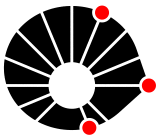

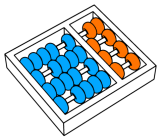

Detecção de Domínios Maliciosos Baseada em Técnicas de Aprendizado de Máquina

Pedro Henrique Barcha Correia *Hélio Pedrini*

Relatório Técnico - IC-PFG-20-08
Projeto Final de Graduação
2020 - Agosto

UNIVERSIDADE ESTADUAL DE CAMPINAS
INSTITUTO DE COMPUTAÇÃO

Detecção de Domínios Maliciosos

Baseada em Técnicas de Aprendizado de Máquina

Pedro Barcha[1], Hélio Pedrini[2]

[1] Instituto de Computação, Universidade Estadual de Campinas (UNICAMP)

13083-852 Campinas-SP, Brasil

pedro.barcha@tutanota.com

[2] Instituto de Computação, Universidade Estadual de Campinas (UNICAMP)

13083-852 Campinas-SP, Brasil

helio@ic.unicamp.br

RESUMO

Páginas da Internet que imitam serviços de bancos, transações monetárias e outras operações financeiras que requerem autenticação ou cadastro de informações sensíveis são chamadas de *phishing websites*. São desenvolvidas por atacantes, com o intuito de obter credenciais de usuários. Neste projeto, um detector de *phishing websites* foi desenvolvido a partir de um classificador treinado com diversas características de uma página, de forma a predizer se a mesma é *phishing* ou não. Acredita-se que o uso desta ferramenta em navegadores, *firewalls* e serviços de *e-mail* possa reduzir drasticamente a quantidade de vítimas desse tipo de fraude. O modelo final produzido apresentou 95% de acurácia e 95% de medida F1.

PALAVRAS-CHAVE

Páginas falsas; phishing websites; classificação; aprendizado de máquina; vetor de características; reconhecimento de padrões.

1. INTRODUÇÃO

Phishing é uma tentativa fraudulenta de obter informações sensíveis, como *login* e senha, de usuários de um sistema [1,2,12,13]. Geralmente, isto é feito por meio de um *website* falso, que imita outro legítimo, em que os usuários digitam seus dados acreditando estar no domínio real.

A fim de mitigar esse problema, este projeto visa desenvolver uma ferramenta que, dada a URL de uma página, prediz se essa é legítima ou não, por meio de um modelo treinado com técnicas de aprendizado de máquina.

A Seção 2 descreve a metodologia proposta neste trabalho. A Seção 3 apresenta e discute os resultados obtidos a partir do método desenvolvido. A Seção 4 apresenta as conclusões do trabalho. Uma lista de referências consultadas durante o desenvolvimento do trabalho é incluída no final do documento.

2. METODOLOGIA

Esta seção apresenta as principais etapas do método desenvolvido para detecção de domínios maliciosos (*phishing websites)*.

2.1 Extrator de Características

Frequentemente, páginas *phishing* possuem características e padrões que as distinguem de páginas legítimas [1,2,10,11,12,13]. Portanto, a partir da extração de diversos atributos ou elementos de uma página (os quais chamaremos de características), pode-se classificar sua legitimidade.

Assim, desenvolveu-se um programa capaz de extrair 24 características de uma página, a partir de sua URL. Algumas delas são diretamente relacionadas à própria cadeia de caracteres da URL (como seu tamanho ou o uso de determinados caracteres), enquanto outras estão relacionadas ao seu conteúdo (como o uso de *pop-ups* ou *cookies*).

É importante ressaltar que nem todas as funções extratoras de características foram utilizadas na versão final do projeto, visto que muitas foram descartadas após avaliadas (vide Seção 3).

As características de *a* até *o,* descritas a seguir, foram selecionadas para o modelo final:

a) Endereço IP na URL

Explicação: *sites* maliciosos podem não possuir domínio, sendo possível acessá-los somente a partir de seu endereço IP público. Exemplo:

Exemplo: https://222.222.222.222/scam.html.

Valor: 0 ou 1.

b) Arroba na URL

Explicação: diversos navegadores ignoram o conteúdo precedente ao arroba. Atacantes utilizam esta técnica para ludibriar seus alvos a acreditarem estar acessando um domínio legítimo.

Exemplo: https://paypal.com@peypal.com leva ao site peypal.com.

Valor: 0 ou 1.

c) "//" na URL

Explicação: de maneira análoga à característica supracitada, em alguns navegadores o conjunto "//" fora do contexto do protocolo de transferência (http/https) faz com que o texto que lhe precede seja ignorado.

Exemplo: https://paypal.com//peypal.com leva ao site peypal.com.

Valor: 0 ou 1.

d) Hífen na URL

Explicação: hífens são raramente utilizados em URLs legítimas. Sites *phishing*, no entanto, utilizam esse caractere para conferir legitimidade ao domínio.

Exemplo: http://login-bradesco.com.br.

Valor: 0 ou 1.

e) Uso de HTTPS

Explicação: HTTPS é um protocolo de transferência que estabelece uma conexão segura entre servidor e cliente. Confere, portanto, credibilidade a uma página e é mais comumente utilizado em páginas legítimas.

