

**WYDZIAŁ ELEKTROTECHNIKI, AUTOMATYKI, INFORMATYKI I INŻYNIERII BIOMEDYCZNEJ**

Projekt dyplomowy

Wykrywanie toksycznych/nienawistnych komentarzy

Detection of toxic/hateful comments

Autor: Bartosz Woziwoda

Kierunek studiów: Automatyka i Robotyka

Opiekun pracy: dr hab. inż. Jerzy Baranowski

Kraków, 2024

**Spis treści**

[**Wstęp** 3](#_Toc152501808)

[**Rozdział 1: Mowa nienawiści** 5](#_Toc152501809)

[**1.1** **Mowa nienawiści** 5](#_Toc152501810)

[**1.2** **Problematyka detekcji mowy nienawiści** 6](#_Toc152501811)

[**1.3** **Aktualne rozwiązania** 7](#_Toc152501812)

[**Rozdział 2: Przetwarzanie języka naturalnego** 8](#_Toc152501813)

[**2.1** **Zbieranie danych** 8](#_Toc152501816)

[**2.2** **Model uczenia maszynowego** 8](#_Toc152501817)

[**2.3** **Podstawowe techniki przetwarzania tekstu** 13](#_Toc152501818)

[**2.4** **Modele klasyfikacji w NLP** 15](#_Toc152501819)

[**Rozdział 3: Opis opracowanego programu** 16](#_Toc152501820)

[**Rozdział 4: Implementacja programu** 19](#_Toc152501821)

[**4.1** **Narzędzia** 20](#_Toc152501822)

[**4.2** **YouTube API** 21](#_Toc152501823)

[**4.3** **Przygotowanie i przetwarzanie danych** 23](#_Toc152501824)

[**a)** **Zbiory danych** 23](#_Toc152501825)

[**b)** **Przetwarzanie danych** 24](#_Toc152501826)

[**c)** **Tokenizacja – Bag of Words & TF-IDF** 25](#_Toc152501827)

[**4.4** **Modele NLP** 25](#_Toc152501828)

[**4.5**  **Aplikacja – funkcjonalności i interfejs** 27](#_Toc152501829)

[**Rozdział 5: Eksperymenty badawcze i testy** 28](#_Toc152501830)

[**5.1** **Porównanie wydajności modeli** 28](#_Toc152501831)

[**5.2** **Testy jednostkowe** 29](#_Toc152501832)

[**Podsumowanie** 30](#_Toc152501833)

[**Bibliografia** 31](#_Toc152501834)

# **Wstęp**

Według portalu Statista, w 2023 roku użytkownicy Internetu na całym świecie spędzali średnio 151 minut dziennie na portalach społecznościowych, co stanowiło prawie 10,5% całego dnia. Dla porównania, w 2012 roku było to 90 minut [1] [2]. Internet stał się kluczowym miejscem do komunikacji, dzielenia się treściami, przemyśleniami i informacjami dotyczącymi życia, zarówno prywatnego, jak i zawodowego. Niestety jednym ze skutków powstania tej pozornie bezpiecznej przestrzeni jest pojawienie się zjawiska określanego jako mowa nienawiści. Anonimowość, którą Internet oferuje, została wykorzystana m.in. do zastraszania, znęcania się i obrażania innych użytkowników wirtualnej sieci bez obawy o poniesienie konsekwencji.

Niezwykle niepokojącym aspektem tego zjawiska jest jego wpływ na dzieci i młodzież. Według raportu „Polskie badanie EU Kids Online 2018” [3], obserwujemy znaczny wzrost korzystania z Internetu przez młodych ludzi w ostatnich latach. Praktycznie wszystkie nastolatki mają dostęp do Internetu, a wiele z nich korzysta z niego wiele razy dziennie bądź niemal przez cały czas. Co bardziej niepokojące, około 72% dzieci i młodzieży deklaruje, że miało do czynienia z treściami uważanymi za szkodliwe. Raport ujawnia także, że co trzeci nastolatek miał styczność z mową nienawiści w Internecie, a aż 27% z nich przyznało się do bycia sprawcami agresji w sieci.

Niniejsza praca koncentruje się na problematyce wykrywania toksycznych i nienawistnych komentarzy w języku polskim. Głównym celem pracy jest opracowanie i implementacja efektywnego algorytmu identyfikacji mowy. Dodatkowo została utworzona aplikacja zapewniająca interfejs graficzny. Program ten umożliwia użytkownikowi analizę sekcji komentarzy pod wybranym filmem na platformie YouTube w celu wykrycia nienawistnych treści.

Pośród popularnych mediów społecznościowych takich jak Facebook, Instagram czy Tik Tok wybrany został YouTube, z uwagi na kilka aspektów:

* Portal ten udostępnia darmowy dostęp do interfejsu programowania aplikacji (popularnie określanego jako API- Application Programming Interface), co umożliwiło ekstrakcję komentarzy.
* Posiada szeroki zasięg – w 2022 roku docierał do 89,36% populacji internautów w Polsce. [4]
* Jest jednym z najpopularniejszych mediów społecznościowych w Polsce – w 2022 roku średnio miesięcznie z platformy korzystało 26,67 mln realnych użytkowników, co stanowiło największą liczbę wśród wszystkich mediów społecznościowych. [4]

Aplikacja oferuje intuicyjny interfejs, który umożliwia obsługę opracowanego algorytmu. Użytkownik ma możliwość dostarczenia własnego adresu internetowego (URL) do filmu umieszczonego na platformie YouTube w celu przeprowadzenia analizy treści zawartej w sekcji komentarzy pod danym filmem. Program przeprowadza analizę tekstu, oceniając go pod kątem obecności szkodliwej treści, a następnie zwraca użytkownikowi informacje zawierające zidentyfikowane przypadki mowy nienawiści. Ten mechanizm pozwala na monitorowanie jakości dyskusji pod danym filmem i ułatwia zwalczanie takich przypadków, co może przyczynić się do tworzenia bezpieczniejszego i bardziej przyjaznego środowiska.

Praca została podzielona na cztery rozdziały, które skupiają się na następujących aspektach:

* Wstęp teoretyczny przedstawiający tematykę identyfikowania i wykrywania mowy nienawiści wraz z towarzyszącymi temu wyzwaniami
* Opis poszczególnych elementów opracowanego programu, gdzie przedstawione są najistotniejsze komponenty rozwiązania
* Szczegółowy opis implementacji programu wraz z pseudokodem kluczowych funkcji i ich dokładnym opisem, pozwalający zrozumieć techniczne detale projektu
* Eksperymenty i testy, które prezentują działanie programu na przykładowych bazach komentarzy, przedstawiając efektywność i użyteczność programu

# **Rozdział 1: Mowa nienawiści**

W tym rozdziale zaprezentowano teoretyczne aspekty dotyczące mowy nienawiści i jej detekcji. Przedstawiono kluczowe definicje i metryki, które stanowiły podstawę dla dalszych rozdziałów zawartych pracy. Podjęto próbę określenia, czym jest mowa nienawiści, a także przyjrzano się stanowisku, jakie zajmują portale społecznościowe wobec tego zjawiska. Omówiony został proces zbierania danych oraz standardowego procesu tworzenia modelu uczenia maszynowego z naciskiem na tematykę przetwarzania języka naturalnego. Na końcu rozważono wyzwania związane z automatyczną detekcją mowy nienawiści oraz przeszkody, które często towarzyszą temu procesowi.

