Universidade Federal de Pernambuco
Centro de informática

Graduação em Engenharia da Computação

Webcrawler para detecção de fraude em fóruns

Pedro Esposito Gomes da Silva

Trabalho de Graduação

Recife
11 de dezembro de 2018

Universidade Federal de Pernambuco
Centro de informática

Pedro Esposito Gomes da Silva

Webcrawler para detecção de fraude em fóruns

Trabalho apresentado ao Programa de Graduação em Engenharia da Computação do Centro de informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Bacharel em Engenharia da Computação.

Orientador: *Luciano de Andrade Barbosa*

Recife
11 de dezembro de 2018

Agradecimentos

A minha família que sempre foi paciente e me apoiou durante toda essa jornada.

Aos meus amigos da faculdade, que estivemos juntos por tanto tempo, e passamos por tantos momentos de alegria, tristeza, conquista e frustração, mas que seguimos em frente e conseguimos chegar ao final dessa jornada.

A meu orientador, Luciano Barbosa, que sempre foi atencioso, e esteve sempre presente em todas as etapas do projeto para me aconselhar e tirar duvidas.

Um agradecimento especial a Gabriel lima, que durante uma discussão sobre a ideia inicial do trabalho, propôs uma das peças que se tornou fundamental para a criação do crawler proposto, e a Jorge que me ajudou na procura pelos fóruns que foram utilizados.

There are more things in heaven and earth, Horatio, Than are dreamt of in your philosophy.

— SHAKESPEARE (Hamlet 1.5.167-8, Hamlet to Horatio)

Resumo

O custo que algumas empresas têm com fraudes é muito grande. Muitas dessas fraudes são vendidas, divulgadas, ou têm seu "produto" vendido na internet. Sendo a internet o maior meio de comunicação entre os fraudadores, a tarefa de monitorá-la é de extrema importância. Porém, a internet possui uma quantidade gigantesca de informação e, por essas informações nem sempre possuírem uma estrutura bem definida, realizar um monitoramento da web de maneira escalável para identificar fraudes se torna um problema bem complexo. Com o objetivo principal desse trabalho sendo automatizar a coleta de informações sobre fraudes em fóruns, nós propomos um web crawler focado que utiliza random forest para realizar a classificação de uma pagina HTML como possuidora ou não de conteúdo que pode ser vinculado à fraude. Os links que levaram para as páginas classificadas positivamente são clusterizados, utilizando k-means para prever quais links da fronteira possuem mais chances de levar a páginas com o mesmo tipo de conteúdo e criar uma fila prioritária com esses links, assim, aumentando a taxa de coleta de paginas relevantes (havest ratio) do crawler proposto. No final, o classificador não obteve um bom resultado, mas mesmo assim o crawler conseguiu ser melhor que o baseline e, apesar de não estar pronto para ser utilizado em aplicações reais, o crawler é promissor.

Palavras-chave: coletor focado, fraude, monitoramento, classificação de texto, aprendizagem não supervisionada

Abstract

The cost that some companies have with fraud is very large. Many of these scams are sold, shared, or have their "product"sold on the internet. As the internet is the largest mean of communication between fraudsters, the task of monitoring the web is of the utmost importance. But the internet has a huge amount of information, and this informations doesn't always have a well-defined structure, making the task of building a scalable web crawler to identify fraud, a very complex one. With the main objective of this work being, create a automated process for collecting forum pages with information about fraud. We propose a focused web crawler that uses random forest to perform the classification of a web page, as if the page possess or not content that can be linked to fraud. The links that lead to positively ranked pages are clustered using k-means, and it's result is used to predict which links on the frontier are more likely to lead to pages with the same content type, and create a priority queue with those links, thereby increasing the havest ratio of the proposed crawler. In the end the classifier did not got a good result, but the crawler was still better than the baseline and although it is not ready to be used in real applications, the crawler is promising.

Keywords: focused web crawler, fraud, text classification, clustering

Sumário

Lista de Figuras

Lista de Tabelas

CAPÍTULO 1

Introdução

O custo que algumas empresas têm com fraudes é muito grande. Uma estatística extraída de um relatório da Association of Certified Fraud Examiners (ACFE) de 2016 [acf16] diz que mais de 6,3 bilhões de dólares foram perdidos em fraudes, nos casos estudados por eles e, em 2018, esse número passou dos 7 bilhões de dólares[acf18]. Os esquemas de fraudes são vendidos e divulgados em toda parte da internet, tanto na Surface Web, que são as páginas que são indexadas pelos engenhos de busca, quanto na Deep Web, que são as páginas que não são indexadas pelos engenhos de busca. Sendo assim o monitoramento da internet para achar possíveis fraudes é de extrema importância para prevenção e contenção dessas fraudes.

A internet possui uma quantidade gigantesca de informação que não segue uma estrutura padronizada e, cada vez mais, mais informações estão sendo inseridas, tornando a recuperação de uma informação específica de maneira automatizada e escalável, uma tarefa extremamente complicada. Ainda mais quando as pessoas que fornecem as informações não querem que elas sejam vistas por qualquer um.

Uma maneira de fazer o monitoramento da internet atrás de fraudes é criando vários coletores específicos para cada site ou plataforma. Porém, a criação e manutenção desses vários coletores são custosas, tornando de fundamental importância a construção de um coletor que seja escalável, no sentido de poder ser utilizado para monitorar vários sites ou plataformas. Com um coletor escalável, os custos tanto de produção quanto de manutenção poderiam ser diminuídos, e a área de coleta poderia ser ampliada com mais facilidade.

Por isso, o objetivo principal desse trabalho é automatizar a coleta de informações sobre fraudes em fóruns. Será utilizado um web crawler focado para coletar as informações de páginas que possuem conteúdo vinculado a fraude, com a ajuda de um classificador de fraudes, e um seletor de links para dar prioridade aos links com maior chance de levarem para paginas de fraude. O classificador de fraude usa aprendizado supervisionado, enquanto o seletor de links usa um aprendizado não supervisionado.

CAPÍTULO 2

Conceitos Básicos

2.1 Conceitos Gerais

2.1.1 Fraude

Existem várias definições do que é fraude, por exemplo, segundo a associação de examinadores de fraudes certificados (The Association of Certified Fraud Examiners - ACFE), fraude se define como: "qualquer ação intencional ou deliberada, que prive alguém de sua propriedade por um meios maliciosos, que engane ou qualquer outro meio injusto"[acf]. Já o direito penal brasileiro se refere à fraude como: "Art. 171 - Obter, para si ou para outrem, vantagem ilícita, em prejuízo alheio, induzindo ou mantendo alguém em erro, mediante artifício, ardil, ou qualquer outro meio fraudulento"[civ18]. O que ambas as definições têm em comum, é que a fraude é realizada para obter um ganho, seja para quem a realiza ou para outra pessoa, e fraude tem que enganar ou utilizar de meios injustos, para obter esse ganho. Para este trabalho foi preciso limitar essa definição e estamos considerando que os documentos que possuem informações de fraudes são aqueles que possuem:

- Tutoriais que ensinam como enganar empresas para obtenção de seus serviços ou produtos de maneira gratuitas.
- Tutoriais para obtenção de informações de terceiros, onde possuem etapas onde pessoas ou organizações são enganadas.
- Contas de terceiros que permitem acesso a um serviço ou plataforma de maneira gratuita.
- Informações de terceiros ou que dizem vender tais informações.

não é considerado fraude:

- Pedido de dados pessoais de terceiros.
- Propagandas de serviços ilícitos ou não.
- Software crackeado, ou técnicas de crakeamento.
- Divulgação de ransomwares ou malwares que não estejam no contexto de um esquema.

2.1.2 Web crawler

Web crawler é um robô que tem como objetivo navegar pelas páginas web e coletar as informações dessas páginas. Os web crawlers têm o seguinte funcionamento:

1. O robô baixa uma página e passa para o seletor de páginas.
2. Se a página for relevante, o seletor de páginas a indexa.
3. O seletor de páginas extrai os links da página visitada e passa para o gerenciador da fronteira.
4. O gerenciador da fronteira escolhe o link da próxima página a ser visitada e passa para o robô.
5. Os passos são repetidos até não possuir links novos a serem visitados ou ao atingir uma certa quantidade de páginas visitadas.

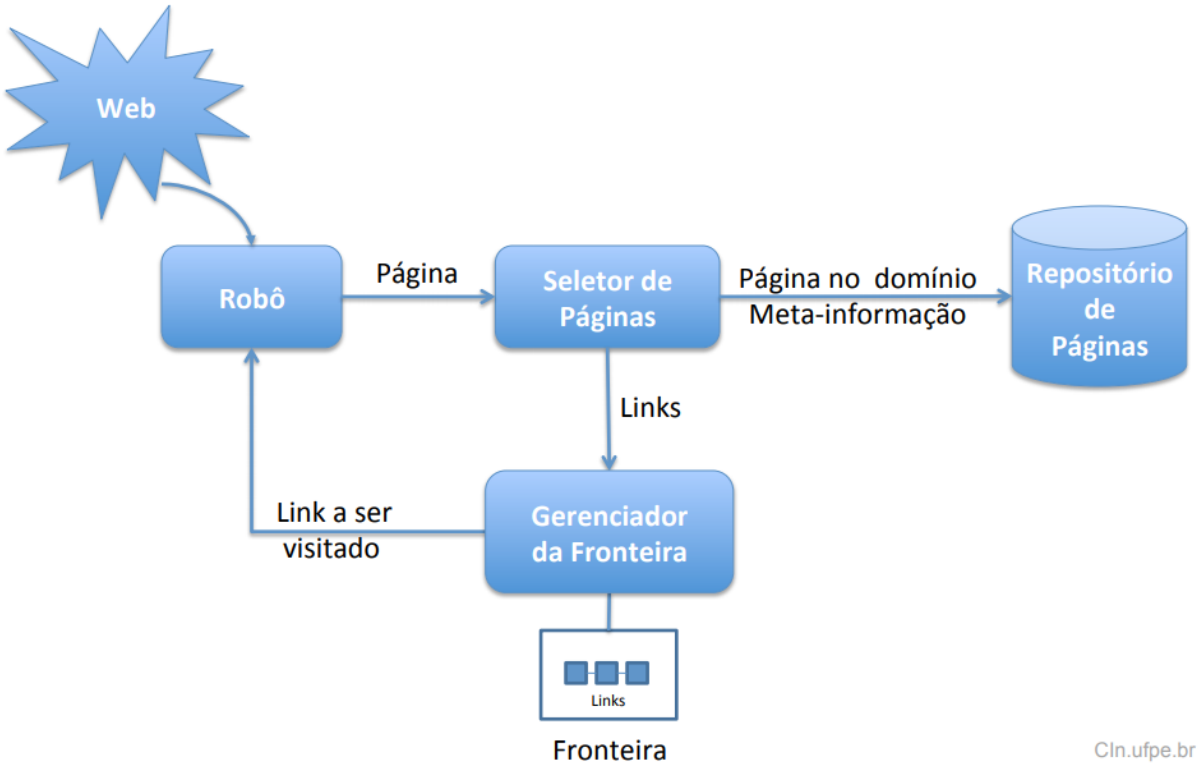


Figura 2.1: Modelo coletor Geral
Fonte: http://www.cin.ufpe.br/ luciano/courses/ir/crawling.pdf

Web crawlers ou simplesmente crawlers são principalmente usados por *search engines* para expandir a quantidade de páginas em seu repositório e atualizar o conteúdo de páginas antigas. O crawler descrito acima é chamado de coletor geral (figura 2.1), onde todas as páginas são

relevantes e o objetivo final é indexar essas páginas de maneira a tornar mais fácil e eficiente as buscas por elas.

Para certas aplicações, são relevantes apenas páginas de um certo domínio. O crawler que faz isso é chamado de coletor focado. Para implementar um coletor focado pode-se inicialmente tentar uma abordagem em que não exista prioridade entre os links e apenas filtrar as páginas que não são do domínio desejado. Porém, a web possui uma quantidade muito grande de páginas, assim tornando essa abordagem uma maneira ineficiente de resolver o problema pois, dessa maneira, o coletor vai visitar uma quantidade muito maior de páginas que não são relevantes para a aplicação.

Para melhorar a eficiência do coletor, é necessário restringir o universo de links para um sub grupo. Esse sub grupo de links deve possuir links com mais chances de encaminhar o coletor para uma página que seja relevante à aplicação. Essa seleção dos links pode utilizar informações como a URL do link, a sua âncora (texto usado para identificar a URL) e o texto de elementos próximos a âncora (vizinhança). Essa seleção pode ser feita de varias maneiras, mas podemos dividir as abordagens em 3 grupos. Aprendizado Online, aprendizado Offline e aprendizado offline-online.