Exemplo: https://abc.com.br uitiliza HTTPS, diferentemente de http://abc.com.br.

Valor: 0 ou 1.

f) *Favicon* proveniente do próprio domínio

Explicação: *favicon* é o ícone, geralmente com o logo da página, exibido na aba do navegador. *Favicons* carregados de um domínio diferente do acessado, podem indicar que a página não é legítima.

Valor: 0 ou 1.

g) Uso de *pop-up*

Explicação: *pop-ups* são janelas criadas pelo site acessado e, atualmente, não é mais tão comum seu uso em *sites* legítimos. Isso se deve principalmente ao fato de que navegadores vêm criando empecilhos ao seu uso, devido à sua capacidade de abrir domínios distintos do acessado. Por outro lado, essa função pode ser conveniente a atacantes.

Valor: 0 ou 1.

h) Tamanho da URL

Explicação: URLs longas podem conter conteúdo malicioso ou serem utilizadas para esconder o domínio *phishing*.

Exemplo:
https://peypal.com/xxxxqqq/content/0018a2125ec6686c9fa0f24ab7312716/?user=&_verify?service=mail&data:text/html;charset=utf-8;base64,PGh0bWw+DQo8c3R5bGU+IGJvZHkgeyBtYXJnaW46IDA7IG92ZXJmbG93OiBoaWRkZW47IH0gPC9zdHlsZT4NCiAgPGlmcmFt

Valor: quantidade de caracteres presentes na URL.

i) Número de *links* próprios

Explicação: *links* próprios são aqueles que apontam para o próprio domínio em que se encontram. Na média, páginas legítimas possuem mais *links* próprios.

Valor: número de *links* próprios.

j) Quantidade de portas abertas

Explicação: o servidor de uma página, se bem configurado, deve possuir abertas apenas as portas 80 (HTTP) ou 443 (HTTPS). Outras portas abertas podem significar uma ameaça ao usuário, pois permitem o uso de serviços não essenciais por parte do servidor.

Implementação: é requisitada conexão às portas 21 (FTP), 22 (SSH) e 3306 (MySQL) do endereço IP da página, a fim de identificar se estão abertas. Outras portas não são testadas para não tornar o tempo de cálculo da característica demasiadamente elevado.

Valor: quantidade de portas abertas dentre 21, 22 e 3306.

k) Idade do certificado

Explicação: geralmente *phishing websites* ficam ativos por pouco tempo, já que após serem notoriamente identificados como maliciosos, são desativados. Assim, quando possuem um certificado, ele costuma ser recente. S*ites* legítimos, no entanto, podem possuir um certificado há anos.

Valor: idade do certificado.

l) Idade do domínio

Explicação: conforme mencionado, *sites phishing* costumam ficar ativos por pouco tempo, assim a idade média de seus domínios é menor do que a de *sites* legítimos.

Valor: idade do domínio.

m) Tempo de registro do domínio

Explicação: como *phishing websites* tendem a ficar ativos por pouco tempo, o atacante geralmente contrata o domínio por um período curto, enquanto *sites* legítimos costumam fazer contratos com duração de vários anos, para evitar renová-los frequentemente.

Valor: tempo de registro do domínio.

n) *Ranking Alexa*

Explicação: o *Ranking Alexa* fornece uma lista em que milhões de *websites* são categorizados por uma combinação de seu número de acessos e de usuários únicos. Na média, *sites* legítimos apresentam uma posição melhor do que *phishing websites*.

Valor: posição no *Ranking Alexa*.

o) *PageRank*

Explicação: o algoritmo *PageRank* fornece um valor inteiro de 0 a 10 que mede a importância de uma página, de acordo com o número e a qualidade de *links* na *web* que apontam para ela.

Valor: valor atribuído pelo *PageRank.*

As características a seguir, após avaliadas, não foram utilizadas no modelo final:

p) Simulação de HTTPS

Explicação: *phishers* podem colocar "https" no início do domínio para simular que o site utiliza HTTPS.

Exemplo: http://https-www-paypal.com simula o uso de HTTPS, mas utiliza HTTP.

Valor: 0 ou 1.

q) Envio de informações por *email*

Explicação: *sites* maliciosos podem enviar credenciais submetidas pelo usuário diretamente para um *e-mail* do atacante.

Implementação: busca pelos métodos *mailto:* (HTML) e *mail()* (PHP) .

Valor: 0 ou 1.

r) Uso de serviços de encurtamento de URL

Explicação: atacantes podem utilizar serviços de encurtamento para esconder URLs suspeitas. Isso aumenta a chance de internautas acessá-las ao encontrá-las, sendo, assim, redirecionados para a página *phishing*.

Exemplo: uma URL com mais de 100 caracteres pode ser encurtada para https://bit.ly/3dqsP2K.

Implementação: a URL é comparada a uma lista de serviços de encurtamento.

Valor: 0 ou 1.

s) Botão direito do *mouse* desativado

Explicação: *phishers* podem desativar o uso do botão direito do *mouse* no *site*, para que o usuário tenha dificuldades em obter seu código fonte malicioso.

Implementação: busca no código fonte por "event.button==2" (*Javascript*), que desabilita o botão direito.

