Judul proyek: *Malicious URL Detector*

Nama anggota kelompok: Kornelis Febriano Kapa Api, I Gede Heryanta Saputra

Penjelasan:

Dalam proyek ini kami menggunakan metode Klasifikasi dengan algoritma *Decision Tree* dan *Random Forest*. Dalam proyek ini kami membuat model *Machine Learning* untuk mendeteksi *malicious* URL. *Dataset* yang kami gunakan terdiri dari 60rb baris URL dengan tipe dari tiap URL. URL dalam *dataset* yang kami gunakan dibedakan dalam 4 kategori/jenis yaitu *benign, defacement, phising, dan malware*.

Klasifikasi merupakan salah satu pendekatan *Supervised Learning*. Pada metode klasifikasi, model dilatih dengan memberikan inputan yang sudah diberi label, sehingga model dapat menemukan patern antara data dan label untuk diberi kategori/diklasifikasi. Kami menggunakan metode klasifikasi karena ingin mengkategorikan inputan berupa URL kedalam salah satu dari 4 kategori *malicious* URL.

Untuk meningkatkan akurasi model untuk mendeteksi, kami menggunakan teknik *Features Engineering*, dimana kami mengekstrak *Lexical Features* dari URL seperti, ada/tidaknya *suspicious word*, panjang karakter, ada/tidaknya angka dalam URL, protokol yang digunakan, dll. Setelah diekstak fitur ini kemudian digunakan saat *training* dan *testing*.

Algoritma yang kami gunakan adalah *Decision Tree* dan *Random Forest*. *Decision Tree* adalah algoritma *Supervised Learning*. *Decision Tree* membuat model yang memprediksi nilai target dengan mempelajari decision/keputusan sederhana yang disimpulkan dari fitur data. *Decision Tree* terdiri dari root node, branches, internal nodes, and leaf nodes. *Decision Tree* melakukan pencarian/penelusuran dari root node kemudian menuju ke internal nodes dan sampai pada leaf nodes berdasarkan fitur yang ada/tersedia. Pada *Decision Tree* tiap leaf node merepresentasikan label dan internal nodes merepresentasikan fitur. *Random Forest* adalah algoritma *Supervised Learning*. *Random Forest* menggabungkan beberapa *Decision Tree* untuk memecahkan masalah yang kompleks dan meningkatkan kinerja model.

Berikut adalah tahapan pembuatan model *Machine Learning Malicious* URL Detector:

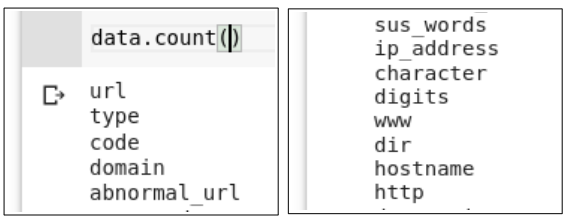
1) Pada tahap ini, kami mengimport *Library* dan melakukan pengecekan terhadap *dataset*

```
[ ] import re
    import numpy as np
    import pandas as pd
    import opendatasets as od
    from sklearn.model_selection import train_test_split
    from sklearn.metrics import classification_report, accuracy_score
    from sklearn.tree import DecisionTreeClassifier
    from sklearn.ensemble import RandomForestClassifier, ExtraTreesClassifier
    from lightgbm import LGBMClassifier
    from sklearn.neighbors import KNeighborsClassifier
    from sklearn.linear_model import SGDClassifier
    import xgboost as xgb
    from tld import get_tld, is_tld
```

```
[ ] #od.download("https://www.kaggle.com/datasets/sid321axn/malicious-urls-dataset")

[ ] data = pd.read_csv("/content/malicious-urls-dataset/malicious_phish.csv")
    data

[ ] data.info()

[ ] data.isnull().sum()

[ ] count = data.type.value_counts()
    count
```

2) Tahap Selanjutnya, kami mengekstrak fitur dari URL sekaligus menkonversinya menjadi angka. Berikut fitur yang kami ekstrak.
   a) kode jenis URL
      Menkonversi beign, defacement, phising, dan malware menjadi 0,1,2,3.
   b) nama domain
      Mengekstrak nama domain dari URL. Contoh:
      - URL: en.wikipedia.org/wiki/Dead_Space_(video_game)
      - domain: en.wikipedia.org
   c) abnormal URL
      Mencari abnormal URL dengan melakukan pencarian terhadap *hostname*.
   *d) suspicious words*
      Mencari ada/tidaknya *suspicious words* dalam URL, seperti:"*PayPal, login, signin, bank, account, update, free, lucky, service, bonus, ebayisapi, webscr*"
   e) IP *address*
      Mencari ada/tidaknya URL yang menggunakan IP *address* sebagai *hostname*.
   f) jumlah karakter dalam URL.
      Menghitung jumlah karakter dalam URL.
   g) jumlah angka dalam URL
      Menghitung jumlah angka dalam URL.
   h) jumlah www
      Menghitung jumlah www dalam URL.
   i) jumlah *dir*
      Menghitung jumlah *directory* dalam URL.
   *j) hostname*
      Menghitung jumlah huruf/panjang *hostname* dalam URL.
   k) http
      Mencari ada/tidaknya penggunaan http dalam URL.

   Hasilnya kami mendapat 13 kolom, 9 diantaranya merupakan fitur.



3) Pada tahap ini kami melakukan definisi terhadap variabel X dan y sebagai *features* dan *prediction* dengan mengaitkan kolom terkait dengan variabel masing-masing.

```
X = data[['abnormal_url','sus_words','ip_address','character','digits','www','dir','hostname','http']]
y = data["code"]
```

4) Tahap selanjutnya, kami melakukan pembagian data untuk train 80 persen dan *testing* 20 persen menggunakan *library* train_test_split
5) Pada tahap ini, kami membuat model *Machine Learning*.

Decision Tree Classifier

```
[ ] dtc = DecisionTreeClassifier()
    dtc.fit(X_train.values, y_train)
    pred = dtc.predict(X_test.values)
    dtc_accuracy = accuracy_score(pred, y_test)
    print(dtc_accuracy)
```

Random Forest Classifier

```
rfc = RandomForestClassifier()
rfc.fit(X_train.values, y_train)
pred = rfc.predict(X_test.values)
rfc_accuracy = accuracy_score(pred, y_test)
print(rfc_accuracy)
print(rfc.feature_importances_)
```

Hasil dari proyek ini adalah:

1) Kami dapat membuat model *Machine Learning* menggunakan metode klasifikasi dengan algoritma *Decision Tree* dan *Random Forest*.
2) Algoritma *Decision Tree* dan *Random Forest* dapat digunakan untuk deteksi *malicious* URL.
3) *Features Engineering* dapat digunakan untuk meningkatkan kualitas model.
4) Akurasi *Decision Tree* pada model ini adalah 0.96, sedangkan *Random Forest* adalah 0.95.