UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO"
FACULDADE DE CIÊNCIAS - CAMPUS BAURU
DEPARTAMENTO DE COMPUTAÇÃO
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

VINICIUS MACHADO COUTINHO

DETECÇÃO DE PÁGINAS DE PHISHING UTILIZANDO APRENDIZADO DE MÁQUINA

BAURU
Janeiro/2023

VINICIUS MACHADO COUTINHO

DETECÇÃO DE PÁGINAS DE PHISHING UTILIZANDO APRENDIZADO DE MÁQUINA

Trabalho de Conclusão de Curso do Curso de Ciência da Computação da Universidade Estadual Paulista "Júlio de Mesquita Filho", Faculdade de Ciências, Campus Bauru.
Orientador: Prof. Dr. Kelton Augusto Pontara da Costa

BAURU
Janeiro/2023

C871d

Coutinho, Vinicius Machado
Detecção de páginas de phishing utilizando aprendizado de máquina / Vinicius Machado Coutinho. -- Bauru, 2023
30 f. : il., tabs.

Trabalho de conclusão de curso (Bacharelado - Ciência da Computação) - Universidade Estadual Paulista (Unesp), Faculdade de Ciências, Bauru
Orientador: Kelton Augusto Pontara da Costa

1. Aprendizado de máquinas. 2. Inteligência Artificial. 3. Fraude na Internet. 4. Árvores de Decisão. I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Biblioteca da Faculdade de Ciências, Bauru. Dados fornecidos pelo autor(a).

Essa ficha não pode ser modificada.

Vinicius Machado Coutinho

Detecção de páginas de phishing utilizando aprendizado de máquina

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

Banca Examinadora

Prof. Dr. Kelton Augusto Pontara da Costa
Orientador
Universidade Estadual Paulista "Júlio de Mesquita Filho"
Faculdade de Ciências
Departamento de Ciência da Computação

Prof. Dra. Simone Prado
Universidade Estadual Paulista "Júlio de Mesquita Filho"
Faculdade de Ciências
Departamento de Ciência da Computação

Prof. Me. Luiz Felipe de Camargo
Universidade Estadual Paulista "Júlio de Mesquita Filho"
Faculdade de Ciências
Departamento de Ciência da Computação

Bauru, _____ de _____ de _____.

Resumo

Com o avanço da Internet e o aumento de serviços digitais, tentativas de fraudes *online*, como o *phishing*, se tornaram um problema cada vez maior. Devido ao contínuo aumento e evolução desses ataques, há a necessidade do desenvolvimento e aprimoramento de métodos para detecção deles. Neste trabalho, foi criado um sistema de detecção de páginas de *phishing*, utilizando técnicas de aprendizado de máquina como Árvore de Decisão, Floresta Aleatória, Árvores Extremamente Aleatórias e XGBoost. Os modelos foram desenvolvidos com um conjunto de dados de 88.647 entradas e mediu-se suas efetividades através de métricas já estabelecidas na área de aprendizado de máquina. Os resultados obtidos foram promissores, com o modelo XGBoost apresentando o melhor resultado, mostrando-se eficaz para a detecção de páginas da internet falsas.

Palavras-chave: Aprendizado de máquinas; Inteligência Artificial; Fraude na Internet; Árvores de Decisão.

Abstract

With the advancement of the Internet and increase use of digital services, online fraud attempts, like phishing, has become an even bigger problem. Because of the ongoing increase and evolution of these attacks, there's a need to develop and improve the methods used to detect them. In this work, a system for detection of phishing pages was developed using machine learning techniques such as Decision Tree, Random Forest, Extremely Randomized Trees and XGBoost. The models were developed using a dataset with 88,647 entries and their effectiveness was measured using metrics already established in the machine learning field. The results were promising, with the XGBoost model presenting the best result, showing to be effective in detecting fake internet pages.

Keywords: Machine Learning; Artificial intelligence; Internet Fraud; Decision Trees.

Lista de figuras

Lista de quadros

Lista de tabelas

Sumário

1 Introdução

A Internet se tornou uma parte indispensável da nossa sociedade. Produtos e serviços digitais estão cada vez mais presentes no dia a dia das pessoas. Cerca de 66.2% da população mundial, aproximadamente 5,2 bilhões de pessoas, tem acesso a Internet (BROADBAND SEARCH, 2022). Entretanto, com empresas e pessoas cada vez mais conectadas, os crimes cibernéticos também aumentam.

Um tipo de crime bastante comum é o *phishing*, onde o atacante utiliza de técnicas de engenharia social para obter dados pessoais de um usuário da Internet. Para James (2005), um ataque de *phishing* pode ser descrito como o ato de mandar um *e-mail* falso para uma possível vítima, com o intuito de obter informações privadas e sensíveis ao se passar por uma entidade confiável.

Segundo o Phishing Activity Trends Report do segundo trimestre de 2022 feito pelo Anti-Phishing Working Group, uma coalizão internacional contra crimes cibernéticos, relata que entre abril e junho de 2022 foram relatados mais de 1 milhão de ataques de *phishing*. O maior número registrado pela organização em um trimestre. (APWG, 2022).

Nesse mesmo contexto, o X-Force Threat Intelligence Index 2022, relatório da empresa IBM que avalia o panorama de crimes cibernéticos no mundo, relatou que o *phishing* foi a maior causa de vetores iniciais de ataques em 2022, com 41%. Um aumento de 8% em relação a 2021 (IBM, 2022). Esse crescimento sugere que os atacantes estão modificando seus métodos de ataque e, assim, acompanhando os avanços nos mecanismos de defesa contra esse tipo de agressão.

Na área do Aprendizado de Máquina foram realizados diversos trabalhos para a detecção desses ataques, pois o *phishing* pode ser naturalmente transformado em uma tarefa de classificação. Uma técnica de aprendizado de máquina é capaz de criar um modelo que consegue prever possíveis endereços de *phishing* com base em dados de endereços já rotulados e, então, o modelo é integrado em algum programa ou navegador da Internet. (ABDELHAMID; THABTAH; ABDEL-JABER, 2017)

Este trabalho busca aplicar conceitos e técnicas de aprendizado de máquina para a detecção de ataques de *phishing*. O objetivo é criar um modelo capaz de classificar se uma página é falsa ou não, através de suas características.

