

# AI-BASED THREAT INTELLIGENCE AND PREDICTION SYSTEM

---

Anshika Tyagi

# CONTENTS

1. Introduction
2. Background
3. Implementation Details
4. Results and Analysis
5. Conclusion

# 1. INTRODUCTION

---

The spread of cyberthreats in the modern era presents serious difficulties for businesses all around the world. In many cases, sophisticated attacks are difficult to detect and mitigate using conventional cybersecurity techniques. The goal of this project is to create an AI-Based Threat Intelligence and Prediction System that will improve cyber threat detection and prediction by utilizing machine learning techniques. By analyzing phishing site URLs, this system will categorize them as benign or malicious, taking a proactive approach to cybersecurity.



## 2. BACKGROUND



The process of obtaining, evaluating, and interpreting data regarding present and future risks to an organization is known as threat intelligence. It seeks to offer practical insights to improve security posture. Conventional approaches frequently depend on manual procedures and static rules, which are inadequate to counter threats that are dynamic and ever-changing.

Advanced threat intelligence capabilities are provided by machine learning (ML) and artificial intelligence (AI). Artificial intelligence (AI)-based systems can improve threat detection speed and accuracy by analyzing large datasets, finding patterns, and making predictions. The goal of this project is to categorize phishing URLs—which are frequently used in cyberattacks to trick users into divulging sensitive information—using artificial intelligence techniques.

# 3. IMPLEMENTATION DETAILS

**3.1 Data Collection and Preprocessing:** The dataset used in this project consists of phishing site URLs, sourced from Kaggle. The implementation involves several key steps:

- Data Loading: The dataset is loaded using Pandas.
- Exploratory Data Analysis (EDA): Understanding the dataset through basic statistics and visualizations.
- Tokenization: Splitting URLs into individual tokens.
- Stemming: Reducing words to their base or root form.
- Joining Tokens: Reconstructing URLs from tokens for further processing.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
matplotlib.inline
import seaborn as sns
import time

from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from nltk.tokenize import RegexpTokenizer
from nltk.stem.snowball import SnowballStemmer
from sklearn.feature_extraction.text import CountVecorizer
from sklearn.pipeline import make_pipeline

from PIL import Image
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
import pickle

df = pd.read_csv("C:\\Users\\anshi\\OneDrive\\Desktop\\phishing_site_urls.csv\\phishing_site_urls.csv")
```

## 3.2 Data PreprocessingTokenization:

- Using RegexpTokenizer to split URLs into tokens.
- Stemming: Using SnowballStemmer to reduce tokens to their root forms.
- Joining Tokens: Combining tokens back into strings.

```
tokenizer = RegexpTokenizer(r'[A-Za-z]+')
[11] ✓ 0.0s

tokenizer.tokenize(df.URL[0]) # this will fetch
[12] ✓ 0.0s

... ['nobell',
     'it',
     'ffb',
     'd',
     'dca',
     'cce',
     'f',
     'login',
     'SkyPe',
     'com',
     'en',
     'cgi',
     'bin',
     'verification',
     'login',
     'ffb',
     'd',
     'dca',
     'cce',
     'f',
     'index',
     'php',
     'cmd',
     'profile',
     'ach',
     ...
     'to',
     'load',
     'nav',
     'login',
     'access']

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```



```
print('Getting words tokenized ...')
t0= time.perf_counter()
df['text_tokenized'] = df.URL.map(lambda t: tokenizer.tokenize(t))
t1 = time.perf_counter() - t0
print('Time taken',t1,'sec')
```

✓ 12s

Getting words tokenized ...  
Time taken 1.2950458000005533 sec

df.sample(10)

✓ 00s

	URL	Label	text_tokenized
126965	royalgateenergy.com/wp-admin/js/images/httpdoc...	bad	[royalgateenergy, com, wp, admin, js, images, ...]
175665	en.wikipedia.org/wiki/Hy_Sandham	good	[en, wikipedia, org, wiki, Hy, Sandham]
301121	cduniverse.com/productinfo.asp?pid=1168879	good	[cduniverse, com, productinfo, asp, pid]
248876	torontobeerweek.com/tbw-store/	good	[torontobeerweek, com, tbw, store]
480909	loot.co.za/index/html/index2144.html	good	[loot, co, za, index, html, index, html]
275979	amazon.com/Locus-1-Patrick-Fillion/dp/1897102577	good	[amazon, com, Locus, Patrick, Fillion, dp]
432577	smhc.qc.ca/en/research	good	[smhc, qc, ca, en, research]
534815	42roza.ru/	bad	[roza, ru]
336883	flickr.com/photos/12150532@N04/4700249211/	good	[flickr, com, photos, N]
490338	dontskid.me/netbot/cp.php?m=login	bad	[dontskid, me, netbot, cp, php, m, login]

stemmer = SnowballStemmer("english")

