UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO"

FACULDADE DE CIÊNCIAS - CAMPUS BAURU

DEPARTAMENTO DE COMPUTAÇÃO

BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

MICHAEL HARUKI NAKATANI

AVALIAÇÃO DE CLASSIFICADORES PARA DETECÇÃO DE PHISHING EM E-MAILS

BAURU
2017

MICHAEL HARUKI NAKATANI

AVALIAÇÃO DE CLASSIFICADORES PARA DETECÇÃO DE PHISHING EM E-MAILS

Trabalho de Conclusão de Curso do Curso de Bacharelado em Ciência da Computação da Universidade Estadual Paulista “Júlio de Mesquita Filho”, Faculdade de Ciências, campus Bauru.

Orientadora: Profa. Dra. Simone das Graças Domingues Prado

Co-orientador: Prof. Dr. Kelton Augusto Pontara da Costa

BAURU

2017

Nakatani, Michael Haruki.

Avaliação de Classificadores para Detecção de Phishing em E-mails / Michael Haruki Nakatani, 2017.

41 p. : il.

Orientadora: Simone das Graças Domingues Prado

Co-orientador: Kelton Augusto Pontara da Costa

Monografia (graduação)-Universidade Estadual Paulista. Faculdade de Ciências, Bauru, 2016.

1. Phishing. 2. Engenharia Social. 3. Inteligência Artificial. 4. Aprendizado de Máquina. 5. Redes Neurais Artificiais. 6. Máquinas de Vetor de Suporte. 7. Árvores de Decisão. 8. Adaptive Boosting. 9. Florestas Aleatórias. I. Universidade Estadual Paulista. Faculdade de Ciências. II. Título.

MICHAEL HARUKI NAKATANI

AVALIAÇÃO DE CLASSIFICADORES PARA DETECÇÃO DE PHISHING EM E-MAILS

Trabalho de Conclusão de Curso do Curso de Bacharelado em Ciência da Computação da Universidade Estadual Paulista "Júlio de Mesquita Filho", Faculdade de Ciências, Campus Bauru.

Banca Examinadora

Prof^{a} Dr^{a} Simone das Graças Domingues Prado
Orientadora
Universidade Estadual Paulista "Júlio de Mesquita Filho"
Faculdade de Ciências
Departamento de Computação

Prof^{a} Dr^{a}. Márcia Aparecida Zanoli Meira e Silva
Universidade Estadual Paulista "Júlio de Mesquita Filho"
Faculdade de Ciências
Departamento de Computação

Prof. Dr. Kleber Rocha de Oliveira
Universidade Estadual Paulista "Júlio de Mesquita Filho"
Faculdade de Ciências
Departamento de Computação

Bauru, 08 de Dezembro de 2017.

Resumo

O *phishing* é uma prática criminosa que utiliza Engenharia Social e faz vítimas pelo mundo todo, especialmente no Brasil causando prejuízos para as pessoas e para as organizações. No entanto, é possível desenvolver sistemas que possam identificar e combater estes ataques utilizando classificadores para reconhecer padrões em mensagens conhecidas através do aprendizado de máquina supervisionado. Neste trabalho foram comparados cinco classificadores utilizando critérios como a medida F1, ou *F-measure,* e as curvas ROC (*Receiver Operating Characteristics*). Dentre os classificadores testados, o Floresta Aleatória (*Random Forest* - RF) obteve maior média de Área Abaixo da Curva ROC (*Area Under* ROC *Curve* - AUC) com AUC = 0,9973 e as Redes Neurais Artificiais (*Artificial Neural Networks* - ANN) obtiveram a maior média de F1 com F1 = 0,9976.

Palavras-chave: Phishing; Engenharia Social; Inteligência Artificial; Aprendizado de Máquina; Redes Neurais Artificiais, Máquinas de Vetor de Suporte; Árvores de Decisão; Adaptive Boosting; Florestas Aleatórias

Abstract

Phishing is a criminal that practice that uses Social Engineering and makes victims all around the world, specially in Brazil causing damage to people and to the organizations. However, it is possible to develop systems that are able to identify and fight against these attacks using classifiers to pattern recognizing on known messages through supervised machine learning. In this work, five classifiers were compared using criterions like F-measure and Receiver Operating Characteristics (ROC) curves. Among the tested classifiers, Random Forest (RF) got the highest Area Under the ROC Curve (AUC) average with AUC = 0,9973 and Artificial Neural Networks (ANN) got the highest F-measure average with F-measure = 0,9976.

Key words: Phishing; Social Engineering; Artificial Intelligence; Machine Learning; Artificial Neural Networks, Support Vector Machines; Decision Trees; Adaptive Boosting; Random Forest

Lista de Ilustrações

Lista de Quadros

Lista de Tabelas

Sumário

1 INTRODUÇÃO

O rápido avanço da Internet e da Tecnologia da Informação (TI) tem trazido muitas vantagens e um impacto positivo para as vidas das pessoas. Porém, com isso houve um crescimento nos cibercrimes. Assim com o avanço da tecnologia, o número de cibercrimes e sua variedade aumentaram concomitantemente (DILEK; ÇAKIR; AYDIN, 2015).

O *phishing*, uma das atividades criminosas mais lucrativas da Internet, é uma fraude eletrônica com o objetivo de obter informações confidenciais de usuários. Bilhões de dólares foram perdidos por empresas e indivíduos devido a ataques de *phishing* (AKINYELU; ADEWUMI, 2014; HENKE et al., 2014; OLIVO; SANTIN; OLIVEIRA, 2015).

Devido ao aumento dos usuários de Internet nos últimos anos e o uso do *e-mail* tanto para tarefas pessoais como corporativas, torna-se necessário pesquisar técnicas computacionais para detecção de ameaças, entre elas o *phishing*, uma vez que a maioria destes novos usuários não possuem conhecimento tecnológico ou são inexperientes o que os torna vulneráveis.

Dentre os métodos de detecção de *phishing*, existem as técnicas baseadas em assinatura, como, por exemplo, o *Domain Keys Identified Mail* (DKIM) que é uma especificação da *Internet Engineering Task Force* (IETF) que define um mecanismo para autenticação de *e-mail* baseado em chaves públicas (OLIVO; SANTIN; OLIVEIRA, 2015).

Outra técnica de detecção é medir a frequência de palavras-chave em um *e-mail* e para aprendizagem podem ser utilizados classificadores como as Máquinas de Vetor de Suporte (*Support Vector Machines* - SVM) ou as Redes Neurais Artificiais (Artificial Neural Networks - ANN) (OLIVO, 2010).

1.1 Problema

Phishing é um mecanismo criminoso que emprega tanto Engenharia Social como Subterfúgio Técnico para roubo de identidade e de dados financeiros e é difundido através de falsos e-mails que direcionam para websites fraudulentos para roubo de dados (APWG[1], 2017).

A quantidade de ataques de phishing em 2016 foi de 1.220.523, 65% maior que em 2015 (APWG, 2017) e o surgimento de novos *sites* de *phishing* entre esses anos mostrou um aumento de mais de 90%, conforme mostra o quadro 1.

Quadro 1 - Sites de phishing detectados por mês 2015-2016

	2015	2016	% de aumento
janeiro	68.185	86.557	26,94%
fevereiro	55.869	79.259	41,87%
março	65.530	123.555	88,55%
abril	64.328	158.988	147,15%
maio	72.709	148.295	103,96%
junho	41.852	158.782	279,39%
julho	64.275	155.102	141,31%
agosto	87.801	104.349	18,85%
setembro	56.196	104.973	86,80%
outubro	48.114	89.232	85,46%
novembro	44.575	118.928	166,80%
dezembro	65.885	69.533	5,54%
Total	735.319	1.397.553	90,06%

Fonte: adaptado de APWG (2017)

1.2 Justificativa

Em 2016, o Brasil sofreu o maior número de ataques de *phishing* correspondendo a 27,61% do total global (SECURELIST, 2017) e cerca de 67,5%

[1] Anti-Phishing Working Group

dos mais de 206 milhões de habitantes brasileiros possuem acesso à Internet (IWS[2], 2017). Com este trabalho, procurou-se estudar formas de se evitar danos às pessoas ou às organizações decorrente da prática do *phishing* por cibercriminosos e pesquisar métodos de combate a esta ameaça.

1.3 Objetivos

Abaixo, são descritos o objetivo geral e os objetivos específicos.

1.3.1 Objetivo Geral

O objetivo deste trabalho foi aplicar os conhecimentos adquiridos ao longo do curso para desenvolver um sistema baseado em computador viável, que pudesse ser implementado e testado até o prazo final, para detecção de *phishing* em *e-mails* utilizando técnicas inteligentes como as ANN usando uma base de dados de *phishing* conhecida e avaliar sua eficácia.