Este Capítulo 1 contém a introdução e objetivos deste trabalho. No Capítulo 2 é apresentado toda a fundamentação teórica do trabalho. O Capítulo 3 descreve as ferramentas e metodologias utilizadas no desenvolvimento dos modelos. O Capítulo 4 apresenta os resultados obtidos e, por fim, no Capítulo 5, está a conclusão do presente trabalho.

1.1 Objetivos

A seguir, estão listados o objetivo geral e os objetivos específicos deste trabalho.

1.1.1 Objetivo Geral

O objetivo geral deste trabalho é utilizar técnicas de aprendizado de máquina para desenvolver um sistema capaz de detectar ataques de *phishing*.

1.1.2 Objetivos Específicos

Os objetivos específicos deste trabalho são:

- Realizar um estudo sobre o conceito e técnicas de aprendizado de máquina;
- Escolher quais técnicas serão usadas para o desenvolvimento dos modelos;
- Selecionar um conjunto de dados para o treinamento dos modelos;
- Treinar os modelos e coletar os resultados obtidos;
- Analisar os resultados e desempenho dos modelos.

2 Fundamentação Teórica

Nesta seção tem como finalidade apresentar as fundamentações teóricas utilizadas no trabalho presente.

2.1 Phishing

Para Whittaker, Ryner e Nazif (2010), uma definição mais abrangente de *phishing* é uma página web que se passa, sem permissão, por uma outra página com o objetivo de induzir o usuário à realizar ações que eles só fariam no site original.

De forma geral, o conceito desse ataque é clonar uma página de um serviço conhecido e confiável, como bancos financeiros, lojas, instituições do governo, etc. e enviar o link dessa página para vítimas via *e-mail*. Ao acessar a pagina, o usuário é requirido a colocar dados como login, senha, nome, endereço, entre outros, para prosseguir. Essas informações então são enviadas para o agressor e ele faz o uso desses dados para benefício próprio.

De acordo com James (2005), o primeiro caso relatado de *phishing* aconteceu no America Online (AOL) em 1995, porém, foi apenas em 2003 que grandes instituições financeiras como CityBank e Wells Fargo começaram a ser alvos deste tipo de ataque. O grande diferencial do *phishing* para os outros tipos de ataques cibernéticos na época é o elemento humano presente nele.

Dhamija, Tygar e Hearst (2006) enfatizam que a falta de conhecimento sobre sistema de computadores e a pouca compreensão a respeito de sistemas de seguranças e seus indicadores, como duas importantes causas no sucesso desses ataques.

Diversas abordagens foram criadas para o combate ao *phishing* e, segundo Khonji, Iraqi e Jones (2013), pode-se separar esses métodos em dois tipos:

- Educação do usuário: educa-se o indivíduo nas características que compõem uma mensagem e página de *phishing*.
- Aprimoramento de software: *softwares* são refinados para melhorar a identificação de tentativas de ataque de *phishing*.

Ainda, de acordo com Khonji, Iraqi e Jones (2013), ambas abordagens possuem desafios em suas implementações. Indivíduos leigos têm dificuldades em aprender e reter as diferentes nuances entre um site original e um site clonado. Soluções envolvendo *softwares* também são dependentes de seus usuários, pois, eles podem ignorar avisos e métodos de segurança colocados na aplicação.

2.2 Aprendizado de Máquina

O Aprendizado de Máquina é uma área da Inteligência Artificial e Ciência da Computação que abrange o desenvolvimento de sistemas que, com o mínimo de intervenção humana, consigam aprender de forma autônoma, identificar padrões e tomar decisões, através de dados e experiências passadas.

Os algoritmos de aprendizado de máquina podem ser divididos em três principais tipos: (i) aprendizado supervisionado; (ii) aprendizado não supervisionado; (iii) aprendizado por reforço.

Raschka (2015) define esses algoritmos como:

- Aprendizado supervisionado: Um algoritmo que recebe dados de entrada e saída já rotulados, com o intuito de criar um modelo que consiga fazer previsões em dados ainda desconhecidos.
- Aprendizado não supervisionado: Um algoritmo que recebe dados não rotulados ou de estrutura desconhecida, com o objetivo de extrair informações e padrões dos dados recebidos.
- Aprendizado por reforço: Um sistema que realiza ações de acordo com seu ambiente, com o objetivo de maximar algum conceito de recompensa.

2.2.1 Aprendizado de Máquina Supervisionado

Algoritmos de aprendizado de máquina supervisionados podem ser divididos em dois tipos: classificação e regressão. Um algoritmo de classificação tem como objetivo calcular em que classe um determinado dado pertence, baseado em experiências anteriores. Já um algoritmo de regressão procura prever um valor contínuo.

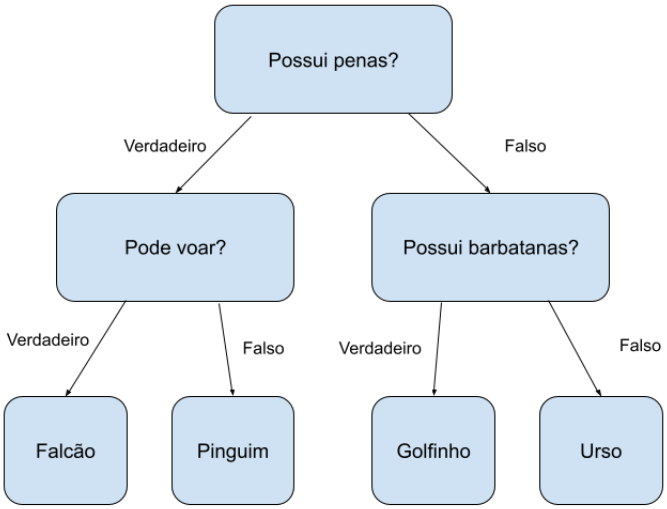
A seguir serão descritos os algoritmos de classificação utilizados neste trabalho.

