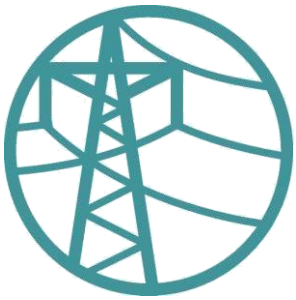


WYDZIAŁ
ELEKTROTECHNIKI
I INFORMATYKI
POLITECHNIKI RZESZOWSKIEJ

Łukasz Borek Paweł Dziedzic

Sytstem detekcji spamu z wykorzystaniem uczenia maszynowego

Projekt inżynierski

Opiekun pracy:
dr inż. Andrzej Paszkiewicz

Rzeszów, 2022

Spis treści

1. Opis danych....4
2. Wstęp....5
3. Przygotowywanie danych....5
3.1. Import danych....5
3.2. Import bibliotek....6
3.3 Łączenie ramek....6
3.3.1 Ramka data_0....6
3.3.2 Ramka data_1....7
3.4 Usunięcie pustych danych....7
3.5 Usunięcie zduplikowanych danych....8
3.6 Podsumowanie danych po modyfikacji....9
4. Transformacja danych tekstowych....10
4.1 Usunięcie liczb oraz zamiana na małe litery....10
4.2 Tokenizacja....10
4.3 Usunięcie znaków specjalnych....11
4.4 Usunięcie wyrazów powszechnych....11
4.5 Lematyzacja....11
4.6 Podział na zestaw do trenowania i testowania....12
4.7 Wektorowanie....12
5. Podsumowanie modyfikacji i transformacji zbioru....12
5.1 Wykresy częstotliwości poszczególnych wyrazów....13
5.1.1 Wykres wiadomości autentycznych....13
5.1.2 Wykres wiadomości spamu....13
6. Algorytmy uczenia maszynowego....14
6.1 Naive Bayes....14
6.1.1 Multinominalny Naive Bayes....15
6.1.2 Gauss Naive Bayes....15
6.1.3 Bernoulli Naive Bayes....16
6.2 K-nearest neighbours....16
6.2.1 K-nearest neighbours w detekcji spamu....17
6.3 Support Vector Machines (SVM)....17
6.3.1 Support Vector Machines w detekcji spamu....18
6.4 Neural Networks....18
6.4.1 Neural Networks w detekcji spamu....19
6.5 ExtraTreeClassifier....19
6.4.1 ExtraTreeClassifier w detekcji spamu....20
6.5 Gradient Boost....20

6.5.1 Gradient Boost w detekcji spamu 20
6.6 Extreme Gradient Boost (XGBoost) 21
6.6.1 Extreme Gradient Boost w detekcji spamu 21
6.7 AdaBoost 22
6.7.1 Adaboost w detekcji spamu 22
7. Prezentacja otrzymanych wyników 23
8. Analiza wyników 24
9. Podsumowanie 24

1. Opis danych

Dane przedstawione w projekcie zostały pobrane ze stron: https://www.kaggle.com/datasets/uciml/sms-spam-collection-dataset oraz https://archive.ics.uci.edu/ml/datasets/sms%2Bspam%2Bcollection.

Pierwsza ramka zawiera kolumnę „type", która posiada wartości tekstowe wyznaczające kategorie danej wiadomości. Kolumna „text" zawiera treść wybranego maila.

	type	text
0	ham	Hope you are having a good week. Just checking in
1	ham	K..give back my thanks.
2	ham	Am also doing in cbe only. But have to pay.
3	spam	complimentary 4 STAR Ibiza Holiday or £10,000 ...
4	spam	okmail: Dear Dave this is your final notice to...
...	...	...
5554	ham	You are a great role model. You are giving so ...
5555	ham	Awesome, I remember the last time we got someb...
5556	spam	If you don't, your prize will go to another cu...
5557	spam	SMS. ac JSco: Energy is high, but u may not kn...
5558	ham	Shall call now dear having food

Rys. 1.1 Pierwsza ramka

Druga ramka zawiera kolumnę „Category", która posiada wartości tekstowe wyznaczające kategorie danej wiadomości. Kolumna „Message" zawiera treść wybranego maila.

	Category	Message
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...
...	...	...
5567	spam	This is the 2nd time we have tried 2 contact u...
5568	ham	Will ü b going to esplanade fr home?
5569	ham	Pity, * was in mood for that. So...any other s...
5570	ham	The guy did some bitching but I acted like i'd...
5571	ham	Rofl. Its true to its name

5572 rows × 2 columns

Rys.1.2 Ramka druga

2. Wstęp

W dzisiejszych czasach znaczącą rolę w naszym życiu odgrywa komunikacja elektroniczna przedstawiana w postaci wiadomości tekstowych, komentarzy na stronach internetowych lub E-maili. Dzięki tym formom komunikacji jesteśmy w stanie w szybki i prosty sposób porozumieć się z osobą znajdującą się niedaleko nas jak i na drugim krańcu świata. Jednakże, wraz z coraz to większą wygodą i rozwojem, wzrasta liczba nadużyć takich jak spam.

Spam to, ogólnie mówiąc, niepożądana przez odbiorców wiadomość tekstowa. Wysyłana jest najczęściej masowo w formie reklamy przez rozmaite firmy lub hakerów mających na celu wyłudzenie danych osobowych lub dostępu do komputera. Może obejmować takie formy jak reklamy, łańcuszki e-mailowe, oszustwa internetowe itp. W tym celu stosuje się filtry pozwalające na wykrycie czy dana wiadomość jest spamem lub nie.

Typowym systemem filtrowania spamu jest taki, który implementuje zestaw reguł i funkcjonuje z zestawem protokołów. Implementowane są odpowiednio filtry treści, nagłówków wiadomości, backlist, reguł i na końcu przydziału i zadania. W zależności od potrzeb użytkownika systemy te są bardziej lub mniej zaawansowane.

Skupmy się jednak na wykorzystaniu uczenia maszynowego w celu opracowania takiego systemu detekcji. Uczenie maszynowe pozwala na przetwarzanie ogromnych ilości danych za pomocą, których jesteśmy w stanie otrzymać szybsze i dokładniejsze wyniki wykrywania niepożądanych treści. Trzeba jednak wziąć pod uwagę aspekt szkolenia modelu ,który może wymagać dodatkowego czasu i wysokiego poziomu wydajności.

3. Przygotowywanie danych

Przed przystąpieniem do samego procesu uczenia maszynowego, przygotowane na rzecz tego projektu dane należy odpowiednio zmodyfikować w celu ich incjalizacji w poszczególnych modelach nauki.

3.1. Import danych

Zawierają one kolumnę dotyczącą wiadomości mailowej oraz kolumnę kategorii tej wiadomości ze względu na to czy jest ona spamem bądź prawdziwą wiadomością.

```
# ham - 0, spam - 1
data_0 = pd.read_csv('mail_spam_or_ham.csv')
data_1 = pd.read_csv('mail_data.csv')
```

Rys. 3.1 Zrzut wczytywanych danych

3.2. Import bibliotek

Biblioteką najczęściej korzystaną w tym projekcie jest scikt-learn, która pozwala na przygotowanie pewnych parametrów pod modele oraz samą ich inicjalizacje. Szczególnie ważnymi bibliotekami są numpy oraz pandas, które pozwalają na tworzenie oraz przekształcanie ramek danych. Odpowiednie moduły z bibliotek nltk oraz collections służą do znajdowania wyrazów powszechnych oraz zliczania wszystkich wyrazów znajdujacych się w ramce. Biblioteka matplotlib.pyplot pozwala na tworzenie wykresów.