1.3.2 Objetivos Específicos

Dentre os objetivos específicos, seguem:

a) Definir os algoritmos para detecção de *phishing*

b) Definir quais classificadores serão testados

c) Definir qual base de dados de *phishing* será mais adequada

d) Avaliar a eficácia dos classificadores para detecção de *phishing*

[2] Internet World Stats

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, são apresentados os conceitos estudados para o desenvolvimento deste trabalho como engenharia social, o aprendizado supervisionado em aprendizado de máquina e as curvas ROC para avaliação de classificadores.

2.1 Engenharia Social

A Engenharia Social é a ciência que estuda como o conhecimento do comportamento humano pode ser usado para manipulação de pessoas (PEIXOTO, 2006 apud KLETTENBERG, 2016) e passa muitas vezes despercebida pelas pessoas, pois as vítimas adquirem confiança no agressor (ROSA et al., 2011 apud KLETTENBERG, 2016).

É, também, utilizada como um método de ataque virtual no qual se aproveita da confiança e da ingenuidade do usuário para obter informações que permitam invadir um sistema e comprometer sistemas de informação (KROMBHOLZ et al., 2014; SILVA, 2011)

Pode-se listar algumas técnicas de fraude utilizando Engenharia Social, como Furto de Identidade, Fraude por Antecipação de Recursos, Boato e *Phishing*.

Furto de Identidade

É o ato pelo qual uma pessoa tenta se passar por outra com o objetivo de obter vantagens indevidas, como o acesso à conta do banco. Com o crescente aumento da disponibilização de informações pessoais nas redes sociais, os cibercriminosos passam a ter mais recursos para roubar a identidade alheia (CERT[3], 2012).

[3]Centro de Estudos para Resposta e Tratamento de Incidentes em Computadores

Fraude por Antecipação de Recursos

Neste tipo de fraude, o golpista induz a pessoa a fornecer informações confidenciais ou realizar um pagamento adiantado, com a promessa de futuramente receber algum benefício. Após fornecer os recursos solicitados, a pessoa percebe que o tal benefício prometido não existe e que foi vítima de um golpe (CERT, 2012).

Um dos golpes mais famosos deste tipo no Brasil é o golpe do bilhete premiado em que o golpista, basicamente, convence a vítima de que não pode resgatar o prêmio por algum motivo, e induz a vítima a lhe dar uma alta quantia em dinheiro em troca do falso prêmio do bilhete.

Boato

Também conhecido como *hoax*, é uma mensagem com conteúdo alarmante ou falso e que, geralmente, tem como remetente, ou aponta como autor, alguma instituição, empresa importante ou órgão governamental. Ao se analisar detalhadamente o texto, pode-se identificar, na maioria das vezes, inconsistências, tentativas de golpe e execução de códigos maliciosos (CERT, 2012).

2.1.1 Phishing

Conforme dito anteriormente, *phishing* é um mecanismo criminoso que emprega tanto Engenharia Social como Subterfúgio Técnico para roubo de identidade e de dados financeiros. Pode ser difundido através de falsos *e-mails* que direcionam para *websites* fraudulentos para roubo de dados. Em 2016, a quantidade de ataques de *phishing* foi de 1.220.523, 65% maior que em 2015 (APWG, 2017) e o Brasil sofreu o maior número de ataques de phishing correspondendo a 27,61% do total global (SECURELIST, 2017).

O golpe de phishing também contém algumas variações como o *Pharming* e o *Spear Phishing.*

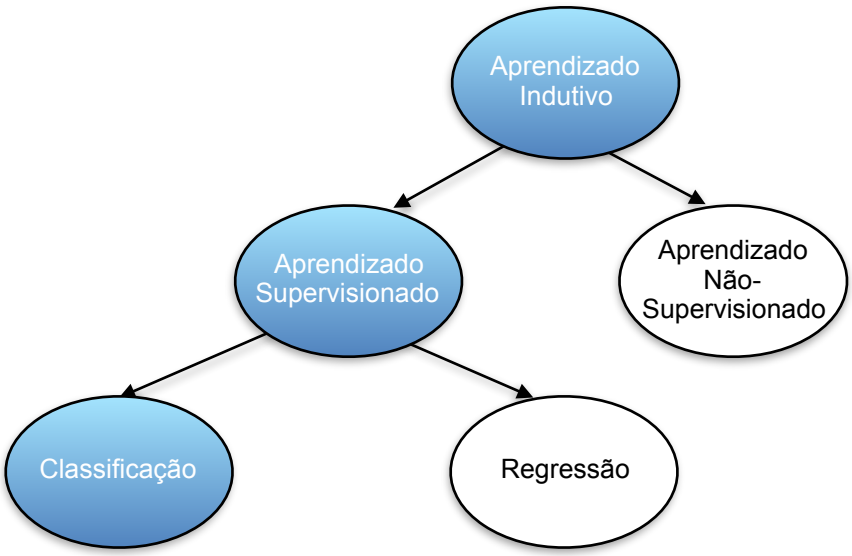
O *Pharming* consiste no redirecionamento da navegação do usuário para

websites falsos, por meio de alterações no Sistema de Nomes no Domínio (*Domain Name Services* - DNS), ou seja, ao tentar acessar um *site* legítimo, o usuário é redirecionado, de forma transparente, para uma página falsa (CERT, 2012). Já o *Spear Phishing* consiste em induzir uma vítima especificamente escolhida a abrir um anexo malicioso ou visitar um *website* malicioso com o intuito de descobrir dados confidenciais ou agir com objetivos nefastos contra a organização da vítima (RSA[4], 2016).

2.2 Aprendizado Supervisionado

O aprendizado supervisionado depende da disponibilidade de uma base de dados de treinamento com exemplos rotulados, sendo que cada exemplo deve conter os dados de entrada e a resposta esperada (HAYKIN, 2009) e a ele é associado um algoritmo de aprendizado, ou indutor, cujo objetivo é construir um classificador que possa determinar a classe de novos exemplos ainda não rotulados. Para rótulos de classe discretos, esse problema é conhecido como classificação e para valores contínuos como regressão (REZENDE, 2003, p. 40). A figura 1 mostra a hierarquia do aprendizado.

Figura 1 - Hierarquia do Aprendizado



Fonte: Rezende (2003, p.41)

[4] Nomeado segundo as iniciais de seu co-fundadores Ron Rivest, Adi Shamir e Leonard Adleman.

Ao induzir, a partir de exemplos disponíveis, é possível que o classificador seja extremamente eficaz para o conjunto de treinamento, mas tenha uma desempenho ruim para o conjunto de teste, sendo incapaz de classificar eficazmente dados não rotulados. Esse problema é conhecido como *overfitting*. Por outro lado, o *underfitting* acontece quando poucos exemplos são dados ao sistema de aprendizado ou o tamanho pré-definido do classificador é muito pequeno ou uma combinação de ambos (REZENDE, 2003, p. 46-47).

2.2.1 Redes Neurais Artificiais

Os estudos com as Redes Neurais Artificiais, ou somente Redes Neurais (Neural Networks - NN), foram motivados inicialmente pelo fato de o cérebro humano trabalhar de um jeito completamente diferente de um computador convencional. O cérebro é um computador altamente complexo, não-linear e que processa informações paralelamente (HAYKIN, 2009).

De forma geral, uma NN é uma máquina, com processamento distribuído massivamente paralelo, projetada para modelar a forma como o cérebro resolverá um problema ou uma função de interesse, empregando interconexões massivas de simples células computacionais conhecidas como unidades de processamento ou neurônios e adquirindo conhecimento através do processo de aprendizado (HAYKIN, 2009).

Uma NN se assimila ao cérebro humano no processo de aprendizado adquirido do ambiente e na atribuição de pesos sinápticos para cada informação adquirida. Dentre suas funcionalidades estão a sua capacidade de trabalhar tanto linear como não-linearmente, a sua adaptabilidade, a sua contextualização de informação, a sua tolerância a falhas e o seu mapeamento de entrada e saída (HAYKIN, 2009).

2.2.2 Máquinas de Vetor de Suporte

Uma SVM é uma máquina de aprendizado para classificação de problemas com saída binária mapeando não-linearmente vetores de entrada para construir uma superfície linear de decisão (CORTES; VAPNIK, 1995). Dada uma amostra para treinamento, a SVM constrói um hiperplano como a superfície de decisão de forma que a distinção entre exemplos positivos e negativos seja maximizada (HAYKIN, 2009).

2.2.3 Árvores de Decisão

Basicamente, as Árvores de Decisão (*Decision Trees*) quebram um problema de decisão complexo em um conjunto de decisões mais simples com soluções, geralmente, mais fáceis de se obter com a expectativa de que solução final obtida com este método se assemelhe à solução esperada (SAFAVIAN; LANDGREBE; 1990).