2.2.1.1 Árvore de Decisão

Árvore de Decisão são um modelo muito popular para realizar tarefas de classificação e regressão. Elas são, fundamentalmente, uma hierarquia de perguntas e respostas que levam a uma decisão.

Na Figura 1, tem-se o caso de uma Árvore de Decisão cujo objetivo é descobrir em qual categoria um animal pertence: falcão, pinguim, golfinho e urso. Nessa imagem, cada nó interno representa uma característica do conjunto de dados, cada galho representa uma regra de decisão e cada nó folha é um possível resultado.

Figura 1 – Exemplo de uma árvore de decisão simples.



Fonte: Adaptado de Müller e Guido (2016).

Árvores de Decisões possuem duas grandes vantagens comparado a outros algoritmos de aprendizado de máquina: elas são fáceis de entender e interpretar; e requerem pouca preparação para os dados (MüLLER; GUIDO, 2016).

Em contrapartida, Árvores de Decisões são propensas a *overfitting*. *Overfitting* é quando um modelo não consegue generalizar todos os dados e fornece previsões precisas apenas para os dados de treinamento.

2.2.1.2 Floresta Aleatória

A Floresta Aleatória é um algoritmo do tipo *ensemble*. Müller e Guido (2016) definem os métodos *ensembles* como "métodos que combinam múltiplos modelos de aprendizado de máquina para criar modelos mais poderosos".

Como citado anteriormente, uma Árvore de Decisão é propensa a *overfitting*. As Florestas Aleatórias são uma maneira de lidar com este problema. Uma Floresta Aleatória é, basicamente, um conjunto de Árvores de Decisões, onde cada árvore é diferente da outra.

Para Raschka (2015), uma Floresta Aleatória pode ser descrita em quatro passos:

1. Selecionar uma amostra aleatória de tamanho *n* dos dados de treino, sendo possível o mesmo dado ser selecionado mais de uma vez;
2. Criar uma Árvore de Decisão desta amostra, onde a cada nó irá:
    - Selecionar aleatoriamente *d* características do conjunto de dados;

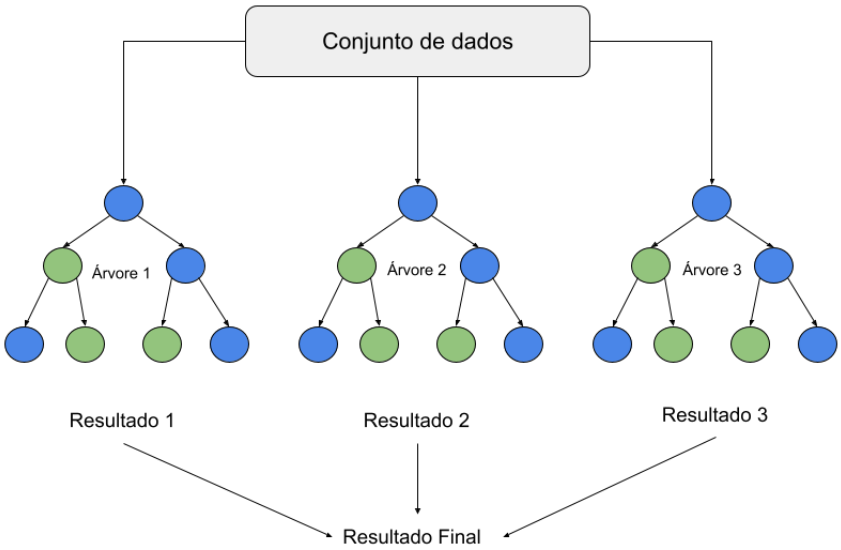
- Separar o nó usando a característica que faz a melhor divisão, de acordo com alguma função objetiva.

3. Os passos 1 e 2 são repetidos k vezes.

4. A previsão final do modelo é obtida através da média dos resultados de todas as árvores ou pelo resultado mais frequente.

A Figura 2 apresenta a estrutura de um modelo de Floresta Aleatória. A partir do conjunto de dados, cria-se múltiplas árvores e, no final, uni-se os resultados de todas árvores para gerar a previsão do modelo.

Figura 2 – Exemplo de estrutura de uma Floresta Aleatória.



Fonte: Elaborado pelo autor.

2.2.1.3 Árvores Extremamente Aleatórias

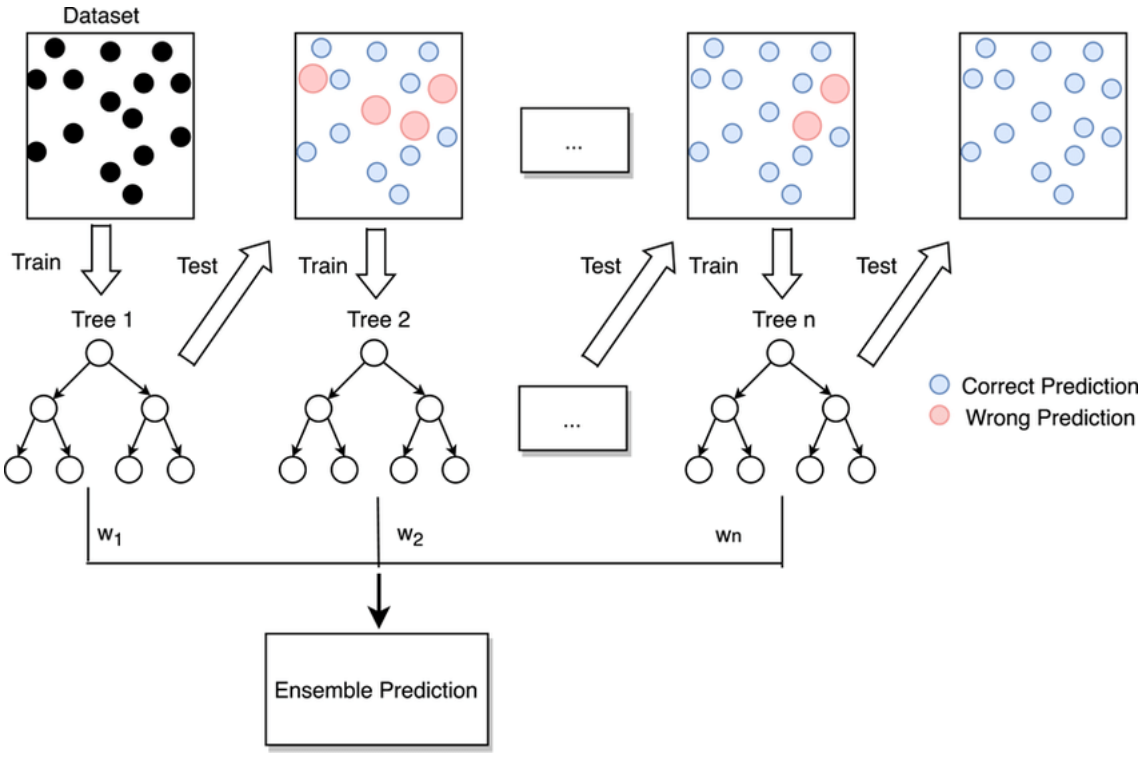
Modelo proposto por Geurts, Ernst e Wehenkel (2006), o Árvores Extremamente Aleatórias, também conhecido como Extra-Trees, tem seu funcionamento igual ao de uma Floresta Aleatória, com duas diferenças: a separação do nó é feita de maneira aleatória e o conjunto de dados de treinamento é usado por completo para criar as árvores.

