Wstęp do Sztucznej Inteligencji

Naive Bayes i klasyfikacja tekstów

mgr inż. Andrii Shekhovstov

Uniwersytet WSB Merito

20 marca 2025

Plan prezentacji

Główne zadania algorytmów SI

Algorytmy sztucznej inteligencji można podzielić na trzy główne kategorie:

- **Klasyfikacja** – przypisywanie nowych danych do jednej z predefiniowanych kategorii (rozpoznawanie pisma ręcznego, analiza sentymentu).
- **Regresja** – przewidywanie wartości liczbowych na podstawie dostępnych danych (przewidywanie cen mieszkań, prognozowanie pogody).
- **Klasteryzacja** – grupowanie podobnych danych bez wcześniejszych etykiet (segmentacja klientów, analiza danych genetycznych).

Podstawowe definicje w zbiorach danych

- **Zbiór danych** – kolekcja próbek wykorzystywanych do uczenia modelu (np. klasyfikatora).
- **Próbka (obiekt)** – pojedynczy element zbioru danych opisany cechami.
- **Atrybut (cecha)** – pojedyncza właściwość próbki (np. wysokość, kolor, wiek).
- **Etykieta** – wartość przypisana próbce w zadaniach nadzorowanych (np. kategoria obiektu).

Przygotowanie zbiorów danych do klasyfikacji

W przypadku zadań klasyfikacji zbiór danych przygotowujemy następująco:

- Dane wejściowe: X – zbiór próbek, gdzie każda próbka jest opisana cechami.
- Etykiety: y – przypisane klasy dla każdej próbki.
- Przykładowy zapis:

Cecha 1	Cecha 2	Cecha 3	Cecha 4	Etykieta
5.1	3.5	1.4	0.2	Iris-setosa
4.9	3.0	1.4	0.2	Iris-setosa
6.3	3.3	6.0	2.5	Iris-virginica

Podział na dane uczące i testowe

W jaki sposób sprawdzić, czy model działa poprawnie na nieznanych próbkach?

Stosujemy podziału zbioru danych na dane uczące i testowe:

- **Zbiór uczący (train set)** – wykorzystywany do trenowania modelu.
- **Zbiór testowy (test set)** – służy do oceny skuteczności modelu.

Często stosuje się podział 80% danych na trening i 20% na test (ale inne proporcje też możliwe).

Ocena dokładności klasyfikacji binarnej

Dokładność (accuracy) to podstawowa miara jakości modelu klasyfikacyjnego:

$$Accuracy = \frac{\text{Liczba poprawnie sklasyfikowanych próbek}}{\text{Liczba wszystkich próbek}}$$

Zakładając, że mamy po równo danych każdej klasy, dokładność jest dobrą miarą.

Natomiast przy niezbilansowanych danych pojawia się problem:

- Jeśli zbiór zawiera 95% klasy A i 5% klasy B, model może przewidywać tylko klasę A i uzyskać 95% accuracy, mimo że nie rozpoznaje klasy B.

Macierz konfuzji

Macierz konfuzji to narzędzie do oceny jakości klasyfikatora. Przedstawia liczbę poprawnych i błędnych klasyfikacji dla każdej klasy.

	Klasa rzeczywista	
	Pozytywne	**Negatywne**
Predykcja: Pozytywne	TP	FP
Predykcja: Negatywne	FN	TN

- **TP**: poprawnie przewidziane pozytywne przypadki.
- **TN**: poprawnie przewidziane negatywne przypadki.
- **FP**: błędnie sklasyfikowane negatywy jako pozytywne.
- **FN**: błędnie sklasyfikowane pozytywy jako negatywne.

Macierz konfuzji - Przykład

Załóżmy że chcemy przygotować klasyfikator, rozpoznający rzadką chorobę, która występuje u 1 osoby z 1000.

Przygotowany model osiąga dokładność 99% - ale spojrzymy na macierz konfuzji:

	Klasa rzeczywista	
	Pozytywne	**Negatywne**
Predykcja: Pozytywne	0	0
Predykcja: Negatywne	1	999

Z macierzy konfuzji widać, że klasyfikator zaznaczył wszystkie próbki jako zdrowe, osiągając w ten sposób wysokiej dokładności.