3.3 Łączenie ramek

Wczytane ramki danych zawierające dane z różnych źródeł należy w pierwszym kroku połączyć w celu lepszej modyfikacje na rzecz modeli.

```
# łączenie ramek data_0 oraz data_1
mail_data = pd.concat([data_0, data_1])

print(mail_data)
```

```
       type                                                text
0         0  Hope you are having a good week. Just checking in
1         0                             K..give back my thanks.
2         0        Am also doing in cbe only. But have to pay.
3         1  complimentary 4 STAR Ibiza Holiday or £10,000 ...
4         1  okmail: Dear Dave this is your final notice to...
...     ...                                                ...
11126     1  This is the 2nd time we have tried 2 contact u...
11127     0              Will ü b going to esplanade fr home?
11128     0  Pity, * was in mood for that. So...any other s...
11129     0  The guy did some bitching but I acted like i'd...
11130     0                         Rofl. Its true to its name

[11131 rows x 2 columns]
```

Rys.3.2 Łączenie ramek data_0 oraz data_1

3.3.1 Ramka data_0

W ramce znajdującej się pod zmienną data_0 zmieniono wartości „ham” oraz „spam” znajdujące się w kolumnie „type” odpowiednio na wartości 0 i 1.

```
# Modyfikacja ramki 'mail_spam_or_ham' pod zmienną data_0
data_0['type'] = data_0['type'].replace({'ham': 0, 'spam': 1})
print(data_0)
```

```
      type                                                text
0        0  Hope you are having a good week. Just checking in
1        0                             K..give back my thanks.
2        0       Am also doing in cbe only. But have to pay.
3        1  complimentary 4 STAR Ibiza Holiday or £10,000 ...
4        1  okmail: Dear Dave this is your final notice to...
...    ...                                                ...
5554     0  You are a great role model. You are giving so ...
5555     0  Awesome, I remember the last time we got someb...
5556     1  If you don't, your prize will go to another cu...
5557     1  SMS. ac JSco: Energy is high, but u may not kn...
5558     0                    Shall call now dear having food

[5559 rows x 2 columns]
```

Rys.3.3 Modyfkiacja ramki data_0

3.3.2 Ramka data_1

W drugiej ramce zmieniono nazwy kolumn a także indeksy w celu lepszego połączenia dwóch ramek. Wykonano również zamianę wartości „ham” oraz „spam” znajdujące się w zmienionej już kolumnie „type” odpowiednio na 0 oraz 1.

```
# Modyfikacja ramki 'mail_data' pod zmienną data_1
# Zamiana nazw kolumn
data_1.rename(columns={'Category': 'type','Message': 'text'}, inplace = True)
# Zamina wartości ham i spam na 1 i 0
data_1['type'] = data_1['type'].replace({'ham':0,'spam':1})
# Zmiana początku i końca indeksu wierszy w celu ustalenia poprawnej numeracji ramki łączonej
data_1.set_index(pd.Index(range(5559,11131)), inplace=True)
print(data_1)
```

```
      type                                               text
5559     0  Go until jurong point, crazy.. Available only ...
5560     0                      Ok lar... Joking wif u oni...
5561     1  Free entry in 2 a wkly comp to win FA Cup fina...
5562     0  U dun say so early hor... U c already then say...
5563     0  Nah I don't think he goes to usf, he lives aro...
...    ...                                                ...
11126    1  This is the 2nd time we have tried 2 contact u...
11127    0              Will ü b going to esplanade fr home?
11128    0  Pity, * was in mood for that. So...any other s...
11129    0  The guy did some bitching but I acted like i'd...
11130    0                       Rofl. Its true to its name
```

Rys.3.4 Modyfikacja ramki data_1

3.4 Usunięcie pustych danych

Punkt ten dotyczy usunięcia danych znajdujących się w połączonej ramce, które zawierają puste wartości w rekordach. Z poniższego zrzutu wynika, że w ramce nie występowały żadne puste rekordy.

```python
# Usuwanie pustych danych
print(mail_data.isnull().sum())
mail_data = mail_data.dropna(how = "any")
print(mail_data)
```

```
type    0
text    0
dtype: int64
       type                                                text
0         0  Hope you are having a good week. Just checking in
1         0                             K..give back my thanks.
2         0        Am also doing in cbe only. But have to pay.
3         1  complimentary 4 STAR Ibiza Holiday or £10,000 ...
4         1  okmail: Dear Dave this is your final notice to...
...     ...                                                 ...
11126     1  This is the 2nd time we have tried 2 contact u...
11127     0                Will ü b going to esplanade fr home?
11128     0  Pity, * was in mood for that. So...any other s...
11129     0  The guy did some bitching but I acted like i'd...
11130     0                         Rofl. Its true to its name

[11131 rows x 2 columns]
```

Rys. 3.5 Usuwanie pustych rekordów

3.5 Usunięcie zduplikowanych danych

Punkt ten dotyczy usunięcia zduplikowanych danych, które w wyniku połączenia dwóch osobnych zbiorów mogły się powielać ze sobą. Z poniższego zrzutu ekrany wynika ,że około połowa rekordów okazała się być zduplikowana. Dane, które funkcja wykryła jako zduplikowane zostały usunięte z ramki danych.

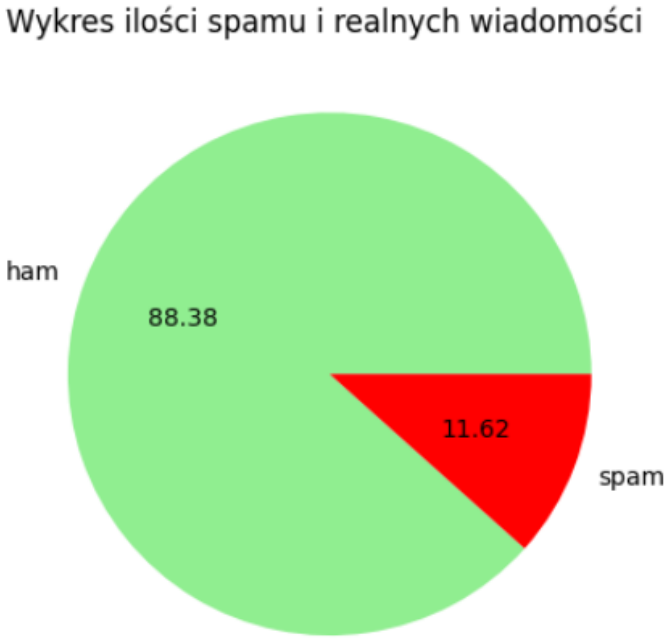
```
# Pozbycie się duplikatów
print(mail_data.duplicated().sum())
mail_data = mail_data.drop_duplicates()
mail_data.set_index(pd.Index(range(0,5962)), inplace=True)
print(mail_data)
```

```
0
      type                                                text
0        0  Hope you are having a good week. Just checking in
1        0                             K..give back my thanks.
2        0        Am also doing in cbe only. But have to pay.
3        1  complimentary 4 STAR Ibiza Holiday or £10,000 ...
4        1  okmail: Dear Dave this is your final notice to...
...    ...                                                ...
5957     0  I'm taking derek &amp;amp; taylor to walmart, if I...
5958     0  No. I meant the calculation is the same. That ...
5959     0  if you aren't here in the next  &lt;#&gt;  hou...
5960     0                Will ü b going to esplanade fr home?
5961     0  The guy did some bitching but I acted like i'd...

[5962 rows x 2 columns]
```

Rys. 3.6 Usunięcie zduplikowanych danych

3.6 Podsumowanie danych po modyfikacji

Poniższy zrzut ekranu prezentuje porównane ilości wiadomości realnych do spamu. Na wykresie można zauważyć że ponad 88% wiadomości jest w kategorii autentycznych, natomiast pozostałe 12% są spamem.



