62977	0.04811	0.06392	0.23669	0.26846

Constata-se, então, os seguintes pontos:

- O modelo com o maior número de VP, Acc_B, Prev_D e sensitividade é o modelo 7, com a última alcançando quase metade de todos os casos. Apesar de ter os melhores valores nas métricas mais importantes, esses benefícios custam de sua precisão. É o único com *Acc* abaixo de 0.9, e apresenta uma substancial queda de *Pr* em relação aos demais, estando mais próximo de um acerto a cada 5, enquanto os outros modelos possuem precisão aproximada de 1 a cada 4. Sendo assim, embora seja um modelo viável, optou-se por desclassificá-lo por ser pouco confiável;
- Pelo F1, sabemos que o modelo 2 é o segundo mais balanceado da tabela. Em comparação com o modelo 1, vemos que ele consegue incrementar suas medidas mais importantes de *Se*, Acc_B e Prev_D com um ligeiro e aceitável comprometimento da sua precisão. Não apenas isso, ele também é mais simples que o modelo 1, possuindo corte maior e usando apenas 1-gram. Portanto, entre os dois, escolhe-se o modelo 2 como superior;
- O modelo 4 é estritamente inferior ao modelo 2 em todos casos relevantes e, além disso, também é um modelo análogo que contém mais informação de (1,2)-grams. Logo, se exclui o modelo 4 automaticamente;
- O modelo 5 é muito próximo do modelo 2, também apresentando valores parecidos. Entretanto, além de sua maior complexidade, ele começa a trocar sua precisão à um nível não tão aceitável, ficando próximo do ponto médio entre um acerto a cada 4 e um acerto a cada 5, por apenas um acerto a mais. Então, entende-se que o modelo 5 oferece poucas vantagens por uma maior complicação, e opta-se por manter o modelo 2;
- Por último, temos o mais exótico de todos, o modelo 8. Ele possui uma diferença considerável de VP, acurácia balanceada e sensitividade em relação aos demais, mas também possui alguns pontos positivos. Além de sua maior simplicidade, sua prevalência de detecção é a que mais se aproxima da verdadeira prevalência, sugerindo melhor concordância com a realidade. Embora sejam apenas vantagens bônus, vale destacar que o modelo 8 é também o de melhor especificidade e *Acc*.

Similarmente aos modelos de árvore, ambos os modelos restantes possuem acurácia inferior à RNI, tornando o teste binomial exato irrelevante. Ainda assim, existe um último teste a ser feito antes da escolha final: o teste de McNemar para concordância do modelo. Logo:

$$H_0 : P(B) = P(C)$$

$$H_1 : P(B) \neq P(C)$$

com as respectivas estatísticas do teste χ^2_{Mc2} e χ^2_{Mc8}:

$$\chi^2_{Mc2} = \frac{(361 - 149)^2}{361 + 149} = 88.125$$

$$\chi^2_{Mc8} = \frac{(258 - 178)^2}{258 + 178} = 14.679$$

Com o cálculo de χ^2_{Mc2}, encontra-se um p-valor de aproximadamente zero, enquanto que com χ^2_{Mc8} temos p-valor de 0.000128 sob H_0. Em ambos os casos, temos evidências suficientes para rejeitar a hipótese nula e acreditar que ambos os modelos estão discordantes da realidade, em um nível de significância de 10%. Isso é, ambos os modelos estão com viés em suas proporções de erro; ao julgar pelos números de FP e FN, constata-se que ambos estão com viés de classificação positiva em relação às classificações negativas.

Desse modo, chegamos ao fim com duas opções equivalentes para o melhor Naive Bayes, nominalmente os modelos 2 e 8. Ambos ostentam estratégias distintas, com o modelo 2 apostando em uma maior complexidade para acertar mais nas áreas importantes, enquanto que o modelo 8 tem uma abordagem mais simples e menos arriscada.

Por um lado, é plausível optar pelo modelo 2 por sua performance superior nas métricas. De outro ponto de vista, pode-se fazer um argumento de que o modelo 8 é superior por ter menos discordância da realidade mas, ainda que vieses sejam indesejados, é possível trabalhar com eles desde que haja cautela, e que estejamos cientes de sua existência. Além disso, o modelo 8 também é um caso mais especial, dado sua distinção dos demais e por incorporar completamente a proposta do presente trabalho, que é predizer a aprovação de PLs na Câmara através do texto de sua redação.

Portanto, a decisão final é de manter ambos os modelos, dada as suas diferentes e complementadoras abordagens.