## **Mowa nienawiści**

Obraz zawierający tekst, zrzut ekranu, diagram, Czcionka

Opis wygenerowany automatycznieMowa nienawiści jest zjawiskiem polegającym na wykorzystywaniu języka w celu pobudzania, szerzenia lub usprawiedliwiania nienawiści, dyskryminacji i przemocy skierowanej przeciwko konkretnym osobom, grupom, lub przedstawicielom mniejszości. Akceptacja mowy nienawiści w społeczeństwie przyczynia się do utrwalania stereotypów, wzmacniania uprzedzeń oraz zmniejszenia akceptacji wobec osób będących przedstawicielami grup ‘hejtowanych’. [5] Na mowę nienawiści składają się bardziej precyzyjne zjawiska związane z poszczególnymi grupami społecznymi- dobrze ilustuje to poniższa grafika (Obraz\_1):

Obraz 1: Grafika przedstawiająca zjawiska zaliczające się do mowy nienawiści [5]

Nie istnieje jedna spójna definicja, która jednoznacznie określa, czym dokładnie jest mowa nienawiści. Termin ten jest zwykle różnie definiowany w zależności od kontekstu, sytuacji prawnej lub kraju. Niemniej jednak, jednomyślnie uznaje się, że jest to zjawisko negatywne i powszechnie szkodliwe. Najczęściej przyjmowaną definicją mowy nienawiści jest ta zawarta w rekomendacji nr R 97 Komitetu Ministrów Rady Europy dotyczącej treści wypowiedzi szerzących nienawiść. Zgodnie z tą rekomendacją, za mowę nienawiści powinna zostać uznana:

„*każda forma wypowiedzi, która rozpowszechnia, podżega, propaguje lub usprawiedliwia nienawiść rasową, ksenofobię, antysemityzm lub inne formy nienawiści oparte na nietolerancji, włączając w to nietolerancję wyrażaną w formie agresywnego nacjonalizmu lub etnocentryzmu, dyskryminacji lub wrogości wobec mniejszości, migrantów lub osób wywodzących się ze społeczności imigrantów*”. [6]

W obecnych czasach media społecznościowe odgrywają kluczową rolę w rozprzestrzenianiu mowy nienawiści. Platformy internetowe, takie jak YouTube, Facebook czy X (dawniej Twitter), stały się przestrzeniami, w których użytkownicy często wyrażają swoje poglądy i opinie. W odpowiedzi na narastający problem mowy nienawiści, platformy te były zmuszone do wprowadzenia odpowiedniej polityki i wypracowania konkretnego stanowiska w zakresie moderacji treści. Każda z witryn wprowadziła zestawy zasad i regulacji, których naruszenie traktuje się jako szerzenie toksycznych treści na ich platformie. Konsekwencją tych działań jest najczęściej ograniczanie swobody korzystania z portalu przez danego użytkownika, np. poprzez zablokowanie możliwości komentowania, lub całkowite zawieszenie dostępu do konta. Niemniej jednak, użytkownicy często próbują omijać te ograniczenia, zakładając nowe konta w celu dalszego, nieograniczonego korzystania z danych stron internetowych..

Portal YouTube, który pełni istotną rolę w niniejszej pracy, również określił swoje stanowisko wobec mowy nienawiści. Zgodnie z polityką firmy, treści promujące przemoc lub nienawiść wobec osób lub grup na podstawie różnych atrybutów, takich jak wiek, kasta, niepełnosprawność, pochodzenie etniczne, tożsamość i wyrażanie płci, narodowość, rasa, status imigracyjny, religia, płeć, orientacja seksualna, ofiary dużego wydarzenia przemocowego i ich krewni oraz status weterana, są usuwane [7].

## **Problematyka detekcji mowy nienawiści**

Automatyczne wykrywanie mowy nienawiści jest tematem, który napotyka na wiele wyzwań. Jednym z głównych problemów jest brak jednoznacznej definicji „mowy nienawiści”, która pozwoliłaby na jednoznaczne określenie, czy dany fragment tekstu zawiera tego rodzaju treści. Chociaż można określić kilka precyzyjnych reguł, to niemożliwe jest stworzenie ogólnego zestawu kryteriów obejmujących wszystkie możliwe przypadki. To sprawia, że klasyfikacja mowy nienawiści zależy często od kontekstu wypowiedzi, a także od odwołań do kultury lub historii.

Ponadto istnieje wiele innych czynników i ograniczeń, które utrudniają klasyfikację tekstu jako toksycznego lub nie. Wśród tych czynników znajdują się:

* Możliwość zamiany obraźliwego słowa na inne wyrażenie, modyfikację lub ukrycie go pod postacią innych słów, zarówno w celu uniknięcia automatycznej moderacji treści, jak i w rezultacie używania specyficznego slang w mediach społecznościowych. Przykładem może być zamiana litery „A” na „4” lub „E” na „3” [8].
* Wyrażenia, które same w sobie nie są ofensywne, ale w odpowiednim kontekście mogą stać się obraźliwe.
* Zmiana znaczenia słów w zależności od wydarzeń historycznych lub kontekstu.
* Subiektywność w klasyfikacji treści, która zależy od osoby dokonującej klasyfikacji. Ze względu na różnice w poglądach, kulturze, wyznawanej religii czy intuicji, jedna osoba może uznać dany tekst za toksyczny, podczas gdy druga wręcz przeciwnie.
* Niska zgodność między osobami klasyfikującymi treści, co oznacza, że ocena, czy dana treść jest toksyczna, może się znacznie różnić [9].

## **Aktualne rozwiązania**

W dziedzinie przetwarzania języka naturalnego nowoczesne rozwiązania coraz częściej wykorzystują techniki głębokiego uczenia oraz zaawansowane modele uczenia maszynowego. Dotyczy to również tematyki detekcji mowy nienawiści, chociaż jest to temat stosunkowo nowy. W ostatnich latach obserwujemy dynamiczny rozwój w automatycznym przetwarzaniu, analizie i generowaniu tekstu. Jest to głównie zasługą modeli opartych na sieciach neuronowych, takich jak transformery, na przykład BERT. Algorytmy te charakteryzują się większą precyzją, wydajnością i skutecznością w porównaniu do tradycyjnych metod NLP. [10]

Istotnym aspektem w badaniach nad automatyczną detekcją mowy nienawiści jest język analizowanego tekstu. Większość prac naukowych w tej dziedzinie jest pisana w języku angielskim, tak samo jak większość dostępnych zbiorów danych powiązanych z tym tematem. Znalezienie badań lub opracowań w mniej popularnych językach, jak polski, często stanowi wyzwanie. Wynika to zarówno z ograniczonej ilości dostępnych zasobów językowych, jak i z mniejszej liczby dostępnych danych czy modeli przystosowanych do mniej powszechnych języków, a także z mniejszej ilości prac naukowych publikowanych w tych językach.

# **Rozdział 2: Przetwarzanie języka naturalnego**



## **Zbieranie danych**

Proces detekcji mowy nienawiści rozpoczyna się, tak jak przy tworzeniu modeli uczenia maszynowego, od dokładnego zbierania danych. Jednak samo odnalezienie kilku tabel nie jest wystarczające. Dogłębne zrozumienie źródeł, struktur, typów danych i metod pozyskiwania ów danych jest niezbędne, aby móc przeprowadzić efektywną analizę lub stworzyć działający model klasyfikujący. Ponadto, należy również wziąć pod uwagę kwestie etyczne i prawne. Wiele danych dostaje się do przestrzeni publicznej Internetu nielegalnie, co skutkuje łatwą dostępnością do informacji, które powinny pozostać poufne lub prywatne. Dlatego warto upewnić się, że dane na których planujemy operować zostały pozyskane w sposób zgodny z przepisami prawa, bez naruszania prywatności czy własności innych osób.

Wszystkie wymienione powyżej czynności stanowią fundamentalny krok na drodze do utworzenia mechanizmu automatycznej detekcji mowy nienawiści.

## **Model uczenia maszynowego**

Przed przystąpieniem do szczegółowego opisu procesu tworzenia modelu uczenia maszynowego, niezbędne jest wprowadzenie podstawowych pojęć i terminów, które będą wykorzystane w dalszych etapach pracy.

* **NLP**

Natural Language Processing, czyli przetwarzanie języka naturalnego, jest dziedziną sztucznej inteligencji, która koncentruje się na interakcji między komputerami a ludzkim językiem. Głównym celem tego obszaru jest umożliwienie komputerom przetwarzania, interpretowania i tworzenia ludzkiego języka w formie tekstowej lub głosowej. [11]

* **Treningowy zbiór danych** (ang. Training data)

Zbiór danych używany do trenowania modelu.. Model używa tego zbioru do nauki rozpoznawania wzorców i wykonywania przewidywań. [12]

* **Testowy zbiór danych** (ang. Testing data)

Zbiór danych służący do oceny ostatecznej wydajności modelu uczenia maszynowego. Jest to niezależny zestaw danych, który nie był używany w trakcie treningu modelu i służy do sprawdzenia, jak model będzie działał w praktyce. [12]

* **Walidacyjny zbiór danych** (ang. Validation data)

Zbiór danych wykorzystywany do dostosowywania parametrów modelu uczenia maszynowego. Dzięki niemu można ocenić jakość modelu na etapie rozwoju i uniknąć zjawiska nadmiernego dopasowania (ang. overfitting). [12]

* **Walidacja krzyżowa** (ang. Cross-validation)

Walidacja krzyżowa to technika, w której zestaw danych jest dzielony na kilka równych części, określanych jako podzbiory. Algorytm uczenia maszynowego wykorzystuje do uczenia wszystkie podzbioru poza jednym. Model jest następnie trenowany na tych danych treningowych i testowany na wyizolowanym podzbiorze. Oznacza to, że każdy podzbiór danych, w pewnym momencie, jest używany jako zestaw testowy, a więc każdy model jest oceniany na danych, z którymi wcześniej nie miał do czynienia. Proces ten jest powtarzany, przy czym za każdym razem inny podzbiór jest traktowany jako zestaw testowy. W efekcie każdy fragment danych jest wykorzystywany zarówno do treningu, jak i do testowania modelu, co pozwala na dokładniejszą ocenę jego skuteczności. [13]

*Obraz zawierający tekst, zrzut ekranu, Czcionka, design

Opis wygenerowany automatycznie*Obraz 2 przedstawia wizualnie opisany powyżej podział.