2.2.4 Adaptive Boosting

Adaptive Boosting (AdaBoost) é um algoritmo de aprendizado que, teoricamente, pode ser usado para reduzir drasticamente o erro de qualquer classificador cujo desempenho seja um pouco melhor do que o aleatório (FREUND; SCHAPIRE, 1996). O AdaBoost aprende um algoritmo forte através da combinação de algoritmos fracos e um conjunto de pesos que são aprendidos através de aprendizado supervisionado. Cada algoritmo fraco só é necessário para fazer as detecções corretas um pouco mais de metade do tempo (MIYAMOTO; HAZEYAMA; KADOBAYASHI, 2008).

2.2.5 Florestas Aleatórias

Florestas Aleatórias (*Random Forest* - RF) são:

> Uma combinação de preditores de árvores tal que cada árvore depende dos valores de um vetor aleatório independentemente amostrado e com a mesma distribuição para todas as árvores da floresta. O erro de generalização das florestas converge para um limite conforme se torna grande o número de árvores na floresta (BREIMAN, 2001, **tradução nossa**).

Cada árvore de decisão é construída baseando-se em um subconjunto aleatório da base de dados de treinamento, então um subconjunto aleatório das variáveis disponíveis é usado para escolher como melhor particionar a base de dados em cada nó (MIYAMOTO; HAZEYAMA; KADOBAYASHI, 2008).

2.3 Matriz de Confusão

A matriz de confusão de uma hipótese mostra o número de classificações corretas *versus* o número de classificações preditas para cada classe sobre um conjunto de exemplos. Em sua diagonal principal, se localizam as quantidades de acertos para cada classe. A tabela 1 mostra uma matriz de confusão para *k* classes diferentes (REZENDE, 2003, p. 47-48).

Tabela 1 - Matriz de Confusão para k classes

Classe	predita C_1	predita C_2	$\cdots$	predita C_k
verdadeira C_1	$M(C_1, C_1)$	$M(C_1, C_2)$	$\cdots$	$M(C_1, C_k)$
verdadeira C_2	$M(C_2, C_1)$	$M(C_2, C_2)$	$\cdots$	$M(C_2, C_k)$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
verdadeira C_k	$M(C_k, C_1)$	$M(C_k, C_2)$	$\cdots$	$M(C_k, C_k)$

Fonte: Rezende (2003, p.41)

Para uma classificação em rótulos positivos ou negativos, utiliza-se uma matriz de confusão de duas classes, ou seja, com saída binária, mostrando exemplos verdadeiros positivos (TP), falsos positivos (FP), verdadeiros negativos

(TN) e falsos negativos (FN) e o total de positivos (P) e o total negativos (N), conforme mostra a figura 2. Alguma medidas podem ser calculadas a partir desta matriz de confusão como, por exemplo, a precisão $p = \frac{TP}{TP + FP}$, o *recall* $r = \frac{TP}{P}$, a acurácia = $\frac{TP + TN}{P + N}$ e a medida F que pode ser calculada com $F = \frac{2.p.r}{p + r}$ (FAWCETT, 2005).

Figura 2 - Matriz de Confusão para duas classes

		Classe Verdadeira	
		p	**n**
Classe Predita	**S**	Verdadeiros Positivos	Falsos Positivos
	N	Falsos Negativos	Verdadeiros Negativos
Totais das Colunas		**P**	**N**

Fonte: Fawcett (2005)

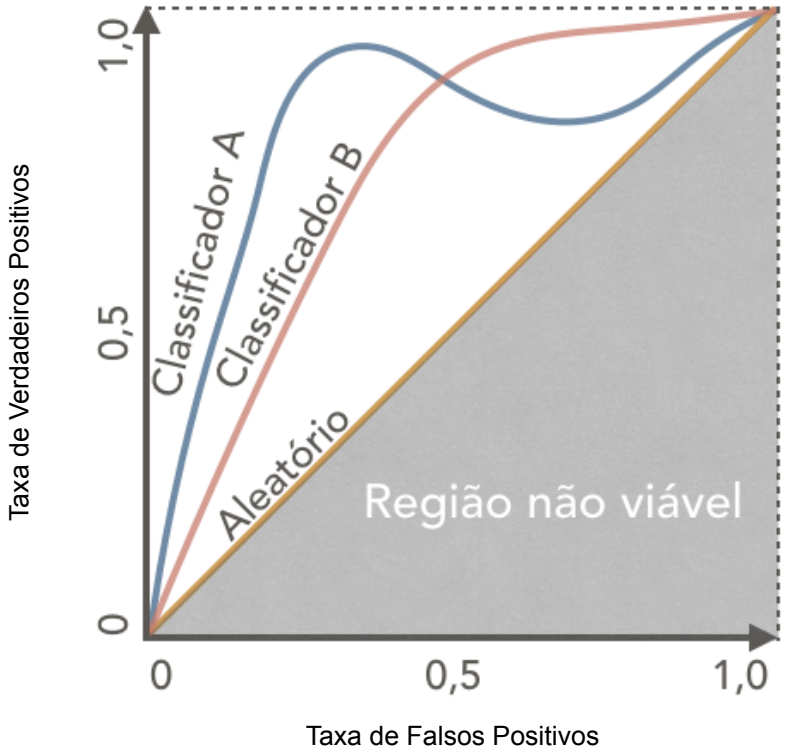
2.4 Curvas ROC

Um gráfico *Receiver Operating Characteristics* (ROC) é uma técnica para visualizar, organizar e selecionar classificadores de acordo com o desempenho de cada um, considerando problemas de classificação para apenas duas classes. Um gráfico ROC possui duas dimensões de forma que a taxa de verdadeiros positivos (TPR) é plotada no eixo Y e a taxa de falsos positivos (FPR) é plotada no eixo X, descrevendo a relação entre custo (TPR) e benefício (FPR) (FAWCETT, 2005).

A figura 3 mostra dois classificadores, a reta diagonal que representa a classificação aleatória e abaixo dela a região de classificadores ruins cujo

desempenho é pior do que o aleatório. Um bom classificador deve estar sempre acima da reta diagonal e quanto mais se aproxima de noroeste, melhor é o seu desempenho (FAWCETT, 2005).

Figura 3 - Curvas ROC



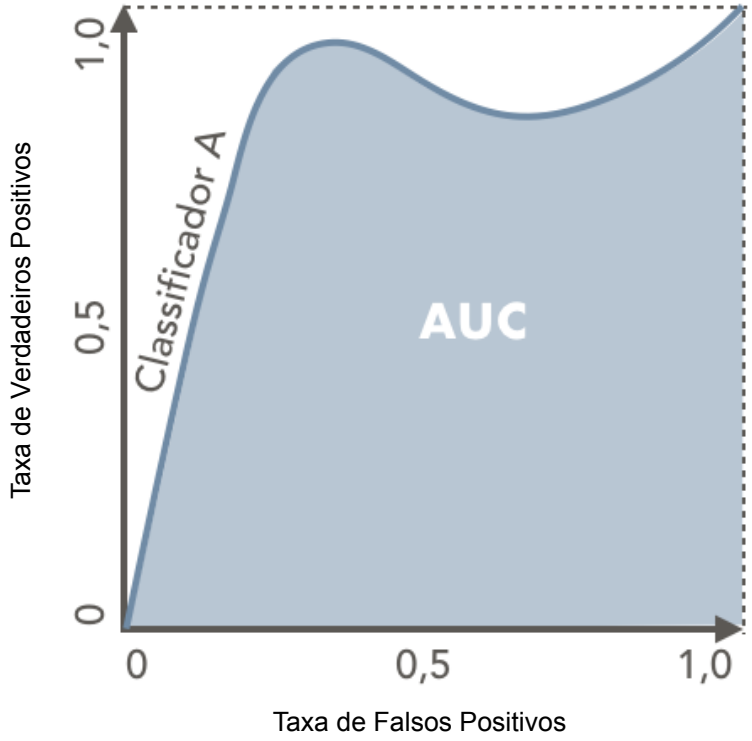
Fonte: elaborado pelo Autor com base em Fawcett (2005)

2.4.1 AUC

A Área Abaixo da Curva ROC (Area Under ROC Curve - AUC) é um número escalar, numericamente equivalente à estatística de Wilconxon, que representa a performance esperada de um classificador. Uma vez que AUC é uma porção da área de um quadrado com lado unitário, seu valor estará compreendido sempre entre 0 e 1,0, conforme mostra a figura 4. No entanto, como a reta diagonal entre os pontos (0,0) e (1,1) que representa a classificação aleatória possui uma área de 0,5, nenhum classificador realístico deve possuir uma AUC menor do que 0,5 (FAWCETT, 2005).