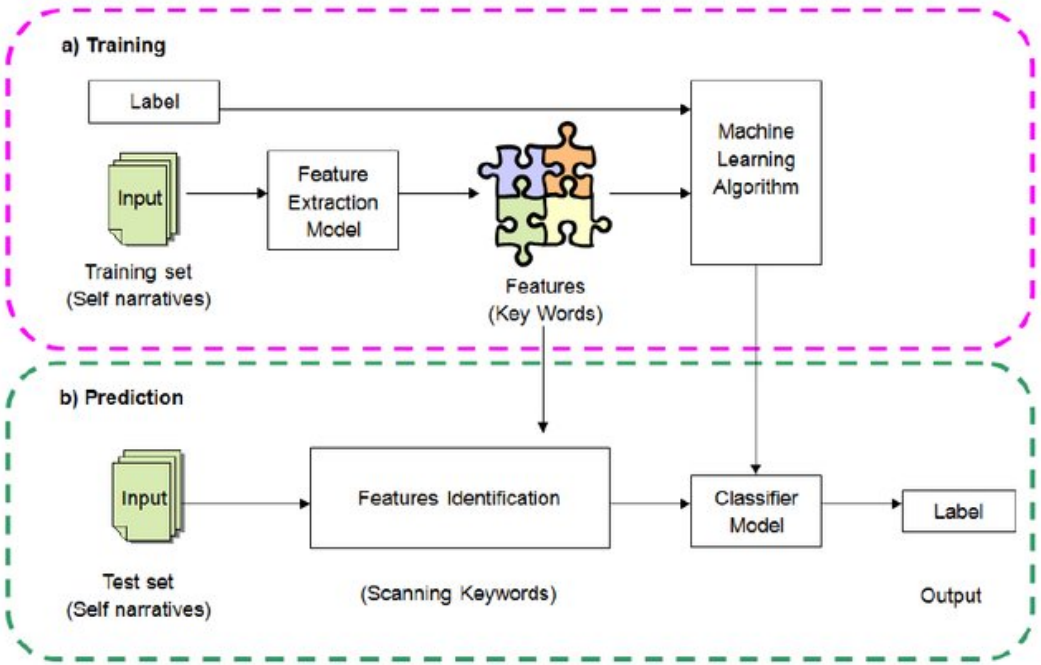
Typowe zadania przetwarzania tekstu

- **Klasyfikacja Tekstu**:
  - Klasyfikacja sentymentów
  - Filtry antyspamowe
  - Kategoryzacja dokumentów
- **Ekstrakcja Informacji**:
  - Rozpoznawanie jednostek nazwanych (NER, ang. Named Entity Recognition)
  - Ekstrakcja relacji
- **Tłumaczenie Maszynowe**:
  - Automatyczne tłumaczenie tekstu między językami
- **Generacja Tekstu**:
  - Tworzenie automatycznych podsumowań
  - Generacja kreatywnych tekstów
- **Analiza Sentymentów**:
  - Ocena emocjonalnego zabarwienia tekstu

Klasyfikacja tekstu – definicja

Definicja

To zadanie polegające na przypisaniu etykiety lub kategorii do całego tekstu lub dokumentu.



[1]

[1] Classifying Unstructured Textual Data Using the Product Score Model: An Alternative Text Mining Algorithm

Analiza wydźwięku

Example

Przykładem klasyfikacji może być analiza wydźwięku, czyli uczuć, pozytywnych lub negatywnych, które piszący wyraża wobec jakiegoś przedmiotu.

- Recenzja filmu, książki lub produktu w sieci wyraża sentyment autora do produktu.
- Tekst redakcyjny lub polityczny wyraża sentyment do kandydata lub działania politycznego.

Analiza wydźwięku cd.

Fragmenty tekstów

1. ...bogato zastosowana satyra, i kilka dużych zwrotów akcji.
2. To było żałosne. Najgorsze były sceny bokserskie...
3. ...wspaniały sos karmelowy i niesamowite tosty z migdałów. Uwielbiam to miejsce!
4. ...wstrętna pizza i zawyżona cena...

Najprostsza wersja analizy wydźwięku jest zadaniem klasyfikacji binarnej, a słowa recenzji stanowią zbiór słów wskaźników. Na przykład frazy zaczerpnięte z pozytywnych i negatywnych recenzji filmów i restauracji. Słowa takie jak dużych, bogato, niesamowite i żałosne oraz okropne i najgorsze są wskaźnikami.

Identyfikacja wiadomości niechcianych

Example

Wykrywanie spamu to ważne zastosowanie komercyjne. Zadanie klasyfikacji polega na przypisaniu poczty elektronicznej do jednej z dwóch klas spamu lub nie-spamu. Do klasyfikacji można wykorzystać wiele funkcji leksykalnych i innych.

Na przykład można być podejrzliwym jeżeli mail zawiera zwroty takie jak

- bezkosztowo
- drogi zwycięzco
- rabat 25%

Identyfikacja języka, wskazanie autora

Example

Wykrywanie języka Na przykład teksty w mediach społecznościowych mogą być napisane w różnych języków i od tego będzie zależała dalsze ich przetwarzanie.