Obraz 2: Schemat przedstawiający sposób podziału danych podczas stosowania walidacji krzyżowej [8]

* **NLP**

Natural Language Processing, czyli przetwarzanie języka naturalnego, jest dziedziną sztucznej inteligencji, która koncentruje się na interakcji między komputerami a ludzkim językiem. Głównym celem tego obszaru jest umożliwienie komputerom przetwarzania, interpretowania i tworzenia ludzkiego języka w formie tekstowej lub głosowej. [11]

* **Klasyfikacja** (ang. Classification)

To proces przypisywania kategorii lub klas do danych. W kontekście uczenia maszynowego, klasyfikacja polega na uczeniu modelu przyporządkowania nowych zestawów danych do jednej z N kategorii. [12]

* **Uczenie nadzorowane** (ang. Supervised learning)

W uczeniu nadzorowanym model jest trenowany na podstawie danych etykietowanych. Oznacza to, że każdemu przykładowi w danych treningowych przypisana jest informacja, której model ma nauczyć się przewidywać. [12]

* **Uczenie nienadzorowane** (ang. Unsupervised learning)

Rodzaj uczenia maszynowego, gdzie model pracuje na nieoznaczonych, nieskategoryzowanych danych. Celem jest znalezienie ukrytych wzorców lub struktur w nieetykietowanych danych bez wcześniejszej znajomości wyników. [12]

* **Macierz pomyłek** (ang. Confusion Matrix)

Tabela, która jest stosowana do oceny jakości modelu klasyfikacyjnego. Przedstawia liczbę poprawnych i błędnych przewidywań, podzielonych według rzeczywistych i przewidywanych kategorii, tak jak na obrazie 3:

Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

Obraz 3: Rysunek przedstawiający strukturę typowej macierzy pomyłek

* + True Positives : Przypadki, których przewidziana wartość to PRAWDA i jest to zgodne z rzeczywistością
  + True Negatives : Przewidziana wartość to FAŁSZ i jest to zgodne z rzeczywistością
  + False Positives : Przewidziano wartość PRAWDA, jednak w rzeczywistości jest to FAŁSZ
  + False Negatives : Przewidziano wartość FAŁSZ, jednak w rzeczywistości jest to PRAWDA
* **Skuteczność** (ang. Accuracy)

Miara oceniająca ogólną wydajność modelu, wyrażona jako stosunek liczby poprawnych przewidywań do całkowitej liczby przewidywań.

|  |  |  |
| --- | --- | --- |
|  |  |  |

* **Precyzja** (ang. Precision)

Miara oceniająca jak duża część identyfikacji modelu jako pozytywnych jest faktycznie prawdziwie pozytywna.

|  |  |  |
| --- | --- | --- |
|  |  |  |

* **Czułość** (ang. Recall)

Określa jaka część rzeczywistych pozytywnych przypadków została poprawnie zidentyfikowana przez model.

|  |  |  |
| --- | --- | --- |
|  |  |  |

* **F1 Score**

Statystyczna miara dokładności testu, która bierze pod uwagę zarówno precyzję, jak i czułość. Jest to harmoniczna średnia z precyzji i czułości, gdzie najlepsza wartość wynosi 1, a najgorsza 0. F1 Score jest szczególnie przydatny, gdy niezbilansowane klasy rozkładów w danych mogą powodować, że inne miary, takie jak skuteczność, są mylące.

|  |  |  |
| --- | --- | --- |
|  |  |  |

Podczas pracy nad tworzeniem modelu uczenia maszynowego w obszarze przetwarzania języka naturalnego (NLP), kluczowym etapem jest zebranie odpowiedniego zestawu danych. Stanowią one podstawę dla wszystkich późniejszych kroków i muszą być odpowiednio dobrane, aby odzwierciedlać różnorodność rzeczywistych przypadków, z którymi model będzie miał do czynienia. Dobrej jakości dane pozwalają na uniknięcie zarówno błędów wynikających z jednostronności czy przekłamań, jak i wielu problemów podczas implementacji.

Kolejny etap to przygotowanie danych, które poddawane są takim procesom jak: czyszczenie, odpowiednie przekształcenia, tokenizacja, lematyzacja. Etap ten opisany został szczegółowiej w następnym podrozdziale.

Następnie budowa modelu rozpoczyna się od wyboru odpowiedniego algorytmu, który jest zdolny uchwycić niuanse języka ludzkiego i skutecznie modelować zadanie, takie jak detekcja mowy nienawiści. Wybór odpowiedniej metody zależy od specyfiki zadania i charakterystyki danych, a także od wymagań stawianych przez finalne zastosowanie modelu.

Po skonstruowaniu modelu przystępuje się do jego trenowania- procesu w większości przypadków wymagającego dużej ilości danych i zasobów obliczeniowych. Trenowanie jest przeprowadzane w sposób, który umożliwia modelowi nie tylko „naukę” na podstawie danych treningowych, ale także generalizację na nowych, nieznanych danych.

Kiedy model jest już wytrenowany, przechodzi on fazę testowania. To etap, na którym oceniana jest jego wydajność, zazwyczaj za pomocą wcześniej niewidzianych danych (zbiór testowy), co pozwala na weryfikację, jak dobrze model radzi sobie w praktycznych zastosowaniach. Analiza wyników jest kluczowa, ponieważ dostarcza informacji zwrotnej niezbędnej do dalszej optymalizacji modelu. W procesie optymalizacji modelu na podstawie wyników testów, dokonuje się korekt w architekturze modelu, wartościach parametrów lub procesie treningowym. Cel stanowi tu poprawa metryk takich jak dokładność, precyzja czy F1-score, które świadczą o jakości utworzonego modelu.

Ostatnim etapem jest wdrożenie modelu do realnego środowiska, gdzie może on być wykorzystywany do wykonywania zadań, dla których został utworzony.

Schemat znajdujący się na następnej stronie pracy w przystępny sposób odzwierciedla opisany powyżej proces.

*Obraz zawierający tekst, zrzut ekranu, diagram, linia

Opis wygenerowany automatycznie* Obraz *4: Typowy przepływ pracy podczas tworzenia modelu uczenia maszynowego [8]*

## **Podstawowe techniki przetwarzania tekstu**

W przypadku tworzenia modeli przetwarzania języka naturalnego (NLP) bardzo istotnym krokiem jest odpowiednie przygotowanie dostępnych danych. Etap ten często jest określany jako preprocessing i obejmuje szereg działań, których celem jest przekształcenie tekstu w surowej postaci do formatu przydatnego dla algorytmów uczenia maszynowego. Poniżej omówione zostały podstawowe techniki przetwarzania tekstu i związane z nimi pojęcia:

* **Czyszczenie Danych**

Polega na usunięciu z tekstu wszelkich niepotrzebnych lub nieistotnych informacji, przekształcaniu tekstu do spójnego formatu i usuwaniu wszelakich nieprawidłowości, które mogłyby zaburzyć wyniki analizy, lub wręcz uniemożliwić działanie algorytmom. Najczęściej obejmuje to usunięcie znaków interpunkcyjnych, niepotrzebnych białych znaków, cyfr, symboli specjalnych oraz konwersję liter na małe, aby uniknąć rozróżnienia tych samych słów napisanych w różnej formie. Czyszczenie danych pomaga w redukcji złożoności danych, poprawia ich jakość i zwiększa efektywność kolejnych etapów przetwarzania tekstu. [14]

* **Tokenizacja**

Proces dzielenia tekstu na mniejsze jednostki, zwane tokenami. Tokeny mogą być pojedynczymi słowami, zdaniami lub nawet częściami słów. Taki podział umożliwia przekształcenie tekstu w strukturę, którą może być łatwiej analizować i przetwarzać, np. poprzez grupowanie według kategorii takich jak: części mowy. [15]

