Detecting Phishing URLs using Machine Learning

HTU UNIVERSITY

REEF AMARIN REEF.AMARIN@HTU.GOV.JO

Reef Amarin | Data Science | 13/09/2021

Contents

Summary

Security threats and attacks have been increased exponentially in last decade affecting businesses in a negative way in overall the world causing a damage and lose in important data lead to lose businesses, information and money as well.

One of big threat we are facing is phishing emails or URLs, what is phishing attack? And how we can detect it to protect us from it.

Phishing threat: is a type of social engineering attack often used to steal user data, including login credentials and credit card numbers. It occurs when an attacker pretended as a trusted authorized entity (ex. your bank), fools a victim into opening an email, instant message, or text message and send their information [1].

This report will discuss this problem by finding the best ML methodology that could give a high accuracy to detect any text whether it's a phishing URL or not.

Here is some Statistic about phishing attacks and their affect globally and in Middle East.

- A security research revealed that the Middle East suffered **more than 2.57 million phishing attacks** in the second quarter of 2020 [2]**.**
- 75% of organizations around the world experienced a phishing attack in 2020, and 74% of attacks targeting US businesses were successful. [3]

1. Introduction

This report will work on detecting phishing with different machine learning methodologies and techniques, to find the best model and techniques for detecting the phishing URLs.

Referring to previous works in this subject, *Tarun Tiwari* concluded that a logistics classifier is the best fitted classifier between different models he had used with accuracy about 95.6% [4].

He used a dataset of different URLs labeled if it is bad or good without cleaning and balancing it, contains about 500.000 records.

His outcome is based on his work without using (grid search) or balancing the data and cross validation techniques.

Other researchers and data scientists used a data set of different independent features that describe the URLs (having_IP,having_IP_Address, URL_Length, Shortining_Service, having_At_Symbol, double_slash_redirecting, Prefix_Suffix, having_Sub_Domain, SSLfinal_State…etc), and the dependent is result of 1 or -1 if phishing or not.

2. Methodology

2.1. EXPERIMENTAL/SAMPLING DESIGN

Since the problem is a classification problem then we will test models with supervised classification algorithms.

I have used four different classifier models to train and get the required results, using a cross validation and specific updated version grid search (HalvingRandomSearchCV) to obtain a high accuracy for which take less time than grid search and gives more performance than regular grid search.

Here below a brief about each model I have used.

Passive aggressive classifier:

The Passive-Aggressive algorithms are a family of Machine learning algorithms that are not very well known by beginners and even intermediate Machine Learning enthusiasts. [5]

However, they can be very useful and efficient for certain applications. [5]

Passive: If the prediction is correct, keep the model and do not make any changes. i.e., the data in the example is not enough to cause any changes in the model. [5]

Aggressive: If the prediction is incorrect, make changes to the model. i.e., some change to the model may correct it [5].

If you want to work on **big data**, this is a very important classifier and since our dataset is over than 500.000 and this is a huge amount, this classifier will be able to handle data of this size.

XGBoost Classifier:

The "**XGBoost**" algorithm has been triumphant in many Machine Learning competitions. [6]

Introduced in 2014 and since then it's been commended ever since. [6]

The beauty of this powerful algorithm lies in its adaptability, which drives quick learning through parallel and distributed computing and offers proficient memory usage [6].

- It is highly flexible

- It uses the power of parallel processing
- It is faster than Gradient Boosting
- It supports regularization
- It is designed to handle missing data with its in-build features.
- The user can run a cross-validation after each iteration.
- It works well in small to medium dataset [6]

Logistic Regression classifier:

- Logistics Regression Is a classification algorithm used to find the probability of event success and event failure.
- It is used when the dependent variable is binary (0/1, True/False, Yes/No) in nature.
- It supports categorizing data into discrete classes by studying the relationship from a given set of labelled data.
- It learns a linear relationship from the given dataset [7].

Stochastic Gradient Descent Classifier (SGD):

- Is one of the key algorithms used in training deep neural networks.
- SGD allows minibatch (online/out-of-core) learning. Therefore, it makes sense to use SGD for large scale problems where it's very efficient.
- The minimum of the cost function of Logistic Regression cannot be calculated directly, so we try to minimize it via Stochastic Gradient Descent, also known as Online Gradient Descent.
- **What is a loss/Cost function?** 'Loss' in Machine learning helps us understand the difference between the predicted value & the actual value [8].

2.2. DATA ANALYSIS

My dataset was about 500.000 records contains two fields, first field is the independent feature of free text to different format to URLs, and the second field was the dependent categorial label data stated whether the URL is (good or bad).

There isn't any missing data, so I didn't do any imputing or replacing the NULLs, while when comparing the bad URLs percentage (29%) to good URLs (71%) it shows that could considered as imbalanced data, so I preferred to do oversampling to make the modules more efficient.