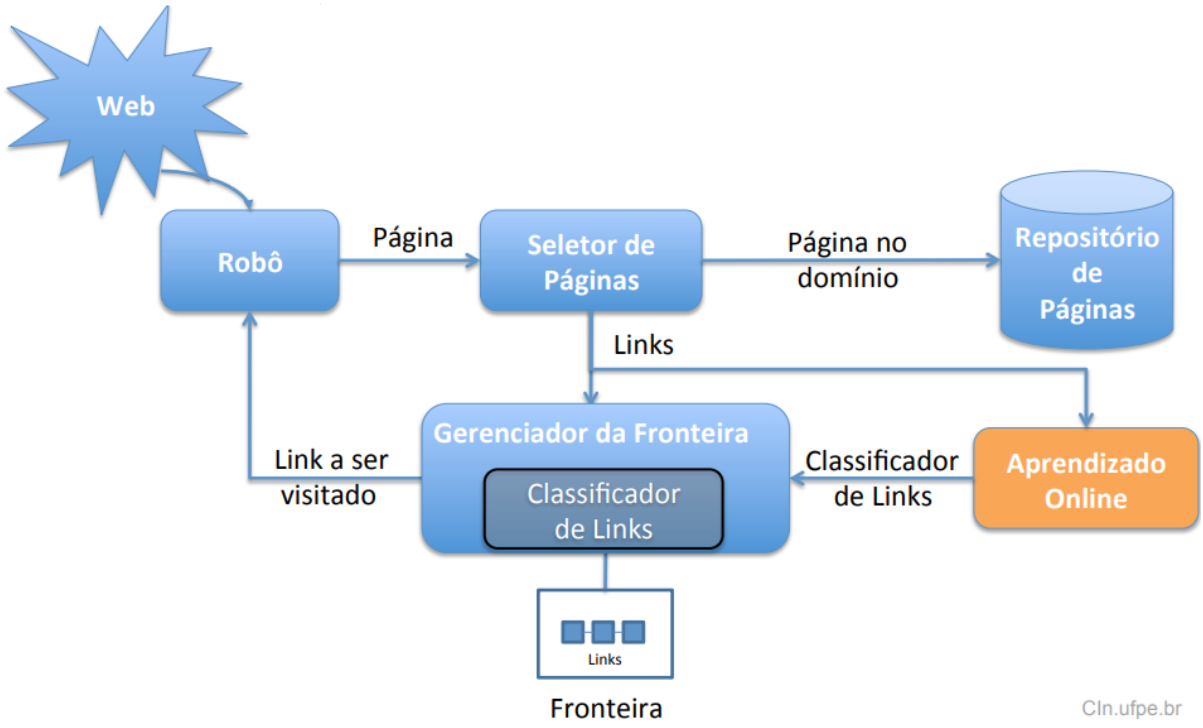


Figura 2.2: Modelo Aprendizado Online
Fonte: http://www.cin.ufpe.br/ luciano/courses/ir/crawling.pdf

Nas abordagens que utilizam o aprendizado online (figura 2.2), o crawler começa sem nenhuma informação a respeito dos links que possuem a maior chance de levar o crawler para páginas do domínio. A medida em que o seletor de paginas vai achando páginas do domínio, o crawler vai aprendendo o conjunto de palavras que indicam que o link é de uma página do

domínio e o conjunto de palavras que indicam que o link não é de uma página do domínio. Com esses conjuntos de palavras, é feita a verificação das chances que cada link tem de levar o crawler para uma página do domínio. Por fim, são priorizados aqueles links que possuem a maior chance.

Nas abordagens que utilizam aprendizado Offline, o crawler começa com um conjunto de palavras que indicam que o link é de uma página do domínio e com o conjunto de palavras que indicam que o link não é de uma página do domínio, e ele utiliza esses conjuntos para priorizar os links.

As abordagens que utilizam o aprendizado offline-online utilizam tanto o conhecimento prévio da abordagem offline, como a capacidade de aprender da abordagem online, que a medida em que mais páginas do domínio são coletadas, o crawler modifica os conjuntos de palavras para obter conjuntos de melhor desempenho.

2.2 Algoritmos utilizados no classificador de fraude

2.2.1 Extração de caraterísticas de texto

Para classificar um texto, é necessário extrair suas características, para que elas possam ser usadas para definir sobre o que o texto se trata. Uma técnica muito utilizada para extrair as características do texto e transformar em uma informação que possa ser utilizado por um classificador é a técnica de "bag of words".

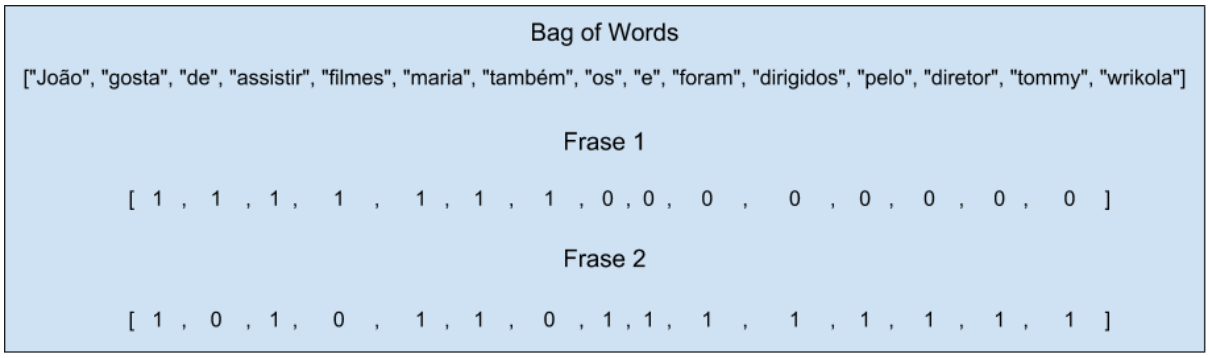


Figura 2.3: bag of words
Fonte: autoria do autor deste trabalho

Bag of words é uma técnica que consiste em criar um dicionário com as palavras que são encontradas em um conjunto de documentos. por exemplo, se pegarmos a frase "João gosta de assistir filmes. maria também gosta de assistir filmes"e a frase "os filmes de João e maria foram dirigidos pelo diretor Tommy Wirkola", nós podemos gerar a bag of words, como é mostrado na figura 2.3, e representar cada frase como um array, onde zero significa que a frase não possui a palavra e 1 que a frase possui a palavra.

Para obter as palavras do exemplo, foi necessário realizar uma tokenização do texto. Tokenizar consiste em identificar e separar todas as palavras do texto. O processo é simples de se entender, porém, a depender do tipo de documento e da língua em que o documento está escrito,

os caracteres que são utilizados como separadores de cada palavra podem não ser triviais como um simples espaço entre as palavras e podem, até mesmo, não existir. Isso torna o simples processo de tokenização algo complexo. Além disso, algumas palavras ou estruturas podem não ser relevantes para a descrição do documento, como uma tag HTML ou as stop words da língua em que o documento está escrito, sendo necessário um processamento adequado para os tipos de documentos que estão sendo classificados.

Para melhorarmos a descrição das nossas frases, ao invés de guardar apenas a informação da existência ou não da palavra na frase, podemos guardar a quantidade de vezes em que cada palavra aparece no texto. Esse processo é chamado de *term frequency* (TF). Um problema nesse tipo de descrição é que não é levado em consideração a quantidade total de palavras que o documento possui e, para melhorar o descritor, o TF pode ser normalizado com o tamanho do documento.

Outro problema é que existem palavras que são naturalmente mais comuns que outras. E o nosso descritor, até então, considera que todas as palavras têm o mesmo peso. Para melhorar isso, o descritor pode usar a frequência inversa do termo nos documentos *inverse document frequency* (IDF), que é um índice que indica o poder descritivo que as palavras tem, em relação ao conjunto total de palavras dos documentos. O IDF usa a quantidade de vezes em que o termo aparece em todos os documentos e faz a inversão disso para achar as palavras com maior poder descritivo.

No entanto, a abordagem TF-IDF não leva em consideração a ordem em que as palavras foram citadas e isso faz com que os documentos sejam descritos de uma forma que o contexto em que as palavras estavam inseridas não seja representado. Uma técnica que pode ser usada junto com as demais citadas para inserir um contexto no descritor é a do n-gram.

O n-gram gera todos os possíveis grupos contendo um numero N de palavras sequenciais e considera esse agrupamento como uma única palavra no documento (figura 2.4). Com N = 1 nós temos o processo de tokenização anteriormente descrito, a medida que N aumenta, maior vai ser a informação do contexto, porém cada vez mais, fica difícil achar conjuntos iguais para comparar os documentos.

2.2.2 Modelos de Classificação

Existem vários modelos de classificação que podem ser utilizados. Para este trabalho foram utilizados os modelos *support vector machine* (SVM) e *random forest* por possuírem um bom desempenho em tarefas de classificação de texto.

2.2.3 SVM

SVM é um algoritmo que faz a separação das classes utilizando o melhor hiperplano que as divide. O melhor hiperplano (linha vermelha da figura 2.5) é escolhido usando *support vectors* e é aquele que possui a maior margem entre os grupos. *Support vectors* são os pontos da fronteira dos grupos com a menor distância para um ponto da fronteira que pertença ao outro grupo (pontos marcados com um X na figura 2.5).

Um problema é que, com a separação sendo feita apenas por um hiperplano o algoritmo só poderia ser usado para classificar dados que são linearmente separáveis. Mas, para resolver

problemas mais complexos o SVM utiliza um Kernel. O kernel é uma função que transforma os dados de um espaço N-dimensional para um espaço M-dimensional, onde M > N, com o intuito de achar um hiperplano no espaço M que consiga separar as classes.

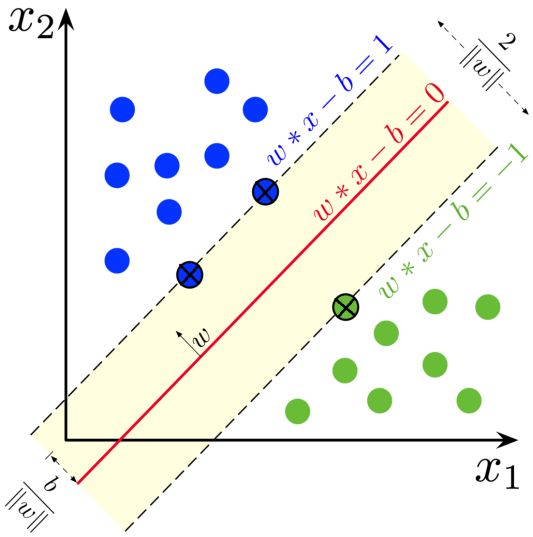


Figura 2.4: Hiperplanos SVM
Fonte: https://en.wikipedia.org/wiki/Support_vector_machine

2.2.4 Random Forest

Para entendermos o algoritmo de *random forest*, primeiro precisamos entender como funciona uma árvore de decisão.

