GNR 652 Course Project

Classifying Phishing & Spam URLs through various ML methods

•••

June 5, 2020

Ritij Saini
17D070027

Karan Chate
17D070023

Himanshu Singh
170110076

Problem Statement:

To use machine learning to classify Spam/Phishing URLs using various ML methods. This project implements various ML models to create a spam classifier using primarily the url string as features.

Introduction

- Spam websites are great problem nowadays, getting in consider the frequency of internet activities
- **"Spam technology"** is advancing and has opportunities to keep site high ranks in search results
- The motive for detection by URL is reducing visiting opportunities to some pages based on a URL to predict that it is harmful or not.

Types of Spams URLs included in this project

Phishing

Identity theft (personal data), most often via special email or chat. These days many IIT Bombay Profs are encountering this type of threats.

Malware

Software which is intended to cause damage to computer and computer networks

Defacement

Change the look (content) existing web pages to steal username passwords of their original accounts

Brief Description of the process:

- Processing URLs for retrieval significant information
- Text data processing via specialized algorithms - CountVectorizer, Multinomial Naive Bayes
- Classification using algorithms Linear SVM, SVM with RBF kernel, RandomForest

DATA SET DESCRIPTION:

1. The data set consists of about 89,000 URLs, of which good (not harmful) make up about 40%
2. The dataset is obtained from the Canadian site Institute for Cybersecurity
3. **Benign URLs:** Over 35,300 benign URLs were collected from Alexa top websites. The domains have been passed through a Heritrix web crawler to extract the URLs.
4. **Spam URLs:** Around 12,000 spam URLs were collected from the publicly available WEBSPAM-UK2007 dataset.
5. **Phishing URLs:** Around 10,000 phishing URLs were taken from OpenPhish which is a repository of active phishing sites.
6. **Malware URLs:** More than 11,500 URLs related to malware websites were obtained from DNS-BH which is a project that maintain list of malware sites.
7. **Defacement URLs:** More than 45,450 URLs belong to Defacement URL category. They are Alexa ranked trusted websites hosting fraudulent or hidden URL that contains both malicious web pages.

1.1 Processing Deep part of URLs

Each deep url was split into words by using various characters as delimiters, such as _, ., :, =, ;, and /. The full regex is shown below:

'| \. | \ / | \ / \ / |: | - | _ |% | \? | = |; | <|> | ~ | \ $ | & | \ +'

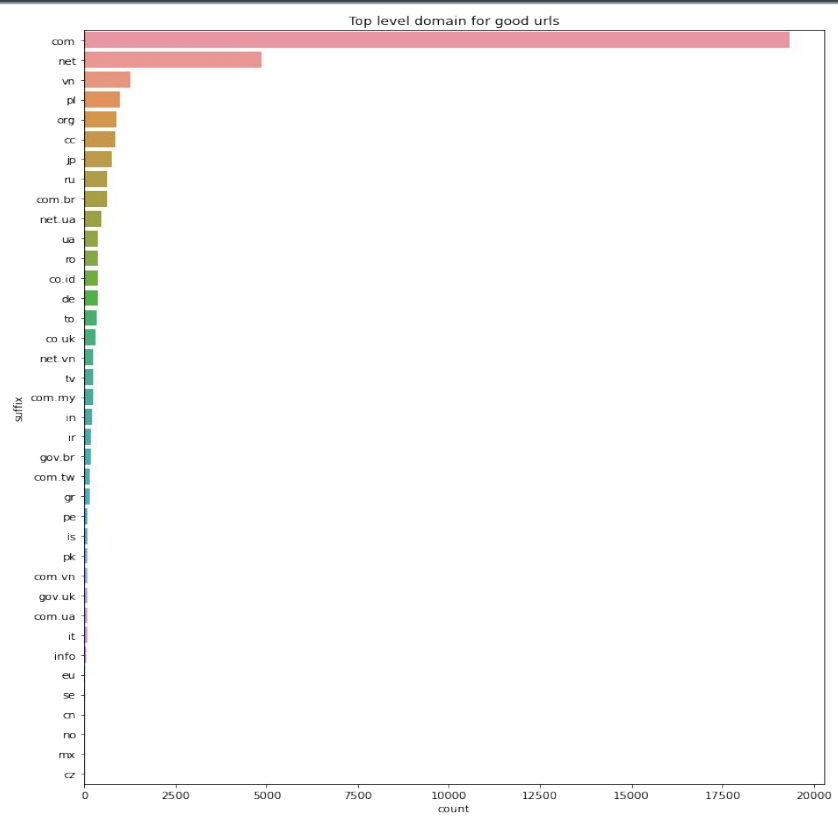
Any characters preceding the domain name were discarded