Example

Określenie autora tekstu (przypisanie autorstwa) lub cechy autora, takie jak płeć, wiek i język ojczysty. Są to zadania klasyfikacji, które są istotne dla nauk humanistycznych, społecznych i językoznawstwa kryminalistycznego.

Określenie tematyki tekstu

Example

Jednym z najstarszych zadań w klasyfikacji tekstu jest przypisanie do tekstu kategorii tematycznej lub etykiety tematu. Algorytm naive-Bayes został wykorzystany w klasyfikacji tekstu w 1961 roku.

Decydowanie, czy praca badawcza dotyczy epidemiologii, czy też może embriologii, jest ważnym elementem pozyskiwania informacji. Istnieją różne zestawy kategorii tematycznych, np. tezaurus MeSH (Medical Subject Headings).

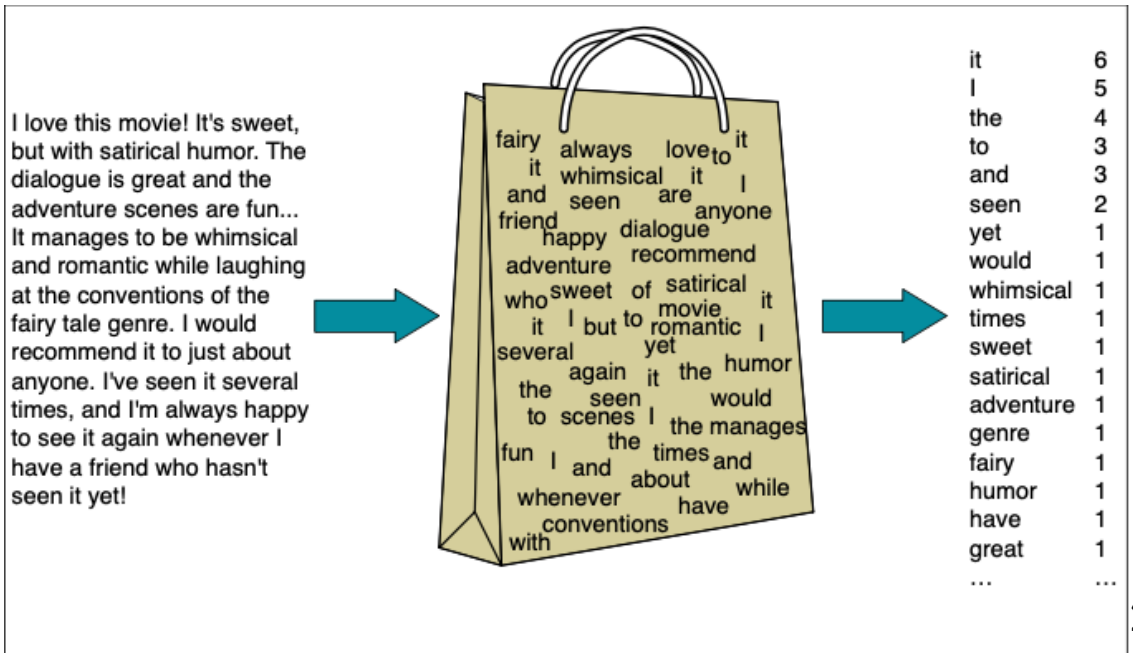
Ekstrakcja cech z tekstu

- Model "bag-of-words"(kosz słów) - Reprezentacja tekstu jako zbioru słów z ich częstotliwościami.
- TF-IDF (Term Frequency-Inverse Document Frequency) - Mierzy ważność słowa w dokumencie w stosunku do całego korpusu.
- Word Embeddings - Reprezentacja słów jako wektorów w przestrzeni wielowymiarowej (np. Word2Vec, GloVe).
- N-gramy - Sekwencje słów o długości n, używane do uwzględnienia kontekstu.
- Cechy syntaktyczne - Informacje o strukturze zdania, takie jak części mowy.
- Cechy semantyczne - Informacje o znaczeniu, takie jak synonimy czy antonimy.

Bag od words

Bag of words

Dokument tekstowy traktujemy jakby był workiem słów (bag-of-words), czyli nieuporządkowanym zbiorem słów z pominięciem ich pozycji. Zapamiętujemy jedynie ich częstotliwość.