Rys. 3.7 Wykres kołowy ilości spamu i hamu

4. Transformacja danych tekstowych

Ta sekcja poświęcona jest transformacji wiadomości tekstowych w celu lepszej reprezentacji numerycznej. Modele uczenia maszynowego są w stanie przetwarzać jedynie dane numeryczne, dlatego ważnym elementem jest wykorzystanie pewnych metod pozwalających na optymalizacje danych tekstowych w celu uzyskania jak najlepszej transformacji numerycznej.

4.1 Usunięcie liczb oraz zamiana na małe litery

Liczby w wiadomościach mailowych nie niosą ze sobą w większości przypadków żadnych istotnych wzorców, dlatego ich usunięcie pozwala skupić się na ważniejszych częściach analizy tekstu przez model. Zamiana liter na małe pozwala uniknąć problemu zróżnicowanej pisowni wyrazu, który ma takie samo znaczenie w obu przypadkach np.: „Word” oraz „word”. Poprzez zamianę każdej litery na małą w całym zbiorze, ułatwiany jest proces porównania oraz grupowania słów.

```
# Zamiana na małe litery
mail_data['text'] = mail_data['text'].str.lower()
# Usunięcie liczb
mail_data['text'] = mail_data['text'].str.replace(r'\d+', '', regex=True)
```

Rys. 4.1 Zamiana liter i usunięcie liczb

4.2 Tokenizacja

Tokenizacja jest to działanie mające na celu podzielenie danego tekstu na mniejsze jednostki. Mogą to być pojedyncze słowa, kawałki kodu czy znaki interpunkcyjne. Proces ten jest wykonywany, w celu wydajniejszego i optymalniejszego transformowania tekstu w kolejnych częściach.

```
# Zmienna pozwala na dzielenie na podstawie wzorca tekstu
tokenizer = nltk.tokenize.RegexpTokenizer(r'\w+')
def tokenize(text):
    tokenize_text = tokenizer.tokenize(text)
    return tokenize_text

mail_data['text'] = mail_data['text'].apply(tokenize)
```

Rys. 4.2 Kod tokenizacji

4.3 Usunięcie znaków specjalnych

Operacja usuwania znaków specjalnych z zbioru ma na celu pozbycie się takich elementów, które często występują w wiadomościach mailowych jak np.: dwukropek bądź średnik.

```
# Usuwanie znaków specjalnych
def special_characters(text_spec_char):
    text_spec_char = [re.sub(r'\W+', '', word) for word in text_spec_char]
    return text_spec_char

mail_data['text'] = mail_data['text'].apply(lambda x: special_characters(x))
```

Rys. 4.3 Usunięcie wyrazów powszechnych

4.4 Usunięcie wyrazów powszechnych

Wyrazy powszechne są to wyrazy często stosowane, ale ich znaczenie w nauce modelu uczenia maszynowego jest relatywnie małe. Można nawet stwierdzić, że mogą one powodować szum lub zakłócenia w otrzymanych wynikach. Przykładami słów powszechnych w języku angielskim są np.: „the” , „a”, ”of”, ”in”.

W tym celu za pomocą biblioteki nltk, która posiada odpowiedni moduł posiadający zbiór takich słów, pozbyto się z całej ramki wyrazów powszechnych.

```
# Usunięcie wyrazów powszechnych
stopwords = stopwords.words('english')
def remove_stopwords(s_words):
    s_words = [word for word in s_words if word.lower() not in stopwords]
    return s_words
mail_data['text'] = mail_data['text'].apply(lambda x: remove_stopwords(x))
```

Rys. 4.4 Usunięcie wyrazów powszechnych

4.5 Lematyzacja

Lematyzacja lub normalizacja tekstu (z ang. „Stemming”) jest procesem, który redukuje słowa przywracając je do podstawowej formy, czyli usuwając fleksyjne końcówki słów np.: „worked” na „work”. Taka operacja pozwala na zmniejszenie liczby słów mających to samo znaczenie, jednak będących w różnej odmianie. Ułatwiony jest dzięki temu proces grupowania oraz identyfikowania wyrazów w zbiorze. Stemming posiada pewne ograniczenia, ponieważ może doprowadzić do utraty części znaczenia słów. Dlatego proces ten nie będzie użyteczny w każdym możliwym przypadku.

```
# Proces lematyzacji
ps = PorterStemmer()
mail_data['text'] = mail_data['text'].apply(lambda x: ' '.join([ps.stem(word) for word in x]))
```

Rys. 4.5 Kod lematyzacji

Tekst, który został podzielony w procesie tokenizacji na mniejsze jednostki, został ponownie złączony w trakcie wykonywania się funkcji lematyzacji.

4.6 Podział na zestaw do trenowania i testowania

Zestaw do trenowania i testowania w uczeniu maszynowym to zbiór danych, który jest wykorzystywany do szkolenia i oceny skuteczności modeli uczenia maszynowego. Składa się z dwóch głównych części: zbioru treningowego i zbioru testowego. Podział danych zazwyczaj opiera się na proporcjach 80% do 20% lub 90% do 10 %, odpowiednio dla modelu trenującego i testującego.

```
X_train, X_test, Y_train, Y_test = train_test_split(mail_data['text'],mail_data['type'],test_size=0.2, random_state = 3)
```

Rys. 4.6 Kod podziału Test/Train

4.7 Wektorowanie

W celu przekształcenia danych do formy liczbowej dokonuje się procesu zwanego wektoryzacją lub przekształcaniem tekstu na macierz liczebności. Operacja polega na zamianie tekstu na wektory liczbowe, które są używane w modelach uczenia maszynowego jako dane wejściowe.

```
from sklearn.feature_extraction.text import CountVectorizer
vectorizer = CountVectorizer()
X_train_transformed = vectorizer.fit_transform(X_train)
X_test_transformed = vectorizer.transform(X_test)
```

Rys. 4.7 Kod wektoryzacji

5. Podsumowanie modyfikacji i transformacji zbioru

W wyniku modyfikacji i transformacji zbioru, ramka danych została znaczenie uszczuplona. W poniższej tabeli zaprezentowana jest różnica pomiędzy pięcioma pierwszymi rekordami zbioru przed i po modyfikacji.