A AUC, também, pode ser interpretada como a capacidade do classificador se adaptar podendo adotar uma estratégia mais conservadora que faz classificações positivas apenas com fortes evidências diminuindo a FPR e, consequentemente a TPR, ou uma estratégia mais liberal em que as classificações são feitas sem a necessidade de fortes evidências aumentando a TPR e, consequentemente a FPR (MIYAMOTO; HAZEYAMA; KADOBAYASHI, 2008; FAWCETT, 2005).

Figura 4 - AUC



Fonte: elaborado pelo Autor com base em Fawcett (2005)

2.5 Revisão Bibliográfica

O quadro 2 mostra trabalhos de detecção de *phishing* com destaque para o PILFER de Fette, Saleh e Tomasic (2007) que extrai as características da mensagem de *e-mail* e utiliza o SVM como classificador para detecção de *phishing*, o CANTINA+ de Xiang et al. (2011) que utiliza SVM, Regressão Logística (Logistic Regression - LR), Redes Bayesianas, Random Forest (RF) e AdaBoost como classificadores para detecção de *phishing* em *websites*, o PhishGILLNET de

Ramanathan e Wechsler (2012) que utiliza Análise Probabilística de Semântica Latente (Probabilistic Latent Semantic Analysis - PLSA) para análise do conteúdo da mensagem de *e*-mail e SVM e AdaBoost como classificadores para detecção de *phishing* e o PhishAri de Aggarwal, Rajadesingan e Kumaraguru (2012) que utilizam Naive Bayes, árvores de decisão e RF como classificadores para detecção de *phishing* no Twitter.

Miyamoto, Hazeyama e Kadobayashi (2008) compararam desempenhos de diversos classificadores para detecção de *phishing* em *websites*, entre eles, as NN, a SVM, o AdaBoost, o *Bagging,* RF e outros. Chegaram à conclusão de que o AdaBoost apresentava o melhor desempenho com F1 igual a 0,8581 e AUC igual 0,9342.

Quadro 2 - Ferramentas para detecção de phishing

Solução	Autores	Ano	Mídia	Classificadores
“Approximate String”	Abraham; Raj	2014	URL	Heurística Customizada
PhishTackle	Ramesh; Krishnamurthi	2014	Website	SVM, Heurística Customizada
“Distributed Software Agents”	Sarika; Varghese	2013	Website	Sistema multi-agentes
PhishCage	Miyamoto et al.	2013	Website	Heurística Customizada
PhishAri	Aggarwal; Rajadesingan; Kumaraguru	2012	Twitter	Naive Bayes, árvores de decisão, RF
PhishGILLNET	Ramanathan; Wechsler	2012	E-mail	SVM, AdaBoost
CANTINA+	Xiang et al.	2011	Website	SVM, LR, Redes Bayesianas, J48, RF, Adaboost
PhishZoo	Afroz; Greenstadt	2011	Website	Heurística Customizada
“Genetic Algorithm”	Sheeram et al.	2010	Website	Algoritmos Evolutivos
GoldPhish	Dunlop; Groat; Shelly	2010	Website	Heurística Customizada
PhishNet	Prakash et al.	2010	URL	Heurística Customizada
PhishCatch	Yu; Nargundkar; Tiruthani	2009	E-mail	Heurística Customizada
Phishpin	Tout; Hafner	2009	Website	Heurística Customizada
Phishwish	Cook; Gurbani; Daniluk	2009	E-mail	Heurística Customizada
CANTINA	Zhang; Hong; Cranor	2007	Website	Heurística Customizada
PILFER	Fette; Saleh; Tomasic	2007	E-mail	SVM

Fonte: adaptado de Falk (2016)

3 DESENVOLVIMENTO

Para o desenvolvimento deste trabalho, foram pesquisadas bases de dados ou *datasets* com exemplos de ataques de *phishing* em *e-mails* e exemplos de *e-mails* legítimos e, então extraiu-se os preditores para criação da base de dados para ser usada pelos classificadores para rotulação de *e-mails* legítimos ou *phishing*.

3.1 Método de Pesquisa

Para o desenvolvimento deste trabalho, decidiu-se utilizar a linguagem de programação Matlab versão *Student*, cuja principal diferença, além do preço viável, em relação à versão completa é a desvantagem de não poder gerar uma versão *standalone* do programa. Também foram instalados os *toolboxes Statistics and Machine Learning* e *Neural Networks* para lidar com as tarefas de aprendizado de máquina.

Para o processamento dos *e-mails*, decidiu-se utilizar a linguagem de programação Perl que é conhecida pela facilidade e agilidade no trabalho com expressões regulares e processamento de textos.

3.2 Características Utilizadas para Classificação de E-mail

Para classificar um *e-mail* em *phishing* ou legítimo, foi utilizada a abordagem de Fette, Sadeh e Tomasic (2007) de análise das características da mensagem como o Localizador Padrão de Recursos (*Uniform Resource Locator* - URL) e o Protocolo de Internet (*Internet Protocol* - IP).

URL Baseado em IP

Alguns ataques de *phishing* são hospedados em computadores comprometidos que provavelmente não possuem entrada de DNS e a maneira mais fácil de acessá-la é através do endereço IP e uma vez que as organizações

raramente usam *links* com endereço IP, este atributo é um forte indicativo de ataque de *phishing*. Este atributo é binário (FETTE; SADEH; TOMASIC, 2007).

URLs Não Correspondentes

A utilização de HTML em uma mensagem de e-*mail* é muito explorada para ataques de *phishing,* por ser possível exibir um *link* legítimo que na verdade redireciona para um *link* malicioso. Este atributo é binário (FETTE; SADEH; TOMASIC, 2007).

Contém HTML

A maioria dos *e-mails* são apenas texto, ou *plain text*, Hypertext Markup Language (HTML) ou uma combinação dos dois. O fato de um e-mail conter HTML não é decisivo para classificá-lo como *phishing* ou legítimo. No entanto, lançar um ataque de *phishing* sem usar HTML é muito difícil, porque a maioria dos ataques técnicos e enganosos não são possíveis se a mensagem contiver apenas texto. Por outro lado, o usuário ainda pode ser ludibriado por nomes de domínio que parecem legítimos. Este atributo é binário (FETTE; SADEH; TOMASIC, 2007).

Quantidade de Links

A quantidade de *links* presente no *e-mail* é um atributo contínuo.

Quantidade de Domínios na URL

Para todas as URLs que começam com *http://* ou *https://* são extraídos os nomes de domínios e são contados distintamente. Este atributo é contínuo (FETTE; SADEH; TOMASIC, 2007).

Quantidade de Pontos na URL

Um dos métodos do agressor construir URLs que parecem legítimas é utilizando sub-domínios como, por exemplo, *http://www.my-bank.update.data.com* ou

utilizando um *script* de redirecionamento como, por exemplo, *http://www.google.com/url?q=http://www.badsite.com* que pode parecer para o usuário ou um filtro ingênuo que este *site* está hospedado em *google.com*, mas, na realidade vai redirecionar seu navegador para *badsite.com.* Seja por incluir uma URL em um *script* de redirecionamento ou por usar uma maior quantidade de sub-domínios, há uma grande quantidade de pontos. Este atributo é simplesmente o máximo número de pontos contidos em qualquer dos *links* presentes no *e-mail* e é contínuo (FETTE; SADEH; TOMASIC, 2007).

Contém Javascript

O agressor pode lançar ataques sofisticados com o uso de Javascript, uma vez que este pode ser usado para muitas coisas desde para criar janelas *popup* até mudar a barra de *status* de um navegador ou cliente de *e-mail*, aparecendo diretamente no *e-mail* ou ser embarcado em um *link*. Este é um atributo binário (FETTE; SADEH; TOMASIC, 2007).

Contém Formulários

Os formulários podem ser utilizados pelo agressor para roubar dados de cartão de crédito e senhas. Este atributo é binário (ZHANG; HONG; CRANOR, 2007).

URL Contém Arroba

O símbolo de arroba (@) em um URL faz com que a *string* à esquerda seja desconsiderada e a *string* à direita seja tratada como o URL de fato para requisitar um *site*. Combinando com a limitação de tamanho da barra de endereço do navegador, torna-se possível escrever URLs que parecem legítimos, mas direcionam para um *site* diferente. Este é um atributo binário (ZHANG; HONG; CRANOR, 2007).

URL Contém Traço

Segundo Zhang, Hong e Cranor (2007), URLs legítimos raramente utilizam o símbolo de traço (-). Este atributo é binário.

3.3 Base de dados

Para compor a base de dados, foi utilizada uma amostra de 2.283 *e-mails* de *phishing* disponível pelo *site monkey.org*[5] coletados entre os anos de 2005 e 2007 e uma amostra de 2.500 *e-mails* legítimos do *site spamassassin.com*[6] coletados no ano de 2002, ambos disponíveis publicamente.

Para tratar o arquivo com extensão *mbox* da base de dados de *phishing*, foi utilizado o programa *Formail*, disponível para ambiente Unix, para separar cada *e-mail* em um respectivo arquivo.

