

Tugas Project 1 Mata Kuliah Sistem Keamanan Cerdas

# **Phishing Email Detection Using Machine Learning**



Disusun oleh:

Kelompok Neural Theft

Seagata Ade Pratama Barus (1301210371)

Mursyid Najib Muhana (1301210411)

**Bandung**

**2024**

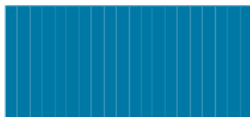
# Deskripsi Proyek

Email phishing telah menjadi ancaman bagi individu ataupun organisasi di seluruh dunia. Email phishing ini bertujuan untuk mengelabui penerima agar membocorkan informasi sensitif atau melakukan tindakan berbahaya. Mendeteksi dan mencegah email phishing sangat penting untuk menjaga keamanan pribadi dan keuangan. Dalam beberapa tahun terakhir, teknik pembelajaran mesin telah muncul sebagai pendekatan yang menjanjikan untuk memerangi ancaman yang terus berkembang ini.

Proyek ini bertujuan untuk membuat model deteksi email phishing menggunakan Phishing Email Detection dataset yang didapatkan dari kaggle. proyek ini melibatkan penggunaan beberapa algoritma machine learning yaitu Random Forest, Support Vector Machine (SVM), Adaboost, Decision Tree

## Dataset

Phishing Email Detection Dataset ini menentukan jenis teks email yang dapat digunakan untuk mendeteksi email phishing dengan analisis ekstensif terhadap teks email yang dapat diklasifikasikan menggunakan pembelajaran mesin.

#	△ Email Text	△ Email Type
	<div>empty3%</div> <div>[null]0%</div> <div>Other (18101)97%</div>	<div>Safe Email61%</div> <div>Phishing Email39%</div>
0	re : 6 . 1100 , disc : uniformitarianism , re : 1086 ; sex / lang dick hudson 's observations on us ...	Safe Email
1	the other side of * galicismos * * galicismo * is a spanish term which names the improper introducti...	Safe Email
2	re : equistar deal tickets are you still available to assist robert with entering the new deal ticke...	Safe Email
3	Hello I am your hot lil horny tex. I am	Phishing Email

Data ini berisi dua fitur yaitu Teks Email dan Jenis Email. Teks Email menentukan isi email dan Jenis Email menentukan jenis email apakah itu Phishing atau Aman

link dataset: <https://www.kaggle.com/datasets/subhajournal/phishingemails/code>

## Algoritma

### 1. Random Forest

Random forests atau random decision forests adalah metode pembelajaran ensemble untuk klasifikasi, regresi, dan tugas-tugas lain yang beroperasi dengan membangun banyak pohon keputusan pada waktu pelatihan.

([https://en.wikipedia.org/wiki/Random\\_forest](https://en.wikipedia.org/wiki/Random_forest))

### 2. Support Vector Machine (SVM)

Mesin vektor pendukung (bahasa Inggris: Support-vector machine atau disingkat SVM) adalah sebuah algoritma klasifikasi untuk data linear dan non-linear. SVM menggunakan mapping non-linear untuk mentransformasikan training data awal ke dimensi yang lebih tinggi.

([https://id.wikipedia.org/wiki/Mesin\\_vektor\\_pendukung](https://id.wikipedia.org/wiki/Mesin_vektor_pendukung))

### 3. Adaboost

AdaBoost, kependekan dari Adaptive Boosting, adalah meta-algoritma klasifikasi statistik yang diformulasikan oleh Yoav Freund dan Robert Schapire pada tahun 1995, yang memenangkan Hadiah Gödel 2003 untuk karya mereka. Algoritma ini dapat digunakan bersama dengan berbagai jenis algoritma pembelajaran lainnya untuk meningkatkan kinerja.

(<https://en.wikipedia.org/wiki/AdaBoost>)

### 4. Decision Tree

Pohon keputusan adalah algoritme pembelajaran non-parametrik yang diawasi, yang digunakan untuk tugas klasifikasi dan regresi. Algoritma ini memiliki struktur pohon yang hirarkis, yang terdiri dari simpul akar, cabang, simpul internal, dan simpul daun.