Ramka przed modyfikacją	Ramka po modyfikacji
Hope you are having a good week. Just checking in	hope good week check
K..give back my thanks.	k give back thank
Am also doing in cbe only. But have to pay.	also cbe pay
okmail: Dear Dave this is your final notice to collect your 4* Tenerife Holiday or #5000 CASH award! Call 09061743806 from landline. TCs SAE Box326 CW25WX 150ppm	okmail dear dave thi is your final notic to collect your tenerif holiday or cash award call from landlin tc sae box cwwx ppm
Aiya we discuss later lar... Pick u up at 4 is it?	aiya we discuss later lar pick u up at is it

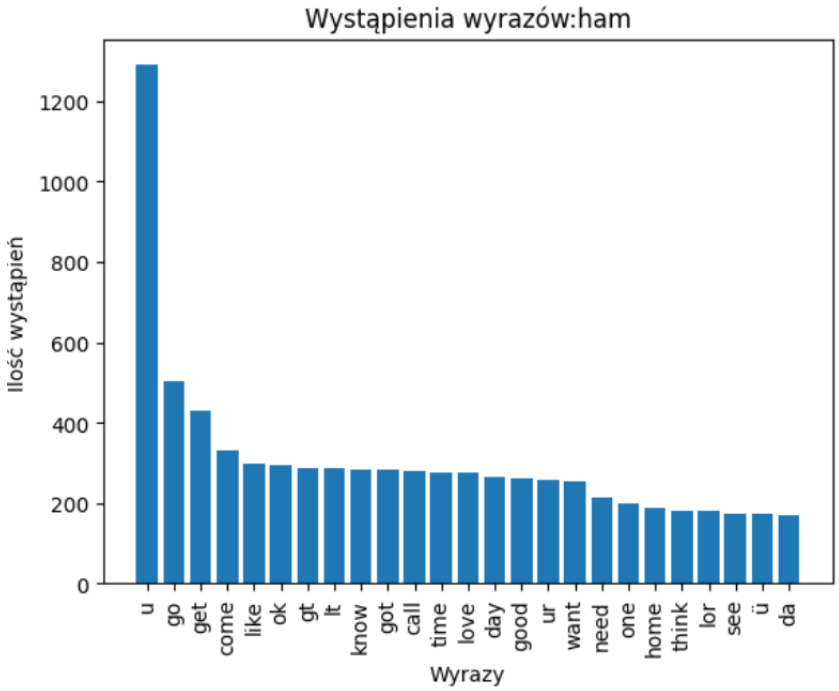
Tabela 5.1 Porównanie ramki przed i po

5.1 Wykresy częstotliwości poszczególnych wyrazów

Usunięcie wyrazów lub znaków mogących powodować szum lub zakłócenia w modelu, pozwala na skupienie się na słowach kluczowych w danej kategorii mailowej. Warto przyjrzeć się zatem wyrazom, które najczęściej występują w zbiorze w celu upewnienia się czy nadają się one do modelu uczenia maszynowego.

5.1.1 Wykres wiadomości autentycznych

Poniższy wykres przedstawia największą częstotliwość występowania dwudziestu wyrazów w wiadomościach autentycznych.

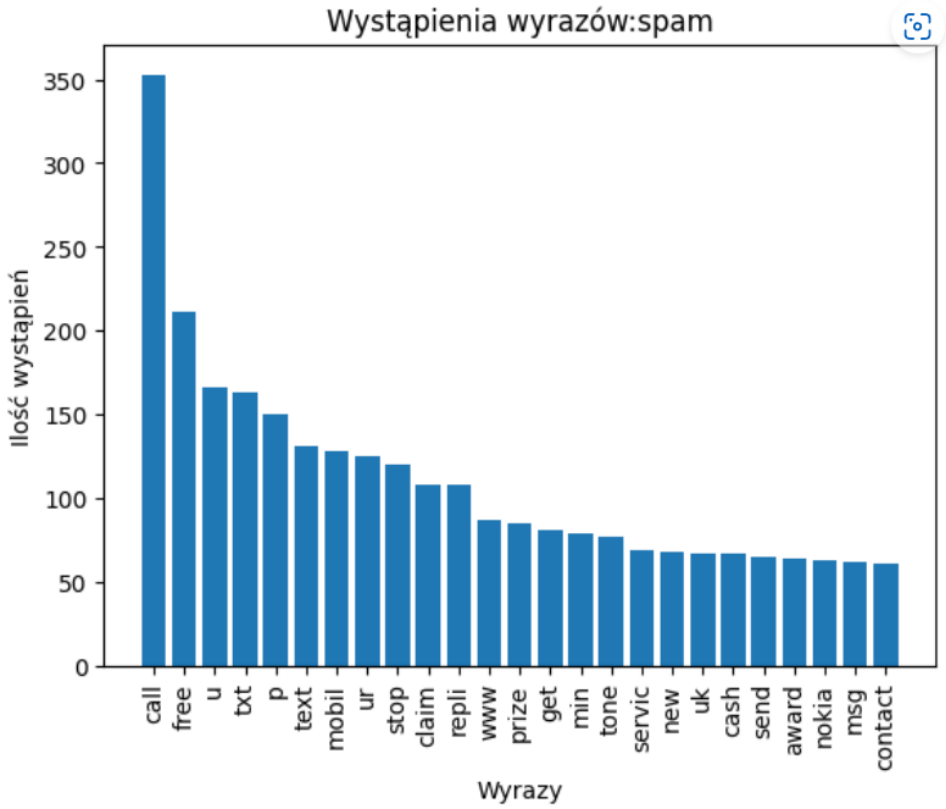


Rys. 5.1 Wykres częstotliwości słów w realnych mailach

Najczęściej występujących wyrazem w podzbiorze wiadomości kategorii „ham” jest „u”. Litera ta jest skrótem od słowa „you” i jej częstość występowania jest dwukrotnie większa od miejsca drugiego, czyli „go”. Wyrazy zamieszczone na wykresie są typowymi słowami pojawiającymi się w wiadomościach mailowych. Można stwierdzić, że dane z tego podzbioru są wartościowe dla modelu.

5.1.2 Wykres wiadomości spamu

Poniższy wykres przedstawia największą częstotliwość występowania dwudziestu wyrazów w wiadomościach spamu.



Rys. 5.2 Wykres częstotliwości słów w mailach spamu

Słowem, który najczęściej pojawia się wiadomościach spamu jest „call”. Różnica pomiędzy występowania najczęstszego wyrazu w porównaniu do reszty słów nie jest tak wielka jak w przypadku wykresu wiadomości realnych. Można zauważyć, że wyrazy typu „free”, „prize” lub „www” są częstymi wyrazami pojawiającymi się w podejrzanych wiadomościach. Wynika to z tego że większość tego rodzaju wiadomości ma za zadanie przekonać swoimi treściami użytkownika do przekierowania na zewnętrzną stronę WWW, która może nieść ze sobą zagrożenia. Można więc stwierdzić, że dane z tego podzbioru są wartościowe dla modelu.

6. Algorytmy uczenia maszynowego

Istnieje wiele algorytmów uczenia maszynowego, które pozwalają na efektywną i wydajną naukę. Nie wszystkie jednak są uniwersalne i będą pasowały do każdego zbioru danych. W przypadku systemów detekcji spamu do najważniejszych należą: Naive Bayes, Support Vector Machines (SVM) lub Sieci neuronowe.