4 Conclusão

No cerne da motivação do presente trabalho, esteve sempre o desejo de construir um modelo preditivo que fosse capaz de apontar a aprovação ou reprovação de um projeto de lei. Na longa jornada até a conclusão, ouve não apenas muitas dificuldades, mas também muito aprendizado e elucidação de perguntas.

Como qualquer trabalho estatístico, o início se deu na obtenção dos dados, disponibilizados em seu formato "bruto"pela interface de programação de aplicações da Câmara dos Deputados (Câmara dos Deputados, 2021). Escolheu-se por importar todas as informações referentes às proposições legislativas e seus respectivos autores e temas da última década. Após algumas necessárias transformações e limpeza nos dados, essa etapa marcou um ponto crucial no estudo: a definição de que os modelos produzidos se resumiriam a predizer a aprovação dos PLs apenas na Câmara dos Deputados. Tal escolha foi feita após a análise exploratória inicial, que também revelou algumas informações interessantes, como produção e aprovação dos projetos por ano de publicação, tema e partido dos autores.

Após a exploração dos dados estruturados, deu-se início a fase verdadeiramente desafiadora do trabalho, o processamento de linguagem natural. Aplicando as ferramentas de inúmeros pacotes da linguagem R, conseguiu-se extrair o texto dos documentos oficiais das propostas de lei, pré-processar e enfim montar representações computacionais dos mesmos. Das técnicas aplicadas, produziu-se 36 possibilidades diferentes de combinação das mesmas, que seriam então levadas à aguardada parte de modelagem. No entanto, antes disso realizou-se diversos estudos e revisões das matrizes construídas, a fim de excluir palavras vazias até então não identificadas, bem como verificar e aprimorar a "saúde"destes dados.

No fim, as matrizes finais foram as apresentadas na seção 3.2, e atenderam de maneira adequada os propósitos necessários do presente trabalho. O principal aprendizado do PLN foram as palavras e expressões mais frequentes dos projetos de lei, como "saúde", "trabalho", "passa_vigorar"e "poder_executivo"; o entendimento desses termos aponta do que se trata as principais tramitações da Câmara, e gerou coerência com a análise exploratória inicial.

Então, deu-se início para a aguardada fase de modelagem, marcada por exaustivas produções, análises e pesquisas sobre os modelos construídos. Com base na adequação a problemática do trabalho, escolheu-se 5 tipos diferentes de modelos para compôr as possibilidades de construção, sendo eles: Logístico, LASSO, Elastic Net, Naive Bayes e Árvore. No total, realizou-se 168 combinações viáveis de matrizes com modelos, com a única impossibilidade sendo de construção das árvores sem nenhuma filtragem dos dados,

devido ao uso excessivo de memória do computador.

Então, com os todos os modelos gerados, calcularam-se algumas medidas descritivas individuais, com o intuito de identificar aqueles de melhor desempenho. Para tal, escolheram-se algumas métricas julgadas como mais relevantes, culminando nas respectivas tabelas 16, 17 e 18 ordenadas pela pontuação F1, o que permitiu a averiguação de inúmeros pontos sobre os modelos e o rápido agrupamento dos melhores e piores modelos. Dentre eles, destacou-se a performance superior das árvore e do naive bayes em relação aos demais, ainda que todos os modelos tenham demonstrado um rendimento estatisticamente baixo. Por fim, a etapa de produção primária também ressaltou uma aparente significância da filtragem e principalmente das covariáveis, conforme explicitado pelas suas concentrações e ausências nos melhores e piores modelos.

Dentre todas as informações angariadas na primeira subetapa da modelagem, algumas careciam de maior investigação; dessa forma, realizou-se duas novas subetapas da modelagem, vistos nas subseções 3.3.2 e 3.3.3. No estudo das técnicas de matrizes, focou-se principalmente em verificar o impacto individual e coletivo de cada um dos métodos de representação dos textos. Portanto, foi constatado novamente a importância das covariáveis e dos filtros, enquanto que o TF-IDF e n-grams não mostraram-se muito relevantes, embora existissem certas combinações que os potencializavam.

Com todo o conhecimento acumulado até o momento, começou-se então a última subetapa do trabalho, que elegeria o modelo mais adequado dentre os produzidos. Como a superioridade das árvores e do naive bayes já era conhecida, a análise resumiu-se apenas aos de melhor eficiência dentre os dois tipo. No caso das árvores, descobriu-se rapidamente que os 12 elegíveis eram idênticos, e que as covariáveis dominaram completamente a estrutura da árvore, tornando as demais técnicas desprezíveis. Portanto, fez-se necessário a escolha do melhor modelo exclusivamente pelo critério de parcimônia, elegendo então o mais simples de todos, contendo covariáveis, 1-gram e corte de 20%. Dentre todas as suas métricas calculadas, descobriu-se que ele é um modelo "confiável", oferecendo um bom balanço entre sensitividade e precisão, ao mesmo tempo em que é coerente com a realidade pelo teste de McNemar.