Para gerar a base de dados, foi escrito um *script* em Perl utilizando expressões regulares para analisar as características de cada *e-mail* legítimo e de *phishing* e extrair o valor de cada preditor e associá-lo um rótulo, conforme mostra o quadro 3.

3.4 Classificação

Foram escolhidos cinco classificadores (NN, SVM, Trees, AdaBoost e RF) conforme utilizados pelos autores Fette, Sadeh e Tomasic (2007), Miyamoto, Hazeyama e Kadobayashi (2008), Ramanathan e Wechsler (2012), Aggarwal, Rajadesingan e Kumaraguru (2012) e Akinyelu e Adewumi (2014) para comparação de desempenho na detecção de *phishing* em *e-mails* com a base de dados construída a partir de casos conhecidos de ataques de *phishing* e *e-mails* legítimos.

---

[5] http://monkey.org/~jose/phishing/phishing3.mbox

[6] http://spamassassin.apache.org/old/publiccorpus/obsolete/

Quadro 3 - Amostra da Base de Dados

IP Based URL	Non Matching URL	Has HTML	Number of Links	Number of Domains	Number of Dots	Has Java Script	Has Forms	URL Has At	URL Has Dash	Is Phishing
1	1	1	1	1	3	0	0	0	0	1
0	1	1	1	1	2	0	0	0	0	1
0	1	1	1	2	3	0	0	0	0	1
1	1	1	8	2	3	0	0	0	0	1
1	1	1	8	2	3	0	0	0	0	1
0	1	1	2	1	6	0	0	0	0	1
0	1	1	1	3	4	0	0	0	0	1
0	1	1	3	1	3	0	0	0	0	1
0	1	1	1	2	3	1	0	0	0	1
1	0	1	2	1	6	0	0	1	0	1
0	0	0	4	3	3	0	0	0	0	0
0	0	0	2	3	3	0	0	0	0	0
0	0	0	2	3	3	0	0	0	0	0
0	0	0	5	1	3	0	0	0	0	0
0	0	0	3	3	4	0	0	0	0	0
0	0	0	2	3	3	0	0	0	0	0
0	0	0	2	3	3	0	0	0	0	0
0	0	0	4	3	3	0	0	0	0	0
0	0	1	3	2	3	0	0	0	0	0
0	0	1	6	2	3	0	0	0	0	0

Fonte: elaborado pelo Autor

3.5 Experimentos

A partir dos 4.783 registros da base de dados, foram separados 30% para o conjunto de teste e 70% para o conjunto de treinamento dos quais obrigatoriamente 20% são amostras de *phishing* e os 50% restantes são amostras de *e-mails* legítimos. Para cada classificador, foram feitas 10 rodadas de testes com novos registros da base de dados sendo sorteadas a cada rodada.

Para realização dos testes, foi utilizado um *Macbook Air* com processador Intel Core i5 de 1,3GHz, 8GB de Memória RAM e Sistema Operacional macOS Sierra v.10.12.6.

No geral, foi usada a configuração padrão do Matlab com algumas alterações. Para treinamento da NN, utilizou-se apenas uma camada escondida. Foram utilizados 100 aprendizes para os classificadores de aprendizado em conjunto AdaBoost e RF. Para as RF, o número de preditores que foi usado é o padrão que é igual à raiz quadrada do total de preditores que é quatro. O quadro 4 mostra mais detalhes.

Quadro 4 - Códigos Utilizados para Treinamento

Classificador	Código
NN	`[net,tr] = train(net,numData,target);`
SVM	`mdl = fitcsvm(phishTrain,'IsPhishing');`
Tree	`mdl = fitctree(phishTrain,'IsPhishing');`
AdaBoost	`mdl = fitensemble(phishTrain,'IsPhishing','AdaBoostM1', 100,'Tree');`
RF	`mdl = TreeBagger(100,phishTrain,'IsPhishing');`

Fonte: elaborado pelo Autor

As tabelas de 2 a 6 mostram os resultados observados e calculados para os classificadores NN, SVM, Tree, AdaBoost e RF, respectivamente.

Tabela 2 - Rodadas de Teste para NN

R	Tempo de Treino (s)	Acurácia em Treino (%)	Acurácia em Teste (%)	TP	FP	P	TN	FN	N	Precisão	Recall	F1	TPR (%)	TNR (%)	AUC
1	3,6319	98,1	99	1326	6	1332	93	9	102	0,9955	0,9955	0,9955	99,5495	91,1765	0,9818
2	3,7112	98	99,1	1325	3	1328	96	10	106	0,9977	0,9977	0,9977	99,7741	90,5660	0,9896
3	5,3826	97,9	99,2	1332	5	1337	90	7	97	0,9963	0,9963	0,9963	99,6260	92,7835	0,9869
4	3,1554	98	99,2	1326	3	1329	96	9	105	0,9977	0,9977	0,9977	99,7743	91,4286	0,9919
5	4,2974	98	99	1319	0	1319	100	15	115	1,0000	1,0000	1,0000	100,0000	86,9565	0,9957
6	5,5544	98	99,1	1327	2	1329	94	11	105	0,9985	0,9985	0,9985	99,8495	89,5238	0,9870
7	4,6096	98,1	99	1319	4	1323	100	11	111	0,9970	0,9970	0,9970	99,6977	90,0901	0,9753
8	0,9961	97,2	99,3	1332	3	1335	92	7	99	0,9978	0,9978	0,9978	99,7753	92,9293	0,9760
9	5,6782	98	99,2	1326	2	1328	96	10	106	0,9985	0,9985	0,9985	99,8494	90,5660	0,9931
10	4,2009	98	99	1324	4	1328	96	10	106	0,9970	0,9970	0,9970	99,6988	90,5660	0,9763

Fonte: elaborado pelo Autor

Tabela 3- Rodadas de Teste para SVM

R	Tempo de Treino (s)	Acurácia em Treino (%)	Acurácia em Teste (%)	TP	FP	P	TN	FN	N	Precisão	Recall	F1	TPR (%)	TNR (%)	AUC
1	0,2325	97,9994	96,4798	1324	10	1334	95	5	100	0,9925	0,9925	0,9925	99,2504	95,0000	0,9796
2	0,2856	97,9098	99,0546	1324	11	1335	98	1	99	0,9918	0,9918	0,9918	99,1760	98,9899	0,9893
3	0,2241	97,9098	99,0414	1326	11	1337	96	1	97	0,9918	0,9918	0,9918	99,1773	98,9691	0,9886
4	0,2098	97,7904	99,7909	1325	8	1333	101	0	101	0,9940	0,9940	0,9940	99,3998	100,0000	0,9976
5	0,2549	97,9098	97,0420	1331	8	1339	91	4	95	0,9940	0,9940	0,9940	99,4025	95,7895	0,9590
6	0,2316	97,9994	95,5953	1331	9	1340	88	6	94	0,9933	0,9933	0,9933	99,3284	93,6170	0,9681
7	0,2725	97,9994	96,4111	1326	10	1336	93	5	98	0,9925	0,9925	0,9925	99,2515	94,8980	0,9696
8	0,2684	97,9098	98,3356	1331	10	1341	91	2	93	0,9925	0,9925	0,9925	99,2543	97,8495	0,9922
9	0,3111	97,9695	97,7772	1322	11	1333	98	3	101	0,9917	0,9917	0,9917	99,1748	97,0297	0,9753
10	0,2615	97,8501	99,1196	1324	9	1333	100	1	101	0,9932	0,9932	0,9932	99,3248	99,0099	0,9897

Fonte: elaborado pelo Autor

Tabela 4 - Rodadas de Teste para Trees

R	Tempo de Treino (s)	Acurácia em Treino (%)	Acurácia em Teste (%)	TP	FP	P	TN	FN	N	Precisão	Recall	F1	TPR (%)	TNR (%)	AUC
1	0,0395	98,2383	98,2523	1320	16	1336	96	2	98	0,9880	0,9880	0,9880	98,8024	97,9592	0,9825
2	0,0307	98,2383	98,2393	1315	18	1333	99	2	101	0,9865	0,9865	0,9865	98,6497	98,0198	0,9959
3	0,0956	98,0591	98,1875	1316	19	1335	97	2	99	0,9858	0,9858	0,9858	98,5768	97,9798	0,9923
4	0,0187	98,0591	98,9597	1313	16	1329	104	1	105	0,9880	0,9880	0,9880	98,7961	99,0476	0,9907
5	0,0193	98,2681	98,1093	1313	22	1335	97	2	99	0,9835	0,9835	0,9835	98,3521	97,9798	0,9943
6	0,0202	98,4174	96,8148	1318	19	1337	93	4	97	0,9858	0,9858	0,9858	98,5789	95,8763	0,9909
7	0,0215	98,3279	98,2386	1321	16	1337	95	2	97	0,9880	0,9880	0,9880	98,8033	97,9381	0,9880
8	0,0211	98,4174	94,3098	1322	14	1336	90	8	98	0,9895	0,9895	0,9895	98,9521	91,8367	0,9585
9	0,0192	98,3279	97,4406	1318	20	1338	93	3	96	0,9851	0,9851	0,9851	98,5052	96,8750	0,9887
10	0,0182	98,2383	98,8117	1311	21	1332	101	1	102	0,9842	0,9842	0,9842	98,4234	99,0196	0,9902