6.1 Naive Bayes

Klasyfikacja Bayesowska jest przykładem nadzorowanej techniki uczenia i jednocześnie statystycznej techniki klasyfikacji. Naiwny klasyfikator Bayesa jest prostym klasyfikatorem probabilistycznym, który opiera się na twierdzeniu Bayesa z solidnymi założeniami, które są niezależne z natury [3]. Naiwne klasyfikatory Bayesa wymagają niewielkiej liczby punktów danych do wytrenowania i mogą radzić sobie z wysokowymiarowymi punktami danych, a także są szybkie i wysoce skalowalne [4]. Wzór przedstawia się w poniższej formie:

$$P(A|B) = \frac{P(B|A)}{P(A)}$$

Wzór ten można przekształcić żeby przedstawiał w jaki sposób algorytm zachowuje się w przypadku systemu detekcji spamu. Token T oznaczający spamminess (ocenę spamu) jest obliczany zgodnie z równaniem :

$$S|T| = \frac{C_{spam}(T)}{C_{spam}(T) + C_{ham}(T)}$$

Gdzie:

$C_{spam}(T) =$ Liczba wiadomości spam posiadający token T

$C_{ham}(T) =$ Liczba wiadomości ham posiadający token T. [3]

6.1.1 Multinominalny Naive Bayes

MultinomialNB implementuje naiwny algorytm Bayesa dla wielomianowo rozłożonych danych i jest jednym z dwóch klasycznych naiwnych wariantów Bayesa używanych w klasyfikacji tekstu. [5]

```
from sklearn.naive_bayes import MultinomialNB
model_MB = MultinomialNB()
model_MB.fit(X_train_transformed, Y_train)
y_pred_MB = model_MB.predict(X_test_transformed)
```

Rys. 6.1 Kod modelu MultinomialNB

6.1.2 Gauss Naive Bayes

GaussianNB implementuje algorytm Gaussian Naive Bayes do klasyfikacji. Zakłada się, że prawdopodobieństwo cech jest gaussowskie:

$$P(x_i|y) = \frac{1}{\sqrt{2 * \pi * \sigma_y^2}} * \exp(-\frac{(x_i - \mu_y)^2}{2 * \sigma_y^2})$$

Parametry μ_y i σ_y są szacowane przy użyciu maksymalnego prawdopodobieństwa [5].

```
from sklearn.naive_bayes import GaussianNB
model_GB = GaussianNB()
model_GB.fit(X_train_transformed.toarray(), Y_train)
y_pred_GB = model_GB.predict(X_test_transformed.toarray())
```

Rys. 6.2 Kod modelu GaussianNB

6.1.3 Bernoulli Naive Bayes

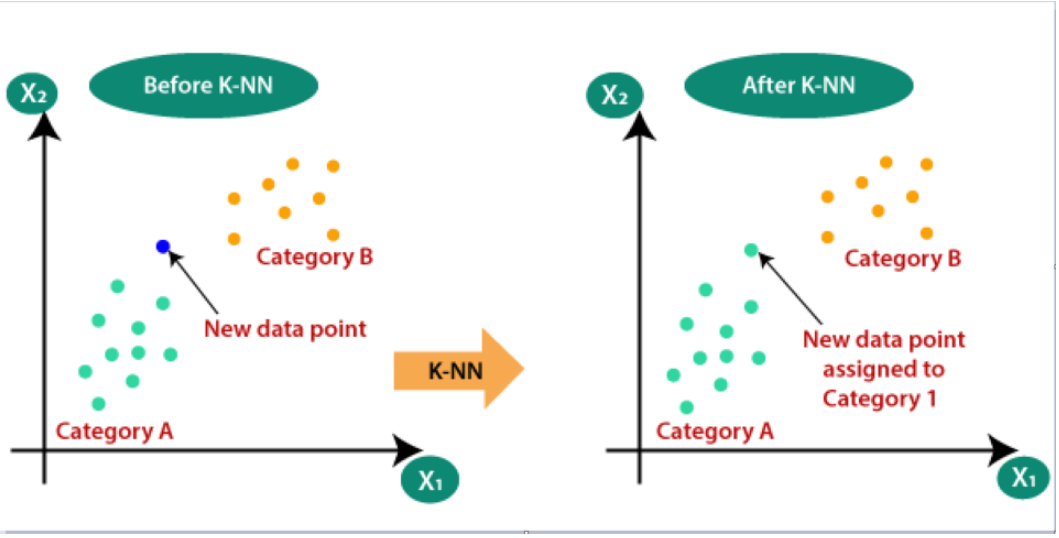
BernoulliNB implementuje naiwne algorytmy uczenia i klasyfikacji Bayesa dla danych, które są dystrybuowane zgodnie z wielowymiarowymi rozkładami Bernoulliego; tj. może istnieć wiele cech, ale zakłada się, że każda z nich jest zmienną o wartości binarnej (Bernoulli, boolean). Dlatego też klasa ta wymaga, aby próbki były reprezentowane jako wektory cech o wartościach binarnych. [5]

```
from sklearn.naive_bayes import BernoulliNB
model_BB = BernoulliNB()
model_BB.fit(X_train_transformed.toarray(), Y_train)
y_pred_BB = model_BB.predict(X_test_transformed.toarray())
```

Rys. 6.3 Kod modelu BernoulliNB

6.2 K-nearest neighbours

W K-Nearest Neighbors (KNN) celem jest sklasyfikowanie nowego, niewidocznego punktu danych poprzez przyjrzenie się K danym punktom danych w zbiorze uczącym, które są najbliżej niego w przestrzeni wejściowej lub przestrzeni cech [4]. Podstawowym parametrem algorytmu KNN jest wartość "k". Wybór odpowiedniej wartości "k" jest ważny i zależy od charakterystyki danych. Małe wartości "k" mogą prowadzić do nadmiernego dopasowania do szumu w danych, podczas gdy duże wartości "k" mogą prowadzić do utraty lokalnych szczegółów i zdolności do rozróżnienia klas.



Rys. 6.4 Działanie modelu K-NN

6.2.1 K-nearest neighbours w detekcji spamu

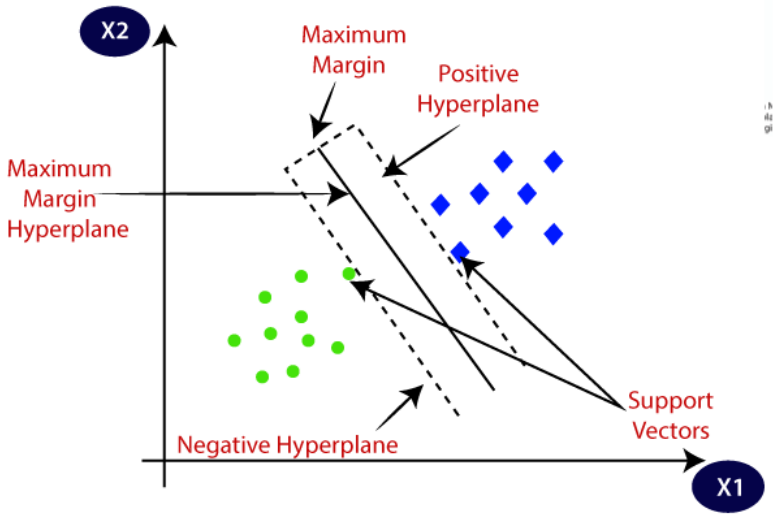
W przypadku detekcji spamu, klasyfikacja elementu może polegać na przypisaniu dokumentu do klasy "spam" lub "nie-spam" na podstawie etykiet najbliższych sąsiadów. Należy jednak zauważyć, że KNN ma swoje ograniczenia w kontekście analizy tekstu. Jednym z ograniczeń KNN jest to, że wymaga przechowywania całego zbioru treningowego, co sprawia, że KNN jest nieskalowalna do dużych zbiorów danych [4].

```
from sklearn.neighbors import KNeighborsClassifier
model_KNN = KNeighborsClassifier(10)
model_KNN.fit(X_train_transformed,Y_train)
y_pred_KNN = model_KNN.predict(X_test_transformed)
```