Valor: 0 ou 1.

t) Uso de *iframe*

Explicação: *iframe* exibe uma outra página dentro da atual. Pode ser utilizada para fins nocivos, como carregar *sites* maliciosos sem o consentimento do usuário.

Valor: 0 ou 1.

u) Análise de âncoras

Explicação: em HTML, âncoras são elementos que, quando clicados, levam a outros pontos da mesma página ou a páginas diferentes (*links*). Um alto índice de âncoras que levam a outros domínios ou que não levam a nada pode indicar intenções maliciosas.

Valor: percentagem de âncoras suspeitas, conforme descrito acima.

v) Quantidade de redirecionamentos

Explicação: *phishing websites* podem apresentar uma quantidade média maior de redirecionamentos ao acessá-lo, em relação a *sites* legítimos.

Valor: quantidade de redirecionamentos.

w) Quantidade de subdomínios

Explicação: na URL, subdomínios precedem o domínio de um *site*. Em geral, páginas legítimas não possuem mais de um, além do "www".

Exemplo: http://www.hostpoint.ch.17a902ef.tcorner.com, cujo domínio é *tcorner*, possui 3 subdomínios, além do "www".

Valor: a quantidade de subdomínios presente na URL, fora "www".

As características de *a* até *w*, descritas acima, foram apresentadas por Mohammad et al. [1,2], sendo todas originalmente binárias ou ternárias (indicando comportamento malicioso, suspeito ou legítimo). A característica relacionada ao tamanho da URL, por exemplo, era originalmente ternária, recebendo determinado rótulo se sua quantidade de caracteres estivesse entre valores preestabelecidos. Por ser uma prática pouco robusta, nós a substituímos pelo próprio valor extraído, que, no exemplo acima, é a quantidade de caracteres.

Introduziu-se também a seguinte característica de teste*:*

x) Valor aleatório

Explicação: característica utilizada para avaliar as demais, visto que características efetivas não deveriam apresentar importância menor ou igual à ela. Ela não é introduzida no modelo final.

Valor: 0 ou 1, determinado de maneira aleatória.

2.2 Base de Dados

Páginas como *Phishtank* fornecem URLs de páginas *phishing*, enquanto outros serviços como o *Umbrella* fornecem listas de *websites* supostamente legítimos. Desta forma, obteve-se 7120 URLs etiquetadas como *phishing* ou legítimas, em igual proporção.

Outrossim, garantiu-se a unicidade de domínios nesta seleção de *websites*, para evitar enviesamento do *dataset*. Assegurou-se também que as URLs retornavam uma resposta válida.

Em seguida, realizou-se a extração de características das páginas obtidas, por meio do programa descrito na seção anterior. Obteve-se, ao fim do processo, uma planilha em que cada linha possui os valores das características e a etiqueta (legítimo ou *phishing*) de dado *site*.

2.3 Treinamento do Modelo Classificador

Para treinar o modelo classificador de *URLs phishing*, dividiu-se o *dataset* obtido na etapa anterior em 25% para teste e 75% para treinamento. O procedimento foi aplicado de maneira estratificada, garantindo, portanto, que ambos os conjuntos possuíssem metade de seus dados *phishing* e metade legítimos.

Em seguida criou-se um *pipeline*, responsável por garantir que um conjunto de dados providos a ele, em qualquer momento (treinamento, teste ou uso real), seja normalizado e, em seguida, passado para o algoritmo classificador desejado. Para esse projeto testou-se as normalizações por MinMax [3] e por Z-score [4]. Já os algoritmos classificadores avaliados foram Árvore de Decisão, Regressão Logística, Máquina de Vetores de Suporte e Floresta Aleatória [5]. Ao final, selecionou-se o algoritmo Floresta Aleatória, com normalização Z-score (vide Seção 3).

Para cada algoritmo, selecionou-se possíveis valores para seus hiperparâmetros e, em seguida, aplicou-se uma busca em grade [6], com validação cruzada de 5-*Folds* [7], a fim de determinar os hiperparâmetros ótimos. Devido ao *pipeline*, garante-se que cada *fold*

possuirá normalização separada para seu treinamento e teste. Ao fim do processo, obtém-se o modelo treinado com o melhor resultado na validação cruzada.

A seguir, aplicando o conjunto de teste (ainda não utilizado) ao modelo recém-treinado, adquire-se os valores de sua acurácia, precisão, revocação e medida *F1* [8,9].

Repetindo o processo descrito nos dois parágrafos anteriores para os distintos algoritmos classificadores e formas de normalização, obteve-se sua melhor combinação.

Por fim, foram avaliados os impactos da remoção de determinados conjuntos de características do *dataset*, com o intuito de determinar qual conjunto deveria compor, de fato, o modelo final. Realizou-se essa avaliação por meio da análise do *dataset* e pelo cálculo da Importância de Gini [10] das características.

2.4 Uso do Modelo Classificador Treinado

Para utilizar o modelo treinado para classificar um *site* ainda não visto, suas características são extraídas e analisadas pelo modelo treinado. Em seguida, o classificador prediz se o *site* é legítimo ou não.