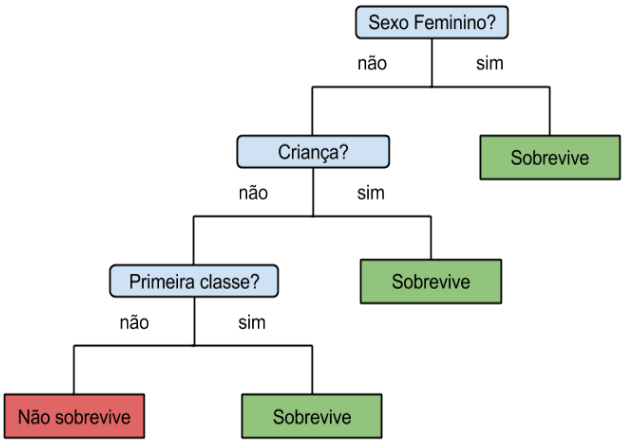


Figura 2.5: Decision tree
Fonte: http://blog.caelum.com.br/r-titanic-e-data-science/

Árvore de decisão é um modelo usado para separação de classes. O modelo realiza perguntas sobre o objeto a ser classificado e, dependendo de cada resposta, novas perguntas são feitas. Esse processo continua até que seja possível determinar, com um certo nível de certeza, a classe ou a probabilidade do objeto de pertencer a classe (figura 2.6). O algoritmo de *random*

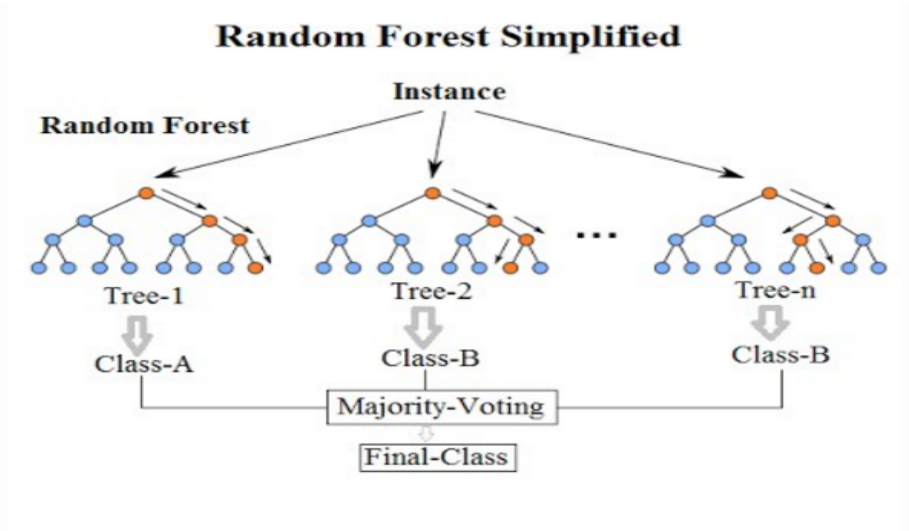


Figura 2.6: Random Forest
Fonte: https://medium.com/@williamkoehrsen/random-forest-simple-explanation-377895a60d2d

forest usa diferentes árvores de decisões e os resultados de cada árvore pode ser utilizado como um voto e a classificação mais votada é a retornada (figura 2.7), ou pode ser retornado a média das probabilidades retornadas por cada árvore.

2.3 Algoritmo utilizado no processo de seleção de links

2.3.1 Clustering

Clustering consiste em agrupar pontos que possuem características semelhantes. Devido a isso, varias técnicas de *clustering* são utilizadas para criar modelos de classificação, nos quais a rotulagem dos dados não é necessária (aprendizagem não supervisionada). Existem vários algoritmos de clustering como K-Means Clustering, Hierarchical Clustering, Mean-Shift Clustering, DBSCAN Clustering, entre outros. Mas, para esse trabalho, utilizamos apenas o K-Means Clustering, que será descrito abaixo.

2.3.2 K-Means

A letra 'K' no k-means é o numero de *clusters* que o algoritmo vai gerar. Para o exemplo, vamos adotar K = 3.

Para obter os *clusters*, o k-means inicialmente escolhe aleatoriamente 3 pontos que pertençam ao mesmo domínio dos dados. Esses pontos serão os primeiros centros dos clusters.

(figura 2.7).

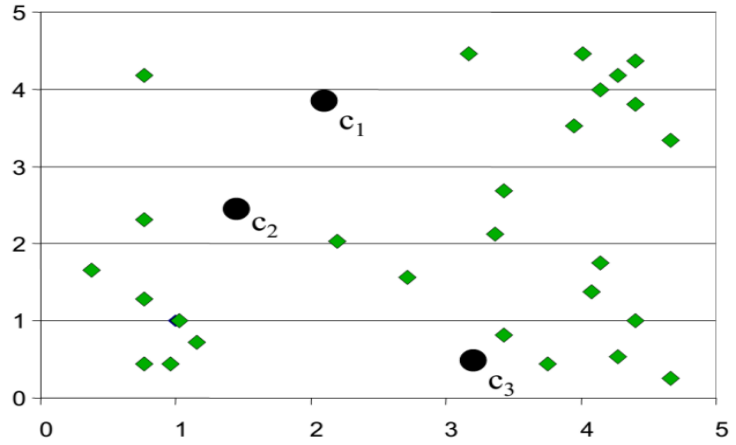


Figura 2.7: k-means: Estado inicial
Fonte: http://www.mit.edu/ 9.54/fall14/slides/Class13.pdf

Com os *clusters* iniciais escolhidos, será calculada a distância entre os centros dos *clusters* e cada ponto do conjunto de dados. Cada ponto é, então, designado ao *cluster* que possuir a menor distância calculada(figura 2.8).

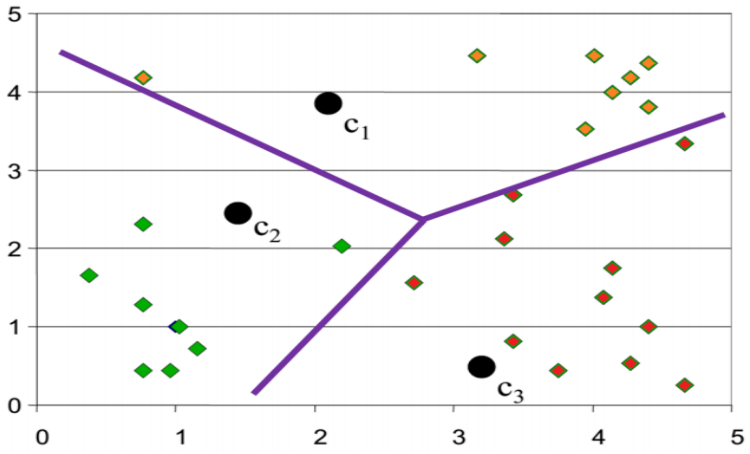


Figura 2.8: k-means: Atribuição do cluster
Fonte: http://www.mit.edu/ 9.54/fall14/slides/Class13.pdf

Depois que todos os pontos forem designados a um *cluster*, é calculado o ponto médio de cada *cluster*, que passa a ser o novo centro (figura 2.9). As distâncias são calculadas novamente e os pontos são designados ao cluster com a menor distância. Esse processo é repetido até que não ocorra nenhuma mudança nos *clusters* figura(2.10).

Um problema que ocorre com o k-means é que nem sempre o melhor conjunto de *clusters* é escolhido. Por causa, disso é recomendado que o processo seja repetido mais de uma vez para que retorne, o melhor conjunto de *clusters* obtido pelas repetições. Para escolher o melhor

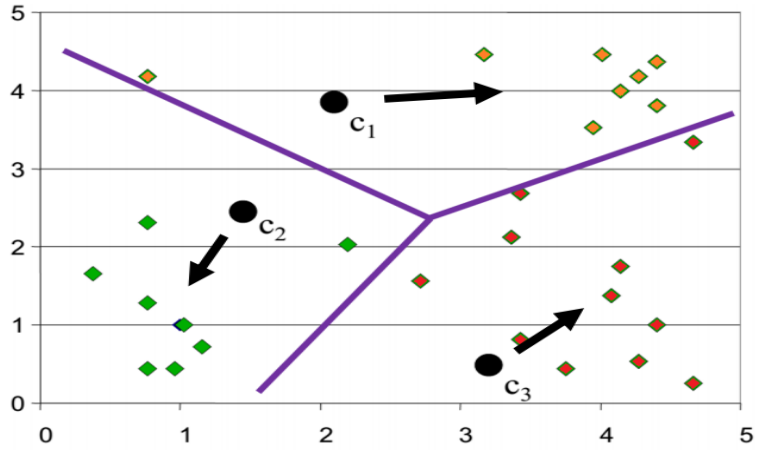


Figura 2.9: k-means: Mudança dos centros dos clusters
Fonte: http://www.mit.edu/ 9.54/fall14/slides/Class13.pdf

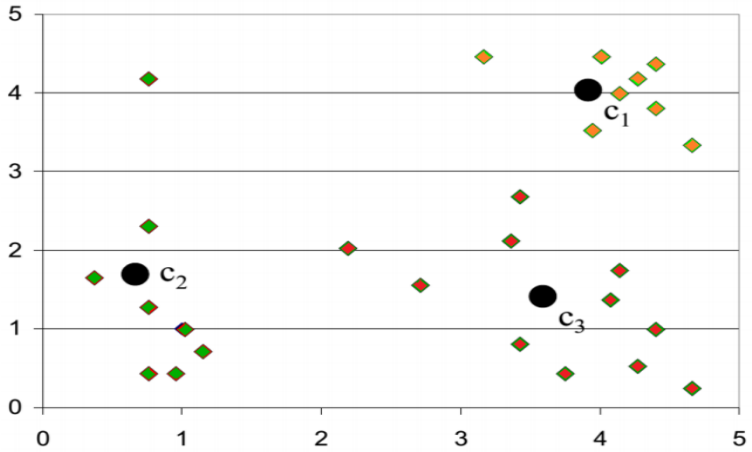


Figura 2.10: k-means: estado final
Fonte: http://www.mit.edu/ 9.54/fall14/slides/Class13.pdf

conjunto de clusters, é calculada a variância entre os pontos de cada cluster final e são somadas essas variâncias. O conjunto que possuir a menor variância total é o melhor conjunto.

2.4 Ferramenta utilizada na execução do web crawler

2.4.1 Rede Tor

"A rede Tor é um grupo de servidores provido por voluntários, que permite que as pessoas melhorem a sua privacidade e segurança na internet"[Pro18].

A rede Tor utiliza uma arquitetura de roteamento onion, que faz as mensagens passarem por N nós antes de chegar no destino. A rede Tor usa 3 nós para fazer o roteamento (figura 2.12) e tem o seguinte funcionamento:

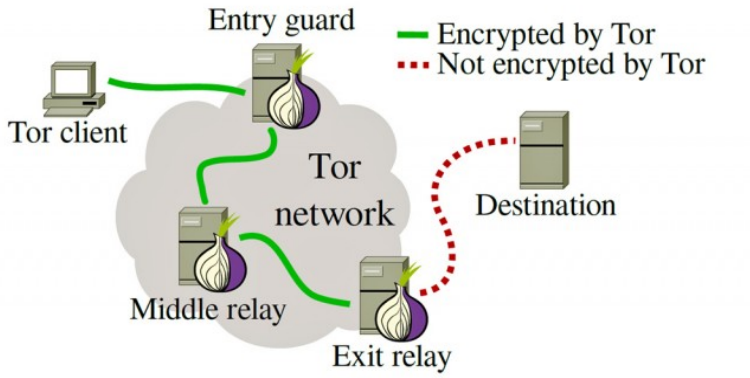


Figura 2.11: Rede Tor
Fonte: https://arstechnica.com/information-technology/2014/01/scientists-detect-spoiled-onions-trying-to-sabotage-tor-privacy-network

/

Um cliente, que queira mandar uma mensagem pela rede Tor para um servidor, vai se conectar com 3 nós da rede Tor e troca com cada um dos nós um conjunto diferente de chaves, que vão ser usadas para encriptar as mensagens.

Em seguida, o cliente monta a mensagem da seguinte maneira: Primeiro, uma mensagem é encriptada com a chave de algum nó e, depois, são adicionadas as informações do nó em que essa chave foi gerada, isso é uma camada da onion. Esse processo é, então, repetido 3 vezes, usando a camada que foi gerado pelo processo anterior como a mensagem da nova camada, assim, gerando uma pilha de mensagens (figura 2.13), dos quais apenas o cliente e o nó que possui a chave usada na camada conseguem descifra-la.

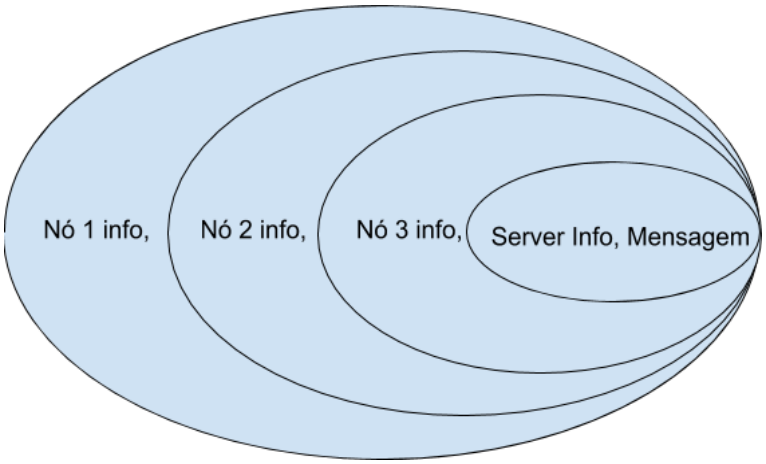


Figura 2.12: Pacote onion
Fonte: autoria do autor deste trabalho

Esse pacote de mensagens é então enviado para o nó da ultima camada que, por sua vez,

decifra a mensagem e envia para o próximo nó. Esse processo ocorre até que o terceiro nó decifre a ultima mensagem e envie a mensagem original para o servidor (figura 2.12).