Rys. 6.5 Kod modelu K-NN

6.3 Support Vector Machines (SVM)

SVM to grupa algorytmów zaprojektowanych do rozwiązywania problemów klasyfikacji i regresji. SVM znalazł zastosowanie w rozwiązywaniu problemów programowania kwadratowego, które mają ograniczenia nierówności i równości liniowej poprzez różnicowanie różnych grup za pomocą hiperpłaszczyzny. Chociaż SVM może nie być tak szybki jak inne metody klasyfikacji, algorytm ten czerpie swoją siłę z wysokiej dokładności ze względu na jego zdolność do modelowania wielowymiarowych granic, które nie są sekwencyjne lub proste [4]. Algorytm SVM działa poprzez znalezienie optymalnej hiperpłaszczyzny, która może dokładnie rozdzielić dane wejściowe na różne klasy lub jak najlepiej je odseparować.



Rys. 6.6 Działanie modelu SVM

6.3.1 Support Vector Machines w detekcji spamu

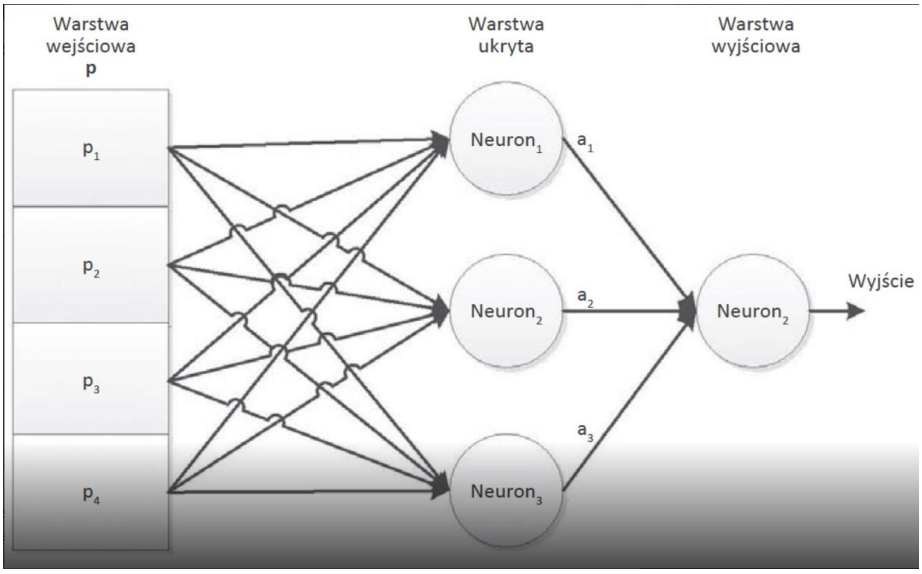
Algorytm SVM (Support Vector Machine) może być używany do detekcji spamu, jako technika klasyfikacji binarnej. Parametr kernel określa rodzaj funkcji jądrowej używanej w algorytmie SVM. Parametr gamma kontroluje elastyczność krzywej funkcji jądrowej. Wpływ gamma zależy od wybranej funkcji jądrowej.

```
from sklearn.svm import SVC
model_SVM = SVC(kernel='sigmoid', gamma=1.0)
model_SVM.fit(X_train_transformed,Y_train)
y_pred_SVM = model_SVM.predict(X_test_transformed)
```

Rys. 6.7 Kod modelu SVM

6.4 Neural Networks

Głębokie uczenie to nowy, rozwijający się obszar, który wykorzystuje sztuczną inteligencję i uczenie maszynowe do uczenia się funkcji bezpośrednio z danych, przy użyciu wielu nieliniowych warstw przetwarzania[3]. Sieci neuronowe są modelami matematycznymi inspirowanymi biologicznymi układami nerwowymi. Składają się z połączonych ze sobą sztucznych neuronów, które przetwarzają i przekazują sygnały wzdłuż ścieżek połączeń.



Rys. 6.8 Przykładowa sieć neuronowa

6.4.1 Neural Networks w detekcji spamu

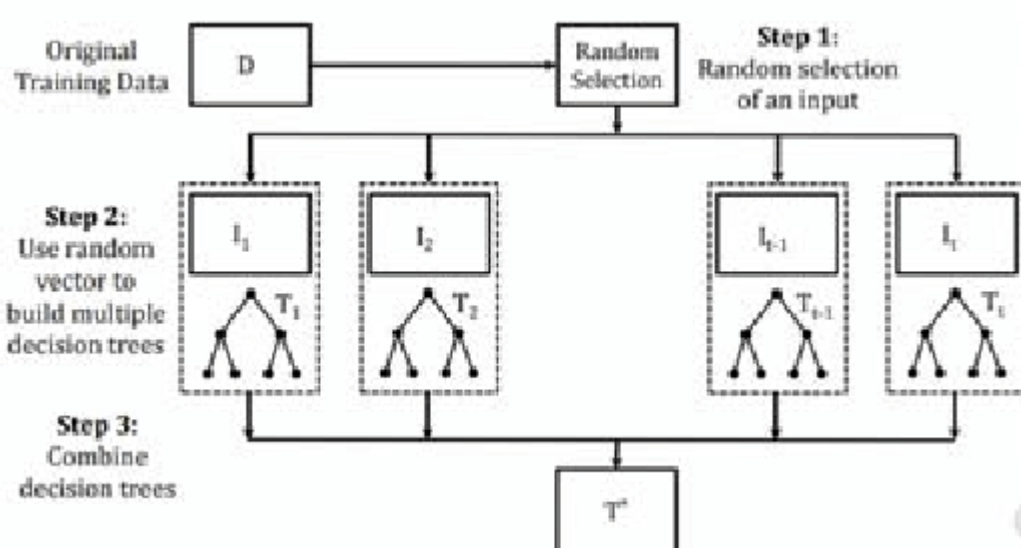
Sieci neuronowe są wykorzystywane w detekcji spamu z powodzeniem od wielu lat. Wykorzystuje się je do analizy treści wiadomości e-mail, identyfikacji wzorców i cech charakterystycznych, które wskazują na spam.

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

model_NN = Sequential()
model_NN.add(Dense(16, activation='relu', input_shape=(X_train_transformed.shape[1],)))
model_NN.add(Dense(1, activation='sigmoid'))

model_NN.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
model_NN.fit(X_train_transformed.toarray(), Y_train, epochs=10, batch_size=32)

# Predykcja na danych testowych
y_pred_NN = model_NN.predict(X_test_transformed.toarray())
# Zaokrąglanie do najbliższej wartości 0 lub 1
y_pred_NN = np.round(y_pred_NN).flatten()
# Zamiana z float na int
y_pred_NN = list(map(int,y_pred_NN))
```

Rys. 6.9 Kod sieci neuronowej

6.5 ExtraTreeClassifier

Extra Trees Classifier , znany również jako Extremely Randomized Trees Classifier, to technika zespołowego uczenia maszynowego, która integruje wiele nieskorelowanych wyników drzew decyzyjnych zebranych w lesie w wyniku ich klasyfikacji [6]. Model w celu zminimalizowania nadmiernego uczenia się oraz zbędnego dopasowania losuje niektóre decyzje i podzbiory danych. Extra Trees buduje wiele drzew i dzieli węzły przy użyciu losowych podzbiorów cech i bootstrapuje obserwacje.



Rys. 6.10 Działanie modelu EXT

6.4.1 ExtraTreeClassifier w detekcji spamu

ExtratreesClassifier jest algorytmem klasyfikacji o dużej dokładności i zdolności do generalizacji. Może on efektywnie wyodrębnić istotne cechy z danych treningowych i nauczyć się rozpoznawać wzorce charakterystyczne dla spamu.