* **Lematyzacja**

Proces przekształcania słów do ich podstawowej formy, czyli lematów (brak odmiany przez przypadki, osoby, formy czasu i tym podobne). Dzięki temu możliwe jest przetworzenie różnych form słowa, takich jak „pisał” i „pisze”, na jedną formę podstawową – „pisać”. Lematyzacja jest szczególnie ważna w językach, w których dany wyraz może być odmieniony na wiele sposobów, takich jak polski. [15]

* **Stop Words**

Zbiór słów typu „i”, „ale”, „to”, „żeby” i tym podobnych. Są to często występujące słowa, które w większości przypadków nie wnoszą znaczącej wartości semantycznej. Często usuwa się je z tekstu aby zmniejszyć złożoność danych, poprawiając tym samym skuteczność algorytmów. [16]

* **Wektoryzacja**

Proces przekształcania tekstu na wektory liczby rzeczywistych, które mogą być użyte jako dane wejściowe dla algorytmów uczenia maszynowego.. Istnieje wiele metod wektoryzacji- w tej pracy wybrano dwa z nich: Bag of Words oraz TF-IDF. [15]

* **BoW (Bag of Words)**

Technika wektoryzacji, w której tekst jest przekształcany w wektor, gdzie każdy element odpowiada częstotliwości występowania danego słowa w tekście, ignorowana natomiast jest ich kolejność [17]. Metoda ta jest stosunkowo prosta zarówno koncepcyjnie jak i implementacyjnie, charakteryzuje się jednak niską precyzją i stosunkowo dużą podatnością na błędnie sklasyfikowanie treści. Na przykład, jeśli użytkownik umieści w swojej wypowiedzi cytat zawierający ironię lub obraźliwe określenie, cała wypowiedź może zostać uznana przez model za treść toksyczną, gdyż model nie analizuje głębszego znaczenia wypowiedzi.

* **TF-IDF (Term Frequency-Inverse Document Frequency)**

Bardziej rozbudowana wersja metody Bag of Words. Skupia się dodatkowo na ważności słowa w kontekście całego tekstu. W zależności od częstotliwości występowania słów jest im nadawane odpowiednie znaczenie. Pozwala to między innymi zredukować wpływ często występujących słów, które nie wnoszą przydatnych informacji, np. zaimków czy spójników. [17]

## **Modele klasyfikacji w NLP**

Przetwarzanie języka naturalnego (NLP) można podzielić na dwie częściowo pokrywające się dziedziny:

* Rozumienie języka naturalnego, które koncentruje się na analizie semantycznej lub określaniu znaczenia danego tekstu.
* Generowanie języka naturalnego, które koncentruje się na generowaniu tekstu przez odpowiedni model. [18]

W tej pracy mamy do czynienia z klasyfikacją tekstu, w celu określenia toksyczności danej treści, a zatem kwalifikuje się to do pierwszej z wymienionych dziedzin. Istnieje wiele modeli uczenia maszynowego, które znajdują zastosowanie w przetwarzaniu języka naturalnego w celu jego „zrozumienia”. Wybrane zostały modele: Regresja Logistyczna, Maszyny Wektorów Nośnych (SVM) oraz Naiwny Klasyfikator Bayes’a. Każdy z dostępnych modeli posiada swoje unikalne cechy i najlepiej sprawdza się w różnych scenariuszach. Poniżej krótko opisano wybrane modele:

* **Regresja Logistyczna**

Regresja logistyczna to nadzorowany algorytm klasyfikacji, którego celem jest przewidzenie prawdopodobieństwa wystąpienia zdarzenia na podstawie danych wejściowych [18]. Jest stosowana w klasyfikacji binarnej i opiera się na funkcji sigmoidalnej. Jest to model prosty w implementacji i interpretacji, który znajduje zastosowanie w wielu problemach klasyfikacyjnych w NLP.

* **Maszyny Wektorów Nośnych (SVM)**

SVM jest nadzorowanym algorytmem uczenia maszynowego, który jest stosowany zarówno do klasyfikacji, jak i regresji. W kontekście klasyfikacji tekstu, SVM kategoryzuje klasy danych poprzez określenie najlepszej hiperpłaszczyzny, która dzieli dane tekstowe na zdefiniowane wcześniej grupy. Algorytm SVM tworzy wiele hiperpłaszczyzn, a celem jest znalezienie najlepszej hiperpłaszczyzny, która dokładnie dzieli obie klasy. W tym celu wybierana jest hiperpłaszczyzna o maksymalnej odległości od punktów danych obu klas. Wektory lub punkty danych bliżej hiperpłaszczyzny nazywane są wektorami nośnymi, które w dużym stopniu wpływają na położenie i odległość optymalnej hiperpłaszczyzny. [19]

* **Naiwny Klasyfikator Bayesa**

Naive Bayes to nadzorowany algorytm klasyfikacji, który znajduje rozkład prawdopodobieństwa warunkowego na podstawie następującego wzoru Bayesa [20]:

|  |  |  |
| --- | --- | --- |
|  |  |  |

, gdzie:

P(A|B) – prawdopodobieństwo wydarzenia A, pod warunkiem B,

P(A|B) – prawdopodobieństwo wydarzenia B, pod warunkiem A,

P(A) – prawdopodobieństwo wydarzenia A,

P(B) – prawdopodobieństwo wydarzenia B

Klasyfikuje teksty opierając się na prawdopodobieństwu występowania zdarzeń. Naiwnym założeniem modelu Naive Bayesa jest to, że poszczególne słowa są niezależne. [18]

Wybór odpowiedniego modelu dla danego problemu klasyfikacji w NLP zależy od specyfiki zadania, jakości i ilości dostępnych danych, jak również od wymagań dotyczących wydajności i interpretowalności modelu. Wybrane modele stanowią popularne i sprawdzone rozwiązania w tym obszarze, a zastosowanie kilku modeli daje możliwość porównywania wyników i efektywności poszczególnych algorytmów.

# **Rozdział 3: Opis opracowanego programu**

Niniejszy rozdział poświęcony został przedstawieniu opracowanego rozwiązania. Utworzona została aplikacja graficzne, umożliwiająca użytkownikom przeskanowanie wybranej sekcji komentarzy na portalu YouTube w celu wykrycia potencjalnie szkodliwej treści.

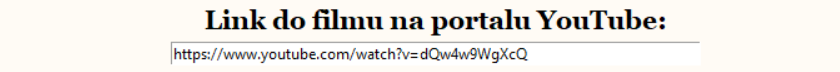
Na następnej stronie znajduje się główny ekran aplikacji (obraz 5), który zostaje wyświetlony po uruchomieniu programu.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, Strona internetowa

Opis wygenerowany automatycznie *Obraz 5: Główny ekran utworzonej aplikacji*

Interfejs użytkownika został zaprojektowany z myślą o prostocie i intuicyjności. Poniżej zostały omówione poszczególne elementy i fragmenty aplikacji, wraz z

* **Pole do wprowadzenia adresu URL (obraz 6 oraz obraz 7)**



*Obraz 6: Pole interfejsu aplikacji przeznaczone do zamieszczania adresu URL*

Użytkownik ma możliwość wkleić w to pole link do wybranego filmu na portalu YouTube, w celu przeanalizowania sekcji komentarzy. Uwzględnione zostały przypadki kiedy użytkownik poda link zawierający odnośnik do konkretnego fragmentu filmu. W przypadku wprowadzenia błędnego adresu użytkownik zostanie o tym poinformowany:

Obraz zawierający tekst, Czcionka, zrzut ekranu, linia

Opis wygenerowany automatycznie

*Obraz 7: Przykład błędnego wprowadzenia adresu URL*

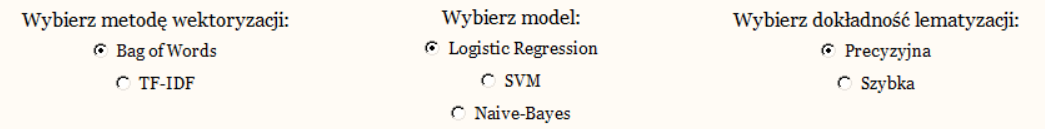
* **Limit komentarzy oraz tytuł filmu (obraz 8)**



*Obraz 8: Pola interfejsu pozwalające ograniczyć ilość pobieranych komentarzy, oraz wyświetlające tytuł filmu*

Aby nie pobierać dla każdego filmu tysięcy komentarzy wprowadzono możliwość ograniczenia ilości pobieranych komentarzy. Oprócz tego po pobraniu komentarzy program wyświetli tytuł wskazanego filmu.

