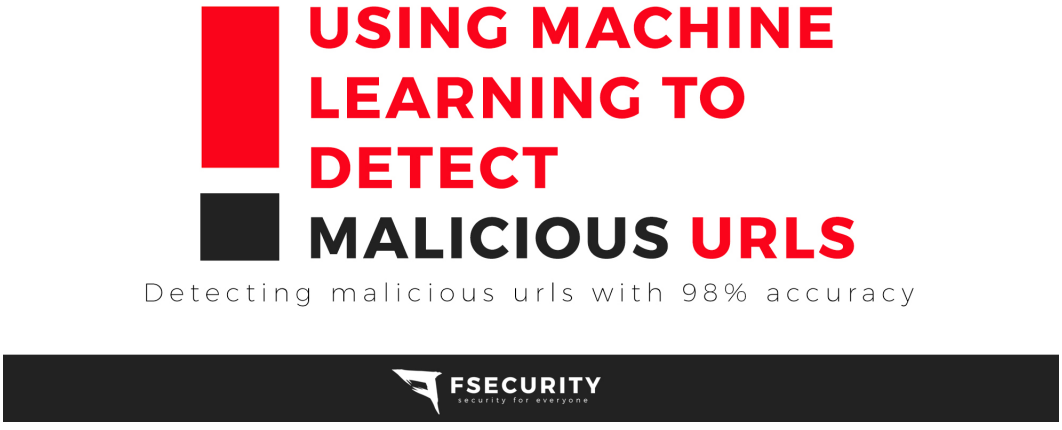
Using Machine Learning to Detect Malicious URLs

By Faizan Ahmad. Published on October 22, 2016. 16



With the growth of Machine Learning in the past few years, many tasks

FSECURIFY
security for everyone

SECURITY FOR EVERYONE

A few days ago, I had this idea about what if we could detect a malicious URL from a non-malicious URL using some machine learning algorithm. There has been some research done on the topic so I thought that I should give it a go and implement something from scratch. So lets start.

Machine Learning and Security | Using Machine Learning to detect Malicious URLs with 98% accuracy

Gathering Data

The first task was gathering data. I did some surfing and found some websites offering malicious links. I set up a little crawler and crawled a lot of malicious links from various websites. The next task was finding clear URLs. Fortunately, I did not have to crawl any. There was a data set available. Don't worry if I am not mentioning the sources of the data. You'll get the data at the end of this post.

So, I gathered around 400,000 URLs out of which around 80,000 were malicious and others were clean. There we have it, our data set. Lets move next.

Analysis

We'll be using **Logistic Regression** since it is fast. The first part was tokenizing the URLs. I wrote my own tokenizer function for this since URLs are not like some other document text. Some of the tokens we get are like *'virus','exe','php','wp','dat'* etc.

SECURITY FOR EVERYONE

```
            tokensByDot = tokensByDot + tempTokens
        allTokens = allTokens + tokens + tokensByDot
    allTokens = list(set(allTokens))    #remove redundant tokens
    if 'com' in allTokens:
        allTokens.remove('com') #removing .com since it occurs a lot
return allTokens
```

The next step is to load the data and store it into a list.

```
allurls = 'C:\\Users\\Faizan Ahmad\\Desktop\\Url Classification Projec
allurlscsv = pd.read_csv(allurls,',',error_bad_lines=False) #reading f
allurlsdata = pd.DataFrame(allurlscsv)  #converting to a dataframe

allurlsdata = np.array(allurlsdata) #converting it into an array
random.shuffle(allurlsdata) #shuffling
```

Now that we have the data in our list, we have to vectorize our URLs. I used **tf-idf** scores instead of using bag of words classification since there are words in urls that are more important than other words e.g *'virus', '.exe','.dat'* etc. Lets convert the URLs into a vector form.

```
y = [d[1] for d in allurlsdata] #all labels
corpus = [d[0] for d in allurlsdata]    #all urls corresponding to a l
vectorizer = TfidfVectorizer(tokenizer=getTokens)   #get a vector for
X = vectorizer.fit_transform(corpus) #get the X vector
```

We have the vectors. Lets now convert it into test and training data and go right about performing logistic regression on it.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.

lgs = LogisticRegression()  #using logistic regression
lgs.fit(X_train, y_train)
print(lgs.score(X_test, y_test)) #pring the score. It comes out to be
```

That's it. See, its that simple yet so effective. **We get an accuracy of 98%.** That's a very high value for a machine to be able to detect a malicious URL with.

Want to test some links to see if the model gives good predictions? Sure. Lets do it.

