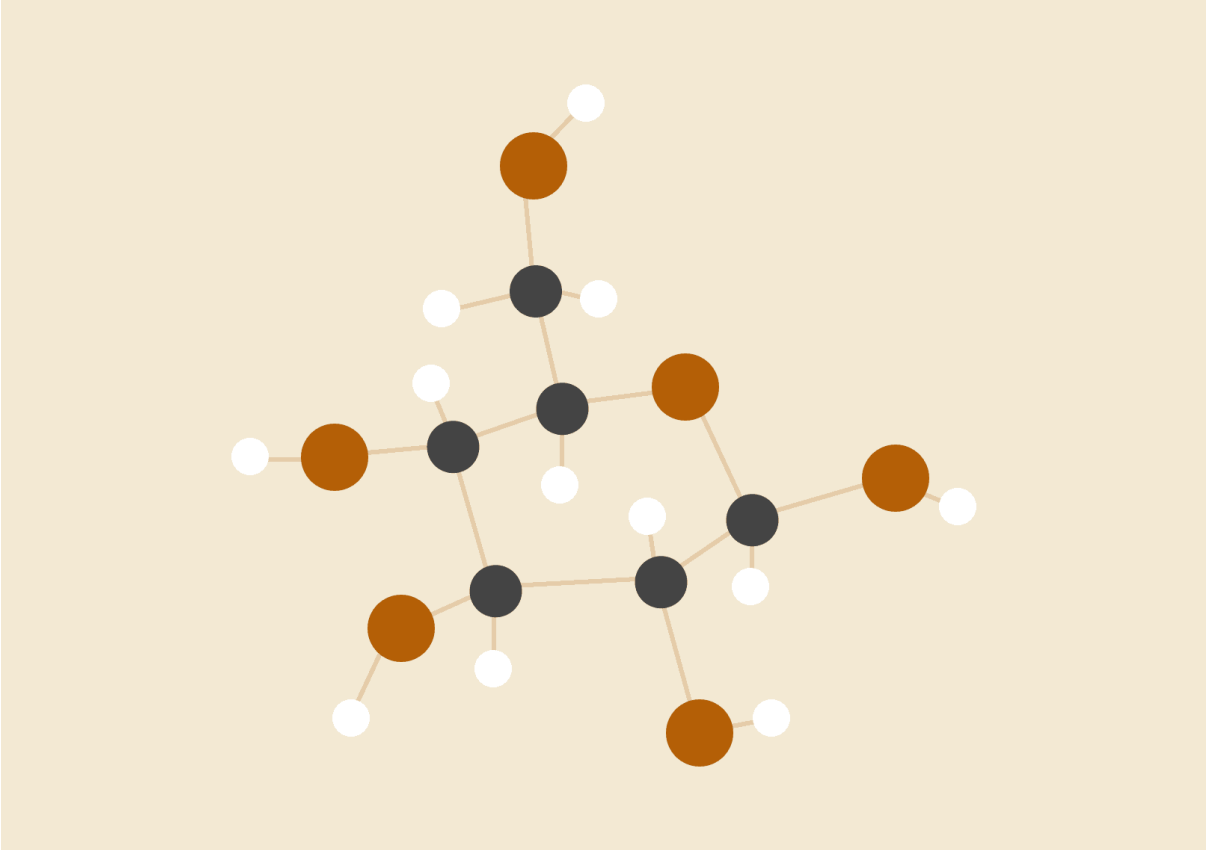
PROJECT WORK

PHISHING WEBSITE DETECTION



Jelena Jakovljevic, Madina Zhakina, Abdulrahman Zebn

SOFTWARE PACKAGES FOR STATISTICS

INTRODUCTION

The topic of this research is Phishing website detection. A phishing website is a domain similar in name and appearance to an official website. They're made in order to fool someone into believing it is legitimate and try to steal any data such as personal data, card or account information.

The purpose of our model is to determine the legality of the site by analyzing characteristics. To accomplish this, the model is trained on a dataset of labeled websites, where each website is represented by a set of features such as the length of URLs, the presence of uncharacteristic characters or the replacement of Latin letters with Cyrillic. Once trained, the model can take in the feature set of a new website and predict whether it is phishing or legitimate.