Se o servidor responder a mensagem, o processo inverso será feito. Cada nó, então, adicionará a sua camada de encriptação e passará para o nó anterior, chegando ao cliente uma mensagem que apenas ele possui todas as chaves necessárias para decifrá-la.

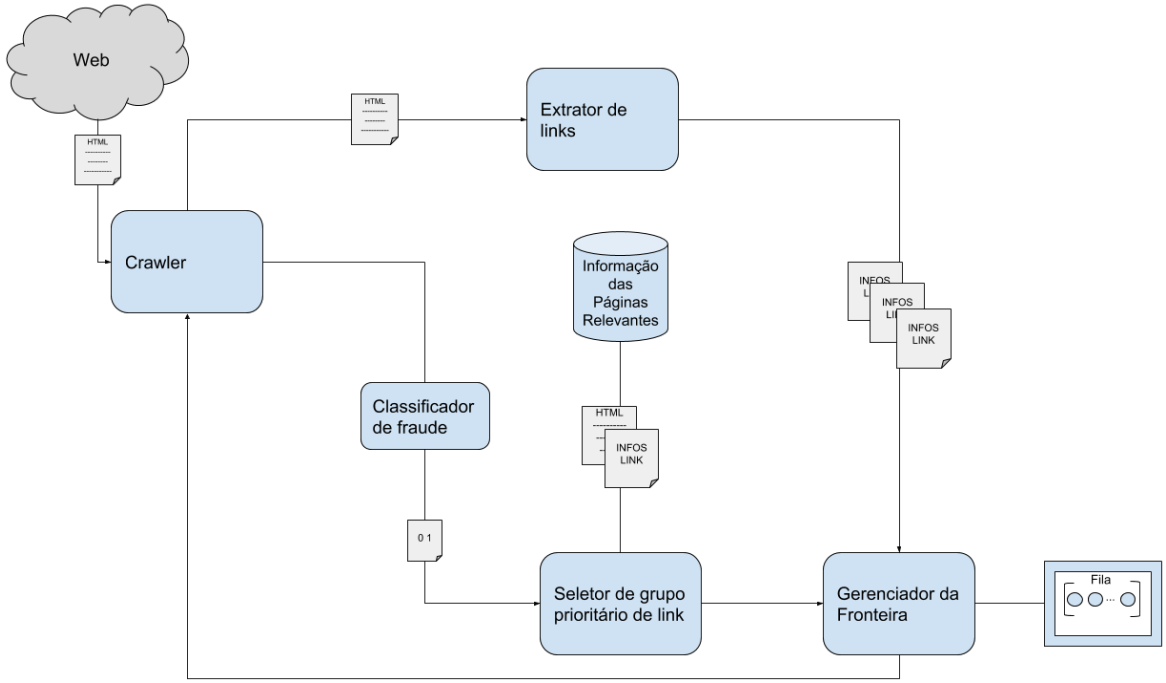
Com esse funcionamento, a rede Tor permite que apenas o primeiro nó da rede saiba quem foi que enviou a mensagem e que apenas o ultimo nó da rede saiba pra quem foi enviada a mensagem, assim, tornado a navegação anônima.

CAPÍTULO 3

Metodologia

Nesse capítulo, explicaremos o funcionamento geral do crawler proposto, representado pela figura 3.1, e depois os módulos do crawler serão explicado em mais detalhes.

Figura 3.1: Esquema geral do crawler proposto



Fonte: autoria do autor deste trabalho

3.1 Funcionamento geral do crawler

O crawler, ao visitar uma página, manda o seu código HTML para o extrator de links e para o classificador de fraude. O extrator de links vai identificar todos os links desta página e irá filtrar os links, para que apenas aqueles que sejam do mesmo domínio da página sejam retornados. Esses links serão passados para o gerenciador da fronteira, que colocará todos os links novos ao final da fila. Em paralelo, o classificador de fraude vai tratar o HTML da página e fará a sua classificação. Caso a página seja de fraude, o seletor de grupo prioritário de links irá salvar o conteúdo da página e as informações do link que levou o crawler a visitá-la. Se certas condições

forem atingidas, o seletor de grupo prioritário de links fará a escolha de um grupo utilizando o cluster. Ao final, o gerenciador de fronteia irá escolher a próxima URL a ser visitada, de acordo com o grupo selecionado pelo seletor de grupo prioritário de links.

3.2 Classificador

Pela área de segurança ser uma área que lida com informações sensíveis e por muitas das pesquisas serem realizadas em organizações privadas existem poucos datasets com disponibilidade publica, ainda mais quando as informações envolvem fraude.

Por isso, para esse trabalho, tivemos que montar o dataset que foi utilizado para treinar o classificador de fraudes usado no crawler proposto.

Devido a natureza dos fóruns utilizados para a criação do dataset e realização dos experimentos, foi necessário tomar algumas providencias para garantir o nosso anonimato e para conter eventuais problemas que poderiam surgir.

- Para toda a navegação realizada na criação do dataset e coleta realizada pelo web crawler, foi utilizada a rede Tor, que possibilita a nossa anonimidade.
- Foi criada uma máquina virtual(VM) para realização de todos os testes e experimentos, dessa maneira, se durante algum teste ou experimento a máquina fosse comprometida, esse comprometimento estaria contido na VM.

3.2.1 Montagem do dataset

Seguindo a definição de fraude abordada na sessão 2.1, começamos a navegar pelos fóruns rotulando as páginas como fraudulentas ou não. Quando o dataset acumulou 300 páginas, as primeiras versões do modelo foram executadas, e as páginas retornadas pelo crawler foram reclassificadas e adicionadas ao dataset.

Nesse momento, o dataset passou a ter 2200 páginas classificadas, sendo 800 páginas fraudulentas e 1400 páginas não fraudulentas. Ao final, foi necessário revisitar todo o dataset para averiguar as classificações. O dataset final possui 781 páginas fraudulentas e 1030 páginas não fraudulentas, e essas páginas foram coletadas de 3 dos 7 fóruns analisados. Uma word cloud com as 100 palavras que mais aparecem no nosso conjunto positivo de páginas está descrita na figura 3.2

Por ser um assunto do qual não somos especialistas, o dataset, apesar de pequeno, tomou uma grande quantidade de tempo para ser gerado e pode possuir páginas classificadas de maneira errada.

3.2.2 Tratamento do HTML

Para processar o HTML, primeiramente, será retirado da página original(Figura 3.3) todos os elementos HTML que possuírem uma tag de comentário, script, ou style (Figura 3.4). Em seguida, serão retiradas todas as tags restantes, e será deixando apenas o texto da página (Figura

Figura 3.2: Word cloud das palavas encontradas nas páginas de fraude do dataset

Fonte : autoria do autor deste trabalho

3.5). Por ultimo, todos os caracteres especias de pular linha, tabulação e o excesso de espaços são retirados (Figura 3.6).

Com o HTML tratado as stop words da língua inglesa são retiradas. As palavras restantes tem que ser transformadas de maneira que consigam descrever a página, como foi previamente abordado na sessão de conceitos básicos. Foi escolhido usar o TF-IDF com 3-gram para gerar o descritor da página.

3.2.3 Treinamento do classificador

O classificador foi treinado com o dataset que montamos. Foi utilizado 10% do dataset como conjunto de teste. Os documentos, tanto do conjunto de treinamento quanto do conjunto de teste foram distribuídos de maneira aleatória.

Devido ao tamanho e qualidade do dataset, foram usados os classificadores SVM e Random Forest, que retornam a probabilidade do documento pertencer à uma dada classe. Com o intuito de nos ajudar a verificar qual seria o melhor valor probabilístico mínimo, para considerarmos a página como fraudulenta e o modelo que seria utilizado nos experimentos, foi realizado o seguinte procedimento: O conjunto de teste foi previsto, considerando como páginas fraudulentas os documentos que possuíam probabilidades maiores que, 0.5, 0.6, 0.7, 0.8 e 0.9, e foram tiradas as métricas de Accuracy, Precision, Recall e F-mesure para cada um desses

Figura 3.3: Html original

```
<body>
  <div class="limit">
    <aside class="sidebar">
      <!-- Something before the submenus -->
    </aside>
    <nav class="navside" id="menu-html">
      <h3><a href="#menu-html">HTML</a></h3>
      <ul>
        <li><a href="index.html">O que é HTML?</a></li>
        <li><a href="oquesemantica.html">O que é semântica?</a></li>
        <li><a href="oquetags.html">O que são Tags, atributos e elementos?</a></li>
        <li><a href="estruturabasica.html">Estrutura básica</a></li>
        <li><a href="oquemetatags.html">O que são metatags?</a></li>
      </ul>
    </nav>

    <article class="content">
      <h1>Estrutura básica</h1>
      <h2>Iniciando o código básico de HTML</h2>
      <p>É possível compreender o documento em HTML de uma maneira muito simples, através de uma divisão de blocos das tags essenciais, conforme a a seguinte estrutura:</p>
      <ul>
        <li>Definição do documento (doctype)</li>
        <li>Cabeça (head)</li>
        <li>Corpo (body)</li>
      </ul>
      <h3>Doctype - Definindo o documento</h3>
      <p>Uma coisa importante: SEMPRE deve existir o doctype, que é este código <code>&amp;lt;!DOCTYPE html&amp;gt;</code>.</p>
      <p>O doctype não é uma tag HTML, mas uma instrução para o navegador e outros programas que podem ler seu site, que o código encontrado ali é um código HTML. Assim eles sabem o que fazer para
      mostrar seu site da melhor forma possível. Lembre-se: o doctype é OBRIGATÓRIO e deve ser sempre a PRIMEIRA LINHA do seu documento.</p>
      <h3>HEAD</h3>
      <p>Contém informações que não são transpostas visivelmente para o usuário/leitor do documento. São dados implícitos, de uso e controle do documento: vinculação com outros arquivos, aplicação de
      lógica de programação de scripts e metadados. Na prática, todo o conteúdo do cabeçalho fica delimitado entre a abertura e fechamento tag <code>head</code>.</p>
    </article>
  </div>

  <script>
    (function(i,s,o,g,r,a,m){i['GoogleAnalyticsObject']=r;i[r]=i[r]||function(){
    (i[r].q=i[r].q||[]).push(arguments)},i[r].l=1*new Date();a=s.createElement(o),
    m=s.getElementsByTagName(o)[0];a.async=1;a.src=g;m.parentNode.insertBefore(a,m)
    })(window,document,'script','//www.google-analytics.com/analytics.js','ga');

    ga('create', 'UA-335027-18', 'tableless.github.io');
    ga('send', 'pageview');

  </script>
  <script src="//ajax.googleapis.com/ajax/libs/jquery/2.0.3/jquery.min.js"></script>
  <script type="text/javascript" src="/iniciantes/assets/javascripts/prettify/src/prettify.js"></script>
  <script type="text/javascript" src="/iniciantes/assets/javascripts/scripts.js"></script>
</body>
```

Fonte:source html da página
http://tableless.github.io/iniciantes/manual/html/estruturabasica.html