O aumento na aleatoriedade faz com que este algoritmo possua uma variância menor e um viés maior, em relação ao Floresta Aleatória. Geurts, Ernst e Wehenkel (2006) também mostraram que essa aleatoriedade pode ser ajustada, levando a variância do modelo a quase desaparecer e o viés aumentar levemente em relação a outros tipos de árvores.

2.2.1.4 XGBoost

Desenvolvido por Chen e Guestrin (2016), o XGBoost é um modelo de aprendizado de máquina do tipo *gradient boosting*. *Boosting* é uma técnica *ensemble* que cria modelos de maneira sequencial, onde cada modelo é ajustado de acordo com os erros dos modelos anteriores. Este processo é exemplificado na Figura 3. *Gradient Boosting* é um algoritmo do tipo *boosting*, no qual novos modelos são criados para calcular os resíduos dos modelos anteriores e, então, esses modelos são somados para determinar o resultado final. Seu nome decorre do fato deste algoritmo utilizar um sistema de gradiente descendente para minimizar a perda quando novos modelos são adicionados (OGUNLEYE; WANG, 2019).

Figura 3 – Exemplo de estrutura de um modelo do tipo Boosting.



Fonte: Zhang et al. (2021).

Também conhecido como Extreme Gradient Boosting, o XGBoost é um algoritmo que busca otimizar o *gradient boosting* por meio de um método de regularização mais avançado e utilizando gradientes de segunda ordem para calcular a função de perda. O modelo também traz melhoras no desempenho, treinamentos podem ser feitos rapidamente através de paralelismo e uso de memória em cache.

2.3 Métodos Para Avaliação de Desempenho

Nesta seção serão apresentadas as métricas de desempenho a serem usadas neste trabalho.

2.3.1 Classificação Binária

Segundo Fawcett (2006), em um classificador binário existem quatro possíveis resultados para cada dado no modelo:

- TP (verdadeiro positivo): Quando uma instância é verdadeira e o classificador identifica ela como verdadeira;
- TN (verdadeiro negativo): Quando uma instância é falsa e ela é identificada como falsa;
- FP (falso positivo): Quando uma entrada é falsa e o modelo prevê ela como verdadeira;
- FN (falso negativo): Quando uma instância é verdadeira e o classificador identifica ela como falsa.

2.3.1.1 Acurácia

A acurácia é a métrica que calcula a quantidade de resultados que foram classificados corretamente. O cálculo desta métrica é dada pela seguinte equação:

$$\text{acurácia} = \frac{TP + TN}{TP + FP + TN + FN} \tag{2.1}$$

2.3.1.2 Precisão

A precisão é a métrica que calcula a quantidade de resultados que foram classificados como positivos são de fato positivos. Calculada pela função:

$$\text{precisão} = \frac{TP}{TP + FP} \tag{2.2}$$

2.3.1.3 Matriz de Confusão

Uma matriz de confusão é uma matriz de duas dimensões que representa todos resultados (TP, TN, FP e FN) de um classificador. A Figura 4 apresenta a estrutura de uma matriz de confusão simples.

Figura 4 – Uma matriz de confusão.

		Classe Verdadeira	
Classe Predita		Positivo	Negativo
	Positivo	TP	FP
	Negativo	FN	TN

Fonte: Adaptado de Fawcett (2006).

3 Metodologia

A primeira etapa para o desenvolvimento do trabalho presente foi o levantamento de materiais sobre *phishing* e técnicas e conceitos de aprendizado de máquina. A seguir, foi feita a escolha dos algoritmos a serem usados no projeto.

Após a escolha dos algoritmos, um conjunto de dados foi selecionado e, em seguida, o conteúdos desses dados foi analisado para que, então, ocorresse o treinamento dos modelos.

Por último, os resultados obtidos foram coletados e analisados.

A seguir estão os materiais e métodos usados para o desenvolvimento do trabalho presente.

3.1 Ferramentas

Nesta seção estão descritas todas as ferramentas utilizadas no desenvolvimento do trabalho.

3.1.1 Python

O Python é uma linguagem de programação interpretada que se tornou uma das linguagens mais populares em toda área de ciências de dados devido a sua simplicidade, consistência, grande comunidade ativa e por possuir diversas bibliotecas e *frameworks* para desenvolvimento na área.