Fonte: elaborado pelo Autor

Tabela 5 - Rodadas de Teste para AdaBoost

R	Tempo de Treino (s)	Acurácia em Treino (%)	Acurácia em Teste (%)	TP	FP	P	TN	FN	N	Precisão	Recall	F1	TPR (%)	TNR (%)	AUC
1	1,6896	98,2084	96,3237	1325	12	1337	92	5	97	0,9910	0,9910	0,9910	99,1025	94,8454	0,9884
2	1,7604	98,0293	97,7166	1325	11	1336	95	3	98	0,9918	0,9918	0,9918	99,1766	96,9388	0,9972
3	1,6870	98,089	97,659	1321	14	1335	96	3	99	0,9895	0,9895	0,9895	98,9513	96,9697	0,9949
4	1,6148	97,9397	99,0854	1319	11	1330	103	1	104	0,9917	0,9917	0,9917	99,1729	99,0385	0,9982
5	1,8023	98,2084	95,5526	1316	19	1335	93	6	99	0,9858	0,9858	0,9858	98,5768	93,9394	0,9919
6	1,7923	98,1487	97,5832	1331	11	1342	89	3	92	0,9918	0,9918	0,9918	99,1803	96,7391	0,9976
7	1,8183	98,0591	98,1866	1319	18	1337	95	2	97	0,9865	0,9865	0,9865	98,6537	97,9381	0,9916
8	1,6993	98,089	98,9239	1324	15	1339	94	1	95	0,9888	0,9888	0,9888	98,8798	98,9474	0,9984
9	1,6433	98,2084	96,281	1321	15	1336	93	5	98	0,9888	0,9888	0,9888	98,8772	94,8980	0,9894
10	1,6729	97,9695	98,9688	1317	15	1332	101	1	102	0,9887	0,9887	0,9887	98,8739	99,0196	0,9946

Fonte: elaborado pelo Autor

Tabela 6 - Rodadas de Teste para RF

R	Tempo de Treino (s)	Acurácia em Treino (%)	Acurácia em Teste (%)	TP	FP	P	TN	FN	N	Precisão	Recall	F1	TPR (%)	TNR (%)	AUC
1	0,9555	97,9695	98,2664	1319	16	1335	97	2	99	0,9880	0,9880	0,9880	98,8015	97,9798	0,9978
2	0,9025	97,9695	97,1319	1320	12	1332	98	4	102	0,9910	0,9910	0,9910	99,0991	96,0784	0,9935
3	0,9710	97,9695	98,3441	1319	14	1333	99	2	101	0,9895	0,9895	0,9895	98,9497	98,0198	0,9966
4	1,0088	97,9695	99,0349	1327	11	1338	95	1	96	0,9918	0,9918	0,9918	99,1779	98,9583	0,9988
5	0,9260	97,9695	98,9628	1318	15	1333	100	1	101	0,9887	0,9887	0,9887	98,8747	99,0099	0,9989
6	0,9537	97,9695	98,9688	1317	15	1332	101	1	102	0,9887	0,9887	0,9887	98,8739	99,0196	0,9984
7	0,9623	97,9695	95,765	1319	14	1333	95	6	101	0,9895	0,9895	0,9895	98,9497	94,0594	0,9949
8	0,8440	97,9695	98,3186	1321	14	1335	97	2	99	0,9895	0,9895	0,9895	98,9513	97,9798	0,9983
9	0,9865	97,9695	97,6093	1328	12	1340	91	3	94	0,9910	0,9910	0,9910	99,1045	96,8085	0,9975
10	1,0103	97,9695	98,3315	1320	14	1334	98	2	100	0,9895	0,9895	0,9895	98,9505	98,0000	0,9979

Fonte: elaborado pelo Autor

3.6 Resultados

A tabela 7 mostra as médias dos campos mais relevantes e o respectivo desvio padrão calculado para cada célula.

Tabela 7 - Médias das Rodadas de Classificação

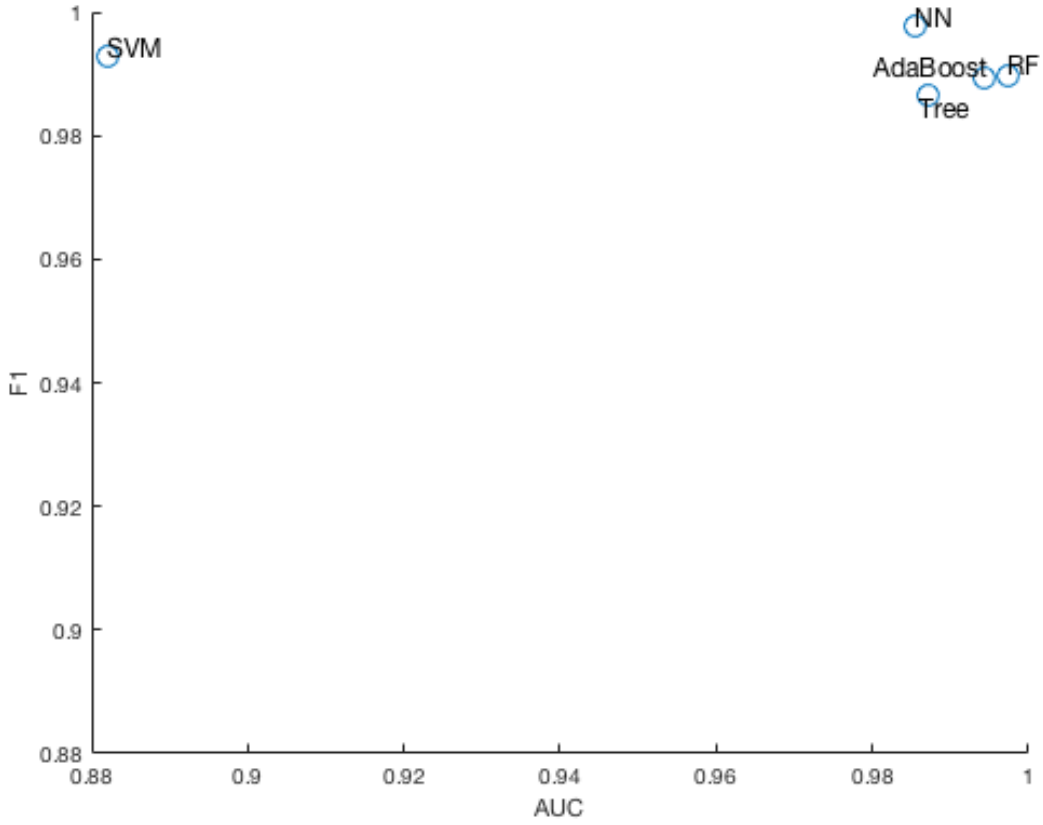
Classificador	Tempo de Treino (s)	Acurácia em Treino (%)	Acurácia em Teste (%)	F1	TPR (%)	TNR (%)	AUC
NN	4,1218 ± 1,3925	97,9300 ± 0,2627	99,1100 ± 0,1101	0,9976 ± 0,0013	99,7595 ± 0,1265	90,6586 ± 1,6956	0,9854 ± 0,0076
SVM	0,2552 ± 0,0310	97,9247 ± 0,0693	97,8648 ± 1,4210	0,9927 ± 0,0009	99,2740 ± 0,0870	97,1152 ± 2,1806	0,8820 ± 0,0126
Tree	0,0304 ± 0,0239	98,2592 ± 0,1252	97,7664 ± 1,3517	0,9864 ± 0,0019	98,6440 ± 0,1916	97,2532 ± 2,1168	0,9872 ± 0,0107
AdaBoost	1,7180 ± 0,0706	98,0950 ± 0,0983	97,6281 ± 1,2359	0,9894 ± 0,0022	98,9445 ± 0,2161	96,9274 ± 1,8665	0,9942 ± 0,0037
RF	0,9521 ± 0,0506	97,97 ± 0	98,0733 ± 1,0102	0,9897 ± 0,0012	98,9733 ± 0,1184	97,5914 ± 1,5617	0,9973 ± 0,0018

Fonte: elaborado pelo Autor

Segundo Miyamoto, Hazeyama e Kadobayashi (2008), pode-se entender que as medidas mais relevantes para avaliação de um classificador são a medida F1 que mede a acurácia e a AUC que é uma métrica para avaliar a capacidade de ajuste do classificador.

