

# Relatório do Projeto III

## Introdução

Este projeto consiste em uma análise de dados que tem como objetivo prever se um projeto de lei proposto por um deputado estadual na câmara, será aprovado ou não; para isso, utilizar-se-á um meio de treinamento e *machine learning*, através do uso do Python e dos recursos disponíveis na biblioteca *scikit learn*.

Para fazer tal análise, será levado em conta algumas vertentes de dados, retirados dos dados abertos da câmara dos deputados, baseando-se na própria ementa do projeto (pode-se ver uma descrição completa dos dados no dicionário de dados) Essa análise será feita com dados desde a virada do milênio, em 2000, passando por 5 mudanças na composição da câmara, nas eleições de 2002, 2006, 2010 e 2014, analisando em duas vertentes: a primeira na qual a proposta é arquivada (não vira de fato uma lei) e a segunda na qual a proposta é transformada em norma jurídica.

## Métodos

O primeiro problema enfrentado foi o de que o número de propostas arquivadas é cerca de 10 vezes maior que o número de propostas transformadas em norma jurídica. Para a resolução, foi montado um data frame auxiliar para classificação, com cerca de 2600 projetos de lei, distribuídos igualmente entre arquivados e aprovados, como uma forma de não viciar o classificador. A seguir, montamos o código, da seguinte forma:

Primeiramente, todas as planilhas foram unificadas, montando um DF único. A seguir, foram selecionadas as colunas que serão relevantes à análise de dados, fazendo o mesmo para a proposição: apenas projetos de lei criado por deputados foram selecionados. A seguir, foi utilizado o método descrito, de criar um data frame artificial para obter maior precisão. A seguir, esse data frame foi separado em duas vertentes: uma de treinamento e outra de teste. Feito isso, o classificador em si foi configurado. Para fazer esse classificador, fizemos testes com dois tipos diferentes de Vetorizadores e quatro tipos de classificadores: Count Vectorizer e TF IDF Vectorizer, Multinomial NB, Decision Tree Classifier, Random Forest Classifier e AdaBoost Classifier, e compararmos a acurácia de cada combinação possível de Vectorizer + Classifier. Para melhorar a iteração, utilizou-se um Voting classifier, que para cada uma das combinações, estabelece a melhor possível combinando várias delas. Para isso, estabelecemos um peso para cada um deles de acordo com a confiabilidade que o classifier nos dá, dando peso 2 para Multinomial, 1 para Decision Tree e Random Forest e 3 para o AdaBoost. Os valores observados de acurácia para cada uma das combinações estão apresentados na tabela a seguir:

---

Método	Count Vectorizer	TF IDF Vectorizer
Multinomial NB.	72,82%	72,82%
Decision Tree	71,73%	71,01%
Random Forest	74,27%	74,09%
AdaBoost	74,81%	73,01%
Voting Classifier	75,91%	

## Conclusão

Apresentando essa acurácia relativamente alta, podemos dizer que o modelo pode classificar, com certa confiabilidade se um projeto de lei será aprovado ou reprovado utilizando-se apenas de sua ementa, apresentando, porém, alguns pequenos erros, pois provavelmente o número de variáveis consideradas é muito pequeno, uma vez que a aprovação de leis é um processo que leva muito tempo para ser concluído e depende de vários fatores que também são altamente variáveis.

## Como melhorar?

Para melhorar o modelo, deve-se colocar mais variáveis a serem consideradas, como por exemplo, o partido de quem propõe a lei, o tamanho da bancada e o governo que estava vigente no momento em que a lei foi votada, além da criação de um sistema padrão para a ementa da lei, estabelecendo uma variável que considera a ordenação mais comum pra cada projeto de lei, melhorando assim os fatores para a classificação, a confiabilidade do projeto e muito possivelmente, a acurácia.