3. RESULTADOS E DISCUSSÃO

Nesta seção, os resultados obtidos a partir da aplicação do método proposto são apresentados e discutidos.

3.1 Primeiro Experimento

3.1.1 Seleção das Características Binárias

Após a preparação do *dataset*, conforme descrito na Seção 2.2, suas características binárias foram analisadas com o intuito de averiguar se produziam o resultado esperado. Assim, gerou-se a Tabela 1, em que verdadeiros positivos correspondem às características que demonstraram resultado esperado para páginas *phishing*. Já os falsos positivos são as características que apresentaram valor oposto ao esperado em páginas legítimas, isto é, valor que deveria ocorrer em páginas maliciosas.

Característica	Verdadeiros Positivos	Falsos Positivos
IP na URL	81	0
Serviço de encurtamento	16	6
@ na URL	91	0
// na URL	50	0
Uso de hífen	503	121
Uso de HTTPS	2776	771
Favicon proveniente do próprio domínio	3247	2224
Simulação HTTPS	3	0
Envio de informações por *e-mail*	352	428
Botão direito do *mouse* desativado	6	0
Uso de *pop-up*	310	77
Uso de iframe	1	1

Tabela 1: Contagem de verdadeiros positivos e falsos positivos para cada característica binária.

As características marcadas em amarelo na Tabela 1 foram removidas do *dataset,* pois não apresentaram o resultado esperado; em "envio de informações por e-mail" obteve-se resultado contrário ao desejado, isto é, mais falsos positivos do que verdadeiros positivos. Para as demais características em amarelo, obteve-se poucos verdadeiros positivos, sendo, portanto, características pouco úteis na identificação da legitimidade de uma página *web*.

As características supracitadas foram, portanto, descartadas e deixaram de ser utilizadas no projeto.

3.1.2 Seleção do Algoritmo de Treinamento

Utilizando as características não eliminadas na etapa anterior, foram treinados 10 modelos para cada algoritmo proposto na Seção 2.3.

Algoritmo	Floresta Aleatória	Árvores de Decisão	Regressão Logística	Máquina de Vetores de Suporte
Acurácia	95,03%	87,13%	78,6%	45,08%

Tabela 2: Acurácia do melhor modelo treinado, para cada algoritmo.

Conforme discriminado na Tabela 2, atingiu-se maior acurácia com o algoritmo de Floresta Aleatória, o qual passou a ser utilizado em todas as etapas seguintes do projeto.

3.1.3 Análise da Importância das Características

Para o melhor modelo obtido na etapa anterior, calculou-se a importância de cada característica, conforme mostrado na Tabela 3.

Característica	**Importância**
Tamanho da URL	38,5%
Page Rank	12,6%
Portas abertas	10,6%
Ranking Alexa	10,4%
Idade do certificado	4,8%
Uso de HTTPS	4,7%
Tempo de registro do domínio	4,3%
Número de subdomínios	3,9%
Idade do domínio	3,4%
Quantidade de redirecionamentos	1,9%
Análise de Âncoras	1,6%
Número de *links* próprios	1,4%
Favicon proveniente do próprio domínio	0,5%

Uso de hífen	0,3%
Uso de *pop-up*	0,3%
// na URL	0,1%
IP na URL	0%
@ na URL	0%

Tabela 3: Importância de Gini, em ordem decrescente, das características do melhor modelo treinado.

De acordo com a Tabela 3, a característica relacionada ao tamanho da URL demonstrou importância muito maior do que as demais: 38,5%. Há uma acentuada discrepância mesmo em relação à segunda característica mais importante, que possui importância de 12,6%. Além disso, ela representa mais de um terço da importância total das 18 características.

Conclui-se, portanto, que há enviesamento. Como consequência, na prática, URLs legítimas longas analisadas por esse modelo são erroneamente classificadas como *phishing*, independentemente do valor das demais características.

Identificou-se que o viés é proveniente do *dataset* gerado, em que, diferentemente das URLs legítimas, diversas URLs *phishing* contêm caminhos. Um exemplo de endereço com caminho é: *https://atagucsea.com/wp-content/themes/twentyfifteen/cloud9/gucemail/*. Sem nenhum caminho, seria: *https://atagucsea.com*. Portanto, a parte legítima do *dataset*, em sua atual conjectura, não representa um cenário real, em que internautas navegam por diversos caminhos de dada página.

3.2 Segundo Experimento

3.2.1 Correção do Enviesamento do *Dataset*

A fim de mitigar o enviesamento relatado no experimento anterior, um novo *dataset* foi gerado, com URLs legítimas contendo caminhos. Isto foi realizado por meio de uma busca por *links* próprios no HTML das páginas. Desse modo, o tamanho médio dessas URLs aumentou e, consequentemente, o valor da característica associada.

3.2.2 Seleção das Características Não-Binárias

A seguir, calculou-se a média dos valores das características não-binárias do *dataset*. Deste modo, foi possível identificar características com resultados fora do esperado, assim como ocorreu com as características binárias no experimento anterior.