A figura 5 mostra a posição dos classificadores testados em um plano bidimensional de acordo com seus respectivos valores de AUC, no eixo das abcissas, e F1 no eixo das ordenadas. De acordo com esta métrica, o melhor classificador é o RF, porque mais se aproxima de nordeste do gráfico.

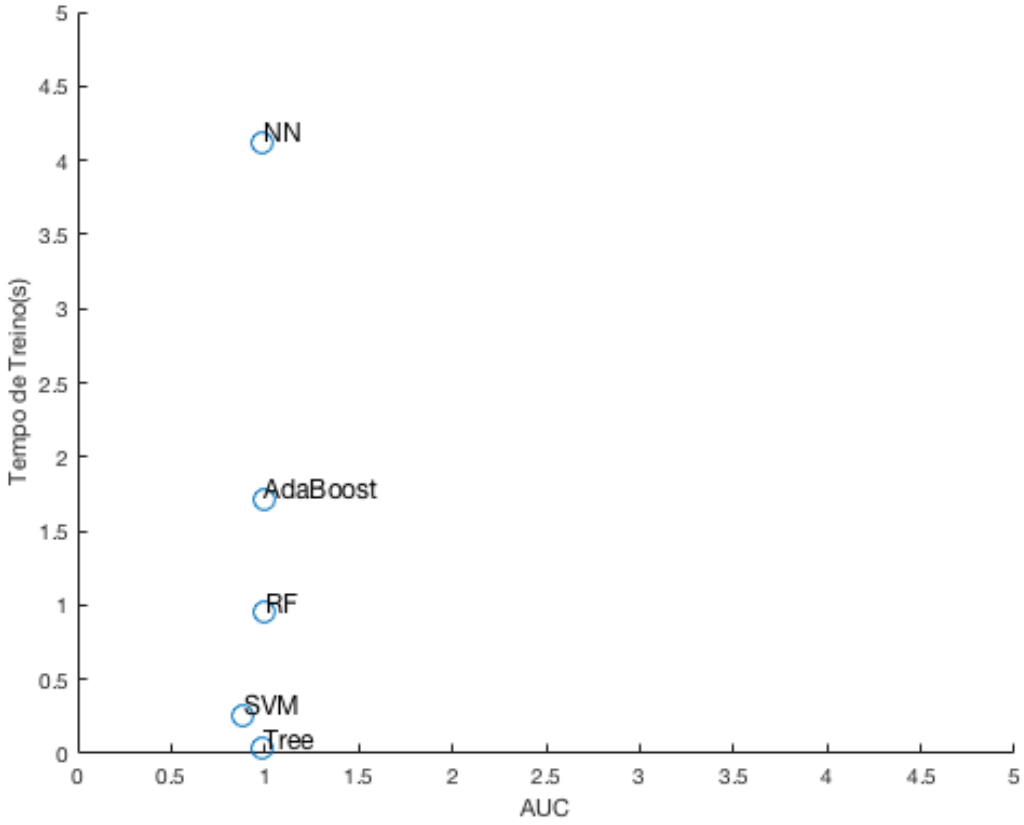
Figura 5 - Gráfico F1 x AUC



Fonte: elaborado pelo Autor

A figura 6 insere o tempo de treinamento do classificador no eixo das ordenadas. Mesmo que neste teste nenhum tempo tenha superado seis segundos, é relevante levá-lo em consideração ao supor que o sistema possa rodar com bases de dados muito grandes. Neste caso, o melhor classificador é a Árvore de Decisão porque mais se aproxima de sudeste.

Figura 6 - Gráfico Tempo de Treino x AUC



Fonte: elaborado pelo Autor

Na tabela 8, foram calculadas as distâncias de cada classificador em relação ao ponto ótimo de cada figura. O campo D1 contém a distância entre o posição do classificador e o ponto (1,1) da figura 5 e o campo D2 mede a respectiva distância de cada um em relação ao ponto (1,0) da figura 6.

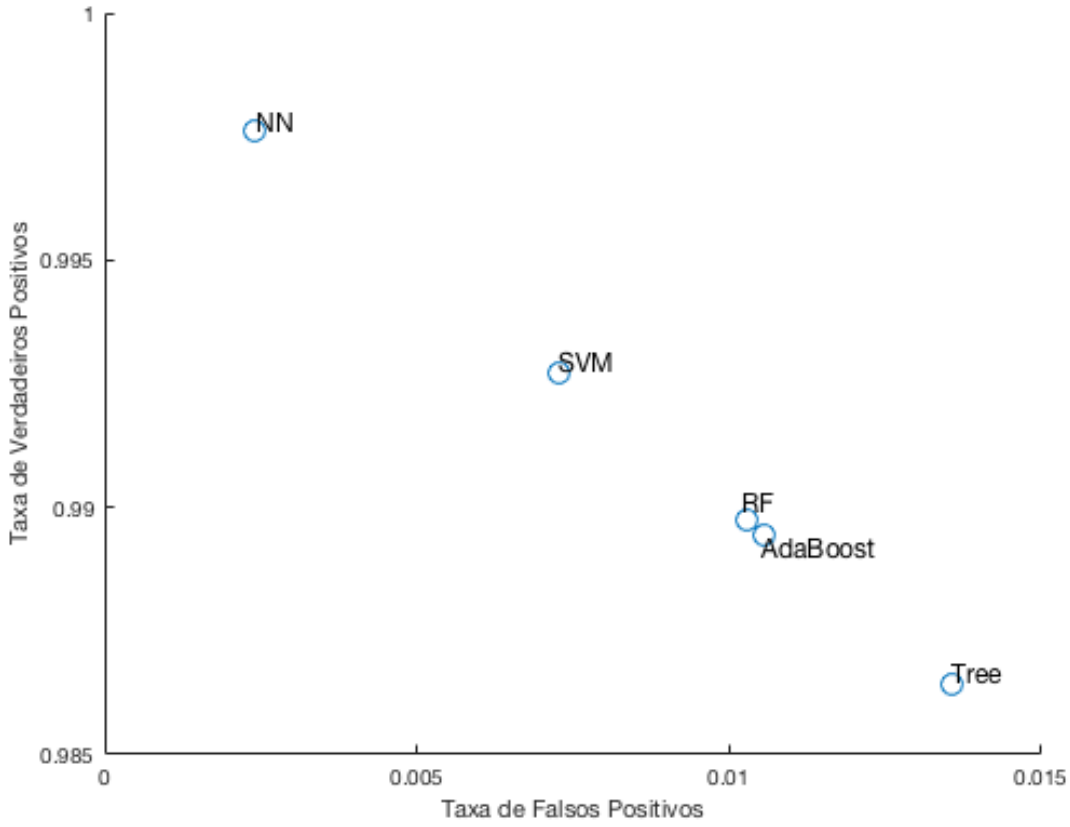
Tabela 8 - Cálculos de Distâncias para cada Classificador

Classificador	D1	D2
NN	0,0148	4,1218
SVM	0,1182	0,2812
Tree	0,0147	0,0330
AdaBoost	0,0121	1,7180
RF	0,0106	0,9521

Fonte: elaborado pelo Autor

Por fim, a figura 7 mostra os classificadores no espaço ROC em função de seus respectivos valores de TPR no eixo das ordenadas e FPR no eixo das abcissas, conforme detalhados na tabela 9. Conforme dito anteriormente, quanto mais próximo de noroeste melhor é o desempenho do classificador, porque significa uma alta TPR e uma baixa FPR. Neste caso, o melhor classificador é a NN.

Figura 7 - Gráfico de Classificadores no Espaço ROC Ampliado



Fonte: elaborado pelo Autor

Tabela 9 - Médias de TPR e FPR de cada Classificador

Classificador	TPR (%)	FPR (%)
NN	99,7595	0,2408
SVM	99,2740	0,7260
Tree	98,6440	1,3560
AdaBoost	98,9445	1,0555
RF	98,9733	1,0266

Fonte: elaborado pelo Autor

4 CONCLUSÃO

É possível notar que as RF obtiveram a maior AUC com AUC = 0,9973 e F1 = 0,9897. Por outro lado, as NN obtiveram a maior F1 com F1 = 0,9976 e AUC = 0,9854. Nota-se também a semelhança de desempenho entre as RF e o AdaBoost com AUC e F1 próximos.

Considerando o gráfico da figura 5, pode-se concluir que virtualmente o melhor classificador é aquele que mais se aproxima do ponto ótimo (1,1). Pode-se dizer então que os classificadores que melhor desempenharam para detecção de *phishing* em *e-mails* são decrescentemente RF, AdaBoost, árvores de decisão e NN quase empatados e, por último, o SVM.