Do outro lado, os naive bayes se mostraram modelos mais "arriscados", comprometendo sua precisão em favor de sensitividade mais elevada. Embora 7 dos 9 modelos fossem bem próximos entre si, outra distinção é que eles não são totalmente idênticos, tanto em estrutura quanto em resultados das métricas empregadas. Por sua vez, isso refletiu em regras de desclassificação dos modelos mais robustas, fazendo-se necessário levar em consideração a estrutura e os resultados dos modelos, além do critério de parcimônia. Exatamente por isso, essa análise concluiu que existem dois modelos elegíveis como o melhor, nominalmente o modelo Naive Bayes com covariáveis, TF-IDF, corte 20% e 1-gram; e o modelo 1-gram sem covariáveis, TF-IDF e filtro.

Ao fim, isso leva à pergunta "Qual dos três modelos elegidos é o melhor?"; na verdade, resposta em si depende da abordagem adotada para o problema. Caso o intuito seja simplesmente predizer da melhor forma possível a aprovação dos projetos de lei, a sugestão é trabalhar com modelos de árvore focado nas covariáveis. Também é possível utilizar o naive bayes e obter resultados próximos ou até aparentemente melhores, mas sua inferior precisão e confiabilidade podem comprometer os resultados obtidos. Por outra perspectiva, se a ideia é, assim como nesse projeto, predizer a aprovação de PLs com o texto de sua redação, então o Naive Bayes demonstrou-se absolutamente superior às demais opções.

No mais, a análise dos melhores modelos evidenciou algumas congruências de fatores significantes na aprovação dos projetos, no caso o regime de urgência de artigo 155 e de tema "Homenagens e Datas Comemorativas". Já individualmente, as árvores destacaram a autoria de projetos do poder executivo e do TST para a aprovação, enquanto o naive bayes mostrou que projetos de deputados de SP e do partido MDB parecem mais propensos ao arquivamento. Já em relação as palavras, o naive bayes ilustrou que projetos que abordam "saúde"e "decreto"são mais prováveis de serem arquivadas, enquanto que se conterem conteúdo de "cargos", "brasil"e "valor"possuem chances aumentadas de sucesso.

Mesmo com tudo isso, devemos lembrar que, embora tenha sido possível predizer a aprovação de projetos de lei com a abordagem adotada, os modelos produzidos foram estatisticamente fracos. A suspeita de motivação disso, é que a aprovação de um PL é um fator político, e não objetivo; por exemplo, um projeto pode nunca ser aprovado simplesmente porque o Presidente da Câmara não pauta o projeto, devido à desavenças com o(s) autor(es). O principal apontamento para essa sugestão é o fato das covariáveis terem dominado a maior parte dos modelos, e elas são muito mais representativas dos propósitos políticos do que a redação do PL.

4.1 Sugestões de trabalhos futuros e outras abordagens

Existem inúmeras estratégias diferentes que podem ser adotadas na predição de um projeto de lei. Caso haja interesse de predizer projetos de lei com base no texto, encontram-se abaixo algumas das possibilidades:

- Balanceio dos dados, aproximando a proporção de arquivados e aprovados;
- Utilização de outra variável predita;
- Outras formas de representação de texto, como *support machine vector* (SVM) e *word2vec*;
- Alterações no conjunto de palavras vazias.

Referências

AGRESTI, A. *An Introduction to Categorical Data Analysis*. 3º. ed. University of Florida, United States: John Wiley & Sons Inc., 2019.

BECKER, L. *Algoritmo de Classificação Naive Bayes*. 2019. Website. Último acesso em 25 de abril de 2022. Disponível em: ⟨https://www.organicadigital.com/blog/algoritmo-de-classificacao-naive-bayes/⟩.

BENOIT, K. et al. quanteda: An r package for the quantitative analysis of textual data. *Journal of Open Source Software*, v. 3, n. 30, p. 774, 2018. Último acesso em 26 de março de 2022. Disponível em: ⟨https://quanteda.io⟩.

BREIMAN, L. et al. *Classification and Regression Trees*. 1º. ed. [S.l.]: Chapman & Hall, 1984.

CONOVER, W. J. *Practical Nonparametric Statistics*. 3º. ed. Texas Tech University, United States: John Wiley & Sons Inc., 1999.