Característica	***Sites Phishing***	***Sites* Legítimos**
Tamanho da URL	64,74	47,11
Número de subdomínios	0,28	0,28
Idade do certificado (dias)	120,99	387,01
Tempo de registro do domínio (dias)	427	913,93
Portas abertas	1,46	0,16
Análise de âncoras	42,43	41,98
Redirecionamentos	0,37	0,49
Idade do domínio (dias)	1863,91	4292,09
Ranking Alexa	1103538,48	680310,13
Page Rank	0,75	3,60
Número de *links* próprios	3,94	16,44

Tabela 4: Média dos valores das características não-binárias para *sites phishing* e *sites* legítimos.

As características em amarelo na Tabela 4 apresentaram valor médio próximo para *sites phishing* e legítimos e, portanto, não são capazes de fornecer indícios da procedência da página. Assim, foram removidas do *dataset* e deixaram de ser utilizadas no projeto.

3.2.3 Treinamento Intensivo

Utilizando apenas as características mantidas após os dois processos de seleção (Seções 3.2.1 e 3.2.2), 500 modelos foram treinados.

A fim de avaliar os modelos produzidos para além de sua acurácia, foram introduzidos os cálculos de precisão e revocação e medida F1, reportados na Tabela 5.

	Precisão	Revocação	F1-score
Legítimo	93%	96%	95%
Phishing	95%	93%	94%

Acurácia
94,38%

Tabela 5: Resultados do modelo com maior acurácia, após treinamento intensivo.

A partir dos resultados da precisão, obtém-se a informação de que, quando uma amostra é classificada como legítima, essa classificação está correta 93% das vezes. Já as classificações como *phishing* estão corretas 95% da vezes.

Da revocação, obtém-se que 96% dos *sites* legítimos são reconhecidos como tais. Para *phishings,* o valor é de 93%.

Observa-se que a melhor acurácia obtida entre os 500 modelos treinados (94,38%) é menor do que a obtida no primeiro experimento (95,03%, Tabela 2), para 30 modelos. No entanto, a acurácia apresentada na Tabela 5 é mais próxima da real, uma vez que o modelo não está mais enviesado.

Para demonstrar a última afirmação, as importâncias do modelo em questão foram calculadas. Obteve-se que a importância da característica de tamanho da URL, originalmente enviesada, diminuiu de 38,5% para 18,2%, conforme apresentado a seguir.

Característica	Importância
Tamanho da URL	18.2%
Ranking Alexa	17,2%
Page Rank	15,8%
Portas abertas	13,1%
Idade do certificado	9,6%
Idade do domínio	7,4%
Uso de HTTPS	5,9%
Tempo de registro do domínio	5,9%
Número de *links* próprios	4,5%

Favicon proveniente do próprio domínio	1%
Uso de hífen	0,5%
Uso de *pop-up*	0,5%
IP na URL	0,1%
@ na URL	0,1%
// na URL	0,1%

Tabela 6: Importância, em ordem decrescente, das características do modelo das Tabelas 5a e 5b.

Nota-se que, apesar de a característica de tamanho da URL continuar sendo a mais importante, sua discrepância em relação à segunda colocada diminuiu sensivelmente. No experimento anterior a diferença entre ambas era 25,9%, enquanto no presente experimento é 1%.

Por outro lado, as últimas colocadas apresentam pouca importância. No experimento a seguir, será analisado o impacto de removê-las do *dataset* (Seção 3.3.3).

3.3 Terceiro Experimento

3.3.1 Normalização

Como primeiro passo do experimento, foi introduzido o processo de normalização. Conforme mencionado na seção 2.3, distintas normalizações foram feitas para as etapas de treinamento, validação cruzada e teste. Desse modo, evita-se que haja interferência dos dados de uma etapa nas demais, o que causaria enviesamento.

A fim de avaliar o desempenho dos algoritmos *MinMax* e *Z-score*, 20 modelos foram treinados para cada. A mesma quantidade de modelos foi gerada sem normalização, para fins comparativos.

Normalização	**Acurácia Máxima**
MinMax	93,76%
Z-Score	93,76%
-	93,15%

Tabela 7: Acurácia do melhor modelo treinado com cada algoritmo de normalização.

Obteve-se melhor acurácia quando o processo de normalização foi aplicado e, portanto, foi incorporado ao treinamento das etapas subsequentes.

No entanto, para ambos os algoritmos de normalização obteve-se igual acurácia para seus melhores modelos. Assim, não foi possível descartar o uso de nenhum por enquanto.

3.3.2 Análise de Componentes Principais

Tendo em vista as características com pouca importância apresentadas na Tabela 6, realizou-se a Análise de Componentes Principais (PCA) [11]. Esse procedimento visa a redução de dimensionalidade do modelo, isto é, a redução da quantidade de características.

20 modelos foram treinados com PCA para cada algoritmo de normalização. Para *MinMax* foi possível obter redução de dimensionalidade preservando 99% da variância original das características. Para *Z-score*, a variância mínima para a qual houve redução foi 97%.

Normalização	Variância	Redução	Acurácia Máxima
Z-score	97%	1	91,85%
MinMax	99%	2	89,38%

Tabela 8: Preservação da variância original das características, número de dimensões reduzidas e acurácia. Todos os valores são relativos ao melhor modelo treinado com PCA, de cada algoritmo de normalização.