```
from sklearn.ensemble import ExtraTreesClassifier
from sklearn.metrics import accuracy_score,precision_score
model_EXT = ExtraTreesClassifier()
model_EXT.fit(X_train_transformed,Y_train)
y_pred_EXT = model_EXT.predict(X_test_transformed)

test_accuracy_EXT = accuracy_score(Y_test,y_pred)
test_precision_EXT = precision_score(Y_test,y_pred)
```

Rys. 6.11 Kod modelu EXT

6.5 Gradient Boost

Gradient Boosting Classifier wykorzystuje słabą hipotezę lub metodę treningową i ulepsza ją poprzez sekwencję modyfikacji [6]. Metoda rozpoczyna się od dopasowania początkowego modelu np. drzewa lub regresji liniowej do danych. Następnie budowany jest kolejny model który stara się przewidzieć przypadki w których pierwszy model zadziałał słabo. Zakłada się, że połączenie tych dwóch modeli działa lepiej niż każdy z nich osobno. Proces ten jest powtarzany wielokrotnie, kolejne modele są dodawane do wcześniejszych modeli. Każdy kolejny model koryguje niedociągnięcia poprzednich modeli.

6.5.1 Gradient Boost w detekcji spamu

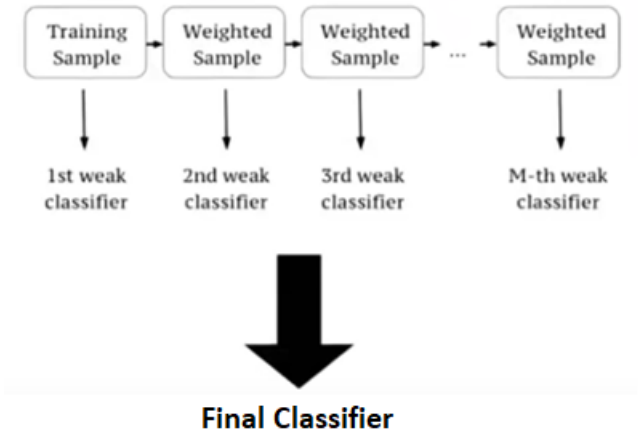
Algorytmy Gradient Boosting są preferowane, ponieważ mogą być wykorzystywane nie tylko do zadań klasyfikacji binarnej, ale także do wieloklasowej regresji i problemów klasyfikacji [6].

```
from sklearn.ensemble import GradientBoostingClassifier
model_GBC=GradientBoostingClassifier()
model_GBC.fit(X_train_transformed,Y_train)
y_pred_GBC = model_GBC.predict(X_test_transformed)
```

Rys. 6.12 Kod modelu Gradient Boost

6.6 Extreme Gradient Boost (XGBoost)

XG-Boost to gradientowe podejście boostingowe, które wykorzystuje algorytm uczenia maszynowego oparty na drzewie decyzyjnym. Jest to zdecentralizowana struktura, która została opracowana tak, aby była wysoce skuteczna, elastyczna i dostępna [6]. Analizuje rozkład cech we wszystkich punktach danych w liściu i wykorzystuje te informacje do zmniejszenia przestrzeni wyszukiwania możliwych podziałów cech. Ogranicza nadmierne dopasowanie, poprzez szybkie zbadanie ustawień hiperparametrów.



Rys. 6.13 Działanie modelu XGBoost

6.6.1 Extreme Gradient Boost w detekcji spamu

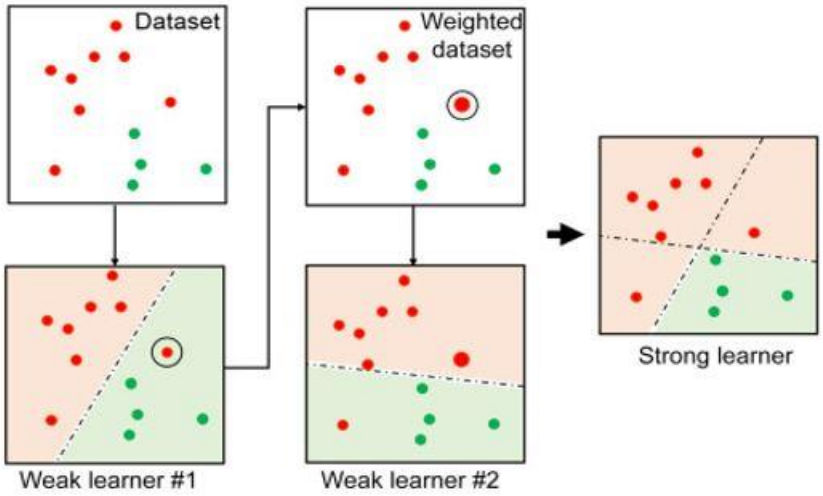
Powodem wyboru tego klasyfikatora w przypadku systemu detekcji spamu jest to, że XG-Boost oferuje równoległą metodę wzmacniania drzewa, która rozwiązuje kilka zadań nauki o danych w szybki i precyzyjny sposób [6].

```
from xgboost import XGBClassifier
model_XGBC=XGBClassifier()
model_XGBC.fit(X_train_transformed,Y_train)
y_pred_XGBC = model_XGBC.predict(X_test_transformed)
```

Rys. 6.14 Kod modelu XGBoost

6.7 AdaBoost

AdaBoost lub adaptive boosting to powtarzalna metoda kombinacji. Klasyfikator Ada-Boost łączy wiele klasyfikatorów o niskiej wydajności, aby stworzyć potężną metodę, która daje mu silną klasyfikację o wysokiej dokładności [6]. Algorytm zaczyna działanie od przypisania wag równych dla każdego przykładu w zbiorze treningowym. Następnie sprawdzane są te przykłady ,które zostały zostały niepoprawnie sklasyfikowane przez słabe klasyfikatory. Przy każdej iteracji algorytm wylicza błąd wagowy dla każdego uczącego przykładu. Elementy które zostały poprawnie sklasyfikowane otrzymują mniejszą wagę, natomiast błędne otrzymują wyższą wagę. W wyniku tego procesu Adaboost nadaje większą wagę dla tych klasyfikatorów, które radzą sobie lepiej w klasyfikacji trudnych przykładów.



Rys. 6.15 Sposób działania modelu AdaBoost

6.7.1 Adaboost w detekcji spamu

AdaBoost jest znany z wysokiej skuteczności klasyfikacji, nawet w przypadku zastosowania słabych klasyfikatorów. Poprzez kombinację wielu słabych klasyfikatorów, AdaBoost tworzy silny klasyfikator, który może efektywnie rozróżniać między spamem a hamem.

