Phishing Website detection using Deep Neural Network (NLP)

A typical Phishing Attack!

A victim opens a compromised link that poses as a credible website. The victim is then asked to enter their credentials, but since it is a “fake” website, the sensitive information is routed to the hacker and the victim gets ”‘hacked.”

How to identify a phishing URL?

- ✓ Is it misspelled?
- ✓ Is it Pointed to the wrong top-level domain?
- ✓ Does one part seems real and one part seems fake?
- ✓ Is it incredibly long?
- ✓ Is it just an IP address?
- ✓ Has a low PageRank?
- ✓ Has a young domain age?
- ✓ Ranks poorly on the Alexa Top 1 Million Sites?

Two different approaches were tried out for the project:

1. Deep Neural Network

- Using a pre-processed dataset from Kaggle.
- To compare the performance of the model, Logistic Regression and Multilayer perceptron have been applied.

2. Deep Neural Network + NLP

- By focusing just on address bar, a NLP model has been applied

What does URL features extraction mean? And why we have used preprocessed Dataset?

Address Bar-Based

- Adding a prefix or suffix separated by (-) to the domain
- Having sub-domain and multi-sub-domains
- Existence of HTTPS
- Domain registration age
- Favicon loading from a different domain
- Using a non-standard port

Abnormal Features

- Loading images loaded in the body from a different URL
- Minimal use of meta tags
- The use of a Server Form Handler (SFH)
- Submitting information to email

HTML and JavaScript-Based Features

- Website forwarding
- Status bar customization typically using JavaScript to display a fake URL
- Disabling the ability to right-click so users can't view page source code
- Using pop-up windows

Domain-Based Features

- Unusually young domains
- Suspicious DNS record
- Low volume of website traffic
- PageRank, where 95% of phishing webpages have no PageRank
- Whether the site has been indexed by Google

A dataset with 0 and 1!

	Domain	Have_IP	Have_At	URL_Length	URL_Depth	Redirection	https_Domain	TinyURL
0	eevee.tv	0	0	0	4	0	0	0
1	appleid.apple.com-sa.pm	0	0	0	1	0	0	0
2	grandcup.xyz	0	0	0	0	0	0	0
3	villa-azzurro.com	0	0	0	1	0	0	0
4	mygpstrip.net	0	0	0	2	0	0	0

Figure 1.1: Some extracted features with their values

First Approach:

Using a Deep Neural Network

The pre-processed dataset from kaggle consisted of 17 features with 10000 URLs

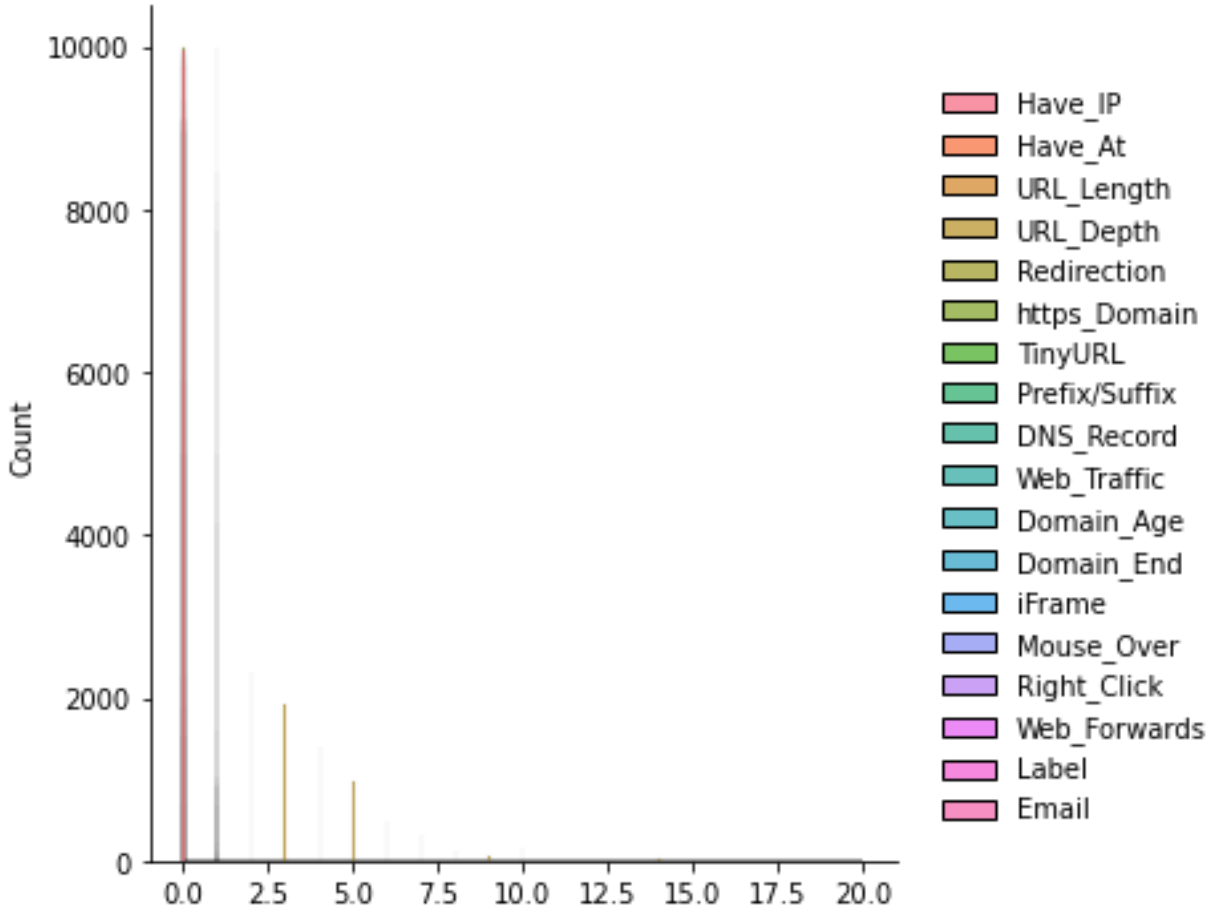

Figure 1.2: Graphical representation of the Features

- Model was trained with Logistic Regression and Multilayer perceptron,

- Training accuracy: 86%
- Test accuracy: 85%
- But the model was not really learning so much!

- Next step was to find the problem and increase the training and test accuracy

- In order to do this more features needed to be extracted from the URL's

- Two more features were added to the dataset namely the existence of subdomains and submitted to email.