Para *Z-score* houve redução de dimensionalidade em 1, enquanto para *MinMax* a redução foi de 2 dimensões. Obteve-se melhor acurácia para PCA com *Z-score*, no entanto o valor (91,85%) é menor do que o obtido sem PCA (93.76%, Tabela 7), para a mesma quantidade de modelos treinados.

Conclui-se, portanto, que a técnica PCA não se mostrou uma opção viável para seleção de características. Assim, um método alternativo foi estudado e descrito na seção a seguir.

3.3.3 Análise das Características com Baixa Importância

Introduziu-se a característica x (Seção 2.1), até então não utilizada. Essa característica atribui aleatoriamente o valor 0 ou 1 para cada amostra do *dataset*. Assim, a utilizaremos no treinamento para determinar características com importância menores que a sua. Entende-se que essas, ao menos separadamente, possuam impacto pouco consistente na

identificação da procedência de *websites*, já que se apresentam menos importantes que uma característica de caráter aleatório.

Treinando um modelo com a característica *x*, obteve-se as seguintes importâncias:

Característica	**Importância**
Ranking Alexa	19,4%
Tamanho da URL	19,1%
Portas abertas	13,1%
Page Rank	12,7%
Idade do certificado	9,2%
Idade do domínio	7,3%
Tempo de registro do domínio	6,4%
Uso de HTTPS	5,1%
Número de *links* próprios	4,5%
Valor aleatório	1,1%
Favicon proveniente do próprio domínio	0,9%
Uso de hífen	0,5%
Uso de *pop-up*	0,4%
IP na URL	0,1%
@ na URL	0,1%
// na URL	0,1%

Tabela 9: Importância, em ordem decrescente, das características do melhor modelo treinado.

Analisando a Tabela 9, identifica-se as características de baixa importância (abaixo da característica aleatória): // na URL, @ na URL, IP na URL, Uso de *pop-up*, Uso de hífen, *Favicon* proveniente do próprio domínio.

Cumprida sua finalidade, a característica aleatória *x* não é mais utilizada em nenhuma etapa subsequente.

Em seguida, testou-se a remoção de cada uma das características pouco importantes separadamente, assim como a remoção de todas concomitantemente. Para isso, foram criadas 5 estruturas de modelos*;* cada uma com uma separação específica do *dataset* em treinamento, teste e validação cruzada. Além disso, possuem uma construção específica da Floresta Aleatória. Assim, para cada combinação de normalização e características, pôde-se criar 5 modelos consistentes entre as experimentações.

Normalização	**Características(s) Removida(s)**	**Acurácia Máxima**
Z-Score	-	92,98%
MinMax	-	93,2%
Z-Score	// na URL	92,81%
MinMax	// na URL	93,03%
Z-Score	@ na URL	92,87%
MinMax	@ na URL	92,92%
Z-Score	IP na URL	92,87%
MinMax	IP na URL	92,87%
Z-Score	Hífen na URL	93,03%
MinMax	Hífen na URL	92,87%
Z-Score	Uso de *pop-up*	93,09%
MinMax	Uso de *pop-up*	93,20%
Z-Score	*Favicon* proveniente do próprio domínio	93,15%

MinMax	*Favicon* proveniente do próprio domínio	93,15%
Z-Score	Todas com baixa importância	92,87%
MinMax	Todas com baixa importância	92,98%

Tabela 10: Acurácia do melhor modelo treinado (dentre 5), para distintos algoritmos de normalização e distintos conjuntos de características.

Os conjuntos de características e normalizações marcados em amarelo na Tabela 10 apresentaram os melhores resultados. Em particular o uso de *MinMax* sem remoção de características e com remoção da característica de *pop-up* atingiram a melhor acurácia (92,3%). A seguir, foi realizado o treinamento de 100 modelos para cada conjunto destacado na Tabela 10.

Normalização	**Característica(s) Removida(s)**	**Acurácia Máxima**
MinMax	-	93,93%
Z-Score	-	94,04%
MinMax	// na URL	94,44%
Z-Score	Hífen na URL	94,04%
Z-Score	Uso de *pop-up*	94,44%
MinMax	Uso de *pop-up*	94,55%
Z-Score	*Favicon* proveniente do próprio domínio	93,93%
MinMax	*Favicon* proveniente do próprio domínio	93,99%
MinMax	Todas com baixa importância	93,65%

Tabela 11: Acurácia do melhor modelo treinado (dentre 100), para distintos algoritmos de normalização e distintos conjuntos de características.

Na Tabela 11, estão destacados em amarelo os conjuntos de características e normalizações que apresentaram melhor acurácia, após o treinamento de 100 modelos para cada. Em

particular, esses resultados foram melhores do que o melhor modelo não enviesado gerado até então no projeto (Tabela 5), com acurácia de 94,38%, após 500 treinamentos.

Observa-se também que o pior resultado ocorreu com a remoção de todas as 6 características menos importantes. Uma possível explicação é que suas importâncias são baixas separadamente, porém determinadas combinações que as envolvem podem ser relevantes para a identificação da procedência de um *website*.