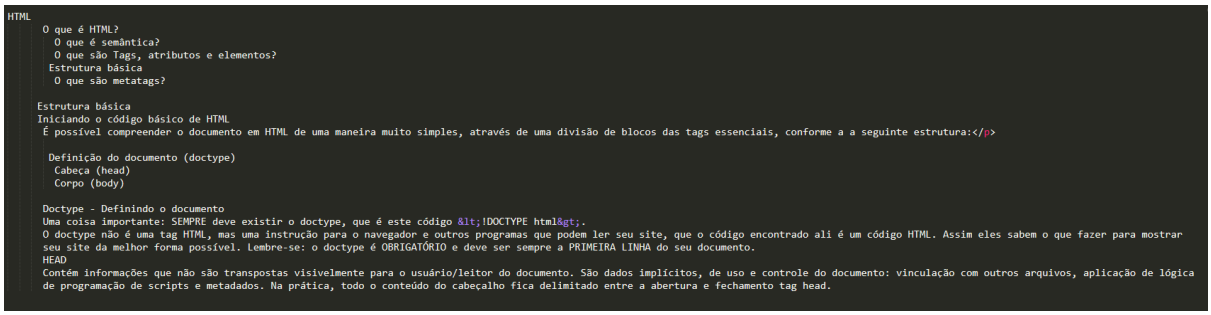
Figura 3.4: Html sem os elementos indesejados

```
<body>
  <div class="limit">
    <aside class="sidebar">
    </aside>
    <nav class="navside" id="menu-html">
      <h3><a href="#menu-html">HTML</a></h3>
      <ul>
        <li><a href="index.html">O que é HTML?</a></li>
        <li><a href="oquesemantica.html">O que é semântica?</a></li>
        <li><a href="oquetags.html">O que são Tags, atributos e elementos?</a></li>
        <li><a href="estruturabasica.html">Estrutura básica</a></li>
        <li><a href="oquemetatags.html">O que são metatags?</a></li>
      </ul>
    </nav>

    <article class="content">
      <h1>Estrutura básica</h1>
      <h2>Iniciando o código básico de HTML</h2>
      <p>É possível compreender o documento em HTML de uma maneira muito simples, através de uma divisão de blocos das tags essenciais, conforme a a seguinte estrutura:</p>
      <ul>
        <li>Definição do documento (doctype)</li>
        <li>Cabeça (head)</li>
        <li>Corpo (body)</li>
      </ul>
      <h3>Doctype - Definindo o documento</h3>
      <p>Uma coisa importante: SEMPRE deve existir o doctype, que é este código <code>&amp;lt;!DOCTYPE html&amp;gt;</code>.</p>
      <p>O doctype não é uma tag HTML, mas uma instrução para o navegador e outros programas que podem ler seu site, que o código encontrado ali é um código HTML. Assim eles sabem o que fazer para
      mostrar seu site da melhor forma possível. Lembre-se: o doctype é OBRIGATÓRIO e deve ser sempre a PRIMEIRA LINHA do seu documento.</p>
      <h3>HEAD</h3>
      <p>Contém informações que não são transpostas visivelmente para o usuário/leitor do documento. São dados implícitos, de uso e controle do documento: vinculação com outros arquivos, aplicação de
      lógica de programação de scripts e metadados. Na prática, todo o conteúdo do cabeçalho fica delimitado entre a abertura e fechamento tag <code>head</code>.</p>
    </article>
  </div>

</body>
```

Fonte:source html da página
http://tableless.github.io/iniciantes/manual/html/estruturabasica.html

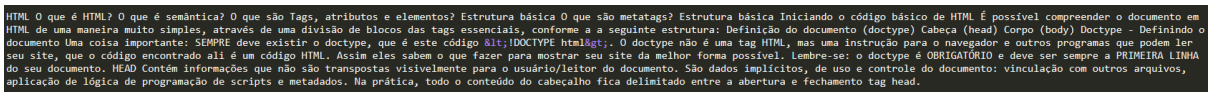
conjuntos. Dessa maneira, poderíamos avaliar qual classificador e qual probabilidade miníma seria escolhido para realizar o experimento final.

Figura 3.5: Html apenas com o texto dos elementos restantes



Fonte:source html da página
http://tableless.github.io/iniciantes/manual/html/estruturabasica.html

Figura 3.6: Html final



Fonte:source html da página
http://tableless.github.io/iniciantes/manual/html/estruturabasica.html

3.3 Seletor de grupo prioritário de links

O seletor de grupo prioritário de links é onde se encontra a inteligência do nosso crawler e seu funcionamento geral está representado na figura 3.7 .

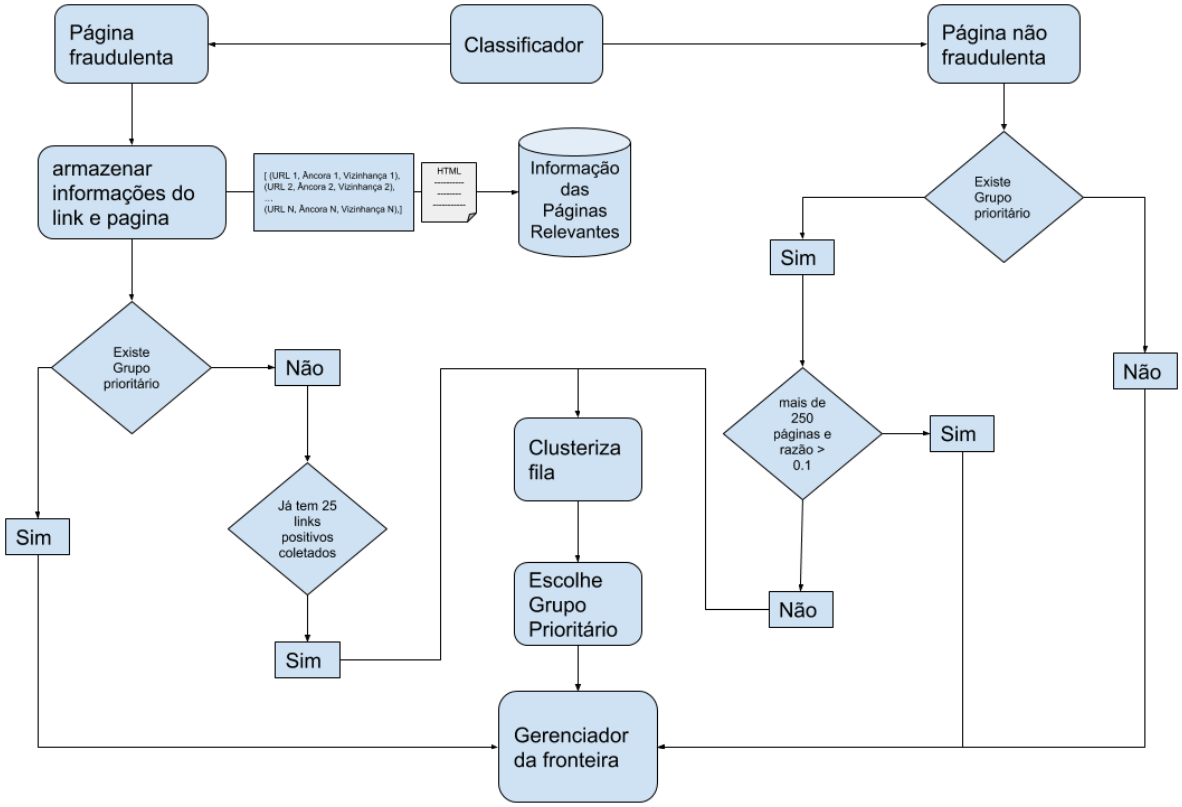
A depender de como a página foi classificada e do estado que o crawler se encontra, alguns caminhos podem ser tomados.

1. caso seja uma página de fraude, as informações do link são armazenadas em um array que contem todas os links de páginas classificadas positivamente.

2. Caso a página possua informações de fraude e não exista nenhum grupo de links prioritário selecionado, será incrementado um contador de fraudes pré grupo. Assim que esse contador atingir um valor de 25 páginas positivas, a fila de links a ser visitados é concatenada com os links das páginas classificadas positivamente, e esse array é clusterizado com k-means, fazendo com que seja atribuído um grupo para cada link. Apos o processo de clusterização, a fila de links a serem visitados é separada dos links de páginas que contem fraude. Os grupos atribuídos aos links de paginas que contem fraude são ranqueados de acordo com a sua ocorrência, e é selecionado o grupo que possua o maior numero de links que levaram o crawler para páginas fraudulentas e que exista pelo menos um link pertencente a esse grupo na fila, caso nenhum grupo satisfaça essas regras, nenhum grupo é selecionado.

3. caso a página possua informações de fraude e exista um grupo de links com prioridade, será incrementado o contador de fraudes do grupo.

Figura 3.7: seletor de grupo prioritário de links



Fonte : autoria do autor deste trabalho

4. caso a página não possua informações de fraude e exista um grupo de links com prioridade. O contador de páginas visitadas no grupo é incrementado. Caso o grupo já tenha visitado pelo menos 250 páginas. Será verificado se a razão entre quantidade de páginas fraudulentas e a quantidade de páginas visitadas com o grupo priorizado (i.e harvest ratio do grupo), é maior que 0.1 . caso seja maior, o grupo continua selecionado, caso não seja, será realizada a clusterização da nova fila e escolhido um novo grupo da mesma forma descrita no caminho 2.

5. caso a página não possua informações de fraude e não exista um grupo de links com prioridade. nenhum dos contadores é alterado.

3.3.1 Clustering

Para gerar o descritor do links nos utilizamos a URL, a âncora, e vizinhança da âncora (Figura 3.8). primeiramente a URL precisa ser tokenizada de maneira diferente, pois diferente da linguagem natural, a URL possui mais alguns separadores importantes. Foram utilizados como separadores ';', ',', '*', '/', '\', '|', '=', '.', '-', '&', ':', '_', '?', e '#'. Assim que a URL foi tokenizada, usando esses separadores, o TF-IDF com 3-gram, foi utilizado para caracterizar a URL. Com a âncora e a vizinhança da âncora, a tokenização não utilizou separadores espe-

Figura 3.8: Exemplo de URL, âncora e vizinhança

```
<dir>
    Tudo isso
    <a href="https://www.exemplo.com.br" class="exemplo"> Isso é a ancora</a>
    <p> é a vizinhança da ancora </p>
</dir>
```

Fonte: autoria do autor deste trabalho

ciais, e só foi utilizado o TF-IDF para gerar o descritor. Para cada um dos elementos (URL, âncora e vizinhança da âncora), foi utilizado um dicionario diferente. Depois de cada uma das informações do link receber um descritor, as matrizes esparsas dos descritores foram empilhadas horizontalmente para formar um descritor único para os links. Com os descritores do link, utilizamos K-means, com k = 15 e random state setado para 5.

3.4 Gerenciador da fronteira

O gerenciador da fronteira tem o trabalho de armazenar os links na fila e escolher qual será o próximo link a ser visitado. Com relação ao trabalho de armazenar os links, o gerenciador de fronteira só coloca os links com URLs que não foram visitadas e que ainda não estão na fila. Dessa maneira, a mesma página não é visitada duas vezes.

Com relação a escolha do próximo link a ser visitado, o gerenciador de fronteira tem 2 modos de escolha, um para, o caso não exista grupo prioritário selecionado e outro para caso exista um grupo selecionado. No caso de não existir um grupo selecionado, o gerenciador da fronteira irá escolher de maneira aleatória um link entre os 20% primeiros links da fila. Isso é feito para tentar fazer o crawler começar a pegar links de postagens mais rapidamente e variar a região do site. Caso haja um grupo selecionado, o seletor de links cria uma fila prioritária para links pertencentes ao grupo escolhido e retorna o primeiro link dessa fila. Esse grupo é escolhido por possuir links que tenham um maior potencial de levar a páginas que contenham informações de fraudes nelas.

CAPÍTULO 4

Resultados e Experimentos

Nesse capítulo, será abordado como os experimentos foram realizados e quais foram os seus resultados. Os experimentos rodam 2 crawlers que possuem abordagens diferentes, o primeiro crawler é o crawler que segue o modelo que foi proposto no capitulo 3, e o segundo crawler, que vai ser utilizado como baseline, possui o mesmo classificador do modelo, mas nenhum grupo prioritário de links é escolhido. Os experimentos foram rodados para 7 fóruns diferentes, e o classificador foi treinado usando páginas de apenas 3 destes 7 fóruns. Os fóruns A, B, e C são os fóruns que possuem páginas rotuladas para o treinamento do classificador de fraude.

Para anonimizar os fóruns usados nos experimentos, foram lhes dados os nomes de fórum A, B, C, D, E, F, e G. Na tabela 4.1, é mostrada a soma das paginas coletadas no experimento 2, com tamanho da fila restante, para que possamos ter uma ideia da quantidade de páginas que cada fórum possui e uma descrição do assuntos abordados em cada um deles.

O algorítimo do classificador escolhido foi o random forest com a probabilidade maior que 0.7. O classificador foi escolhido usando como base as métricas tiradas do conjunto de treinamento (tabela 4.2). Porém, com o decorrer do experimento, verificamos que o range de probabilidade das páginas classificadas com o random forest estava variando em sua maioria entre 0.3 e 0.6, desta maneira, o crawler proposto não estava conseguindo achar uma quantidade miníma de páginas positivas para ser testado. Com isso, apesar das métricas do random forest com probabilidade maior que 0.7 terem tido os melhores resultados no conjunto de teste, na execução dos experimentos, foi considerado como fraude as páginas que tinha probabilidade maior que 0.5.

Deve ser observado que, como em [eJF07] as páginas de fraude nos fóruns são bem esparsas, o que faz a taxa de coleta (havest ratio) em geral ser baixo e que demore para achar informações boas o suficiente para guiar o crawler. Além disso, ainda temos a baixa performance do classificador de fraudes que, quando executado em um ambiente real, pode dar informações inconsistentes para o seletor de grupo prioritário de links, podendo fazer com que demore ainda mais tempo para que informações boas sejam coletadas.