BACKGROUND RESEARCH

	Column names	Description
1	having_IPhaving_IP_Address	using or existence of IP address in URL
2	URLURL_Length	if the length of the URL is greater than or equal 54 characters then the URL classified as phishing
3	Shortining_Service	a necessary web page can still be accessed using a shorter URL
4	having_At_Symbol	a legal website can’t contain @ symbol
5	double_slash_redirecting	an URL starts with “HTTPS” then the “//” should appear in 7th position
6	Prefix_Suffix	a dash character is rarely used in legitimate URL
7	having_Sub_Domain	if the number of dots is greater than one, then the URL is classified as suspicious, if the dots are greater than two, it is classified as phishing
8	SSLfinal_State	an existence of SSL(Secure Sockets Layer) and information about certificate signed with HTTPS
9	Domain_registeration_length	time of domain expiration should be more than 1 year
10	Favicon	a web page is probably a Phishing attempt if the favicon loads from a different domain than what is displayed in the URL bar
11	port	firewalls block all or most of the ports and only open the ones selected
12	HTTPS_token	using of the “HTTPS” token in the domain part is a attacker's trick
13	Request_URL	most of the objects that are embedded on legal websites share the same domain as the webpage address
14	URL_of_Anchor	checking whether anchor link to any webpage
15	Links_in_tags	percentage of tags in the URL must be less than 17% of the total address

16	SFH	SFHs with "about:blank" or an empty string are viewed with suspicion
17	Submitting_to_email	using "mail()\" or "mailto:\" Function to Submit User Information
18	Abnormal_URL	a presence of a URL in a large WHOIS database
19	Redirect	a number of Redirect Page not should be more than 1
20	on_mouseover	onMouseOver event in source code might show a fake URL
21	RightClick	checking whether right-clicking is enabled so that users can view and save the source code of a web page
22	popUpWidnow	using pop-up window to submit personal data is suspicious
23	Iframe	using of the “iframe” tag and hide frame borders so user doesn't know even that is a different window
24	age_of_domain	from WHOIS database it’s easy to know an age of domain and it should be more than 6 month
25	DNSRecord	DNS record shouldn’t be empty or not found
26	web_traffic	this feature measures the popularity of the website in Alexa database
27	Page_Rank	measure how important a web page is on the Internet
28	Google_Index	examines whether a website is in Google’s index or not
29	Links_pointing_to_page	a number of links pointing to the webpage indicates its legitimacy level
30	Statistical_report	using statistical records to identify fishing website

There are several theories about why phishing is so effective. One theory is that phishing takes advantage of the human tendency to trust familiar branding and logos. Another theory is that phishing takes advantage of people's trust in their friends and acquaintances, as phishing emails and messages often appear to be from someone the victim knows.

There are several approaches to detect phishing websites, the first one is rule-based systems that are based on predefined rules and heuristics, these systems are easy to implement but they can be easily bypassed by phishers. The second one is based on machine learning. These systems are able to learn from examples and can adapt to new phishing patterns, but they require a large labeled dataset to be trained. Another approach is based on the use of browser extensions, these systems can block phishing websites when they are accessed by the user but they are not able to detect new phishing websites.

The most recent and promising approach is the use of deep learning algorithms, these systems can learn from features of the website such as the website's content, structure, and the URL, these systems are able to detect new phishing websites and have high accuracy rates, but they require a large dataset to be trained.