✓ 00s

```
print('Getting words stemmed ...')
t0= time.perf_counter()
df['text_stemmed'] = df['text_tokenized'].map(lambda l: [stemmer.stem(word) for word in l])
t1= time.perf_counter() - t0
print('Time taken',t1,'sec')
```

✓ 22s

Getting words stemmed ...  
Time taken 22.331301799999892 sec

```
print('Getting words stemmed ...')
t0= time.perf_counter()
df['text_stemmed'] = df['text_tokenized'].map(lambda l: [stemmer.stem(word) for word in l])
t1= time.perf_counter() - t0
print('Time taken',t1,'sec')
```

✓ 22s

Getting words stemmed ...  
Time taken 22.331301799999892 sec

df.sample(10)

✓ 00s

	URL	Label	text_tokenized	text_stemmed
272644	allstarpics.net/pic-gallery/lisa-robin-kelly-p...	good	[allstarpics, net, pic, gallery, lisa, robin, ...]	[allstarp, net, pic, galleri, lisa, robin, kel...
323702	eventective.com/USA/Illinois/Wheaton/Party-Ren...	good	[eventective, com, USA, Illinois, Wheaton, Parti...	[eventect, com, usa, illinol, wheaton, parti, ...]
115509	nortagem.cl/ws/rg/es/auth/view/document/	bad	[nortagem, cl, ws, rg, es, auth, view, document]	[nortagem, cl, ws, rg, es, auth, view, document]
94740	www.larp.com/WashDC/	good	[www, larp, com, WashDC]	[www, larp, com, washdc]
496950	105.117.152.176/upload/_dispatch.php	bad	[upload, dispatch, php]	[upload, dispatch, php]
527099	peterdickem.com/87745g	bad	[peterdickem, com, g]	[peterdickem, com, g]
17002	www.ccswy.com/css/secure.runescape.com/m=webl...	bad	[www, ccswy, com, css, secure, runescape, com, m...	[www, ccswi, com, css, secur, runescap, com, m...
165279	dictionary.sensagent.com/united+states+army+co...	good	[dictionary, sensagent, com, united, states, arm...	[dictionari, sensag, com, unit, state, armt, co...
322921	epodunk.com/cgi-bin/genInfo.php?locIndex=20390	good	[epodunk, com, cgi, bin, genInfo, php, locIndex]	[epodunk, com, cgi, bin, geninfo, php, locindex]
274795	amazon.com/Donkey-Heart-Monkey-Mind-ebook/dp/B...	good	[amazon, com, Donkey, Heart, Monkey, Mind, ebo...	[amazon, com, donkey, heart, monkey, mind, ebo...

```
print('Get joiningwords ...')
t0= time.perf_counter()
df['text_sent'] = df['text_stemmed'].map(lambda l: ' '.join(l))
t1= time.perf_counter() - t0
print('Time taken',t1,'sec')
```

✓ 01s

Get joiningwords ...  
Time taken 0.145803299999974967 sec

```
bad_sites = df[df.Label == 'bad']
good_sites = df[df.Label == 'good']
```

✓ 01s

bad\_sites.head()

✓ 00s

	URL	Label	text_tokenized	text_stemmed	text_sent
0	robk33.it/affiliados/aweb/aweb/aweb/aweb/aweb/...	bad	[robk33, it, affi, do, doc, coo, f, login, stype...	[robk33, it, affi, do, doc, coo, f, login, stype...	robk33 it affi do doc coo f login stype com wr ag...
1	www.dghdgt.com/pajpal/co.uk/cygi-bin/welwa...	bad	[www, dghdgt, com, pajpal, co, uk, cygi, bin...	[www, dghdgt, com, pajpal, co, uk, cygi, bin...	www dghdgt com pajpal co uk cygi bin welwa...
2	www.kidsy.com/pajpal/co.uk/cygi-bin/welwa...	bad	[www, kidsy, com, pajpal, co, uk, cygi, bin...	[www, kidsy, com, pajpal, co, uk, cygi, bin...	www kidsy com pajpal co uk cygi bin welwa...
3	mail.pntelakid.com/www.online.americanerpro...	bad	[mail, pntelakid, com, www, online, americaner...	[mail, pntelakid, com, www, online, americaner...	mail pntelakid com www online americanerpro...
4	thawh3aydng3.com/wp-content/themes/wides...	bad	[thawh3aydng3, com, wp, content, theme, wide...	[thawh3aydng3, com, wp, content, theme, wide...	thawh3aydng3 com wp content theme wides...