```
from sklearn.ensemble import AdaBoostClassifier
model_ABC=AdaBoostClassifier()
model_ABC.fit(X_train_transformed,Y_train)
y_pred_ABC = model_XGBC.predict(X_test_transformed)
```

Rys. 6.15 Kod modelu AdaBoost

7. Prezentacja otrzymanych wyników

W celu przedstawienia wyników działanie każdego z modeli, została napisana funkcja „results”, która zwracała listę metryk danego modelu. Po wykonaniu funkcji do każdego modelu w projekcie, wyniki zostały wpisane do ramki danych.

```
def results(x):
    accuracy = accuracy_score(Y_test, x)
    precision = precision_score(Y_test, x)
    recall = recall_score(Y_test, x)
    f1 = f1_score(Y_test, x)
    confusion_mat = confusion_matrix(Y_test, x)
    no_spam_true, no_spam_false = confusion_mat[0][0], confusion_mat[1][1]
    yes_spam_true, yes_spam_false = confusion_mat[1][0], confusion_mat[0][1]
    return list(map(float,[accuracy,precision,recall,f1,no_spam_true,no_spam_false,yes_spam_true,yes_spam_false]))
```

Rys. 7.1 Funkcja zwracająca metryku

Poniższy zrzut prezentuje tabele z wynikami dla poszczególnych modeli uczenia maszynowego

	Accuracy	Precision	Recall	F1-Score	Ham_True	Ham_False	Spam_True	Spam_False
MultinomialNB	0.966471	1.000000	0.716312	0.834711	1052.0	0.0	101.0	40.0
GaussianNB	0.869237	0.470120	0.836879	0.602041	919.0	133.0	118.0	23.0
BernouliNB	0.982397	0.991803	0.858156	0.920152	1051.0	1.0	121.0	20.0
K-NearstNeigbours	0.894384	1.000000	0.106383	0.192308	1052.0	0.0	15.0	126.0
SVM	0.979883	0.946565	0.879433	0.911765	1045.0	7.0	124.0	17.0
NeuralNetworks	0.984074	0.962121	0.900709	0.930403	1047.0	5.0	127.0	14.0
ExtraTreeClassifier	0.982397	0.991803	0.858156	0.920152	1051.0	1.0	121.0	20.0
GradientBoostClassifier	0.966471	0.971963	0.737589	0.838710	1049.0	3.0	104.0	37.0
XGradientBoostClassifier	0.974015	0.958333	0.815603	0.881226	1047.0	5.0	115.0	26.0
AdaBoost	0.974015	0.958333	0.815603	0.881226	1047.0	5.0	115.0	26.0

Rys. 7.2 Wyniki modeli

Każda kolumna oznacza inną metrykę, oceniającą wydajność danego modelu.

1) Accuracy (Dokładność) – określa stosunek poprawnie zidentyfikowanych przypadków do wszystkich w zbiorze.
2) Precison (Precyzja) – określa stosunek prawdziwie pozytywnych przykładów do wszystkich pozytywnych predykcji.
3) Czułość (Recall) – określa stosunek prawdziwie pozytywnych przykładów do wszystkich rzeczywiście pozytywnych.
4) F1-score – jest to średnia harmoniczna precyzji i czułości.
5) Ham_true, Ham_false – są to ilości wiadomości prawdziwych, które zostały prawidłowo oraz błędnie zakwalifikowane.

6) Spam_true, Spam_false - są to ilości wiadomości spam, które zostały prawidłowo oraz błędnie zakwalifikowane

8. Analiza wyników

Najlepszym modelem pod względem trzech metryk oceny okazały się być sieci neuronowe, które osiągnęły najlepsze wyniki dla dokładności (0.984064), czułości (0.900709) oraz F1-score (0.930403). Zaraz po nim znalazły się dwa modele ExtraTreeClassifier oraz BernouliNB, które osiągnęły takie same wyniki dla wszystkich metryk z tą różnicą że ich poziom precyzji był większy od tego w sieciach neuronwych. Najgorzej wypadającym algorytmem pod względem metryk okazał się być K-nearest neighbours , którego metryki dokładności, precyzji, czułości oraz f1-score odpowiednio wyniosły 0.894384, 1.000000, 0.106383, 0.192308

9. Podsumowanie

Celem projektu było znalezienie takiego modelu uczenia maszynowego, który najlepiej będzie radził sobie z detekcją spamu.

Zostało przeprowadzone 10 eksperymentów na różnych modelach uczenia maszynowego. Zastosowano różne techniki przetwarzania danych, selekcji cech oraz algorytmów uczenia maszynowego jak np.: maszyny wektorów nośnych, drzewa decyzyjne lub głębokie sieci neuronowe.

Pierwsze czynności dotyczące zebrania oraz przygotowania danych do treningu nie sprawiły większych trudności. Zbiór wiadomości mailowych miał początkowo składać się z około 11 tysięcy rekordów jednakże w wyniku usunięcia duplikatów, blisko połowa danych musiała zostać usunięta. Sprawiło to, że wybranych modeli nie udało się wytrenować na planowej liczbie danych.

Proces samej transformacji danych oraz uczenia modelu przebiegł bez problemów. Pomyślnie udało się przetrenować wybrane modele na zbiorze, oraz zebrać potrzebne do ich analizy statystyki. Wyniki eksperymentów pokazały że najlepszym modelem okazał się być sieć neuronowa, której to wyniki są bardzo zadowalające.

Źródła

[1] Naeem Ahmed, Rashid Amin, Hazmza Aldabbas, Deepika Koundal, Bader Alouffi, Tariq Shah, "Machine Learning Techniques for Spam Detection in Email and IoT Platforms: Analysis and Research Challenges", rok 2022, Dostępne pod: https://www.hindawi.com/journals/scn/2022/1862888/, Data dostępu: 17 maja 2023

[2] Javaid Nabi, „Machine Learning — Text Processing", 13 września 2018, Dostępne pod: https://towardsdatascience.com/machine-learning-text-processing-1d5a2d638958, Data dostępu: 12 maja 2023

[3] Emmanuel Gbenga Dada , Joseph Stephen Bassi , Haruna Chiroma , Shafi'i Muhammad Abdulhamid , Adebayo Olusola Adetunmbi , Opeyemi Emmanuel Ajibuwa , „Machine learning for email spam filtering: review, approaches and open research problems", czerwiec 2019, Dostępne pod: https://www.sciencedirect.com/science/article/pii/S2405844018353404, Data dostępu: 21 maja 2023

[4] Mohammad Saeid Mahdavinejad, Mohammadreza Rezvan , Mohammadamin Barekatain , Peyman Adibi , Payam Barnaghi , Amit P. Sheth b , „Machine learning for internet of things data analysis: a survey", lipiec 2018, Dostępne pod: https://www.sciencedirect.com/science/article/pii/S235286481730247X, Data dostępu: 21 maja 2023

[5] Scikit-learn. „Naive Bayes", Dostępne pod: https://scikit-learn.org/stable/modules/naive_bayes.html#naive-bayes, Data dostępu 25.05.2023

[6] Humaira Yasmin Aliza ,Kazi Aahala Nagary, Eshtiak Ahmed, Kazi Mumtahina Puspita, Khadiza Akter Rimi, Ankit Khater, Fahad Faisal, „A Comparative Analysis of SMS Spam Detection employing Machine Learning Methods", rok 2022, Dostępne pod: https://ieeexplore.ieee.org/abstract/document/9754002/authors#authors , Data dostępu 25.05.2023