Em todos os gráficos apresentados a baixo, os pontos verdes indicam que foi escolhido um grupo de links para ser priorizado, e os pontos azuis, que o crawler não possui mais grupo de links com prioridade.

Para melhorar a visualização dos gráficos do harvest ratio, a quantidade de páginas visitadas pelo modelo e baseline foram aproximadamente igualadas, mas as informações completas das execuções podem ser encontradas nas tabelas 4.3 e 4.4 . Para melhorar a visualização dos gráficos das médias das probabilidades do classificador de fraude, foram retiradas as 100 primeiras páginas, por possuir uma variação muito grande, o que atrapalhava na escala.

	Assuntos abordados no fórum	tamanho do fórum
Fórum A	venda de informações de terceiros, serviços e tutoriais, entre outros assuntos	115.000 páginas
Fórum B	venda de informações de terceiros, serviços e tutoriais, entre outros assuntos	155.000 páginas
Fórum C	venda de informações de terceiros, serviços e tutoriais, entre outros assuntos	180.000 páginas
Fórum D	venda de informações de terceiros e serviços	43.000 páginas
Fórum E	divagação dos mais diversos tipos softwares	21.000 páginas
Fórum F	divagação dos mais diversos tipos softwares, entre outros assuntos	500.000 páginas
Fórum G	tutoriais de engenharia social, entre outros assuntos	400.000 páginas

Tabela 4.1: Overview dos Fóruns

	Random forest	SVM
padrão	Accuracy = 0.946524 Precision = 0.962500 Recall = 0.916667 F-measure = 0.939024	Accuracy = 0.550802 Precision = 0.000000 Recall = 0.000000 F-measure = 0.000000
probabilidade >= 0.5	Accuracy = 0.946524 Precision = 0.962500 Recall = 0.916667 F-measure = 0.939024	Accuracy = 0.780749 Precision = 0.811594 Recall = 0.666667 F-measure = 0.732026
probabilidade >= 0.6	Accuracy = 0.925134 Precision = 0.960526 Recall = 0.869048 F-measure = 0.912500	Accuracy = 0.759358 Precision = 0.809524 Recall = 0.607143 F-measure = 0.693878
probabilidade >= 0.7	Accuracy = 0.919786 Precision = 0.972603 Recall = 0.845238 F-measure = 0.904459	Accuracy = 0.754011 Precision = 0.865385 Recall = 0.535714 F-measure = 0.661765
probabilidade >= 0.8	Accuracy = 0.887701 Precision = 0.984615 Recall = 0.761905 F-measure = 0.859060	Accuracy = 0.754011 Precision = 0.931818 Recall = 0.488095 F-measure = 0.640625
probabilidade >= 0.9	Accuracy = 0.855615 Precision = 1.000000 Recall = 0.678571 F-measure = 0.808511	Accuracy = 0.737968 Precision = 1.000000 Recall = 0.416667 F-measure = 0.588235

Tabela 4.2: Medições do conjunto de treino

	Total	Páginas Positivas	Páginas Negativas	Harvest Ratio
Fórum A	19540	1266	18274	0.064790
Fórum A Baseline	23358	619	22739	0.026501
Fórum B	33994	3053	30941	0.089810
Fórum B Baseline	30498	1087	29411	0.035642
Fórum C	9826	1507	8319	0.153369
Fórum C Baseline	15377	1106	14271	0.071926
Fórum D	7896	1361	6535	0.172366
Fórum D Baseline	11378	567	10811	0.049833
Fórum E	20442	26	20416	0.001272
Fórum E Baseline	20441	26	20415	0.001272
Fórum F	10755	153	10602	0.014226
Fórum F Baseline	12502	144	12358	0.011518
Fórum G	21685	16	21669	0.000738
Fórum G Baseline	16295	8	16287	0.000491

Tabela 4.3: Overview do experimento 1

4.1 Experimento 1

O primeiro experimento teve uma duração de 3 dias , onde o crawler, utilizando o modelo, e o crawler com a baseline rodaram por aproximadamente 36h cada.

Como pode ser observado os fóruns possuem tempos de resposta muito diferentes entre si, o que trás uma variação para a quantidade de páginas coletadas em um mesmo período de tempo. Com relação a desigualdade de páginas entre o baseline e o crawler proposto, isto é devido ao seletor de links que, por precisar clusterizar a fila, mais tempo é necessário para completar a sua execução.

O fórum 'E' teve a sua fila zerada, assim, terminando sua coleta com 20442 páginas.

Os fóruns E e G, quando rodados com o crawler proposto, não chegaram a achar páginas o suficiente para iniciar o processo de clusterização e a selecionar um grupo prioritário de links. Sendo assim, a diferença dos seus resultados é devido apenas ao fator aleatório do modelo e por terem sido rodados em tempos diferentes, a árvore do site pode ter sofrido variações.

Na execução dos crawlers para o fórum A, que possui a sua execução representada pelas figuras 4.1 e 4.2, conseguimos ver que, no inicio, ocorrem varias mudanças nos grupos escolhidos, até que um grupo bom seja encontrado, o que indica que a maneira que o grupo é escolhido e o critério para a mudança de grupo podem ser melhorados. Contudo, no momento em que um grupo bom é escolhido, a média dos scores gerados pelo classificador aumenta, indicando que uma boa região foi encontrada, como pode ser visto na figura 4.2 . Outro ponto a ser observado são as duas quedas na média e no havest ratio, que ocorrem após a página 15000. Essas quedas ocorrem devido ao fato do crawler não ter um grupo selecionado, e mostra que todos os links do grupo foram visitados.

Na execução dos crawlers para o fórum B, que possui a sua execução representada pelas

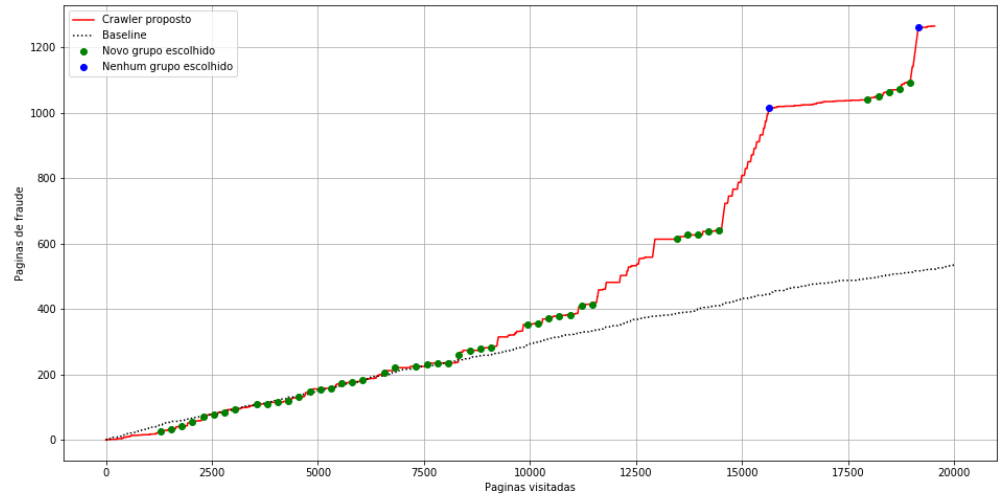


Figura 4.1: Harvest ratio do fórum A Exp-1.

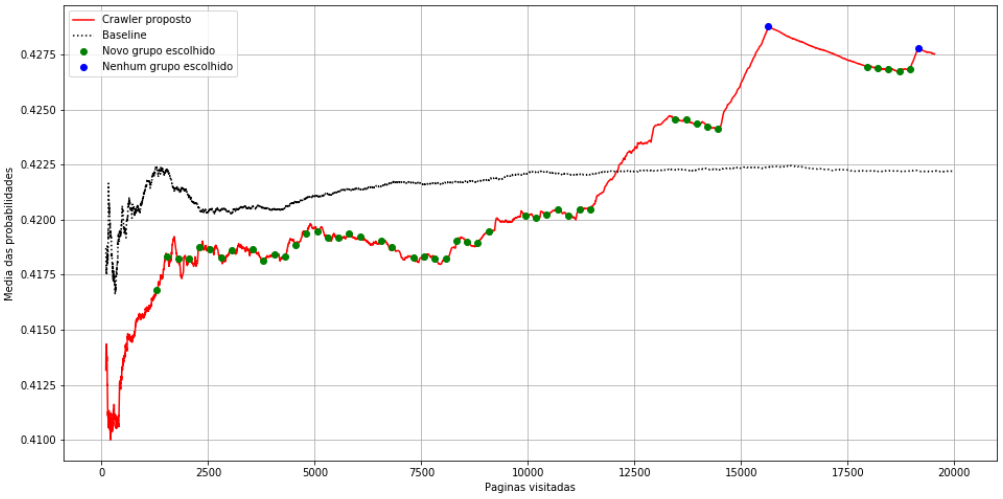


Figura 4.2: Média das probabilidades do classificador de fraudes fórum A Exp-1.

figuras 4.3 e 4.4, podemos ver que ocorreu uma mudança de grupos ainda maior do que no fórum A, reforçando que é preciso melhorar o critério de seleção de grupos, pois o custo de clusterizar a fila pode ser bem grande, a depender de seu tamanho. Porém, a constante mudança de grupos não causa uma perda significativa na taxa de coleta e, quando levamos em consideração a figura 4.4, podemos ver que, apesar dos clusters não terem aumentado muito a taxa de coleta ate marca de 22000 páginas, eles aumentaram a média das probabilidades das páginas

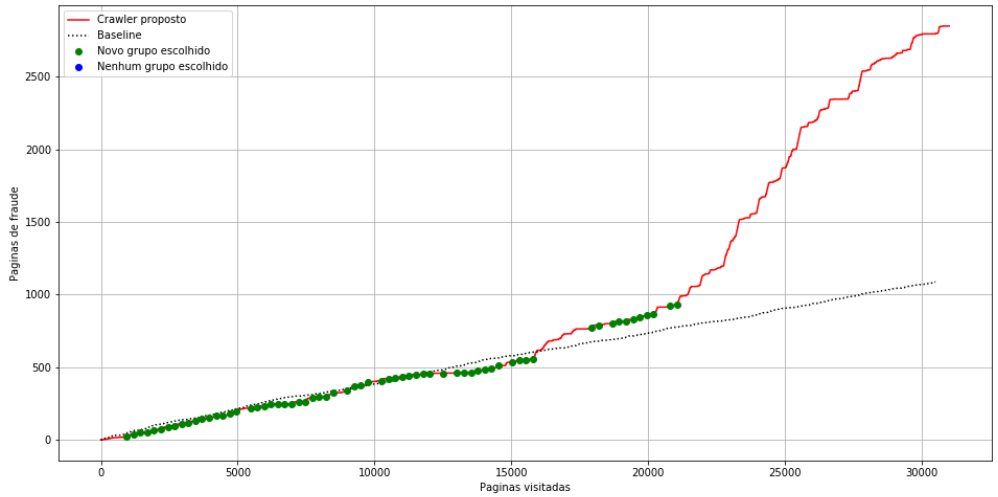


Figura 4.3: Harvest ratio do fórum B Exp-1.

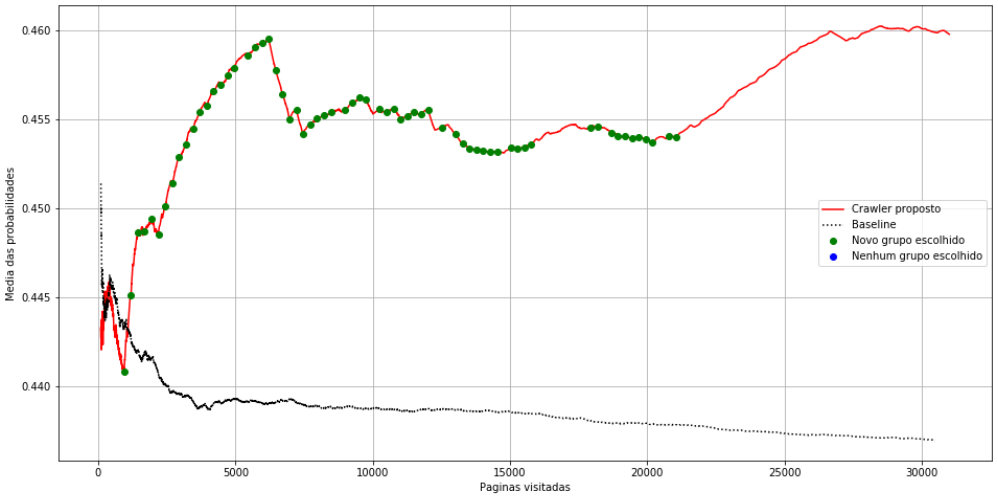


Figura 4.4: Média das probabilidades do classificador de fraudes fórum B Exp-1.

em 0.02, o que indica que, mesmo não achando uma taxa desejável, os grupos direcionaram o crawler para páginas com maior chance de serem de fraude.