A principal razão que levou a escolha dessa linguagem para o desenvolvimento do trabalho são as bibliotecas de aprendizado de máquina que ela possuí. As bibliotecas utilizadas foram:

- **Scikit-Learn:** Scikit-Learn é uma biblioteca de código aberto voltada para o aprendizado de máquina. Ela possui um vasto conjunto de métodos para classificação, regressão, pré-processamento de dados e métricas de desempenho (KRAMER, 2016).
- **Pandas:** Criada em 2008, Pandas é uma biblioteca de análise e manipulação de dados em Python. Suas grandes vantagens estão em sua flexibilidade, fácil uso, alto desempenho e ser código aberto (MCKINNEY, 2011). Essas características levaram ao Pandas se tornar uma ferramenta essencial em toda área de ciência de dados e Inteligência Artificial.
- **XGBoost:** A biblioteca XGBoost foi utilizada para implementar o XGBClassifier, o classificador XGBoost escolhido no desenvolvimento do trabalho.

- **Matplotlib:** Matplotlib é uma biblioteca para visualização de dados. Com ela é possível criar e exportar gráficos 2D em poucos comandos, sendo muito utilizada na área de aprendizado de máquina (BARRETT et al., 2005).

3.1.2 Jupyter Notebook

Um *notebook* Jupyter é um ambiente de programação interativa que mistura códigos executáveis e elementos textuais como imagens, equações ou parágrafos de texto. Além de ser uma ferramenta de código aberto, o Jupyter possui suporte a diversas linguagens de programação como, R, Julia, Python, etc. (BARBA et al., 2019).

Utilizou-se *notebooks* Jupyter para todo o desenvolvimento deste trabalho.

3.1.3 SHAP

SHAP é uma ferramenta usada para interpretar modelos de aprendizado de máquina, onde é dado um valor de relevância para cada característica do conjunto de dados para determinar sua influência no classificador. A partir dela, é possível analisar, por meio de gráficos, o impacto que cada característica teve no modelo criado.

O SHAP for utilizado para analisar o modelo que obteve o melhor resultado.

3.2 Conjunto de Dados

Para a realização do treinamento e teste dos modelos utilizou-se o conjunto de dados apresentado por Vrbančič, Jr e Podgorelec (2020). O conjunto possui no total 88,647 entradas de dados. Sendo elas:

- 30.647 páginas rotuladas como *phishing* (valor 1)
- 58.000 páginas rotuladas como legítimas (valor 0)

3.2.1 Características do Conjunto

Os dados deste conjunto possuem 112 características, sendo uma delas o atributo *phishing*. Essas características podem ser divididas em três grupos:

- Atributos baseados no endereço completo: As características desta parte do conjunto têm como base o endereço completo da página, sendo alguns desses atributos apresentados no Quadro 1.

Quadro 1 – Exemplo de atributos baseados em todo endereço

Atributo	Descrição	Formato
qty_comma_url	Número de ',' no endereço	Inteiro
qty_asterisk_url	Número de '*' no endereço	Inteiro
length_url	Número de caracteres no endereço	Inteiro
email_in_url	Se um e-mail está presente no endereço	Booleano

Fonte: Elaborado pelo autor.

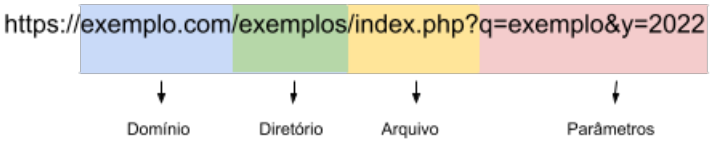
- Atributos baseados em uma parte do endereço: Exemplificado na Figura 5, o endereço de uma página é dividido em quatro partes: domínio, diretório, arquivo e parâmetros. Os atributos deste grupo têm como base uma parte do endereço. O Quadro 2 apresenta alguns desses atributos.

Quadro 2 – Exemplo de atributos baseados em uma parte do endereço

Atributo	Descrição	Formato
qty_dot_domain	Número de '.' no domínio	Inteiro
qty_plus_domain	Número de '+' no endereço	Inteiro
domain_in_ip	Se o domínio está no formato de um endereço IP	Booleano
qty_comma_directory	Número de ',' no diretório	Inteiro
directory_length	Número de caracteres no diretório	Inteiro
file_length	Número de caracteres no arquivo	Inteiro
qty_params	Número de parâmetros	Inteiro

Fonte: Elaborado pelo autor.

Figura 5 – Separação de um endereço



Fonte: Adaptado de Vrbančič, Jr e Podgorelec (2020).

- Atributos baseados em métricas externas e do endereço: Os atributos aqui têm como base elementos como a existência de certificados de segurança no endereço e se a página está indexada no Google Search Index. Exemplos desses atributos estão evidenciados no Quadro 3, com o atributo de classificação do modelo, o *phishing* incluso.

Quadro 3 – Exemplo de atributos baseados em métricas externas e do endereço

Atributo	Descrição	Formato
url_shortened	Se o endereço está encurtado	Booleano
tls_ssl_certificate	Se possui certificados tls/ssl válidos	Booleano
time_domain_activation	Tempo em que o domínio está ativo (em dias)	Inteiro
url_google_index	Se o endereço está indexado no Google	Booleano
phishing	Se o endereço é rotulado como phishing	Booleano

Fonte: Elaborado pelo autor.

3.3 Tratamendo dos Dados

Foram necessários alguns ajustes para a utilização do conjunto de dados escolhido.

Para tratar dos dados ausentes no conjunto de dados escolhido, utilizou-se a classe SimpleImputer do Scikit-Learn. Esta classe substitui os valores ausentes usando uma estatística descritiva dos dados de cada atributo, como média, mediana ou valor mais frequente. A mediana foi escolhida para o desenvolvimento deste trabalho.

Após o processamento de dados ausentes, foi feita a normalização de todos atributos do conjunto de dados. O objetivo da normalização é colocar todos os dados em uma mesma escala numérica. Aplicou-se a classe MinMaxScaler, também do Scikit-Learn, para reduzir os dados a uma escala de 0 a 1.

3.4 Desenvolvimento dos Modelos

Com o conjunto de dados selecionado, realizou-se a etapa de treinamento e teste dos modelos. Para evitar problemas de sobre-ajuste, os dados foram separados em dados de treino e dados de teste. Sendo 70% para treinamento e 30% para testes ou 62052 e 26595 entradas respectivamente.