good\_sites.head()

✓ 00s

	URL	Label	text_tokenized	text_stemmed	text_sent
98021	480c.com/j/index.htm/us/battle.net/noghn/w...	good	[480c, com, j, index, itm, us, battle, net, ...]	[480c, com, j, index, itm, us, battle, net, ...]	480c com j index itm us battle net noghn w...
98020	www.dghdgt.com/pajpal/co.uk/cygi-bin/welwa...	good	[www, dghdgt, com, pajpal, co, uk, cygi, bin...	[www, dghdgt, com, pajpal, co, uk, cygi, bin...	www dghdgt com pajpal co uk cygi bin welwa...
98009	www.kidsy.com/pajpal/co.uk/cygi-bin/welwa...	good	[www, kidsy, com, pajpal, co, uk, cygi, bin...	[www, kidsy, com, pajpal, co, uk, cygi, bin...	www kidsy com pajpal co uk cygi bin welwa...
98004	mail.pntelakid.com/www.online.americanerpro...	good	[mail, pntelakid, com, www, online, americaner...	[mail, pntelakid, com, www, online, americaner...	mail pntelakid com www online americanerpro...
98026	thawh3aydng3.com/wp-content/themes/wides...	good	[thawh3aydng3, com, wp, content, theme, wide...	[thawh3aydng3, com, wp, content, theme, wide...	thawh3aydng3 com wp content theme wides...

df.head()

✓ 00s

	URL	Label	text_tokenized	text_stemmed	text_sent
0	robk33.it/affiliados/aweb/aweb/aweb/aweb/aweb/...	bad	[robk33, it, affi, do, doc, coo, f, login, stype...	[robk33, it, affi, do, doc, coo, f, login, stype...	robk33 it affi do doc coo f login stype com wr ag...
1	www.dghdgt.com/pajpal/co.uk/cygi-bin/welwa...	bad	[www, dghdgt, com, pajpal, co, uk, cygi, bin...	[www, dghdgt, com, pajpal, co, uk, cygi, bin...	www dghdgt com pajpal co uk cygi bin welwa...
2	www.kidsy.com/pajpal/co.uk/cygi-bin/welwa...	bad	[www, kidsy, com, pajpal, co, uk, cygi, bin...	[www, kidsy, com, pajpal, co, uk, cygi, bin...	www kidsy com pajpal co uk cygi bin welwa...
3	mail.pntelakid.com/www.online.americanerpro...	bad	[mail, pntelakid, com, www, online, americaner...	[mail, pntelakid, com, www, online, americaner...	mail pntelakid com www online americanerpro...
4	thawh3aydng3.com/wp-content/themes/wides...	bad	[thawh3aydng3, com, wp, content, theme, wide...	[thawh3aydng3, com, wp, content, theme, wide...	thawh3aydng3 com wp content theme wides...

df.head()

✓ 00s

	URL	Label	text_tokenized	text_stemmed	text_sent
0	robk33.it/affiliados/aweb/aweb/aweb/aweb/aweb/...	bad	[robk33, it, affi, do, doc, coo, f, login, stype...	[robk33, it, affi, do, doc, coo, f, login, stype...	robk33 it affi do doc coo f login stype com wr ag...
1	www.dghdgt.com/pajpal/co.uk/cygi-bin/welwa...	bad	[www, dghdgt, com, pajpal, co, uk, cygi, bin...	[www, dghdgt, com, pajpal, co, uk, cygi, bin...	www dghdgt com pajpal co uk cygi bin welwa...
2	www.kidsy.com/pajpal/co.uk/cygi-bin/welwa...	bad	[www, kidsy, com, pajpal, co, uk, cygi, bin...	[www, kidsy, com, pajpal, co, uk, cygi, bin...	www kidsy com pajpal co uk cygi bin welwa...
3	mail.pntelakid.com/www.online.americanerpro...	bad	[mail, pntelakid, com, www, online, americaner...	[mail, pntelakid, com, www, online, americaner...	mail pntelakid com www online americanerpro...
4	thawh3aydng3.com/wp-content/themes/wides...	bad	[thawh3aydng3, com, wp, content, theme, wide...	[thawh3aydng3, com, wp, content, theme, wide...	thawh3aydng3 com wp content theme wides...