* **Wybór parametrów (obraz 9)**



*Obraz 9: Pola interfejsu dające możliwość wybrania poszczególnych parametrów*

Analizy komentarzy można dokonać dobierając 3 parametry. Są to kolejno: wybór metody wektoryzacji tekstu, wybór modelu NLP oraz wybór dokładności lematyzacji.

* **Przycisk „Skanuj komentarze” (obraz 10)**

Obraz zawierający tekst, Czcionka, logo, biały

Opis wygenerowany automatycznie

*Obraz 10: Przycisk uruchamiający główną funkcjonalność programu*

Pod tym przyciskiem kryje się mechanizm wykonujący większość operacji. Wywołana zostaje kluczowa funkcja programu, która zostanie dokładniej opisana w następnym rozdziale. Na podstawie podanych przez użytkownika informacji (adres URL oraz parametry) zostaje pobrana zawartość sekcji komentarzy pod wybranym filmem na portalu YouTube. Następnie jest ona analizowana i poszczególnym komentarzom przypisywana jest etykieta „toksyczny” lub „neutralny”. Klasyfikacja komentarzy odbywa się dzięki zastosowaniu modeli NLP, które zostały odpowiednio wytrenowane na prawie 40 000 przykładowych komentarzach.

* Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

  Opis wygenerowany automatycznie**Pole wyświetlające rezultaty (obraz**

*Obraz 11: Fragment interfejsu wyświetlający sklasyfikowane komentarze*

Po wciśnięciu przycisku „Skanuj komentarze” w tej sekcji pojawiają się rezultaty działania programu w formacie tabeli. W kolumnie „Komentarz” możemy przeczytać jak użytkownicy komentowali dany film. Natomiast w kolumnie „Toksyczność” znajdziemy informację, czy dany wpis został uznany przez model NLP za toksyczny czy też nie. Dodatkowo komentarze uznane za toksyczne zostały wyróżnione poprzez wytłuszczenie czcionki i podświetlenie wpisu wyróżniającym się kolorem.

# **Rozdział 4: Implementacja programu**

W tym rozdziale skoncentrowano się na implementacji programu, kluczowym elemencie tej pracy. Rozdział ten został podzielony na pięć podrozdziałów, które szczegółowo omawiają poszczególne etapy implementacji. Kolejno opisują:

1. Wykorzystane narzędzia wraz z ich krótkim opisem
2. Podłączenie do API portalu YouTube- proces nawiązania połączenia z interfejsem programistycznym YouTube, co umożliwiło dostęp do danych z tej platformy.
3. Pobieranie, przetwarzanie i przygotowanie danych- omówienie procesu zbierania danych z YouTube oraz ich odpowiedniego przetwarzania, tak aby były gotowe do analizy.
4. Tworzenie modeli przetwarzania języka naturalnego- opisano etapy tworzenia modeli przetwarzania języka naturalnego, których rolą jest identyfikacja mowy nienawiści w tekście.
5. Proces tworzenia aplikacji zawierającej interfejs graficzny

## **4.1 Narzędzia**

Część techniczną projektu rozpoczęto od instalacji Anaconda Navigator, który cieszy się popularnością w projektach powiązanych z analizą danych. Zawiera on w sobie język programowania Python wraz z licznie dostępnymi bibliotekami oraz środowisko Jupyter Notebook. Poniżej krótki opis poszczególnych narzędzi:

* **Anaconda Navigator**

Jest to graficzny interfejs użytkownika (GUI), który upraszcza zarządzanie środowiskami Pythona i pakietami [21]. Zapewnia również dostęp do popularnych narzędzi do analizy danych oraz zintegrowanych środowisk programistycznych (IDE)

* **Python**

Interpretowany, obiektowy język programowania wysokiego poziomu z dynamiczną semantyką. Jego wbudowane struktury danych wysokiego poziomu w połączeniu z dynamicznym pisaniem i dynamicznym wiązaniem czynią go bardzo atrakcyjnym do szybkiego tworzenia aplikacji, a także do wykorzystania jako język skryptowy lub klejący do łączenia ze sobą istniejących komponentów. Prosta, łatwa do nauczenia składnia Pythona podkreśla czytelność i dlatego zmniejsza koszty utrzymania programu. [22]

* **Jupyter Notebook**

To aplikacja webowa typu open source, która umożliwia tworzenie i udostępnianie dokumentów zawierających kod, równania, wizualizacje oraz tekst. Jest intuicyjna w użyciu i umożliwia wykonanie kodu partiami (komórkami), co ułatwia proces tworzenia algorytmów. [23]

W ramach projektu skorzystano z następujących modułów języka Python, kluczowych dla analizy danych i uczenia maszynowego:

* NumPy: Podstawowa biblioteka do obliczeń numerycznych i algebry liniowej.
* Scikit-learn: Bogata biblioteka do uczenia maszynowego, zawierająca różnorodne algorytmy klasyfikacji, regresji i grupowania, które mogą być wykorzystane do detekcji mowy nienawiści.
* Matplotlib: Moduł do tworzenia wykresów i wizualizacji
* Pandas: Biblioteka dostarczająca funkcje do analizy i manipulacji danymi. Oprócz dostarczania elastycznych i wydajnych struktur danych, takich jak DataFrame, oferuje również bogaty zestaw funkcji do czyszczenia, transformacji i analizy danych.
* SpaCy: Biblioteka do przetwarzania języka naturalnego. Udostępnia narzędzia do takich zadań jak lematyzacja czy tokenizacja
* Regex: Biblioteka udostępniająca funkcje do operowania na tekście za pomocą wyrażeń regularnych, jak np. wyszukiwanie wzorców.
* Tkinter: Biblioteka umożliwiająca tworzenie interfejsów graficznych

Każde z tych narzędzi wnosi kluczowe funkcjonalności niezbędne do efektywnego przeprowadzenia procesu detekcji mowy nienawiści w Internecie, od wstępnej obróbki danych, poprzez analizę i modelowanie, aż po wizualizację wyników.

Dodatkowo, w ramach projektu wykorzystano GitHub, platformę hostingową wykorzystującą system kontroli wersji Git, do zarządzania kodem źródłowym. GitHub jest standardem w branży dla kontroli wersji oraz współpracy, oferując możliwości śledzenia postępów w pracy, dokumentacji zmian oraz współpracy z innymi użytkownikami. Dla ułatwienia obsługi repozytoriów przez interfejs graficzny, zastosowano również GitHub Desktop. To narzędzie pozwala na wykonywanie operacji na repozytorium bez konieczności używania wiersza poleceń, co może znacząco przyspieszyć pracę i uczynić proces zarządzania kodem bardziej intuicyjnym.

## **4.2 YouTube API**

Proces podłączenia do API portalu YouTube oraz ekstrakcji danych w postaci komentarzy został zrealizowany poprzez wykonanie następujących kroków:

1. Założenie konta Google: Rozpoczęto od utworzenia nowego konta Google.
2. Użycie Google Developers Console: Następnie w Google Developers Console wybrano "APIs and services/Library", a następnie "YouTube Data API v3" i utworzono nowy projekt.
3. Generowanie klucza API: Klucz API został wygenerowany i umieszczony w zmiennej środowiskowej w celu zapewnienia bezpieczeństwa i ochrony klucza przed umieszczeniem go w kodzie źródłowym.
4. Instalacja biblioteki klienta API Google: Przeprowadzono instalację oficjalnej biblioteki Google wspierającej obsługę API za pośrednictwem języka Python.
5. Implementacja funkcji do ekstrakcji komentarzy: Zaimplementowano funkcję w Pythonie, która wykorzystuje metody zawarte w bibliotece google-api-python-client do pobierania komentarzy z filmów na YouTube.
6. Obsługa polskich znaków: Podczas zapisu danych do pliku w formacie Excel zastosowano kodowanie "utf-8", aby uniknąć problemów z polskimi znakami.
7. Ograniczenie liczby komentarzy: W celu ominięcia ograniczenia API YouTube dotyczącego liczby zapytań, które wynosi 1000, wprowadzono metody umożliwiające kontynuację zapytań po osiągnięciu tego limitu. **[TODO]**

Kroki te pozwoliły na skuteczne połączenie się z API YouTube i realizację procesu ekstrakcji danych niezbędnych do analizy mowy nienawiści w komentarzach.