Levando-se em consideração o tempo de treinamento, as NN tiveram o pior com t = 4,1218 segundos e as árvores de decisão o melhor com t = 0,0304 segundos. Se for calculada a distância de cada posição dos classificadores em relação ao ponto ótimo que é (1,0) da figura 6, pode-se dizer que, virtualmente, os melhores classificadores decrescentemente são Árvores de Decisão, SVM, RF, AdaBoost e NN.

Ao analisar as posições dos classificadores no espaço ROC que leva em consideração FPR e TPR, pode-se dizer claramente que, apesar das RF terem obtido a maior AUC, as NN apresentam o melhor desempenho médio com a maior TPR e menor FPR seguida por SVM, RF, AdaBoost e Árvores de Decisão.

É possível perceber com todo este estudo que o desempenho dos classificadores varia de acordo com o problema e a forma como ele vai ser resolvido. O aprendizado de máquina é muito versátil para resolver problemas que envolvam o reconhecimento de padrões ou que não possuam um algoritmo que possa ser pensado.

5 TRABALHOS FUTUROS

Para avaliar a melhora da eficácia dos classificadores, pode-se abordar outras técnicas como a PLSA para analisar o conteúdo dos *e-mails* e o aprendizado semi-supervisionado para treinamento dos classificadores assim como fizeram Ramanathan e Weschsler (2012) com o PhishGILLNET.

Pode-se, também, avaliar a eficácia dos classificadores em um ambiente de produção com *e-mails* não rotulados, construir uma base de dados própria de *e-mails* de *phishing* e legítimos para re-aprendizagem dos classificadores ou que possa ser utilizada em outros estudos.

Por fim, pode-se estudar maneiras e a viabilidade de aplicar a detecção de *phishing* em *e-mails* criptografados.

REFERÊNCIAS

AGGARWAL, A.; RAJADESINGAN, A.; KUMARAGURU, P.. PhishAri: Automatic realtime phishing detection on Twitter. **In ecrime researchers summit (ecrime)**. 2012. p. 1–12.

AKINYELU, A. A.; ADEWUMI, A. O. **Classification of Phishing Email Using Random Forest Machine Learning Technique**. University of KwaZulu-Natal. Durban, 2014. Acesso em: <https://www.hindawi.com/journals/jam/2014/425731/> . Acesso em: 24 abr. 2017.

APWG. **Phishing Activity Trends Report 4th Quarter 2016**. 23 Fev. 2017. Disponível em: <https://docs.apwg.org/reports/apwg_trends_report_q4_2016.pdf>. Acesso em: 30 jun. 2017.

BREIMAN, L. Random forests. **Machine Learning**. Universidade da California. Berkeley, 2001.

CERT.br. **Cartilha de Segurança para Internet**. São Paulo, 2012. Disponível em: <https:// cartilha.cert.br/livro/cartilha-seguranca-internet.pdf>. Acesso em 7 jul. 2017.

CORTES, C.; VAPNIK, V. Support-Vector Networks. **Machine Learning**. Boston: Kluwer Academic Publishers, 1995. p. 273-297.

DILEK, S.; ÇAKIR, H.; AYDIN, M. **Applications Of Artificial Intelligence Techniques To Combating Cyber Crimes: A Review**. Gazi University. Ankara, 2015. Disponivel em: <https://arxiv.org/pdf/1502.03552.pdf>. Acesso em: 24 abr. 2017.

FALK, C. Knowledge Modeling of Phishing Emails. **CERIAS Tech Report 2016-3**. Purdue University. West Lafayette, 2016.

FAWCETT, T. An Introduction to ROC Analysis. **Pattern Recognition Letters**. Institute for the Study of Learning and Expertise. Palo Alto, Dez. 2005.

FETTE, I.; SADEH, N. TOMASIC, A. Learning to Detect Phishing Emails. World Wide Web Conference Committee. Banff, Mai. 2007. Disponível em: <http://www2007.wwwconference.org/htmlpapers/paper550/>. Acesso em 30 jun. 2017.

FREUND, Y.; SCHAPIRE, R. E. Experiments with a New Boosting Algorithm. **Proceedings of the 13th International Conference on Machine Learning (ICML'96)**. 1996.

HAYKIN, S. **Neural Networks and Learning Machines**. 3. ed. Upper Saddle River: Pearson Education. 2009.

HENKE, M.; SANTOS, C.; NUNAN, E.; FEITOSA, E.; SANTOS, E.; SOUTO, E. **Aprendizagem de Máquina para Segurança em Redes de Computadores: Métodos e Aplicações**. Universidade Federal do Amazonas. Manaus, 2014. Disponível em: <https://www.researchgate.net/profile/Eulanda_Santos/publication/228447003_Aprendizagem_de_Maquina_para_Seguranca_em_Redes_de_Computadores_Metodos_e_Aplicacoes/links/0fcfd510a7af45b479000000.pdf>. Acesso em 24 abr. 2017.

IWS. **Brazil Internet Stats and Telecom Market Report**. 2017. Disponível em: <http:// www.internetworldstats.com/sa/br.htm>. Acesso em: 7 jul. 2017.

KLETTENBERG, J. **SEGURANÇA DA INFORMAÇÃO: Um estudo sobre o uso da engenharia social para obter informações sigilosas de usuários de Instituições Bancárias**. Universidade Federal de Santa Catarina, Florianópolis, 2016. Disponível em: <https://repositorio.ufsc.br/xmlui/bitstream/handle/123456789/172575/343623.pdf? sequence=1&isAllowed=y>. Acesso em: 30 jun. 2017.

KROMBHOLZ, K.; HOBEL, H.; HUBER, M.; WEIPPL, E. Advanced social engineering attacks**. Journal of information security and applications,** n. 22, 2014. Disponível em: <https://pdfs.semanticscholar.org/3266/f05e2e5e785cbab72d2e378059ecc62ef706.pdf>. Acesso em: 30 jun. 2017.

MIYAMOTO, D.; HAZEYAMA, H.; KADOBAYASHI, Y. **An Evaluation of Machine Learning-based Methods for Detection of Phishing Sites**. Nara Institute of Science and Technology. Nara, 2008. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.1013.5518&rep=rep1&type=pdf>. Acesso em: 24 abr. 2017.

OLIVO, C. K. **Avaliação De Características Para Detecção De Phishing De Email**. Pontifícia Universidade Católica do Paraná. Curitiba, 2010. Disponível em: <http://www.inf.ufpr.br/lesoliveira/download/CleberOlivoMSC.pdf>. Acesso 24 abr. 2017.

OLIVO, C. K.; SANTIN, A. O.; OLIVEIRA, L. E. S. **Abordagens para Detecção de Spam de E-mail**. XV Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais. 2015. Disponível em: <https://secplab.ppgia.pucpr.br/files/papers/2015-7.pdf>. Acesso em 24 abr. 2017.

RAMANATHAN, V.; WECHSLER, H. PhishGILLNET—phishing detection methodology using probabilistic latent semantic analysis, AdaBoost, and co-training. **EURASIP Journal on Information Security**. 2012. Disponível em: <https://jis-eurasipjournals.springeropen.com/articles/10.1186/1687-417X-2012-1>. Acesso em: 30 jun. 2017

REZENDE, Solange Oliveira. Sistemas Inteligentes: Fundamentos e Aplicações. In: MONARD, M. C.; BARANAUSKAS, J. A. **Conceitos sobre Aprendizado de Máquina**. Barueri, SP: Editora Manole Ltda, 2003. p.39-56.

RSA. **See Everything, Fear Nothing. 2016**. Disponível em: <https://www.rsa.com/content/dam/rsa/PDF/2016/09/spearphishing-use-case.pdf>. Acesso em: 7 jul. 2017.

SAFAVIAN, S. R.; LANDGREBE, D. **A Survey of Decision Tree Classifier Methology**. Universidade Purdue. West Lafayette, 1990.

SECURELIST. **Spam and Phishing in 2016**. Disponível em: <https://securelist.com/kaspersky-security-bulletin-spam-and-phishing- in-2016/77483/>. Acesso em: 30 jun. 2017.

SILVA, A. de O. e. Engenharia social: o fator humano na segurança da informação. Coleção Meira Mattos - **Revista das Ciências Militares**. Rio de Janeiro, n. 23, nov. 2011. Disponível em: <http://portal.eceme.ensino.eb.br/meiramattos/index.php/RMM/article/viewFile/16/49>. Acesso em: 30 jun. 2017.

Xiang, G.; Hong, J.; Rose, C. P.; Cranor, L.; CANTINA+: A feature-rich machine learning framework for detecting phishing web sites. **ACM Transactions on Information and System Security**. 2001.

ZHANG, Y.; HONG, J.; CRANOR, L. CANTINA: A Content-Based Approach to Detecting Phishing Web Sites. **International World Wide Web Conference Committee (IW3C2)**. 2007.