Na execução dos crawlers para o fórum C, que possui a sua execução representada pelas figuras 4.5 e 4.6, a mudança de grupos se mantém da mesma maneira que ocorreu nos fóruns A e B, porém é menos recorrente e um grupo bom é encontrado rapidamente. Um trecho interessante de se observar é o que começa na marca de 2000 páginas e vai ate 8000 páginas.

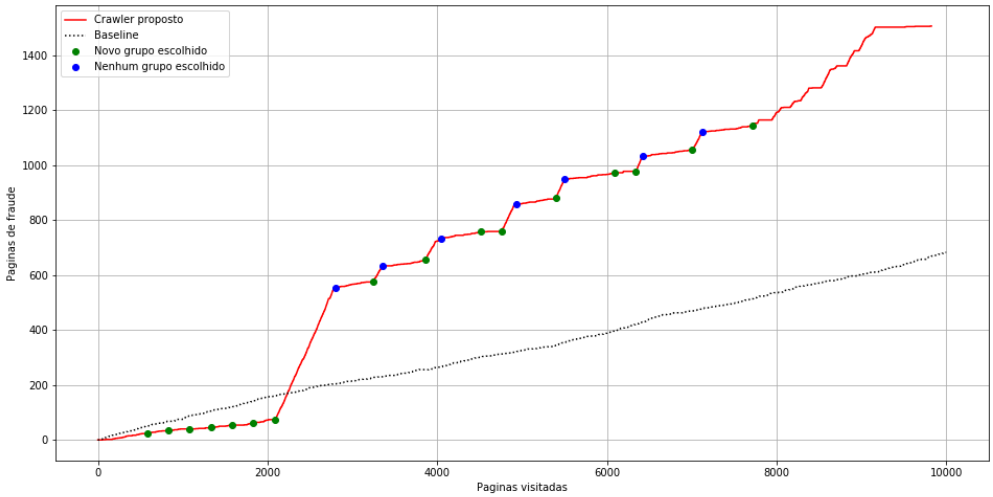


Figura 4.5: Harvest ratio do fórum C Exp-1 .

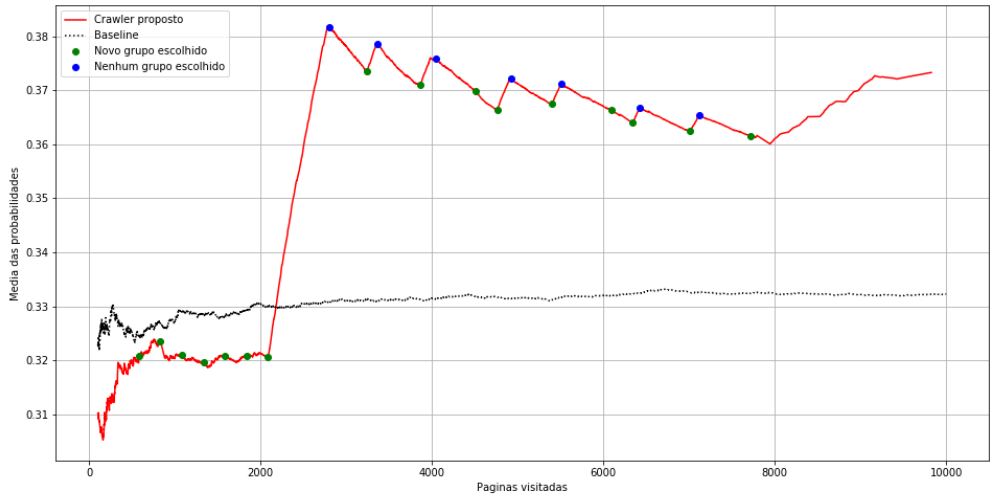


Figura 4.6: Média das probabilidades do classificador de fraudes fórum C Exp-1.

Nesse trecho, tanto no gráfico da figura 4.5 como na figura 4.6, observamos a escolha de vários grupos bons que possuem uma quantidade pequena de páginas, o que faz o crawler acabar rapidamente com os links do grupo escolhido e voltar a não ter grupo prioritário. Com essa falta de grupo, o crawler acaba sofrendo uma queda tanto na taxa de coleta quanto na média das probabilidades, o que indica que talvez, uma abordagem que escolha outro grupo, nesses casos, possa obter um melhor desempenho.

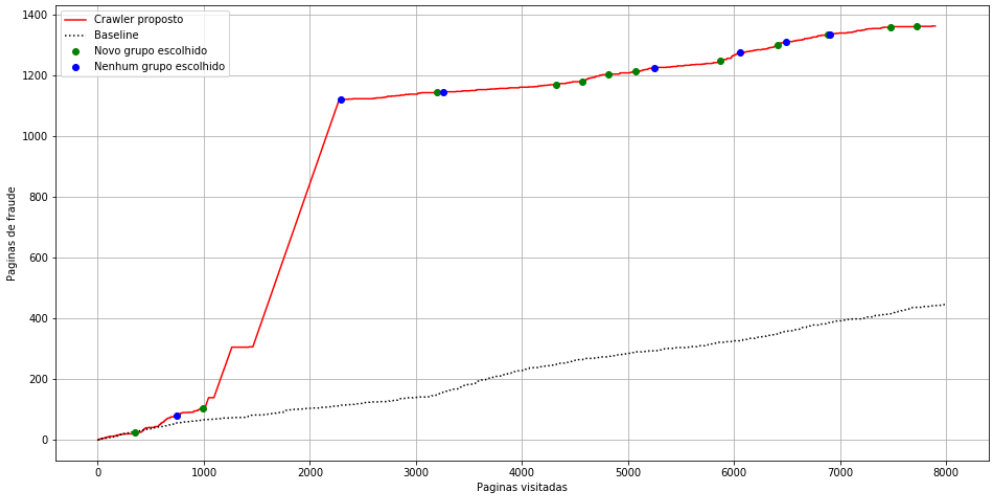


Figura 4.7: Harvest ratio do fórum D Exp-1.

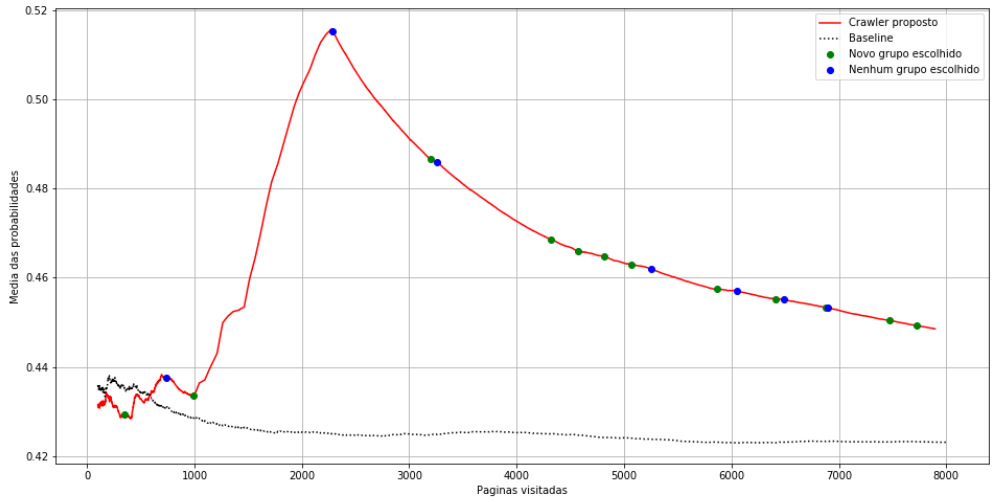


Figura 4.8: Média das probabilidades do classificador de fraudes fórum D Exp-1.

Na execução dos crawlers para o fórum D, que possui a sua execução representada pelas figuras 4.7 e 4.8, o fórum D, por possuir uma grande quantidade de páginas do tipo de fraude que o classificador estava melhor prevendo, conseguiu um desempenho muito bom desde o inicio de sua execução, e fica bem evidente, na figura 4.8, que, mesmo no trecho em que não foi encontrado nenhuma página de fraude, aproximadamente entre 1200 e 1500 páginas coletadas, a média das páginas continuava a subir.

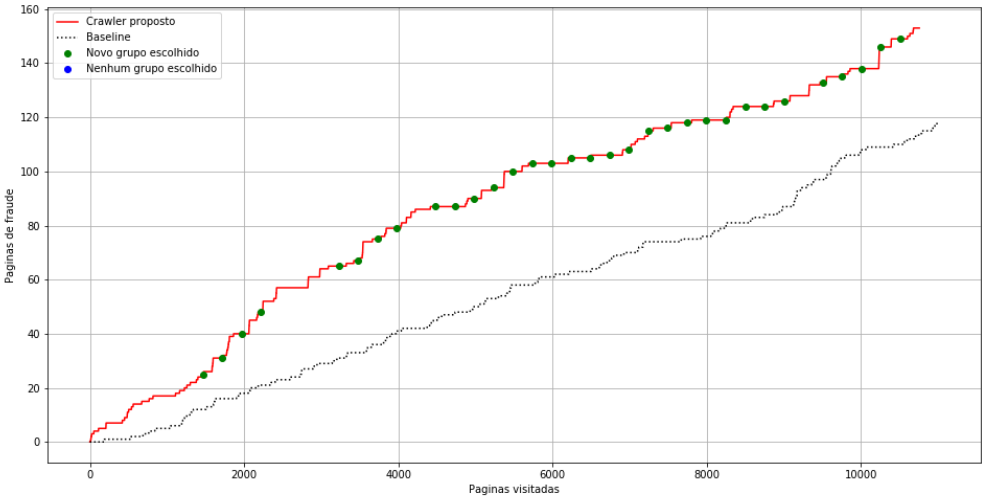


Figura 4.9: Harvest ratio do fórum F Exp-1.

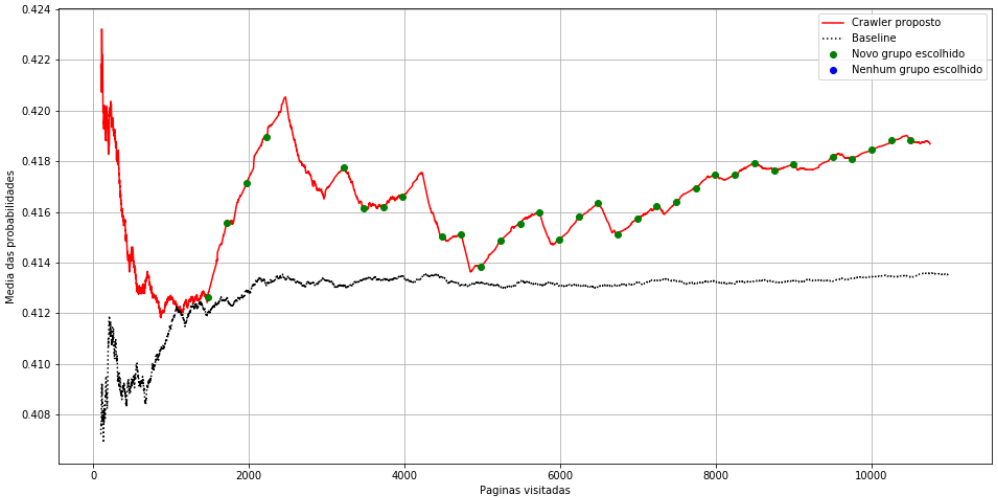


Figura 4.10: Média das probabilidades do classificador de fraudes fórum F Exp-1.

Na execução dos crawlers para o fórum F, que possui a sua execução representada pelas figuras 4.9 e 4.10, o fórum F, quando comparamos com os outros fóruns, achou poucas páginas de fraude, o que faz os clusters não acharem um grupos muito bons, mas mantém o harvest ratio sempre acima da baseline.