Poniżej znajduje się krótki opis najistotniejszych funkcji w tej sekcji:

1. **get\_id( )** – funkcja służy do analizy adresu URL YouTube w celu wyodrębnienia unikalnego identyfikatora wideo.
2. **extract\_comments( )** – funkcja pobiera komentarze z danego filmu na YouTube, identyfikowanego przez video\_id, z opcjonalnym ograniczeniem liczby pobieranych komentarzy do wartości limit. Tworzy ona połączenie z YouTube API, pobiera wątki komentarzy do maksymalnie 100 i iteracyjnie zbiera zarówno główne komentarze, jak i odpowiedzi na nie, aż do osiągnięcia określonego limitu lub końca dostępnych danych.

Obraz zawierający tekst, zrzut ekranu, diagram, Czcionka

Opis wygenerowany automatyczniePonizszy schemat prezentuje działanie tej funkcji:

*Obraz 12: Schemat przedstawiający działanie funkcji extract\_comments()*

## **4.3 Przygotowanie i przetwarzanie danych**

Aby rozpocząć tworzenie modelu NLP służącego do detekcji mowy nienawiści, konieczne jest pozyskanie odpowiednich danych, spełniających określone wymagania dotyczące zawartości, struktury, wielkości oraz wiarygodności. Po wstępnej weryfikacji jakości pozyskanych danych, przystąpiono do ich dostosowywania do potrzeb i ujednolicenia. Następnie przeprowadzono potrzebne modyfikacje i procesowanie, takie jak czyszczenie czy tokenizacja, aby uzyskać finalną strukturę danych, na podstawie której będzie można opracować model uczenia maszynowego. Poniżej przedstawiono szczegółowy opis poszczególnych etapów.

### **Zbiory danych**

Pierwszym krokiem było zdefiniowanie potrzeb i kryteriów jakie musiały spełniać poszukiwane zbiory danych:

* **Licencja**

Zbiór danych musi być dostępny do użytku.

* **Zawartość**

Potrzebne są dane zawierające przykładowe komentarze lub wpisy z Internetu, które zostały już sklasyfikowane na grupy toksyczne oraz nie-toksyczne, i odpowiednio zetykietyzowane.

* **Struktura danych**

Plik musi zawierać co najmniej dwie kolumny: treść komentarza lub wpisu oraz oznaczenie, czy dany tekst został uznany za toksyczny.

* **Język polski**

Treść komentarzy w zbiorach danych musi być w języku polskim.

* **Obszerność**

Aby stworzyć model o wysokiej dokładności, konieczna jest odpowiednia ilość danych. Sumaryczna liczba rekordów powinna wynosić przynajmniej kilka, jeśli nie kilkanaście tysięcy.

* **Jakość danych**

Należy zweryfikować, czy zawarta w danych klasyfikacja tekstów na toksyczne i nie-toksyczne jest prawidłowa. Oczywiście nie jest wykonalnym sprawdzenie ręcznie kilku tysięcy przypadków, jednak warto upewnić się, czy etykiety nie zostały np. przypisane losowo, lub sprawdzić poprawność etykietowania na małej próbce np. 100 rekordów.

Po utworzeniu powyższej listy kryteriów przeszukano Internet w celu odnalezienia odpowiednich zbiorów danych. Oto lista źródeł danych, które zostały wykorzystane podczas tworzenia modelu NLP:

* **Zbiór danych z witryny PolEval** [24]**– edycja 2019, zadanie 6- „Automatic cyberbullying detection”**

Dane są zebranymi z portalu Twitter wpisami, liczące 10 041 komentarzy użytkowników. Zawiera 851 toksycznych przypadków

* **Zbiór danych z witryny huggingface.co** [25]**– „hatecheck-polish”**

Składa się z 3816 rekordów, w tym 2807 sklasyfikowanych jako toksyczne. Znajdujące się tu wyrażenia to proste, często powtarzające się zdania. Przykładowo zawarte jest tu 7 wpisów o identycznej strukturze: „Nienawidzę [wybrana grupa społeczna]”.

* **Zbiór danych z witryny github.com** [26] **– „BAN-PL: a Polish Dataset of Banned Harmful and Offensive Content from Wykop.pl Web Service”**

Zawiera 24 000 zebranych komentarzy z serwisu wykop.pl, które zostały zanonimizowane. Połowę z nich stanowią zwykłe, neutralne komentarze o różnym znaczeniu, natomiast pozostała część to wpisy „szkodliwe”, za których publikację autorom zostały zablokowane konta przez administratorów serwisu

Pobrane pliki wymagały przeprowadzenia kilku operacji. Pierwszą z nich było zmienienie kodowania na „UTF-8”, co pozwoliło na poprawne wyświetlanie polskich znaków. Następnie usunięto zbędne elementy i ujednolicono nazewnictwo kolumn, aby zachować spójność. Pliki połączono w jedną tabelę i zapisano jako nowy plik, który posłuży do dalszych prac nad modelem. Dodano również kolumnę wskazującą źródło każdego rekordu. Po tych operacjach dane były gotowe do dalszego przetwarzania.

### **Przetwarzanie danych**

Etap przetwarzania danych rozpoczęto od przeczyszczenia danych. W ramach tego procesu przeprowadzono następujące czynności:

* Usunięto brakujące wartości
* Zamieniono wszystkie duże litery na małe
* Za pomocą wyrażeń regularnych z biblioteki języka Python (regex), usunięto wszystkie znaki, które nie są spacją, liczbą, literą lub znakiem podkreślenia.

Po wykonaniu powyższych działań przystąpiono do lematyzacji danych tekstowych, czyli w skrócie przekształcenia słów do ich podstawowej, leksykalnej formy. W tym celu wykorzystano funkcje z biblioteki „SpaCy”, która oferuje zaawansowane narzędzia do przetwarzania języka naturalnego.

Powyższe działania miały na celu usunięcie błędnych i zbędnych elementów, takich jak symbole i znaki interpunkcyjne, oraz ujednolicenie postaci, w jakiej występują poszczególne słowa. Efektem tych działań jest większa odporność na potencjalne błędy i zmniejszenie złożoności danych, co często przekłada się na zwiększenie efektywności modelu.

### **Tokenizacja – Bag of Words & TF-IDF**

Ostatnim etapem przygotowania danych przed przystąpieniem do tworzenia modelu NLP jest tokenizacja. Dzięki rozbudowanym i bogatym bibliotekom języka Python, przeprowadzenie procesu tokenizacji nie wymaga implementacji algorytmu procesowania danych od podstaw. Wykorzystano funkcje CountVectorizer (dla metody Bag of Words) oraz TfidfVectorizer (dla metody TF-IDF) ,z biblioteki Scikit-learn, do utworzenia obiektów wektoryzacji. Została opracowana funkcja pozwalająca wybrać, która metoda zostanie wykorzystana do tokenizacji danych.

Warto wspomnieć o jednym z argumentów wymienionych funkcji, czyli *stop\_words.* Jest to zbiór słów często występujących w danym języku, nie wnoszących natomiast istotnej wartości semantycznej. Ich usunięcie pozwala na redukcję złożoności danych i zwiększenie efektywności modeli klasyfikacyjnych, poprzez skupienie się na bardziej istotnych fragmentach tekstów.

W przypadku języka polskiego nie występuje wbudowany zasób takich słów, jednakże wykorzystany został zbiór dostępny na stronie GitHub [27]. Zbiór ten zawiera 351 słów, w tym wyrazy jak na przykład: „a”, „aby”, „bo”, „ci”, „ich”, „które”, „niż”, „oni”, „taki”, „że”, „żeby”.

## **4.4 Modele NLP**

Do zaimplementowania modeli wykorzystano funkcje dostępne w pakiecie Scikit-learn. Są to:

* **Model Regresji Logistycznej**

Funkcja LogisticRegression z pakietu sklearn.linear\_model

* **Model SVM**

Funkcja SVC z pakiety sklearn.svm

* **Model Naive-Bayes**

Funkcja MultinomialNB z pakietu sklearn.naive\_bayes

Obraz zawierający tekst, zrzut ekranu, Czcionka, Równolegle

Opis wygenerowany automatycznieCały mechanizm obejmujący tokenizację, utworzenie odpowiedniego modelu uczenia maszynowego, jego trenowania, predykcji wyników i procesów pośrednich został zrealizowany w postaci jednej z kluczowych funkcji- *make\_prediction()*, poniżej została ona opisana w uproszczony sposób w postaci schematu:

*Obraz 13: Schemat przedstawiający działanie funkcji make\_predictions()*

## **4.5 Aplikacja – funkcjonalności i interfejs**

W celu utworzenia interfejsu graficznego pozwalającego użytkownikowi na obsługę utworzonego oprogramowania użyto wbudowanej biblioteki języka Python – Tkinker. Oferuje ona szereg gotowych rozwiązań, w tym różne widżety takie jak: przyciski, etykiety, pola tekstowe, a także odpowiednie funkcje pozwalające na ich modyfikowanie i stylizację. [28].

**Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie** W ramach kodu odpowiedzialnego za tworzenie interfejsu graficznego zaimplementowano kluczową funkcję scan\_comment\_section(), wywoływaną poprzez kliknięcie przez użytkownika klawisza „Skanuj komentarze”. Wykorzystuje ona elementy zaimplementowane w poprzednich etapach implementacji – pobiera komentarze z portalu YouTube, odpowiednio je przetwarza i przekształca, wczytuje wytrenowane modele NLP i na ich podstawie dokonuje klasyfikacji pobranych komentarzy na kategorie „toksyczny” oraz „neutralny”. Na końcu zwraca komentarze wzbogacone o etykietę „Toksyczność”. Proces ten został w uproszczeniu przedstawiony na poniższym schemacie:

*Obraz 14: Schemat przedstawiający działanie funkcji scan\_comment\_section()*

# **Rozdział 5: Eksperymenty i testy**

## **5.1 Porównanie wydajności modeli**

W projekcie zostały utworzone 3 modele NLP:

* Regresja Logistyczna
* SVM
* Naive-Bayes.

Oprócz tego dostępne są dwie metody tokenizacji tekstu:

* Bag of Words
* TF-IDF

Użytkownik korzystający z opracowanej aplikacji może dowolnie wybrać te parametry. Konsekwencją tego jest 6 istniejących konfiguracji parametrów, na podstawie których może zostać dokonana detekcja toksycznej treści w komentarzach.

Modele można porównywać między sobą bazując na wartościach wybranych metryk, w pracy skupiono się głównie na skuteczności, F1-score, oraz średniej skuteczności z walidacji krzyżowej. W ramach przeprowadzenia testów wydajności poszczególnych modeli przeprowadzone zostały eksperymenty w dwóch scenariuszach.

* **Predykcja danych ze zbioru z witryny huggingface.co** [25]**– „hatecheck-polish”**

Opis zbioru danych został zamieszczony w rozdziale 4.3, w podpunkcie „a)”. Ze względu na względną prostotę zawartych wpisów zbiór ten stanowi dobry test dla modeli. Pozwolił on sprawdzić jakość działania modeli dla przypadku mało skomplikowanych wpisów.

W tabeli 1 zostały zebrane wyniki działania poszczególnych modeli, wyrażone w postaci metryk, używanych standardowo do ocenienia jakości predykcji.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Model | Metoda tokenizacji | Skuteczność | Precyzja | Czułość | F1 Score | Średnia skuteczność z walidacji krzyżowej |
| Regresja Logistyczna | Bag of Words | 95.4 % | 94.3 % | 99.6 % | 96.9 % | 93.5 % |
| SVM | Bag of Words | 95.8 % | 94.5 % | 100% | 97.2 % | 93.8 % |
| Naive-Bayes | Bag of Words | 95.3 % | 94.6 % | 99.1 % | 96.8% | 93.8 % |
| Regresja Logistyczna | TF-IDF | 93.2 % | 91.4 % | 100 % | 95.5 % | 89.8 % |
| SVM | TF-IDF | 95.9 % | 94.8 % | 99.8 % | 97.3 % | 94.8 % |
| Naive-Bayes | TF-IDF | 94.6 % | 93.4 % | 99.6 % | 96.4 % | 91.5 % |

*Tabela 1: Porównanie wyników poszczególnych modeli*

Rezultaty są bardzo zadowalające – wszystkie metryki osiągają bardzo wysokie wartości, a nawet znalazły się przypadki gdzie czułość jest na poziomie 100%. Warto zwrócić uwagę właśnie na tę metrykę – w żadnym przypadku jej wartość nie spadła poniżej 99%, co oznacza, że modele bez trudu radzą sobie z odnajdywaniem wpisów toksycznych.

Różnice między poszczególnymi modelami i metodami tokenizacji są nieznaczne, jednak jest możliwe wyciągnięcie kilku wniosków:

* + Wybór metody tokenizacji Bag of Words skutkował osiągnięciem (uśredniając) lepszych rezultatów niż w przypadku metody TF-IDF
  + Model SVM osiągnął najlepsze wyniki, jednak różnice pomiędzy poszczególnymi modelami NLP są bardzo niewielkie
* **Wszystkie dostępne dane (3 zbiory danych)**

W tym scenariuszu modele są testowane przy wykorzystaniu wszystkich dostępnych zbiorów danych. Dane są zatem dużo bardziej zróżnicowane i zbliżone do postaci, z którą można spotkać się na portalach społecznościowych. Wyniki tego testu są powinny zatem być zbliżone

Opis zbioru danych został zamieszczony w rozdziale 4.3, w podpunkcie „a)”. Ze względu na względną prostotę zawartych wpisów zbiór ten stanowi dobry test dla modeli. Pozwolił on sprawdzić jakość działania modeli dla przypadku mało skomplikowanych wpisów.

W tabeli 2 zostały zebrane wyniki działania poszczególnych modeli, analogicznie do tabeli 1.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Model | Metoda tokenizacji | Skuteczność | Precyzja | Czułość | F1 Score | Średnia skuteczność z walidacji krzyżowej |
| Regresja Logistyczna | Bag of Words | 86.3 % | 87.7 % | 77.3 % | 82.2 % | 85.7 % |
| SVM | Bag of Words | 83 % | 89.3 % | 66.4 % | 76.2 % | 81.9 % |
| Naive-Bayes | Bag of Words | 84.1 % | 77.8 % | 85.6 % | 81.4 % | 83.5 % |
| Regresja Logistyczna | TF-IDF | 84.6 % | 89.6 % | 70.6 % | 78.9 % | 83.8 % |
| SVM | TF-IDF | 86.1 % | 91.2 % | 72.4 % | 81 % | 85 % |
| Naive-Bayes | TF-IDF | 83.5 % | 89.1 % | 68.1 % | 77.2 % | 83 % |

*Tabela 2: Porównanie wyników poszczególnych modeli*

Tym razem, zgodnie z oczekiwaniami, jakość predykcji jest niższa niż w poprzednim scenariuszu. Jednak pomimo znacznie większej złożoności i zróżnicowania danych modele poradziły sobie zaskakująco dobrze.

W tym przypadku obie metody tokenizacji osiągnęły dość porównywalne wyniki, ciężko wskazać faworyta. Wśród modeli najlepszy wynik uzyskała regresja logistyczna, jednak jest to różnica rzędu pojedynczych punktów procentowych. Wśród sprawdzonych przypadków najgorzej poradziły sobie pary parametrów: SVM i Bag of Words oraz Naive-Bayes i TF-IDF.

## **5.2 Testy jednostkowe**

Podczas tworzenia projektu zaimplementowano testy jednostkowe, które umożliwiały ciągłą weryfikację poprawności wprowadzanych zmian w kodzie. Utworzone na platformie GitHub repozytorium zostało zintegrowane z narzędziem Travis CI, co pozwoliło na wykorzystanie mechanizmu „Ciągłej Integracji” (ang. Continuous Integration). Dzięki temu, każdorazowe wprowadzenie zmian w repozytorium - proces znany jako „commit” - inicjowało automatyczne testowanie kodu. Na obrazie 15 przedstawiono przykładowe testy jednostkowe dla kolejnych wersji kodu, udokumentowane za pomocą Travis CI.

Obraz zawierający tekst, Czcionka, linia, zrzut ekranu

Opis wygenerowany automatycznie

*Obraz 15: Fragment sekcji „Build history” w narzędziu Travis CI*

# **Podsumowanie**

Celem pracy inżynierskiej było opracowanie klasyfikatora, który, wykorzystując techniki przetwarzania języka naturalnego (NLP) będzie rozpoznawał komentarze o charakterze toksycznym, mowę nienawiści i treści temu podobne, w języku polskim. Jako źródło takiej treści wybrany został jeden z najpopularniejszych portali społecznościowych – YouTube. W ramach realizacji tego projektu utworzono również prosty interfejs graficzny, pozwalający użytkownikowi „skanować” wybrane komentarze.