Câmara dos Deputados. *Regimento Interno da Câmara dos Deputados*. 1989. Website. Último acesso em 24 de abril de 2022. Disponível em: ⟨https://www2.camara.leg.br/atividade-legislativa/legislacao/regimento-interno-da-camara-dos-deputados/arquivos-1/RICD%20atualizado%20ate%20RCD%2021-2021.pdf⟩.

Câmara dos Deputados. *Dados Abertos da Câmara dos Deputados*. 2021. Website. Último acesso em 09 de agosto de 2021. Disponível em: ⟨https://dadosabertos.camara.leg.br/index.html⟩.

DEVAY, A. *Modelos de Predição — Regressão de Ridge e Lasso*. 2019. Último acesso em 26 de março de 2022. Disponível em: ⟨https://medium.com/turing-talks/turing-talks-20-regress~ao-de-ridge-e-lasso-a0fc467b5629⟩.

DUTTA, N. *Word2Vec For Word Embeddings - A Beginner's Guide*. 2021. Website. Último acesso em 07 de maio de 2022. Disponível em: ⟨https://www.analyticsvidhya.com/blog/2021/07/word2vec-for-word-embeddings-a-beginners-guide/⟩.

EIRÃO, T. A elaboração de projetos de lei: alguns apontamentos à luz da técnica legislativa na câmara dos deputados. *Cajur*, 2016. Último acesso em 28 de agosto de 2021.

EIRÃO, T. A decisão sobre a agenda legislativa da câmara dos deputados: estudo dos projetos de lei apresentados e transformados em lei de 2002 a 2016. *Cajur*, 2018. Último acesso em 28 de agosto de 2021.

FEINERER, I. *Introduction to the tm Package Text Mining in R*. 2020. Website. Último acesso em 26 de março de 2022. Disponível em: ⟨https://cran.r-project.org/web/packages/tm/vignettes/tm.pdf⟩.

FRIEDMAN, J. et al. *glmnet: Lasso and Elastic-Net Regularized Generalized Linear Models*. 2021. Website. Último acesso em 26 de março de 2022. Disponível em: ⟨https://cran.r-project.org/web/packages/glmnet/glmnet.pdf⟩.

KUHN, M. *The caret Package*. 2019. Website. Último acesso em 25 de abril de 2022. Disponível em: ⟨https://topepo.github.io/caret/⟩.

MANNING, C. D.; RAGHAVAN, P.; SCHUTZE, H. *An Introduction to Information Retrieval*. 1º. ed. University of Cambridge, United States: University of Cambridge Press, 2009.

MONTESQUIEU. *O espírito das leis: as formas de governo, a federação, a divisão dos poderes*. [S.l.]: Saraiva, 2010.

OLIVEIRA, D. Compreendendo e prevendo o processo legislativo via ciência de dados. *USP*, 2019. Último acesso em 26 de março de 2022.

OOMS, J. *Text Extraction, Rendering and Converting of PDF Documents*. 2021. Website. Último acesso em 26 de março de 2022. Disponível em: ⟨https://cran.r-project.org/web/packages/pdftools/pdftools.pdf⟩.

República Federativa do Brasil. *Constituição da República Federativa do Brasil*. 1988. Website. Último acesso em 28 de agosto de 2021. Disponível em: ⟨http://www.planalto.gov.br/ccivil_03/constituicao/constituicao.htm⟩.

STANKEVIX, G. *Arvore de Decisão em R*. 2019. Último acesso em 29 de março de 2022. Disponível em: ⟨https://medium.com/@gabriel.stankevix/arvore-de-decis~ao-em-r-85a449b296b2⟩.

THERNEAU, T.; ATKINSON, B.; RIPLEY, B. *rpart: Recursive Partitioning and Regression Trees*. 2022. Website. Último acesso em 29 de março de 2022. Disponível em: ⟨https://cran.r-project.org/web/packages/rpart/rpart.pdf⟩.

VAJJALA, S. et al. *Pratical Natural Language Processing*. [S.l.]: O'Reilly, 2020.

WICKHAM, H. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2016. Disponível em: ⟨https://ggplot2.tidyverse.org⟩.

WICKHAM, H. et al. *dplyr: A Grammar of Data Manipulation*. [S.l.], 2022. Último acesso em 25 de abril de 2022. Disponível em: ⟨https://dplyr.tidyverse.org⟩.

Wikipedia contributors. *Wikipedia, The Free Encyclopedia*. 2022. Último acesso em 29 de março de 2022. Disponível em: ⟨https://en.wikipedia.org/wiki/Main_Page⟩.