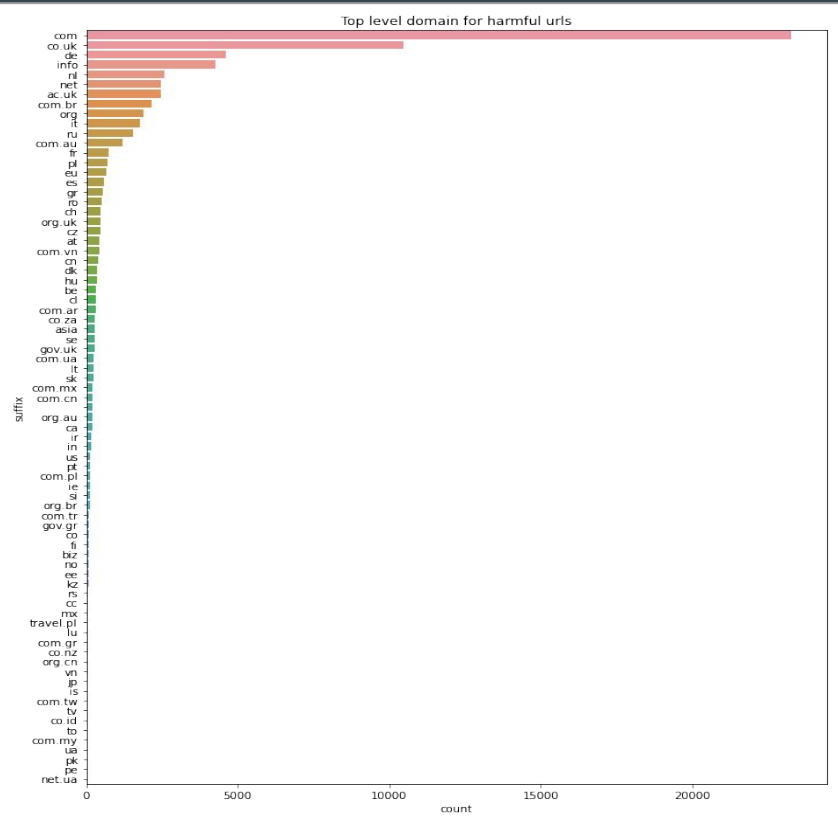
1.2 Processing Deep part of URLs

- Words in deep URL were kept if they were more than 2 characters long, otherwise they were ignored.
- A multinomial event model was fitted on these words with Laplace smoothing and a dictionary limited to words that appeared with a frequency of at least 0.1% times the size of the dataset.
- The output probabilities from above classifier was then used as a feature.