METHODS

First of all, the data exploration code part we imported needed libraries to work with data. Then read the CSV file into a pandas DataFrame. We can see some information about our dataset with functions like head(), columns and shape. Unnecessary data as index we drop from our dataFrame. plot_data_hist() function represents the number of each unique value contained in each column. We can see unique variables of each column. The data contained in these columns are simple and represented in integer format as -1, 0 and 1. -1 means phishing site, 0 means suspicious and 1 is a legitimate one. So we can say that data in this dataset is pretty much balanced, so there is no meaning to remove outliers. However, we can compute correlation values between two phishing website characteristics and make a correlation matrix. For that, I use Cramer's V correlation method for two categorical values: firstly, I convert columns to categorical type and initialize a CamresV object using a dataset, then use fit() function to return correlation matrix. To implement a graphic variant I use the heatmap() function from seaborn library. Also, the upper triangular part of the matrix displays identical information in the lower triangular part, so we can cut off the unnecessary part of the matrix using function triu() function from numpy library that returns a copy of an array with the elements below the k-th diagonal zeroed and ones_like() function that returns an array of ones with the same shape and type as a given array. The last step is to remove high correlated data. In my dataset there were double_slash_redirecting, HTTPS_token, HTTPS_token, port, popUpWidnow, Submitting_to_email, which had a correlation coefficient above 75 percent. We cleaned our data frame for modeling part by drop() function. And splitted data to prepare for models.

Secondly, for Decision Tree code part,

SPLITTING DATASET INTO TRAIN AND TEST SET

We used a test size of 0.2% to split our data.

```
#Splitting data
X = df.drop('Result', axis = 1)
y = df['Result']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

TRAINING THE MODEL

from sklearn.metrics we imported classification_report, confusion_matrix, accuracy_score.

We use 2 models or Algorithms to compare our data, firstly:

DECISION TREE

```
import matplotlib.pyplot as plt
from sklearn.datasets import make_classification
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import accuracy_score
```

These are the decision tree libraries that we used.

```
# Decision Tree
parameters = {'max_depth' : range(1,10),
              'min_samples_leaf': range(1,5)
             }
```

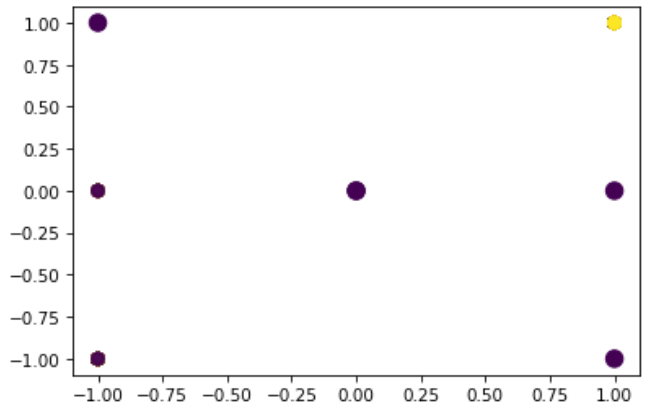
We used max_depth range (1, 10) And min_samples_leaf range (1,5) To find best parameter for decision tree because if we didn't put the parameter, this tree go very deep and the data will be overfitted.

Thirdly, for KNN model code part,

1. Select only the first two columns of the data (This is used for visualization purposes)
2. Plots a scatter plot of the data where each point in the plot represents a sample of the data and its position on the x and y axis corresponds to the values of the first and second columns of the data The color of the points in the scatter plot corresponds to the labels of the samples.
3. Defining a new point and plots it in the same scatter plot
4. Calculating the distance between the new point and all the points in the data. These distances are used to find the k closest points to the new point. The labels of these closest points are used to predict the label of the new point.
5. The code also plots the new points with a color that corresponds to their predicted label.

RESULTS

After cleaning, visualizing, testing and training our data, we used 2 algorithms to determine what our best course of action would be. We started with a decision tree model which gave us an accuracy of Training set accuracy: 95.4%. Test set accuracy: 94% For KNN algorithm result is



DISCUSSION AND CONCLUSIONS

Overall, phishing website detection is a challenging problem, as phishers are constantly coming up with new ways to evade detection. However, by understanding the theories behind why phishing is effective and using a combination of different approaches, it is possible to develop models that can effectively detect phishing websites.

The results of the KNN model are highly dependent on the selection of the value of k, which determines the number of nearest neighbors that was used to make the prediction. The results was affected by the choice of distance metric used to determine the similarity between data points. In general, KNN is a simple but powerful algorithm that can provide good results in detection of phishing websites.