	Total	Páginas Positivas	Páginas Negativas	Harvest Ratio
Fórum A	50000	1399	48601	0.027980
Fórum A Baseline	50000	1349	48651	0.026980
Fórum B	50000	3108	46892	0.062160
Fórum B Baseline	50000	2293	47707	0.045860
Fórum C	15671	974	14697	0.062153
Fórum C Baseline	18105	988	17117	0.054571
Fórum D	43143	1977	41166	0.045824
Fórum D Baseline	42253	1893	40360	0.044802
Fórum E	20436	26	20410	0.001272
Fórum E Baseline	20441	26	20415	0.001272
Fórum F	14723	165	14558	0.011207
Fórum F Baseline	14573	147	14426	0.010087
Fórum G	50000	19	49981	0.000380

Tabela 4.4: overview do experimento 2

4.2 Experimento 2

O segundo experimento tinha como objetivo pegar 50 mil páginas de cada fórum, tanto para o baseline quanto para o crawler e observar se os comportamentos do experimento 1 se repetiriam. O experimento durou uma semana para ser completado, mas os fóruns C e F tiveram que ser interrompidos antes das 50 mil páginas serem coletadas, pois não havia mais tempo disponível.

O fórum D teve sua fronteira de links esvaziada e acabou a sua execução com 43 mil páginas. A diferença entre o total de páginas coletadas entre o baseline e o crawler do fórum D pode ser devida as mudanças na árvore do site e possíveis quedas da internet durante o processo de coleta. O fórum E, como no primeiro experimento, acabou sua execução com 20 mil páginas.

Como no primeiro experimento, os fóruns E e G continuaram sem conseguir rodar devidamente o crawler do modelo proposto e, para economizar tempo e recurso, o baseline do fórum G não foi rodado.

Na segunda execução dos crawlers para o fórum A, que possui a sua execução representada pelas figuras 4.11 e 4.12, o havest ratio do crawler ficou abaixo da baseline ate a marca de 48 mil páginas coletadas e, quando vemos o gráfico da imagem 4.12, conseguimos perceber que o crawler proposto teve o seu começo em uma região ruim do fórum e, quando o crawler escolhe o primeiro grupo, ele acaba não conseguindo variar muito as regiões do site e fica preso em uma área que não é tão boa.

Na segunda execução dos crawlers para os fóruns B, C e F, não tivemos nenhum comportamento novo por parte do crawler, que sempre manteve o havest ratio acima do baseline.

Na segunda execução dos crawlers para o fórum D, que possui a sua execução representada pelas figuras 4.13 e 4.14, temos o caso da fronteira de links ter acabado antes de chegar ao marco de 50 mil páginas. Podemos ver que, logo após o término do primeiro pico, que fica

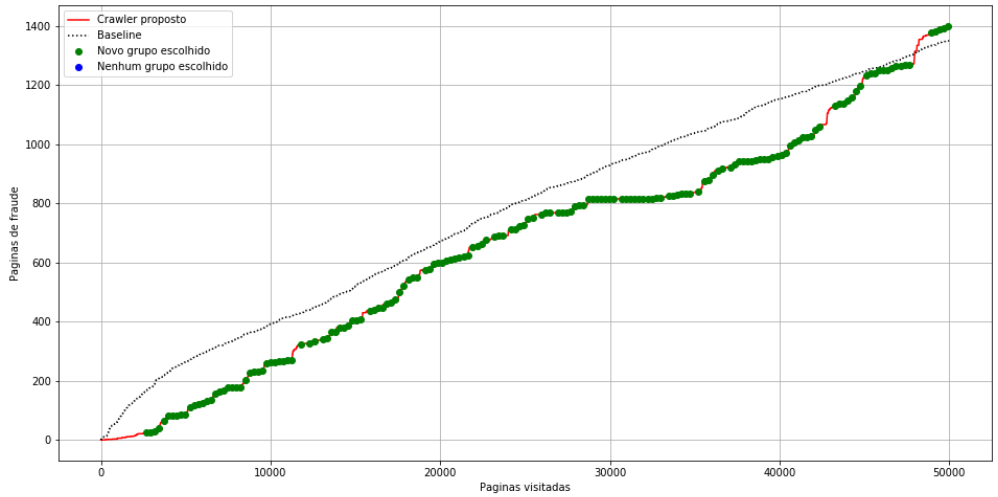


Figura 4.11: Harvest ratio do fórum A Exp-2.

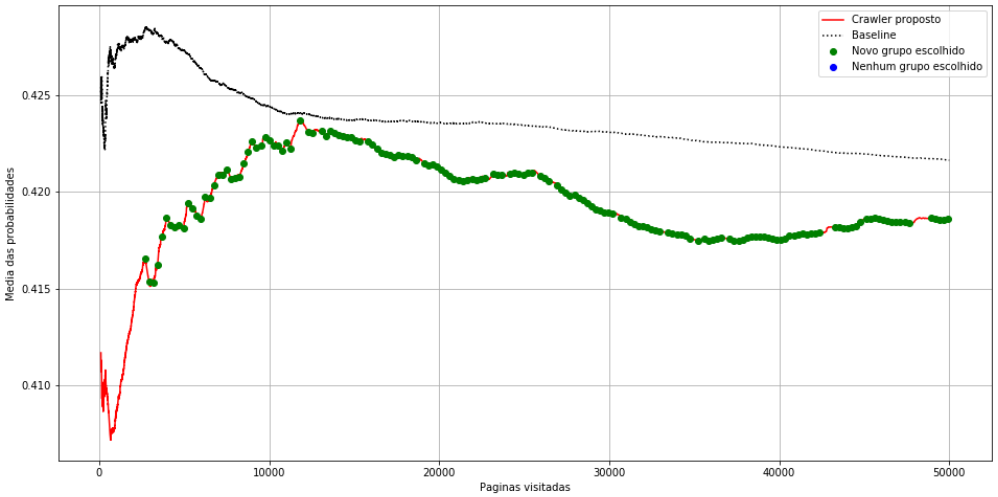


Figura 4.12: Média das probabilidades do classificador de fraudes fórum A Exp-2.

por volta do marco de 5 mil páginas coletadas, ocorre uma queda exponencial até se igualar as médias do baseline. Essa execução é interessante de ser observada, porque nos da uma ideia de qual seria o comportamento do crawler proposto, quando as páginas do site são exauridas.

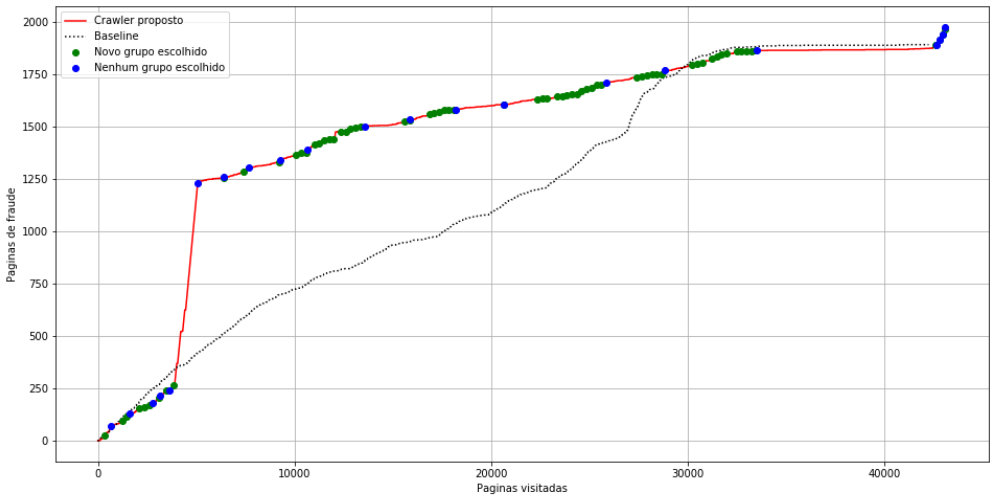


Figura 4.13: Harvest ratio do fórum D Exp-2.

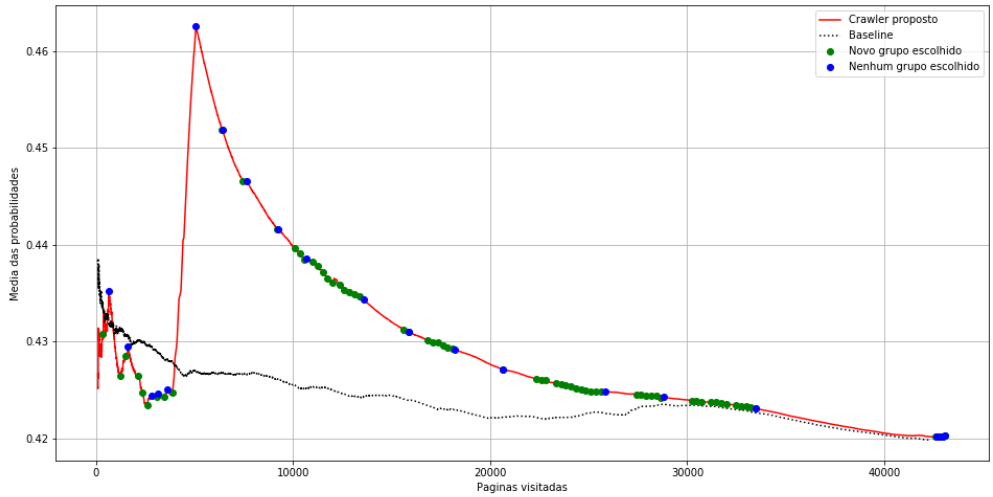


Figura 4.14: Média das probabilidades do classificador de fraudes fórum D Exp-2.

CAPÍTULO 5

Conclusão

Ao final dos 2 experimentos podemos identificar alguns problemas com o crawler proposto, como o uso excessivo do processo de clusterização, oque torna o modelo muito mais custoso que o baseline, e o problema de não conseguir identificar que o crawler está em uma região ruim do site e procurar outra. Porém como nos 2 experimentos realizados, o crawler proposto, conseguiu no final, um desempenho melhor que a baseline, e por ter conseguido esses resultados com um classificador montado com um dataset pequeno, e utilizando uma abordagem diferente, em que se usa uma aprendizagem supervisionada para rotular os resultados da aprendizagem não supervisionada, concluímos que o crawler proposto, apesar de não estar pronto para ser utilizado em aplicações reais, deve ser mais estudado e alterado para resolver ou mitigar os problemas encontrados, para que a sua execução se torne mais eficiente.

Referências Bibliográficas

[acf] acfe. What is fraud? Ultima visita dia 3 Dezembro de 2018, https://www.acfe.com/fraud-101.aspx.

[acf16] acfe. Report to the nations on occupational fraud and abuse, 2016. Ultima visita dia 3 Dezembro de 2018,https://s3-us-west-2.amazonaws.com/acfepublic/2018-report-to-the-nations.pdf.

[acf18] acfe. Report to the nations on occupational fraud and abuse, 2018. Ultima visita dia 3 Dezembro de 2018, https://s3-us-west-2.amazonaws.com/acfepublic/2018-report-to-the-nations.pdf.

[civ40] Casa civil. Código penal brasileiro, 1940. Ultima visita dia 3 Dezembro de 2018 http://www.planalto.gov.br/ccivil_03/Decreto-Lei/Del2848compilado.htm.

[CW16] Chih-Wei. A practical guide to support vector classification, 2016. Ultima visita dia 3 Dezembro de 2018, https://www.csie.ntu.edu.tw/ cjlin/papers/guide/guide.pdf.

[eJF07] Luciano Barbosa e Juliana Freire. An adaptive crawler for locating hidden-web entry points, 2007. Ultima visita dia 6 Dezembro de 2018, https://vgc.poly.edu/ juliana/pub/ache-www2007.pdf.

[Koe17] William Koehrsen. Random forest simple explanation, 2017. Ultima visita dia 3 Dezembro de 2018, https://medium.com/@williamkoehrsen/random-forest-simple-explanation-377895a60d2d.

[Pro18] Tor Project. Tor project overview, 2018. Ultima visita dia 3 Dezembro de 2018, https://www.torproject.org/about/overview.html.en.

[Sei18] George Seif. The 5 clustering algorithms data scientists need to know, 2018. Ultima visita dia 3 Dezembro de 2018, https://towardsdatascience.com/the-5-clustering-algorithms-data-scientists-need-to-know-a36d136ef68.

[sla] scikit learn. Random forest classifier. Ultima visita dia 3 Dezembro de 2018, https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html.

[slb] scikit learn. Support vector machines. Ultima visita dia 3 Dezembro de 2018, https://scikit-learn.org/stable/modules/svm.html.

[Ull] Shimon Ullman. Unsupervised learning clustering. Ultima visita dia 3 Dezembro de 2018, http://www.mit.edu/ 9.54/fall14/slides/Class13.pdf.

Este volume foi tipografado em LaTeX na classe UFPEThesis (www.cin.ufpe.br/~paguso/ufpethesis).