In next two slides TLD = Top level domain as shown in 7th slide

1.1 Top Level Domain Distribution





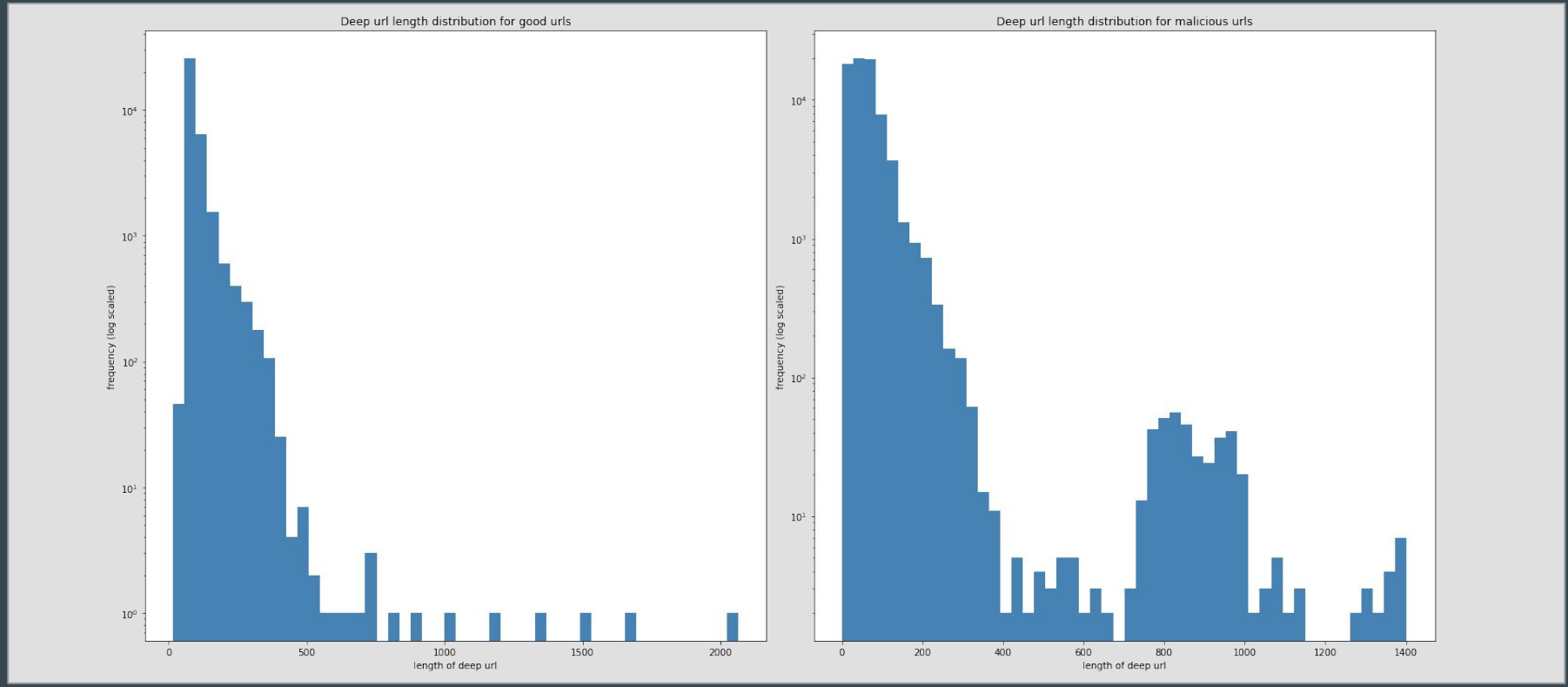
1.2 Top Level Domain Distribution

A. Top-level tld was encoded as a categorical variable with the distributions shown in previous slide for spam and not spam urls.

B. While ".com" is by far the most common tld in the data, we can see that the relative frequencies of other TLDs varies across the spam vs. not spam datasets for spam and not spam urls.

```
g = plt.figure(figsize=(20,12))
fig.add_subplot(1,2,1)
plt.title('Top level domain for good urls')
sns.countplot(y = final_ds_copy[final_ds_copy['label']==0]['suffix'], order = final_ds_copy[final_ds_copy['label']==0]['suffix'].value_counts().index)
fig.add_subplot(1,2,2)
plt.title('Top level domain for harmful urls')
sns.countplot(y = final_ds_copy[final_ds_copy['label']==1]['suffix'],order = final_ds_copy[final_ds_copy['label']==1]['suffix'].value_counts().index)
plt.tight_layout()
plt.show()
```

Deep URL length Distribution:



Methods

We have implemented the following ML techniques to obtain best results for our phishing URL classifier:

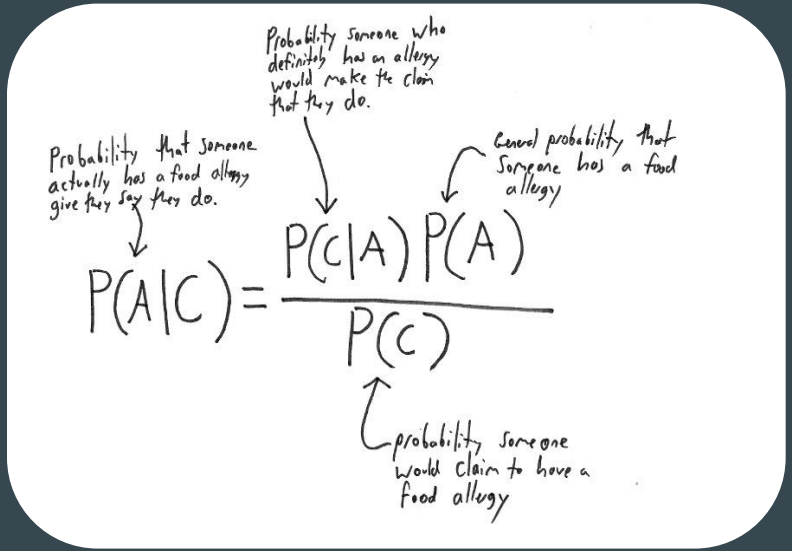
1) *Naive bayes classifier*
2) *Random Forest*
3) *SVM with RBF kernel*
4) *Neural Network*

The description and results of each of these methods are summarized in upcoming slides.