- Still the training accuracy of the model was **not** increased

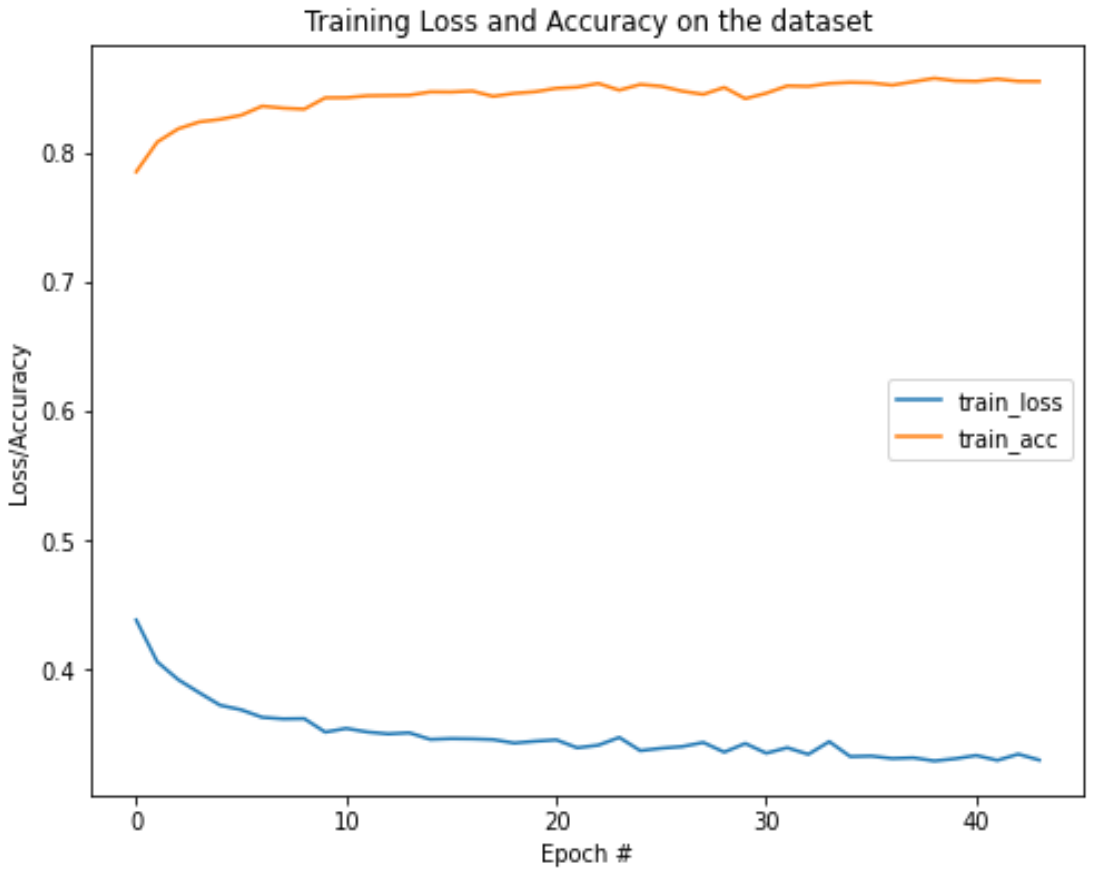

Figure 1.3: Training loss and accuracy on the dataset

Reason for the non-increasing accuracy was :
Most of the Feature values are binary and the model was not learning from them

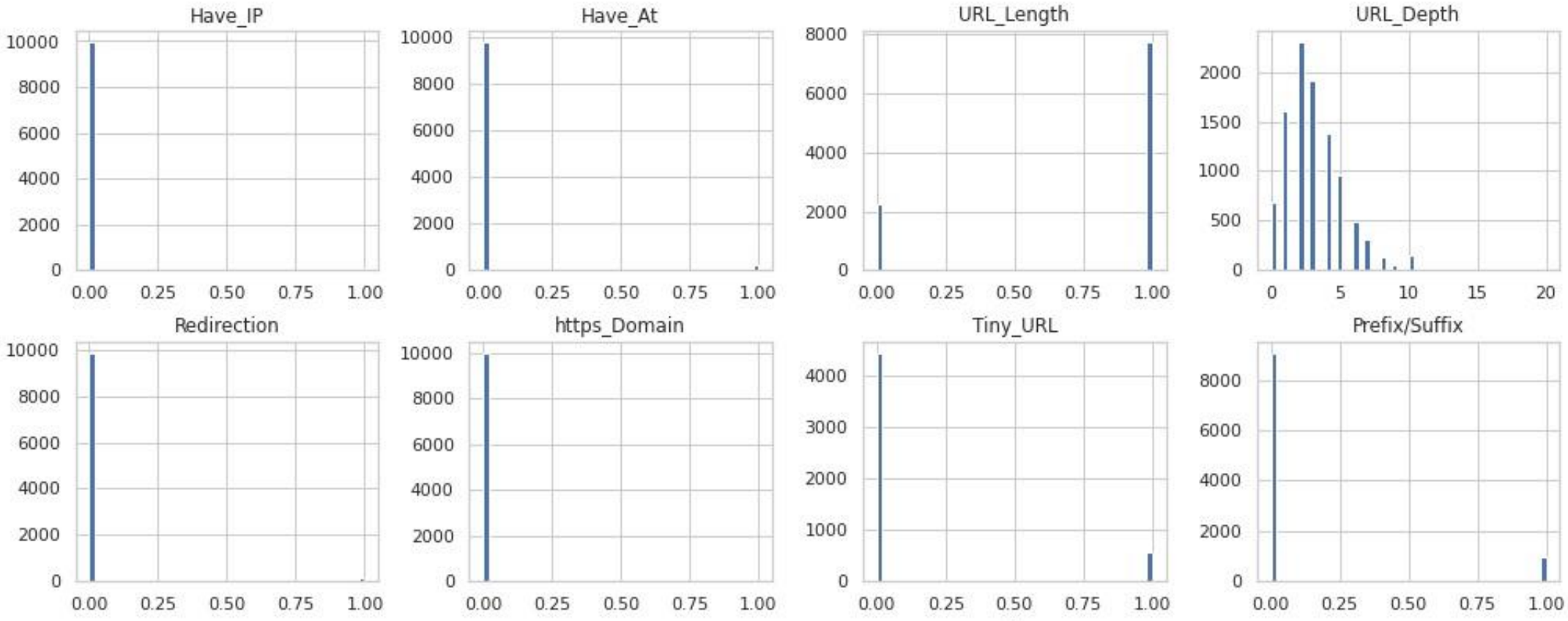

Figure 1.4: Plots of the training data distribution

Second approach: Deep Learning Neural Network + NLP

This **NLP** or Natural Language Processing model emphasis more on “words/phrases” and “how they are grouped together” in a URL, different to the previous model which depend on specific features of URL’s.

This model make use of “Tokenization” which will separate a piece of text in small units of tokens.

A “token” could be a word, a character or sub words.

This approach was able to obtain:

- Training accuracy of 98%
- Testing accuracy of 97%