Para cada modelo foram realizados diversos testes para achar os parâmetros que resultam nos melhores resultados possíveis. Durante os testes na árvore de decisão, o parâmetro de mais significância que apresentou o melhor resultado foi o *max_leaf_nodes* com 600 nós. Nos algoritmos de Floresta Aleatória, Extra-Tree e XGBoost o número de árvores, o parâmetro *n_estimators*, foi o mais importante, e decidiu-se que cada modelo ficaria, respectivamente, com 400, 550 e 600 árvores.

4 Resultados

Neste capítulo serão apresentados e discutidos os resultados obtidos neste trabalho. As métricas de desempenho usadas foram as abordados na Seção 2.3.

Nos modelos desenvolvidos, uma classificação positiva significa um endereço de uma página de *phishing* e uma classificação negativa significa uma página legítima.

Para finalizar este capítulo será feito uma análise do modelo XGBoost por meio da biblioteca SHAP.

4.1 Árvore de Decisão

Apresentado na Tabela 1, o modelo de Árvore de Decisão obteve bons resultados, porém, devido as limitações do algoritmo em relação aos outros utilizados neste trabalho, teve resultado inferior aos demais, com 95.8% de acurácia e 93.9% de precisão.

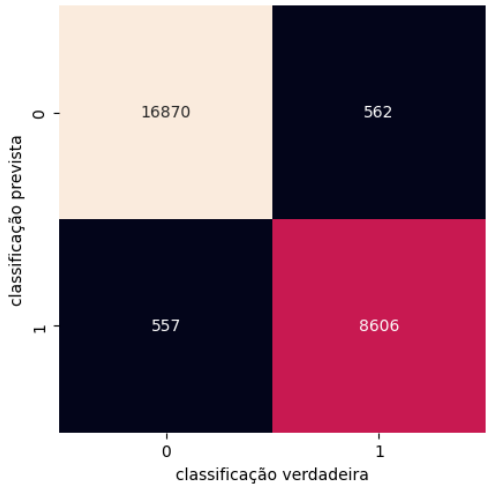
Tabela 1 – Resultados do modelo Árvore de Decisão

	Acurácia	Precisão
Árvore de Decisão	95.80%	93.92%

Fonte: Elaborado pelo autor.

A Figura 6 mostra a matriz de confusão do modelo de Árvore de Decisão. De todas as 26595 entradas de teste, apenas 562 foram classificadas como falsos negativos e 557 como falsos positivos.

Figura 6 – Matriz de confusão da Árvore de Decisão



Fonte: Elaborado pelo autor.

4.2 Floresta Aleatória

A Tabela 2 mostra que o Floresta Aleatória conseguiu resultados cerca de 1% melhor que o modelo de Árvore de Decisão, com 96.9% de acurácia e 95.4% de precisão. Com mais de 100 características no conjunto de dados, um algoritmo de Floresta Aleatória consegue generalizar os dados mais facilmente.

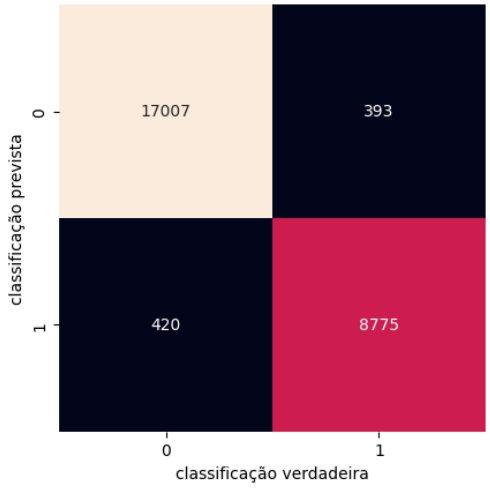
Tabela 2 – Resultados do modelo Floresta Aleatória

	Acurácia	Precisão
Floresta Aleatória	96.94%	95.43%

Fonte: Elaborado pelo autor.

A Figura 7 mostra a matriz de confusão do modelo Floresta Aleatória. É possível observar que a taxa de falsos negativos diminuiu mais do que a taxa de falsos positivos, em relação aos resultados do modelo de Árvore de Decisão.

Figura 7 – Matriz de confusão da Floresta Aleatória



Fonte: Elaborado pelo autor.

4.3 Árvores Extremamente Aleatórias

O modelo de Extra-Trees, ou Árvores Extremamente Aleatórias, é muito semelhante ao de Floresta Aleatória, podendo ser um pouco melhor ou pior dependendo da situação. Isso é refletido nos resultados obtidos, representado pela Tabela 3, que demonstra o Extra-Trees como um modelo levemente melhor que o Floresta Aleatória convencional.

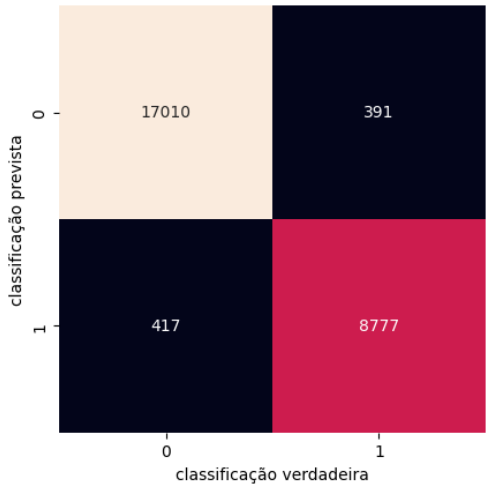
Tabela 3 – Resultados do modelo Extra-Trees

	Acurácia	Precisão
Extra-Trees	96.96%	95.46%

Fonte: Elaborado pelo autor.

A matriz de confusão do Extra-Trees, representada na Figura 8, mostra números extremamente semelhantes aos da matriz de confusão do modelo Floresta Aleatória.

Figura 8 – Matriz de confusão do Extra-Trees



Fonte: Elaborado pelo autor.

4.4 XGBoost

Os resultados desse modelo, apresentados na Tabela 4, revela que o XGBoost foi o modelo com melhor resultado, chegando a quase 97.5% de acurácia e 96.4% de precisão. Esse resultado pode ser explicado pelo fato do XGBoost ser uma implementação de um *gradient boosting* e também possuir otimizações adicionais para melhorar sua performance.