Zakładany efekt został osiągnięty, a rezultaty są satysfakcjonujące. Początkowo problem stanowiła realizacja projektu w języku polskim, ze względu na ograniczoną dostępność danych i materiałów źródłowych w tym języku, jednak końcowo udało się odnaleźć potrzebne materiały do realizacji tego zagadnienia. Opracowany program z powodzeniem wykonuje postawione przed nim zadania, analizując i klasyfikując komentarze pod kątem obecności mowy nienawiści. Łączy się z portalem YouTube za pomocą API i pobiera sekcję komentarzy znajdującą się pod wskazanym przez użytkownika filmem. Utworzona aplikacja pozwala na intuicyjne korzystanie z opracowanego programu. Umożliwia wybór modelu NLP jaki zostanie użyty do klasyfikacji, jak również metody wektoryzacji tekstu oraz dokładności lematyzacji.

Przeprowadzone testy wydajności poszczególnych modeli NLP wskazują, że skuteczność detekcji szkodliwych treści jest na wysokim poziomie, sięgającym nawet 90% dla mniej skomplikowanych komentarzy i około 80% dla bardziej złożonych treści, podobnych do tych, które można spotkać na portalach społecznościowych.

Projekt spełnia wszystkie główne założenia, jednak posiada też potencjał do dalszego rozwoju. Możliwe kierunki rozwoju obejmują rozbudowę istniejących modeli NLP poprzez lepsze dostosowanie parametrów, a także poszerzenie ich puli o bardziej złożone techniki, jak sieci neuronowe. Oprócz tego ciekawym kierunkiem rozwoju byłoby również rozbudowanie aplikacji o nowe funkcjonalności, takie jak wyświetlanie statystyk analizowanych komentarzy lub możliwość ukrycia toksycznej treści.

Przeciwdziałanie szerzącemu się powszechnie zjawisku mowy nienawiści stanowi obecnie bardzo istotny temat. Ograniczenie widoczności szkodliwych treści, a więc w konsekwencji również ich wpływu na odbiorców, stanowi wyzwanie dla popularnych portali społecznościowych, mające realny wpływ na samopoczucie użytkowników. Narzędzia pozwalające na sprawowanie większej kontroli nad takimi wpisami, takie jak opracowana aplikacja, stanowią krok w stronę utworzenia zdrowszego i bardziej przyjaznego środowiska w przestrzeni cyfrowej.

# **Bibliografia**

|  |  |
| --- | --- |
| [1] | M. Z. Shervin Malmasi, *Detecting Hate Speech in Social Media.* |
| [2] | “Statista,” [Online]. Available: https://www.statista.com/statistics/433871/daily-social-media-usage-worldwide/. [Accessed 21 10 2023]. |
| [3] | J. Pyżalski, A. Zbrodowska, Ł. Tomczyk i K. Abramczuk, „Polskie badanie EU Kids Online 2018. Najważniejsze wyniki i wnioski.,” Wydawnictwo Naukowe UAM, Poznań, 2018. |
| [4] | Gemius, Polskie Badania Internetu, IAB Polska, „Social Media 2023,” 2023. |
| [5] | Fundacja Humanity in Action Polska, [Online]. Available: https://uprzedzuprzedzenia.org/czym-mowa-nienawisci/. [Data uzyskania dostępu: 02 11 2023]. |
| [6] | A. Bodnar, „20 rekomendacji RPO. Adam Bodnar o sposobach rozwiązywania problemu mowy nienawiści,” Rzecznik Praw Obywatelskich, 21 01 2019. [Online]. Available: https://bip.brpo.gov.pl/pl/content/20-rekomendacji-rpo-adam-bodnar-o-sposobach-rozwiązywania-problemu-mowy-nienawiści. [Data uzyskania dostępu: 02 11 2023]. |
| [7] | J. Saroar i O. Mourad, „A systematic review of hate speech automatic detection using natural language processing,” p. https://www.sciencedirect.com/science/article/pii/S0925231223003557, 2023. |
| [8] | R. Saini, P. Alonso i G. Kovács, „Challenges of Hate Speech Detection in Social Media. SN COMPUT. SCI. 2, 95,” pp. https://doi.org/10.1007/s42979-021-00457-3, 2021. |
| [9] | I. Kwok i Y. Wang, „Locate the Hate: Detecting Tweets against Blacks,” 2013. |
| [10] | S. Jahan i M. Oussalah, „A systematic review of hate speech automatic detection using natural language processing,” 14 08 2023. |
| [11] | „Oracle,” [Online]. Available: https://www.oracle.com/pl/artificial-intelligence/what-is-natural-language-processing/. [Data uzyskania dostępu: 12 11 2023]. |
| [12] | A. Pant, „towardsdatascience,” [Online]. Available: https://towardsdatascience.com/workflow-of-a-machine-learning-project-ec1dba419b94. [Data uzyskania dostępu: 05 11 2023]. |
| [13] | „Qlik,” [Online]. Available: https://help.qlik.com/pl-PL/cloud-services/Subsystems/Hub/Content/Sense\_Hub/AutoML/holdout-crossvalidation.htm. [Data uzyskania dostępu: 12 11 2023]. |
| [14] | [Online]. Available: https://www.techtarget.com/searchdatamanagement/definition/data-scrubbing. [Data uzyskania dostępu: 02 12 2023]. |
| [15] | „pogromcykodu,” [Online]. Available: https://pogromcykodu.pl/czym-jest-nlp/. [Data uzyskania dostępu: 12 11 2023]. |
| [16] | P. Yadav, „medium,” [Online]. Available: https://medium.com/@pawan329/text-data-preprocessing-made-easy-steps-to-clean-text-data-using-python-81a138a0e0e3. [Data uzyskania dostępu: 12 11 2023]. |
| [17] | P. Singh. [Online]. Available: https://medium.com/analytics-vidhya/fundamentals-of-bag-of-words-and-tf-idf-9846d301ff22. [Data uzyskania dostępu: 16 11 2023]. |
| [18] | [Online]. Available: https://www.deeplearning.ai/resources/natural-language-processing/. [Data uzyskania dostępu: 19 11 2023]. |
| [19] | [Online]. Available: https://www.projectpro.io/article/machine-learning-nlp-text-classification-algorithms-and-models/523. [Data uzyskania dostępu: 19 11 2023]. |
| [20] | [Online]. Available: https://corporatefinanceinstitute.com/resources/data-science/bayes-theorem/. [Data uzyskania dostępu: 19 11 2023]. |
| [21] | [Online]. Available: https://www.edlitera.com/blog/posts/how-to-use-the-anaconda-navigator-for-python-programming. [Data uzyskania dostępu: 02 12 2023]. |
| [22] | [Online]. Available: https://www.python.org/doc/essays/blurb/. [Data uzyskania dostępu: 3 11 2023]. |
| [23] | [Online]. Available: https://www.databricks.com/glossary/jupyter-notebook. [Data uzyskania dostępu: 3 11 2023]. |
| [24] | PolEval, 2019. [Online]. Available: http://2019.poleval.pl/index.php/tasks/task6. [Data uzyskania dostępu: 22 10 2023]. |
| [25] | P. Röttger, 2022. [Online]. Available: https://huggingface.co/datasets/Paul/hatecheck-polish. [Data uzyskania dostępu: 22 10 2023]. |
| [26] | I. Okulska, K. Głąbińska, A. Kołos, A. Karlińska, E. Wiśnios, A. Nowakowski, P. Ellerik i A. Prałat, „BAN-PL: a Novel Polish Dataset of Banned Harmful and Offensive Content from Wykop.pl web service,” 2023. [Online]. Available: https://github.com/ZILiAT-NASK/BAN-PL/tree/main. |
| [27] | [Online]. Available: https://github.com/bieli/stopwords/tree/master. [Data uzyskania dostępu: 12 11 2023]. |
| [28] | [Online]. Available: https://theforcecode.com/help/pl/tworzenie-interfejsow-uzytkownika-w-pythonie-wprowadzenie-do-biblioteki-tkinter/. [Data uzyskania dostępu: 01 12 2023]. |
| [29] | D. Mamakas, Bag of Words BERT-based models for large, 2022. |
| [30] | M. Kostrzewski. [Online]. Available: https://www.kaggle.com/datasets/mateuszkostrzewski/polish-negative-words. [Data uzyskania dostępu: 22 10 2023]. |
| [31] | D. Jakóbczak, „Interaktywna aplikacja do klasyfikacji obrazów za pomocą sieci,” 2021. |