3.3.4 Modelo Final

Por fim, 1000 modelos foram treinado para os conjunto em amarelo da Tabela 11. Também foram treinados modelos sem remoção de características.

Normalização	Característica(s) Removida(s)	Acurácia Máxima
-	-	94,38%
MinMax	-	94,83%
Z-Score	-	95,00%
MinMax	// na URL	94,72%
Z-Score	Uso de *pop-up*	94,44%
MinMax	Uso de *pop-up*	94,55%

Tabela 12: Acurácia do melhor modelo treinado (dentre 1000), para distintos algoritmos de normalização e distintos conjuntos de características.

	Precisão	Revocação	Medida F1
Legítimo	94%	97%	95%
Phishing	96%	93%	95%

Tabela 13: Precisão, Revocação e medida F1 do modelo final (em amarelo na Tabela 12).

Após o treinamento intensivo dos 6 casos apresentados na Tabela 12, obteve-se melhor acurácia (95%) para normalização por *Z-score* e sem remoção de características. Portanto, no modelo final, as características com baixa importância foram mantidas. Isso indica que a

seleção de características binárias e não-binárias realizada nos dois primeiros experimentos (Seções 3.1 e 3.2) foi suficiente.

A pior acurácia na Tabela 12 também ocorreu para um caso em que todas as características foram mantidas, porém sem normalização, o que revela a importância deste procedimento.

A partir dos resultados da precisão do melhor modelo, obtém-se a informação de que quando uma amostra é classificada como legítima, essa classificação está correta 94% das vezes. Já as classificações como phishing estão corretas 96% da vezes.

Da revocação, obtém-se que 97% dos *sites* legítimos são reconhecidos como tais. Para *phishings,* o valor é de 93%

Ao fim, foram mantidas 15 características:

- *Ranking Alexa*
- Tamanho da URL
- Portas abertas
- *Page Rank*
- Idade do certificado
- Idade do domínio
- Tempo de registro do domínio
- Uso de HTTPS
- Número de *links* próprios
- *Favicon* proveniente do próprio domínio
- Uso de hífen
- Uso de *pop-up*
- IP na URL
- @ na URL
- // na URL

3.3.5 Classificador e Desempenho

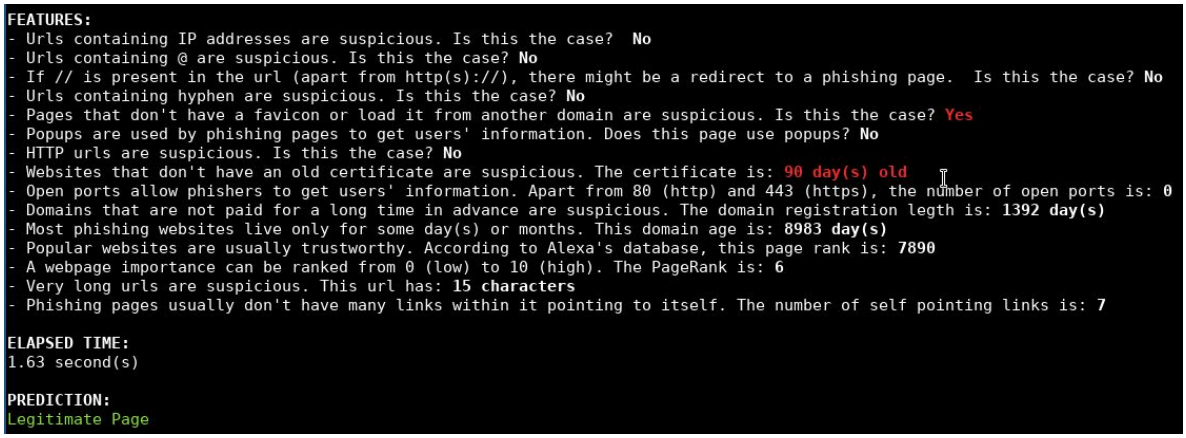

Figura 1: Resposta do classificador, ao receber URL legítima como entrada.

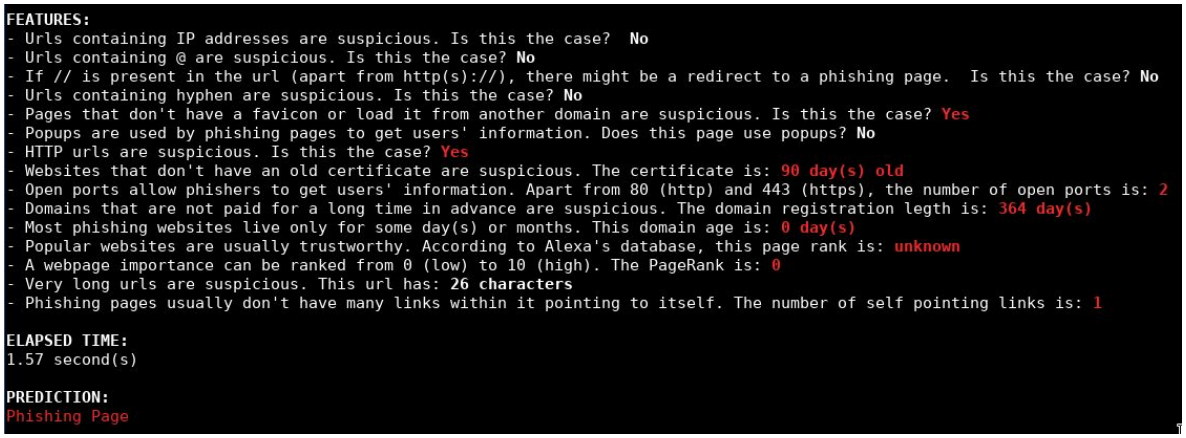

Figura 2: Resposta do classificador, ao receber URL *phishing* como entrada.