(<https://www.ibm.com/topics/decision-trees>)

# Implementasi

## Library

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

#sklearn
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.pipeline import Pipeline
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, accuracy_score, confusion_matrix
from sklearn.ensemble import AdaBoostClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.feature_extraction.text import CountVectorizer
```

## Dataset

```
df = pd.read_csv('Phishing_Email.csv')
```

## Preprocess

Mengambil 1 sampel dari masing masing tipe E-mail lalu melakukan Undersampling untuk menyamakan total row email yang safe dan phishing

```
Safe = df[df["Email Type"]== "Safe Email"]
Phishing = df[df["Email Type"]== "Phishing Email"]

sampleP = Phishing.sample()
Phishing.drop(sampleP.index)
sampleS = Safe.sample()
Safe.drop(sampleS.index)

Safe = Safe.sample(Phishing.shape[0])
df= pd.concat([Safe, Phishing], ignore_index = True)
```

Dikarenakan kolom yang memiliki nilai null tidak ada, oleh karena itu tidak ada proses pembuangan baris/kolom null

## Pembagian Train & Test

```
X_train,x_test,y_train,y_test = train_test_split(X, y, test_size = 0.30,  
random_state = 10)
```

Pembagian df menjadi 2 jenis

```
testS_X = sampleS["Email Text"].values  
testS_Y = sampleS["Email Type"].values
```

```
testP_X = sampleP["Email Text"].values  
testP_Y = sampleP["Email Type"].values
```

## Random Forest

```
classifier = Pipeline([("tfidf",TfidfVectorizer()  
),("classifier",RandomForestClassifier(n_estimators=10))])  
forest = classifier.fit(X_train,y_train)
```

## Hyperparameter Tuning

```
classifier = Pipeline([("tfidf",TfidfVectorizer()  
),("classifier",RandomForestClassifier(n_estimators=1000))])  
classifier.fit(X_train,y_train)
```

## SVM

```
SVM = Pipeline([("tfidf", TfidfVectorizer()),("SVM", SVC(C = 100, gamma =  
"auto"))])  
SVM.fit(X_train,y_train)
```

## Hyperparameter Tuning

C = 1000

```
SVM = Pipeline([("tfidf", TfidfVectorizer()),("SVM", SVC(C = 1000, gamma =  
"auto"))])  
SVM.fit(X_train,y_train)
```

C = 5000

```
SVM = Pipeline([("tfidf", TfidfVectorizer()),("SVM", SVC(C = 5000, gamma =  
"auto"))])  
SVM.fit(X_train,y_train)
```

## Adaboost

```
adaboost_classifier = Pipeline([  
    ("tfidf", TfidfVectorizer()),
```

```

        ("classifier",
AdaBoostClassifier(base_estimator=DecisionTreeClassifier(max_depth=1),
n_estimators=50))
])

adaboost_model = adaboost_classifier.fit(X_train, y_train)

```

## Decision tree

```

DT = Pipeline([("tfidf", TfidfVectorizer()),("Decision Tree",
DecisionTreeClassifier())])

DT.fit(X_train,y_train)

```

## Hasil & Diskusi

Algoritma	Akurasi	Running Time		
		Safe	Phishing	Avg
Random Forest	0.96558	0.09506s	0.09297s	0.09401s
SVM	0.96649	0.02383s	0.02008s	0.02195s
Adaboost	0.93185	0.03678s	0.03482s	0.0358s
Decision Tree	0.90451	0.00465s	0.0042s	0.00442s

Berdasarkan ke-4 algoritma yang telah dipakai kami menemukan algoritma yang memiliki akurasi tertinggi adalah **SVM** dengan 0.96649 dan disusul oleh **Random Forest** dengan 0.96558. Untuk rata-rata running time tercepat adalah **Decision Tree** dengan waktu 0.00442s kemudian dilanjutkan dengan **SVM** dengan 0.02195s

## Kesimpulan

Algoritma	Akurasi	Avg runtime
Decision Tree	0.90451	0.00442s
SVM	0.96649	0.02195s

Kesimpulan dari proyek yang kami kerjakan adalah algoritma yang memiliki akurasi tertinggi adalah **SVM** dengan 0.96649 dan algoritma dengan rata-rata running time tercepat adalah **Decision Tree** dengan waktu 0.00442s.

## Referensi

- Akinyelu, A. A., & Adewumi, A. O. (2014). Classification of phishing email using random forest machine learning technique. *Journal of Applied Mathematics*, 2014, 6, 425731. <https://doi.org/10.1155/2014/425731>.
- Harikrishnan, N. B., Vinayakumar, R., & Soman, K. P. (2018). A machine learning approach towards phishing email detection. *CEN-Security@IWSPA 2018*, 1-6. Retrieved from <http://ceur-ws.org>.
- Aslan, S., & Samet, R. (2019). A comprehensive review on malware detection approaches. *Electronics*, 8(6), 42. <https://doi.org/10.3390/electronics8060042>.