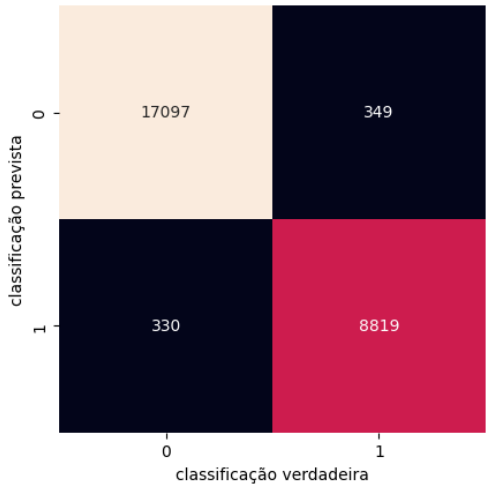
Tabela 4 – Resultados do modelo XGBoost

	Acurácia	Precisão
XGBoost	97.44%	96.39%

Fonte: Elaborado pelo autor.

A Figura 9 mostra a matriz de confusão do modelo XGBoost. Este modelo obteve 330 classificações do tipo falso positivo e 349 do tipo falso negativo, as menores taxas de todos os modelos desenvolvidos neste trabalho.

Figura 9 – Matriz de confusão do XGBoost



Fonte: Elaborado pelo autor.

4.5 Resultados Gerais

No Quadro 4 há todos os resultados obtidos dos modelos desenvolvidos neste trabalho. Todos modelos atingiram resultados satisfatórios, com o classificador XGBoost obtendo as melhores taxas de acurácia e precisão.

Quadro 4 – Resultado de todos os modelos

	Acurácia	Precisão
Árvore de Decisão	95.80%	93.92%
Floresta Aleatória	96.94%	95.43%
Extra-Tress	96.96%	95.46%
XGBoost	**97.44%**	**96.39%**

Fonte: Elaborado pelo autor.

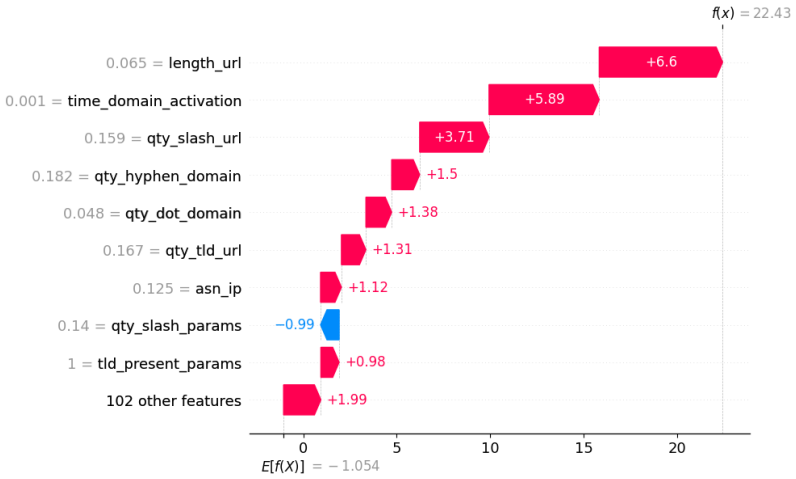
4.6 Análise do Modelo XGBoost

O SHAP é uma ferramenta que interpreta e explica, de maneira intuitiva, a saída de um modelo de aprendizado de máquina. Usando valores *Shapley*, o algoritmo busca explicar a predição de um modelo, computando o quanto cada característica do conjunto de dados contribuiu para esta predição (LUNDBERG, 2022).

O classificador escolhido para análise foi aquele que possui os melhores resultados, o XGBoost.

A Figura 10 apresenta o gráfico de uma classificação positiva do modelo, ou seja, uma página classificada como *phishing*. No eixo Y estão os atributos, ordenados por relevância, que levaram ao resultado.

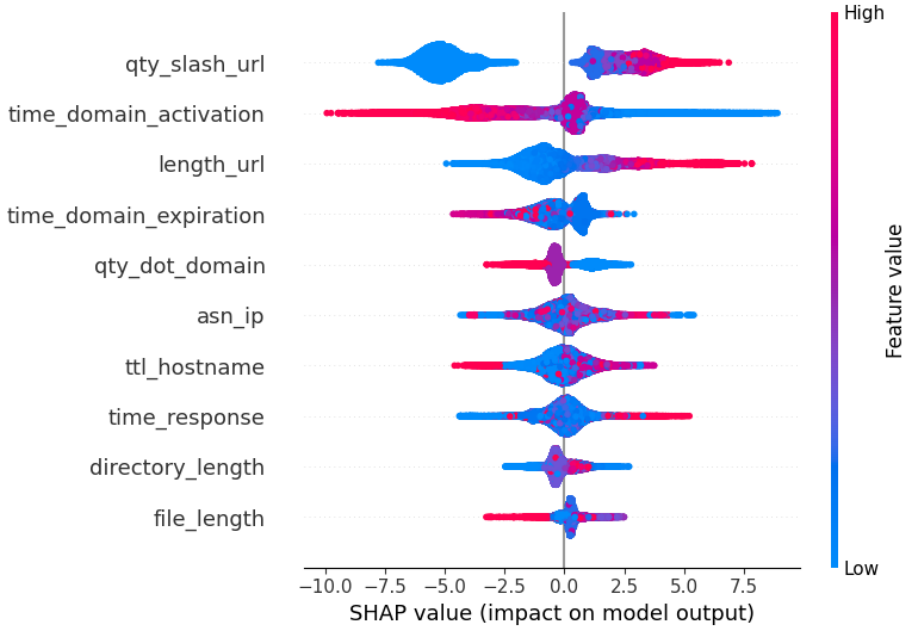
Figura 10 – Gráfico *waterfall* de uma classificação



Fonte: Elaborado pelo autor.