As Figuras 1 e 2 são exemplos da resposta do classificador, que exibe: o resultado de cada característica (em vermelho caso possua valor tipicamente *phishing),* o tempo utilizado para classificar a URL e a predição. Nota-se que no caso em que a predição é *phishing* (Figura 2), diversas características são exibidas em vermelho na saída.

O programa utilizada *threads* para calcular as características que requerem acesso à *internet*, como verificar a idade do domínio ou de seu certificado. O tempo despendido por essas é praticamente o tempo total da classificação da URL. Para uma banda larga com velocidade de *download* de 240 Mb/s, o tempo de classificação foi em média pouco maior que 1,5 segundo. Em momentos de baixa conectividade, no entanto, esse valor pode ser sensivelmente maior, o que revela a direta dependência de uma boa conexão para que a classificação seja rápida.

Por outro lado, a capacidade de processamento necessária para executar o classificador é sempre baixa. Para um computador Intel i5-4200M (2.5GHZ, 64-bit, 4 *cores*), utilizou-se apenas 0,1% da CPU. Quanto à memória, apenas 128 MB foram utilizados.

4. CONCLUSÕES

Neste projeto, desenvolveu-se e treinou-se um classificador de *phishing websites* capaz de classificar corretamente uma quantidade significativa de domínios, com acurácia e medida *F1* de 95%.

Com uma boa conexão à *Internet*, o tempo necessário para classificar uma URL é de cerca de 1,5 segundo. Assim, utilizando a ferramenta em conjunto com navegadores, por exemplo, seria possível evitar fraudes; haveria tempo suficiente para classificar as páginas acessadas por dado internauta e, em caso de *phishing,* alertá-lo antes que preenchesse campos com informações sensíveis.

Em geral, aplicações como programas antivírus, a fim de defender seus usuários, utilizam listas públicas de *URLs phishings* denunciadas por suas vítimas. A alternativa aqui estudada, ao alertar internautas sobre *sites* falsos acessados, poderia também denunciá-los automaticamente a essas listas. Evitaria-se, assim, a necessidade de vítimas para que domínios maliciosos fossem denunciados.

O projeto desenvolvido provou, portanto, a eficácia e validade de técnicas de inteligência artificial no campo da segurança da informação, em especial na prevenção de ataques *phishing*.

REFERÊNCIAS

[1] R. M. Mohammad, F. Thabtah. T. L. Mccluskey. (2013). Predicting phishing websites based on self-structuring neural network. Neural Computing and Applications, vol. 25, pp. 443–458.

[2] R. M. Mohammad, F. Thabtah. T. L. Mccluskey. (2015). An assessment of features related to phishing websites using an automated technique. 7th International Conference for Internet Technology and Secured Transactions, pp. 492-497.

[3] Von Neumann, J. (1928). Zur theorie der gesellschaftsspiele. Mathematische Annalen, vol. 100, pp 295–320.

[4] E. Kreyszig. (1979). Advanced Engineering Mathematics, 4a ed., Wiley, p. 880

[5] S. Russel, P. Norvig. (2020). Artificial Intelligence: A Modern Approach, 4a ed., Pearson, pp. 657, 684, 692, 697.

[6] B. H. Shekar, G. Dagnew. (2019). Grid search-based hyperparameter tuning and classification of microarray cancer data. Second International Conference on Advanced Computational and Communication Paradigms (ICACCP), pp. 1-8.

[7] P. A. Lachenbruch, M. R. Mickey. (1968). Estimation of error rates in discriminant analysis. Technometrics, vol. 10, pp. 1–12.

[8] Van Rijsbergen, C. J. (1979). Information Retrieval , 2a ed., Butterworth-Heinemann.

[9] Chinchor, N. (1992). MUC-4 evaluation metrics. Fourth Message Understanding Conference, pp. 22–29.

[10] G. Louppe, L. Wehenkel, A. Sutera, P. Geurts. (2013). Understanding variable importances in forests of randomized trees. 26th International Conference on Neural Information Processing Systems, vol. 1, pp. 431–439.

[11] Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. Journal of Educational Psychology, vol. 24, pp. 417–441.

[12] Sahingoz, O. K., Buber, E., Demir, O., Diri, B. (2019). Machine learning based phishing detection from URLs. Expert Systems with Applications, 117, 345-357.

[13] Benavides, E., Fuertes, W., Sanchez, S., Sanchez, M. (2020). Classification of Phishing Attack Solutions by Employing Deep Learning Techniques: A Systematic Literature Review. In Developments and Advances in Defense and Security, Springer, Singapore, pp. 51-64.