3. Results

The best classifier I have used was PassiveAgressive classifier and it gave the highest balanced accuracy between all above with result = 98.9% as shown below in figure (1-1):

```
param_grid = {'C' : [0.003, 0.01, 0.03, 0.1], 'loss': ['hinge', 'squared_hinge'], 'max_iter': [5, 10, 30, 100, 300]}
clf = PassiveAggressiveClassifier(random_state = randomState, loss = 'squared_hinge', max_iter = 100, C = 0.01)
clf.fit(X_train, Y_train)
grid_search = GridSearchCV(
    estimator=clf,
    param_grid=param_grid,
    scoring = 'roc_auc',
    n_jobs = -1,
    cv = 2,
    verbose=True
)
grid_search.fit(X_train,Y_train)
pred = clf.predict(X_test)

print(f"Balanced accuracy score of a Passive Aggressive Classifier with Over sampling : {balanced_accuracy_score(Y_test,pred)}")
```

```
Fitting 2 folds for each of 40 candidates, totalling 80 fits
Balanced accuracy score of a Passive Aggressive Classifier with Over sampling : 0.9894509372988385
```

Figure (1-1)

Following the best classifier, the second one was Logistics classifier and it gave a good balanced accuracy between all above with result = 93.6% as shown below in figure (1-2):

```
scoring = ["accuracy", "balanced_accuracy"]
penalty = ['l2']
lr =LogisticRegression(max_iter=200)
C = np.logspace(0, 4, 10)
hyperparameters = dict(C=C, penalty=penalty)
logreg_cv=HalvingRandomSearchCV(lr,hyperparameters,cv=5)
logreg_cv.fit(X_res,y_res)
cv_result = cross_validate(lr, X_res, y_res, scoring = scoring)
print(f"Balanced accuracy score of a LogisticRegression  Classifier with Over sampling : {cv_result['test_balanced_accuracy'].mean():0.3f}")
```

```
Balanced accuracy score of a LogisticRegression  Classifier with Over sampling : 0.936
```

Figure (1-2)

The third classifier in accuracy was Stochastic Gradient Descent Classifier SGD, the second one was Logistics classifier and it gave a good balanced accuracy between all above with result = 88.8% as shown below in figure (1-3):

```
[ ]  1 params = {
     2     "loss" : ["hinge", "log", "squared_hinge", "modified_huber"],
     3     "alpha" : [0.0001, 0.001, 0.01, 0.1],
     4     "penalty" : ["l2", "l1", "none"],
     5 }
     6 scoring = ["accuracy", "balanced_accuracy"]
     7 SGD = SGDClassifier(max_iter=1000)
     8 logreg_cv=HalvingRandomSearchCV(SGD,params,cv=5)
     9 logreg_cv.fit(X_res,y_res)
    10 cv_result = cross_validate(SGD, X_res, y_res, scoring = scoring)
    11 print(f"Balanced accuracy score of a SGD  Classifier with Over sampling : {cv_result['test_balanced_accuracy'].mean():0.3f}")

Balanced accuracy score of a SGD  Classifier with Over sampling : 0.888
```

Figure (1-3)

The final and worst classifier between all proposed one is the XGboost with accuracy around 73.7 as shown below in figure (1-4)

```
 1 scoring = ["accuracy", "balanced_accuracy"]
 2 XG = XGBClassifier(learning_rate=0.02, n_estimators=100, objective='binary:logistic',
 3                     verbose=1)
 4
 5 params = {
 6      'max_depth':range(3,10,2),
 7      'min_child_weight':range(1,10,2)
 8 }
 9 grid_search = HalvingRandomSearchCV(estimator=XG, param_distributions=params, scoring='roc_auc', n_jobs=1,cv=2, verbose=True)
10
11 grid_search.fit(X_res, y_res)
12 grid_search.best_estimator_
13 cv_result = cross_validate(XG, X_res, y_res, scoring = scoring)
14 print(f"Balanced accuracy score of a XGboost Classifier with Over sampling : {cv_result['test_balanced_accuracy'].mean():0.3f}")

Fitting 2 folds for each of 3 candidates, totalling 6 fits
Balanced accuracy score of a XGboost Classifier with Over sampling : 0.737
```

Figure (1-4)

4. Discussion

After experiment and from results we obtained that even if the Logisitics classifier works with dependent variable is binary (0/1, True/False, Yes/No) in nature and gave the highest accuracy in previous studies but after balancing data and working with balanced accuracy in cross validation technique in my work the accuracy decreases to 2% lower than previous.

The best model in my result was PassiveAgressive classifier with balanced accuracy 98.9%, since I have tested it with balanced data and since I'm using a cross validation and because of that this model works very well with large scale dataset (big data) and since my dataset was big and increased to be 700.000 records after balancing the data the accuracy has been increased than model used before to be 98.9%.

The lowest accuracy has been resulted from using XGBoost and this was because this model is more efficient with small-mid-sized dataset.

5. Conclusion

If you want to work on big data, this **PassiveAggressive** is a best important classifier with 98.9% balanced accuracy.

The previous experiment from other data scientists with about 95.6% accuracy using logistics and it took too much
time comparing to my chosen one (**PassiveAggressiveClassifier**).

Since that phishing threats and security field is updated and changed year by year so there will be a huge amount of data coming (new
URLs) in every second and this **PassiveAggressive** will be able to handle data of this size.

6. References

[1] [Online]. Available: https://en.wikipedia.org/wiki/Phishing.

[2] [Online]. Available: https://kdmarc.com/blog/middle-east-hit-by-a-wave-of-phishing-attacks-in-q2-of-2020/ .

[3] [Online]. Available: 26 Phishing Attack Statistics To Keep In Mind In 2021 (softactivity.com).

[4] [Online]. Available: https://www.kaggle.com/.

[5] [Online]. Available: https://www.geeksforgeeks.org/passive-aggressive-classifiers/.

[6] [Online]. Available: https://datascience.foundation/datatalk/xgboost-an-efficient-implementation-of-gradient-boosting.

[7] [Online]. Available: https://www.geeksforgeeks.org/advantages-and-disadvantages-of-logistic-regression/.

[8] [Online]. Available: https://towardsdatascience.com/stochastic-gradient-descent-clearly-explained-53d239905d31.

[9] [Online]. Available: https://en.wikipedia.org/wiki/Phishing.

[10] [Online]. Available: https://www.kaggle.com/.