A Figura 11 mostra um gráfico com o sumário dos atributos mais importantes para o classificador. No gráfico, o eixo Y representa as características, por ordem de relevância. Cada ponto equivale a uma amostra dos dados, sendo sua cor uma representação do valor dela no atributo do eixo Y. No eixo X estão os valores SHAP, que determinam o impacto de cada ponto na classificação do modelo.

Figura 11 – Sumário dos atributos mais importantes



Fonte: Elaborado pelo autor.

Observa-se que os dois atributos mais importantes possuem uma relação forte entre sua cor e seu valor SHAP. Isso demonstra que, para o modelo, endereços que possuem menos "barras"e possuem um domínio ativo a mais tempo, são considerados páginas legítimas.

5 Conclusão

A digitalização transformou a sociedade em um mundo altamente conectado, a área de segurança de redes enfrenta um desafio cada vez maior. Milhares de crimes cibernéticos acontecem todos os dias e, com isso, cresce a necessidade de novos métodos e ferramentas para combater esses ataques.

No trabalho presente foi desenvolvido um sistema de detecção de páginas de *phishing* utilizando conceitos e técnicas de aprendizado de máquina.

Os objetivos propostos foram alcançados. Foi realizada uma pesquisa bibliográfica sobre as áreas relevantes ao trabalho e selecionou-se as técnicas e métricas a serem utilizadas. Em seguida, foram criados os modelos com as técnicas escolhidas e, por fim, coletou-se e analisou-se os resultados obtidos.

Os resultados alcançados foram promissores, com todos os classificadores apresentando uma alta taxa de precisão e acurácia, demonstrando sua eficácia em distinguir uma página legítima de uma falsa.

Trabalhos futuros com esta mesma abordagem podem envolver o uso de outras técnicas de aprendizado de máquina e detecção de ataques de *phishing*. Pode-se utilizar também um outro conjunto de dados mais amplo, para criar modelos que consigam generalizar o problema mais facilmente.

Referências

ABDELHAMID, N.; THABTAH, F.; ABDEL-JABER, H. Phishing detection: A recent intelligent machine learning comparison based on models content and features. In: *2017 IEEE International Conference on Intelligence and Security Informatics (ISI)*. [S.l.: s.n.], 2017. p. 72–77.

APWG. Phishing attack trends report – 2q 2022. 2022. Disponível em: <https://docs.apwg.org/reports/apwg_trends_report_q2_2022.pdf>. Acesso em: 05 jan. 2023.

BARBA, L. A.; BARKER, L. J.; BLANK, D. S.; BROWN, J.; DOWNEY, A. B.; GEORGE, T.; HEAGY, L. J.; MANDLI, K. T.; MOORE, J. K.; LIPPERT, D.; NIEMEYER, K. E.; WATKINS, R. R.; WEST, R. H.; WICKES, E.; WILLING, C.; ZINGALE, M. Teaching and learning with jupyter. *Recuperado: https://jupyter4edu. github. io/jupyter-edu-book*, 2019.

BARRETT, P.; HUNTER, J.; MILLER, J. T.; HSU, J.-C.; GREENFIELD, P. matplotlib–a portable python plotting package. In: *Astronomical data analysis software and systems XIV*. [S.l.: s.n.], 2005. v. 347, p. 91.

BROADBAND SEARCH. Key internet statistics to know in 2022 (including mobile). 2022. Disponível em: <https://www.broadbandsearch.net/blog/internet-statistics>. Acesso em: 05 jan. 2023.

CHEN, T.; GUESTRIN, C. Xgboost: A scalable tree boosting system. In: *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. [S.l.: s.n.], 2016. p. 785–794.

DHAMIJA, R.; TYGAR, J. D.; HEARST, M. Why phishing works. In: *Proceedings of the SIGCHI conference on Human Factors in computing systems*. [S.l.: s.n.], 2006. p. 581–590.

FAWCETT, T. An introduction to roc analysis. *Pattern recognition letters*, Elsevier, v. 27, n. 8, p. 861–874, 2006.

GEURTS, P.; ERNST, D.; WEHENKEL, L. Extremely randomized trees. *Machine learning*, Springer, v. 63, n. 1, p. 3–42, 2006.

IBM. X-force threat intelligence index 2022. 2022. Disponível em: <https://www.ibm.com/downloads/cas/ADLMYLAZ>. Acesso em: 05 jan. 2023.

JAMES, L. *Phishing exposed*. [S.l.]: Elsevier, 2005.

KHONJI, M.; IRAQI, Y.; JONES, A. Phishing detection: A literature survey. *IEEE Communications Surveys Tutorials*, v. 15, n. 4, p. 2091–2121, 2013.

KRAMER, O. Scikit-learn. In: *Machine learning for evolution strategies*. [S.l.]: Springer, 2016. p. 45–53.

LUNDBERG, S. *SHAP*. 2022. Disponível em: https://github.com/slundberg/shap. Acesso em: 05 jan. 2023.

MCKINNEY, W. pandas: a foundational python library for data analysis and statistics. *Python for high performance and scientific computing*, Seattle, v. 14, n. 9, p. 1–9, 2011.

MüLLER, A. C.; GUIDO, S. *Introduction to Machine Learning with Python*. [S.l.]: O'Reilly, 2016.

OGUNLEYE, A.; WANG, Q.-G. Xgboost model for chronic kidney disease diagnosis. *IEEE/ACM transactions on computational biology and bioinformatics*, IEEE, v. 17, n. 6, p. 2131–2140, 2019.

RASCHKA, S. *Python Machine Learning*. [S.l.]: Packt Publishing, 2015.

VRBANČIČ, G.; JR, I. F.; PODGORELEC, V. Datasets for phishing websites detection. *Data in Brief*, Elsevier, v. 33, p. 106438, 2020.

WHITTAKER, C.; RYNER, B.; NAZIF, M. Large-scale automatic classification of phishing pages. 2010.

ZHANG, T.; LIN, W.; VOGELMANN, A. M.; ZHANG, M.; XIE, S.; QIN, Y.; GOLAZ, J.-C. Improving convection trigger functions in deep convective parameterization schemes using machine learning. *Journal of Advances in Modeling Earth Systems*, Wiley Online Library, v. 13, n. 5, p. e2020MS002365, 2021.