### 3.3 Feature Extraction Count Vectorizer : Converting text data into numerical features using Count Vectorizer.c

```
cv = CountVectorizer()
```

✓ 0.0s

```
feature = cv.fit_transform(df.text_sent)
```

✓ 3.7s

```
feature[:5].toarray()
```

✓ 0.0s

```
array([[0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0],
       [0, 0, 0, ..., 0, 0, 0]], dtype=int64)
```

### 3.4 Model Training and Evaluation:

- **Logistic Regression:** Training and evaluating a logistic regression model.
- **Multinomial Naive Bayes:** Training and evaluating a multinomial Naive Bayes model.

```
from sklearn.model_selection import train_test_split
```

✓ 0.0s

```
trainX, testX, trainY, testY = train_test_split(feature, df.Label)
```

✓ 0.0s

```
from sklearn.linear_model import LogisticRegression
```

✓ 0.0s

```
lr = LogisticRegression()
lr.fit(trainX, trainY)
```

✓ 7.3s

`g:\Python\lib\site-packages\sklearn\linear_model\logistic.py:459: ConvergenceWarning: lbfgs failed to converge (status=1): STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.`

Increase the number of iterations (max\_iter) or scale the data as shown in:

<https://scikit-learn.org/stable/modules/preprocessing.html>

Please also refer to the documentation for alternative solver options:

[https://scikit-learn.org/stable/modules/linear\\_model.html#logistic-regression](https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression)

```
n_iter_1 = _check_optimize_result(
```

```
LogisticRegression()
LogisticRegression()
```

## 3.5 Model Comparison

- **Accuracy Comparison: Comparing the accuracy of different models using a bar plot.**

```
from sklearn.naive_bayes import MultinomialNB
```

✓ 0.0s

```
mnb = MultinomialNB()
```

✓ 0.0s

```
mnb.fit(trainX,trainY)
```

✓ 0.9s

▼ MultinomialNB ⓘ ⓘ  
MultinomialNB()

```
mnb.score(testX,testY)
```

✓ 0.2s

0.9586200368436765

```
Scores_ml['MultinomialNB'] = np.round(mnb.score(testX,testY),2)
```

✓ 0.2s

```
from sklearn.linear_model import LogisticRegression
```

✓ 0.0s

```
lr = LogisticRegression()  
lr.fit(trainX,trainY)
```

✓ 7.3s

g:\PythonLib\site-packages\sklearn\linear\_model\\_logistic.py:469: ConvergenceWarning: lbfgs failed to converge (status=1):  
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max\_iter) or scale the data as shown in:  
<https://scikit-learn.org/stable/modules/preprocessing.html>  
Please also refer to the documentation for alternative solver options:  
[https://scikit-learn.org/stable/modules/linear\\_model.html#logistic-regression](https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression)  
n\_iter\_1 = \_check\_optimize\_result(  
 LogisticRegression ⓘ ⓘ  
 LogisticRegression()

```
lr.score(testX,testY)
```

✓ 0.2s

0.9653188871178916

```
Scores_ml = {}  
Scores_ml['Logistic Regression'] = np.round(lr.score(testX,testY),2)
```