1) Naive bayes classifier

Naïve Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naïve) independence assumptions between the features. They are among the simplest Bayesian network models.

Train set accuracy of NB classifier: **97.17 %**
Test set accuracy of NB classifier: **96.17 %**



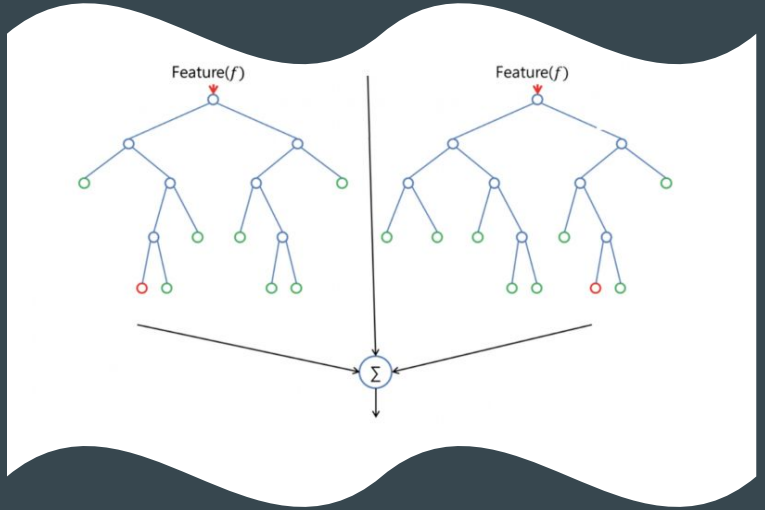
Confusion Matrix

	0	**1**
0	11044	631
1	747	23518

2) Random Forest

Random forest is a supervised learning algorithm. The "forest" it builds, is an ensemble of decision trees, usually trained with the “bagging” method. The general idea of the bagging method is that a combination of learning models increases the overall result.

Validation set accuracy of Random Forest: **99.71 %**
Test set accuracy of Random forest: **99.29 %**



Confusion Matrix

	0	**1**
0	11514	161
1	93	24172

15 Most Important features according to our random forest

('label_proba_1', 38.22204303617866)

('label_proba_0', 37.479463340831124)

('len deep url', 11.93243819523408)

('len of domain', 3.8430349893701)

('suffix_co.uk', 1.3278298714824788)

('suffix_com', 1.3138298906282377)

('suffix_net', 1.0667911267259529)

('suffix_vn', 0.541653935742985)

('suffix_info', 0.5059103352679813)

('suffix_jp', 0.49035731635762475)

('suffix_cc', 0.31053744732707816)

('suffix_tv', 0.29229789412264434)

('suffix_ua', 0.261701341904223)

('suffix_ac.uk', 0.24519333524004708)

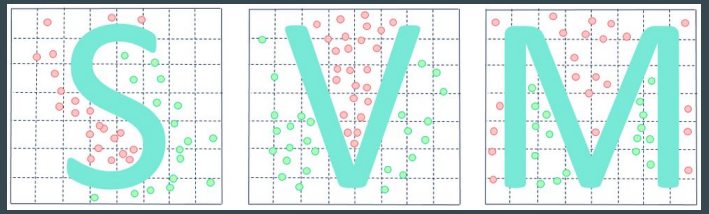
('suffix_nl', 0.23338296465245364)

Our Random forest has given us the best accuracy, far better from human level classification. These top 15 features are derived from the naive bayes probabilities and they depicts the overall dependence of modal on these feature.

To save computation we can eliminate the least important features. Running our random forest with these top 15 features can give an accuracy of above 90% on both Validation and test set.