```
X_predict = ['wikipedia.com','google.com/search=faizanahad','pa
X_predict = vectorizer.transform(X_predict)
```

- google.com/search=faizanahad **(Good Url)**
- pakistanifacebookforever.com/getpassword.php/ **(Bad Url)**
- www.radsport-voggel.de/wp-admin/includes/log.exe **(Bad Url)**
- ahrenhei.without-transfer.ru/nethost.exe **(Bad Url)**
- www.itidea.it/centroesteticosothys/img/_notes/gum.exe **(Bad Url)**

This is what a human would have predicted. No?

The data and code is available at **Github**

That is it. I hope you enjoyed reading.

Your comments are most welcome.

16 Comments ADD YOURS

1

Josh Mathew on October 22, 2016 at 9:00 pm
Great read.

2

x on October 24, 2016 at 5:34 pm
Where is the dataset from? How was it created? What are its usage restrictions (if any)?

3

Ahmad on October 24, 2016 at 6:23 pm
taset was compiled after scraping various websites offering malicious links e.g vxvault.net. The good URLs were obtained from a link given in a research paper. That dataset was public. There are no usage restrictions.

Best Regards

SECURITY FOR EVERYONE

It appears like some of the text in your posts are running off the screen.
Can someone else please comment and let me know
if this is happening to them as well? This may be a issue with
my internet browser because I've had this happen before.
Thanks

5

Yuri on October 28, 2016 at 2:44 pm

I think method getTokens could be simplified using re.split which supports regular expressions.
It could be as simple as:
def getTokens(input):
tokens = set(re.split(r'[.-/]'))
tokens.pop('com')
return tokens

6

Ahmad on October 30, 2016 at 7:18 pm

s you for the suggestions Yuri.

Best Regards.

7

SeongKyu, Park on October 30, 2016 at 2:17 pm

Hi.
It is very interesting tool.
How to use it?
When I execute though command, it happens an error messge.
ImportError: No module named sklearn.feature_extraction.text
Please check it.
Thanks.

8

Kyu, Park on October 30, 2016 at 2:28 pm

I checked dependencies and resolved this problem.

Nice Work. I tried to run your script, but looks like 'train_test_split' is not defined.
Am I missing any file?
-Thanks

10 **n Ahmad** on November 5, 2016 at 8:58 am
ndly do.
from sklearn.cross_validation import train_test_split

11 on November 5, 2016 at 2:25 pm
s, its working now!!!.

12 **Using Machine Learning to Detect Malicious URLs - Use-R!Use-R!** on November 9, 2016 at 1:29 pm
[...] With the growth of Machine Learning in the past few years, many tasks are being done with the help of machine learning algorithms. Unfortunately or fortunately, there has been little work done on security with machine learning algorithms. So I thought of presenting some at Fsecurify. [...]

13 **Data Science with Python | My Data Blog** on November 24, 2016 at 6:26 am
[...] Using Machine Learning to detect malicious URL [...]

14 **Dee** on December 5, 2016 at 2:13 am
Hi Fiazan, The explaination is really good. but i am not able to run the code. I am relatively new to python. i am running code in python 3.5. I am getting this error: ValueError: empty vocabulary; perhaps the documents only contain stop words. It is raising error at this line vectorizer.fit_transform(corpus). Can you please check it?

FSECURIFY
security for everyone

SECURITY FOR EVERYONE

16

IrmaIMclin on December 18, 2016 at 1:40 am

Keep on working, great job!

+ **Leave a Comment**

Your thoughts..

Name... your@email.com... Website...

SUBMIT