1. Opis projektu

Aplikacja służy do wykrywania phishingowych adresów URL. Wykorzystuje algorytm klasyfikacyjny Random Forest, który na podstawie cech wyekstrahowanych z adresu URL przewiduje, czy dany adres jest bezpieczny, czy może być phishingowy. Interfejs aplikacji został zrealizowany za pomocą biblioteki Streamlit, co umożliwia łatwe korzystanie z aplikacji w formie webowej.

---

2. Struktura projektu

1. **Streamlit (interfejs użytkownika):**
    - Główny mechanizm interakcji z użytkownikiem.
    - Umożliwia wprowadzanie URL, prezentację wyników analizy oraz przeglądanie historii sprawdzonych adresów.
2. **PhishingFeatureExtractor (ekstrakcja cech):**
    - Klasa odpowiedzialna za analizę cech URL, które mogą sugerować, że adres jest phishingowy.
3. **Model Random Forest:**
    - Algorytm uczenia maszynowego, który na podstawie cech URL dokonuje klasyfikacji na phishingowy lub bezpieczny.
4. **Dane treningowe:**
    - Plik Training_Dataset.arff zawierający próbki URL oraz ich etykiety (Result: phishingowy lub bezpieczny).
    - Dane te są wykorzystywane do trenowania modelu.

---

3. Szczegółowy opis kodu

A. Interfejs użytkownika

1. **Inicjalizacja Streamlit:**
    - st.session_state przechowuje:
        - **is_safe**: Wynik predykcji dla ostatnio sprawdzonego URL (1 - podejrzany, 2 - bezpieczny).
        - **URL_list**: Lista URL sprawdzonych przez użytkownika.
2. **Opcje w panelu bocznym:**
    - Sprawdź URL: Umożliwia wprowadzanie URL i przeprowadzenie analizy.
    - Zobacz Historię: Wyświetla wszystkie wcześniej sprawdzone URL.
3. **Funkcja add_item:**

- Pobiera URL z pola tekstowego.
- Przeprowadza analizę za pomocą klasy PhishingFeatureExtractor.
- Wynik analizy jest predykowany przez model Random Forest.
- Wynik (is_safe) oraz URL są dodawane do historii.

B. Ekstrakcja cech (PhishingFeatureExtractor)

Klasa zawiera metody, które analizują różne aspekty URL. Wynikiem działania każdej metody jest wartość 1, 0, lub -1, która reprezentuje odpowiednio bezpieczną, podejrzaną lub phishingową charakterystykę.

Przykłady metod:

1. **contains_ip(url)**:
   - Sprawdza, czy URL zawiera adres IP. Adresy IP w URL są częściej używane w phishingu.
   - **Zwraca:** 1 (bezpieczny) lub -1 (podejrzany).
2. **url_length(url)**:
   - Bada długość URL. Długie URL (>54 znaki) są częściej phishingowe.
   - **Zwraca:** 1 (bezpieczny), 0 (średnio podejrzany), -1 (bardzo podejrzany).
3. **uses_shortening_service(url)**:
   - Sprawdza, czy URL korzysta z usług skracania linków (np. bit.ly).
   - **Zwraca:** -1 (podejrzany) lub 1 (bezpieczny).
4. **ssl_final_state(url)**:
   - Weryfikuje certyfikat SSL strony. Brak certyfikatu lub nieważny certyfikat zwiększa ryzyko phishingu.
   - **Zwraca:** 1 (ważny certyfikat), 0 (nieważny), -1 (brak certyfikatu).
5. **domain_registration_length(url)**:
   - Sprawdza, jak długo domena jest zarejestrowana. Krótki okres (<=1 rok) jest bardziej podejrzany.
   - **Zwraca:** 1 (bezpieczny) lub -1 (podejrzany).
6. **favicon(url)**:
   - Sprawdza, czy favicon (ikona strony) pochodzi z tej samej domeny co URL.
   - **Zwraca:** 1 (ten sam domena), -1 (inny domena).
7. **request_url(url)**:

- Analizuje, czy zasoby (np. obrazy, skrypty) ładowane na stronie pochodzą z innych domen.
- **Zwraca:** 1 (większość zasobów lokalnych), 0 (średnio podejrzane), -1 (większość zewnętrznych).

C. Model Random Forest

1. **Dane treningowe:**
   - Dane są ładowane z pliku .arff, dekodowane, a następnie konwertowane do DataFrame.
2. **Przygotowanie danych:**
   - Zbiór danych dzielony jest na treningowy i testowy (train_test_split).
3. **Trenowanie modelu:**
   - Model Random Forest jest trenowany na zbiorze treningowym.
4. **Predykcja:**
   - Model przyjmuje jako wejście cechy wygenerowane przez PhishingFeatureExtractor.
   - Wynik predykcji (1 lub 2) jest zwracany aplikacji Streamlit.

D. Obsługa wyników

1. **Sprawdzenie URL:**
   - Wynik analizy (is_safe) jest prezentowany w aplikacji:
     - **Zielony komunikat:** URL jest bezpieczny.
     - **Czerwony komunikat:** URL jest podejrzany.
2. **Historia URL:**
   - Lista wcześniej sprawdzonych URL jest dostępna w opcji Zobacz Historię.

---

4. Przykład działania

1. **Scenariusz użytkownika:**
   - Użytkownik wybiera opcję Sprawdź URL.
   - Wprowadza URL: http://example.com.
   - Wynik analizy cech URL:

csharp

KopiujEdytuj

[1, -1, 0, 1, -1, 1, 1, 1, 1, -1, 1, 1, 1, 0, 1, -1, -1, 1, 1, 1, 1, 1, 1, 1, -1, 1, 1, 1, 1, 1]

- Model Random Forest przewiduje klasę 2 (bezpieczny URL).

2. **Wynik:**
    - Aplikacja wyświetla komunikat: **"URL wydaje się bezpieczny"**.
    - URL jest dodawany do historii.

Dokładny Opis metod zapisany jest w dokumecie „Phishing Websites Features”

---

5. Instrukcja uruchomienia

1. **Wymagania wstępne:**
    - Python w wersji 3.7 lub wyższej.
    - Zainstalowane biblioteki wymienione w pliku requirements.txt.
2. **Jak uruchomić aplikację:**
    - Zainstaluj wymagane zależności:

bash

KopiujEdytuj

pip install -r requirements.txt

- Uruchom aplikację Streamlit:

bash

KopiujEdytuj

streamlit run Main.py

3. **Plik danych:**
    - Upewnij się, że plik Training_Dataset.arff znajduje się w katalogu Data/.