✓ 0.1s



# 4. RESULTS AND ANALYSIS

## 4.1 Logistic Regression

- Training Accuracy: 98%
- Testing Accuracy: 97%



```
acc = pd.DataFrame.from_dict(Scores_ml, orient='index', columns=['Accuracy'])

acc.reset_index(inplace=True)
acc.rename(columns={'index': 'Model'}, inplace=True)

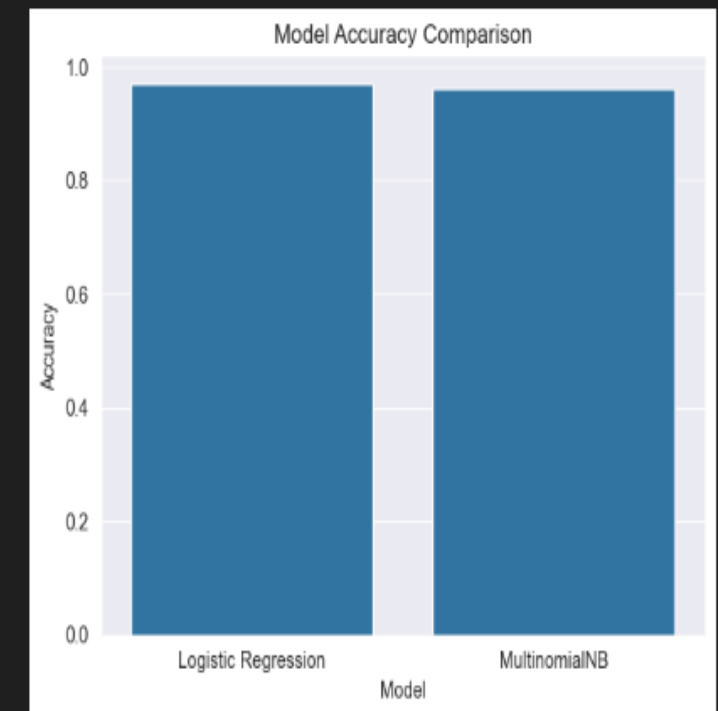
sns.set_style('darkgrid')

sns.barplot(data=acc, x='Model', y='Accuracy')

plt.xlabel('Model')
plt.ylabel('Accuracy')
plt.title('Model Accuracy Comparison')

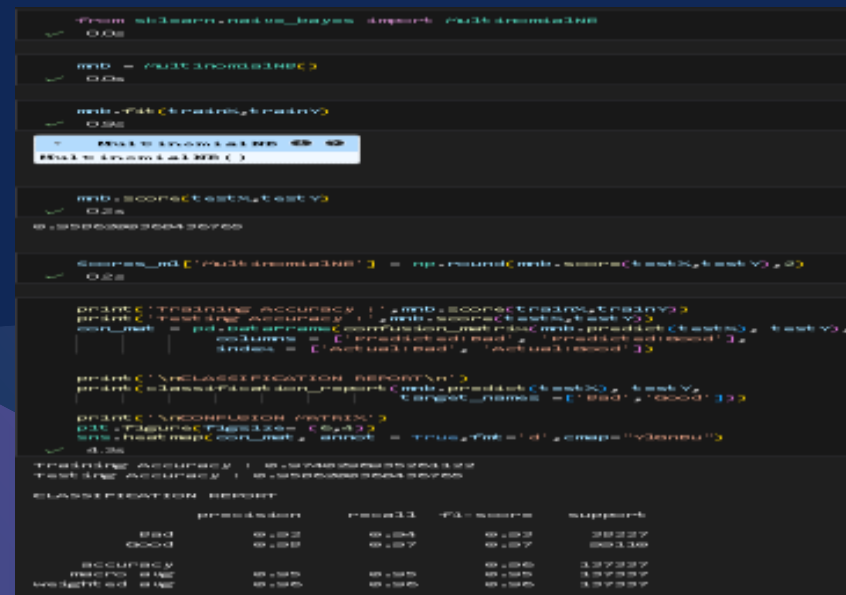
plt.show()
```

✓ 0.0s



## 4.2 Multinomial Naive Bayes

- Training Accuracy: 96%
- Testing Accuracy: 95%



## 5. CONCLUSION

---

The AI-Based Threat Intelligence and Prediction System demonstrated effective performance in classifying phishing URLs using machine learning techniques. Both Logistic Regression and Multinomial Naive Bayes models showed high accuracy, with Logistic Regression slightly outperforming in this context. The system can be further enhanced by incorporating additional features and advanced algorithms to improve prediction accuracy and robustness.



# REFERENCES

---

- Books:

"Artificial Intelligence in Cybersecurity" by Leslie F. Sikos.

"Machine Learning and Security" by Clarence Chio and David Freeman.

- Articles:

"The Role of Artificial Intelligence in Cyber Security" by John A. Clark.

"Predictive Threat Intelligence Using Machine Learning" by Karen Scarfone.

- Websites:

- Darktrace

- QRadar