3) SVM using RBF(Radial Basis Function) kernel

RBF is a popular Kernel method used in SVM models. RBF kernel is a function whose value depends on the distance from the origin or from some point. Gaussian Kernel is of the following format; $K(X_1, X_2) = exponent(-\gamma \|X_1 - X_2\|^2)$



Val set accuracy of SBM with RBF kernel: **99.38 %**
Test set accuracy of sVM with RBF kernel: **96.46 %**

Confusion Matrix

	0	**1**
0	10877	798
1	471	23794

Best SVM Parameters:

C=100

gamma= 0.001

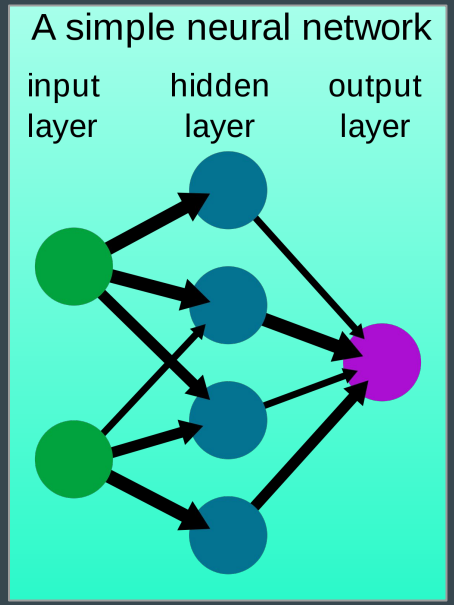
4) 4 Layer Neural Network

We have trained this network using PyTorch framework for over 50 epochs. Parameters are as follows:

Number of units in hidden layer 1: 250
Number of units in hidden layer 2: 200
Number of units in hidden layer 3: 150
Learning Rate: 0.001
Activation for hidden layers: ReLu
Activation for output: Softmax
For optimizer Stochastic gradient descent was used

After 45 Epoch

Train set accuracy of NN: **99.37 %**
Test set accuracy of NN: **98.56 %**



Random Forest without NB classifier probabilities as attributes

In the last part of our code, we had tried to implement Random Forest (our best classifier among our 4 methods) without naive bayes predicted probabilities as attributes. It was expected a drop in accuracy on test set and hence observed in our code's output.

Val set accuracy of RF without NB probabilities: **99.71 %**
Test set accuracy of RF without NB probabilities: **95.39 %**

Conclusion: This proves that naive bayes output probabilities were the most important feature. This is corroborated by the top 15 feature importance ranking from the Random Forest model as shown in previous slides.

Summary of above methods

	Test Accuracy (%)
Naive Bayes	96.17
Random Forest	99.29
SVM with RBF	96.46
4 layer Neural Network	98.56
Random forest without NB probability features	95.39

Conclusion:

A. Several classification algorithms were shown, of which Random forest and SVM with RBF kernel works the best.
B. The most important attributes are probability predictions from Naive bayes and when these attributes were removed then accuracy of Random forest dropped by 0.5% on test set.
C. Spam is a fast growing problem because it constantly discovers new ways even if we analyse them significantly but still they are becoming difficult to detect
D. Machine learning in this area has delivered significant improvement. Even our modal with 89k training examples has achieved an accuracy of 99%.

Individual Contribution

Ritij Saini, 17D070027	Karan Chate, 17D070023	Himanshu Singh, 170110076
Collected, analyzed & Manipulated Data and implemented Naive bayes and its probability attributes. Also, Made this Powerpoint Presentation.	Extracted and analysed top-level domain urls and implemented Random forest, SVM and Neural Network. Performed hyper-parameter search over the models using random-search.	Manipulated data and implemented various Random forest and made graphs related to that. Also described all the functions and its use.

NOTE

One of our team member **(Abhishek Acharya, 180020006) has decided to Drop this course** after “Closure of semester mail” from Dean AP.

Kindly evaluate this project as an effort made by a team of 3 members.

References

1. **Our Colab Notebook “View only” Link**: https://colab.research.google.com/drive/11L0xHPKa4d4CJelczw_ijux3pRggrtQJ?usp=sharing
2. Pedregosa, Fabian, et al. "Scikit-learn: Machine learning in Python." Journal of machine learning research 12.Oct (2011): 2825-2830.
3. Alexandros Ntoulas, Marc Najork, Mark Manasse, and Dennis Fetterly. 2006. Detecting spam web pages through content analysis. In Proceedings of the 15th international conference on World Wide Web (WWW '06). ACM, New York, NY, USA, 83-92. DOI: https://doi.org/10.1145/1135777.1135794
4. http://cs229.stanford.edu/proj2018/report/116.pdf

Thank you

For your attention!