[2]

[2] Speech and Language Processing by D Jurafsky, J.H. Martin

Przykład

Rozpatrzmy przykład z dwoma poniższymi zdaniami:

1. Good food, very nice place.
2. Food wasn't good, but place is nice.
3. Can't recommend. Awful service.

W wyniku powstaje taka tabelka częstotliwości występowania słów:

	good	food	very	nice	place	wasn't	but	is	can't	recommend	awful	service
Fraza 1	1	1	1	1	1	0	0	0	0	0	0	0
Fraza 2	1	1	0	1	1	1	1	1	0	0	0	0
Fraza 3	0	0	0	0	0	0	0	0	1	1	1	1

Przykład

- W naszym przykładzie mieliśmy 3 frazy i 12 unikatowych słów, co sprawia że musimy przechowywać $3 \cdot 12 = 36$ liczb, na $36 \cdot 8 = 288$ bajtach (licząc że używamy tylko 8 bit na jedną liczbę).
- Dla 10 fraz i 100 unikatowych słów potrzebujemy $10 \cdot 100 \cdot 8 = 8000$ - 8 KiB przestrzeni.
- Dla 1000 fraz i 10000 słów: $1000 \cdot 10000 \cdot 8 = 80'000'000$ - 80 MiB przestrzeni.
- Im więcej mamy słów i fraz (a potrzebujemy dużo danych dla dobrego modelu) tym więcej potrzebujemy miejsca.
- Chwila, ale te liczby to są w większości zera?
- Typowe wypełnienie macierzy w przypadku podejścia bag of words wynosi od 1% do 10%, czasami mniej.

Macierze rzadkie

Macierz rzadka (sparse) to macierz, w której większość elementów to zera. Korzyści wynikające z zastosowania macierz rzadkich to efektywność pamięciowa oraz szybsze obliczenia.

- **CSR (Compressed Sparse Row)** - Skompresowane wiersze, efektywne dla operacji wierszowych.
- **CSC (Compressed Sparse Column)** - Skompresowane kolumny, efektywne dla operacji kolumnowych.
- **COO (Coordinate List)** - Lista trójek (wiersz, kolumna, wartość), prosta do implementacji

Więcej informacji o macierzach rzadkich w Python można przeczytać tutaj: `https://docs.scipy.org/doc/scipy/reference/sparse.html`

Format COO (Coordinate List)

Rozpatrzmy przykładową macierz:

$$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

W formacie COO ta macierz będzie zapisana jako:

- Wartości (values): [1, 1]
- Indeksy wierszy (row_indices): [1, 2]
- Indeksy kolumn (col_indices): [0, 2]

Czyli w tym wypadku przechowujemy tylko 6 wartości zamiast 16.

Więcej informacji

Więcej o ekstrakcji cech z tekstów można przeczytać tu:

https://scikit-learn.org/stable/modules/feature_extraction.html#text-feature-extraction

Dyskretny klasyfikator Bayesa

Działanie

Naive Bayes jest klasyfikatorem probabilistycznym. Dla dokumentu d, ze wszystkich klas $c \in C$ klasyfikator zwraca klasę $\hat{c}$, która ma maksymalne prawdopodobieństwo. Innymi słowy wskazuje szacowaną najbardziej prawdopodobną kategorią (Maximum a posteriori (MAP) Hypothesis).

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} P(c \mid d)$$

Stosując regułę Bayesa:

$$P(x \mid y) = \frac{P(y \mid x)P(x)}{P(y)}$$

uzyskujemy następujący wzór na wskazanie klasy:

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} P(c \mid d) = \underset{c \in C}{\operatorname{argmax}} \frac{P(d \mid c)P(c)}{P(d)} \tag{1}$$

Uproszczenie

Można zauważyć, że $P(d)$ jest niezmienne, gdyż liczba dokumentów d jest zadana i niezależna od klasy.

W konsekwencji uproszczamy równanie (1) poprzez opuszczenie mianownika.

To prowadzi do formuły:

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} \overbrace{P(d \mid c)}^{\text{likelihood}} \overbrace{P(c)}^{\text{prior}} \tag{2}$$

W ten sposób obliczamy najbardziej prawdopodobną klasę danego dokumentu, wybierając klasę, która ma najwyższy iloczyn dwóch prawdopodobieństw: prawdopodobieństwa klasy $P(c)$ i prawdopodobieństwa dokumentu $P(d|c)$ w danej klasie.

Reprezentacja dokumentu

Cechy wydobyte z dokumentu

Bez utraty generalizacji możemy przedstawić dokument d jako wektor cech $f_1, f_2, \ldots, f_n$

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} \overbrace{P(f_1, f_2, \ldots, f_n \mid c)}^{\text{likelihood}} \overbrace{P(c)}^{\text{prior}}$$

Niestety równanie jest wciąż trudne do bezpośredniego wyliczenia gdyż oszacowanie prawdopodobieństwa każdej możliwej kombinacji cech (np. każdego możliwego zestawu słów i ich pozycji w dokumencie) wymagałoby ogromnej liczby parametrów i niemożliwie dużych zestawów treningowych. Naiwny klasyfikator Bayes'a przyjmuje dwa założenia upraszczające.

Dwa założenia w NCB

- **Bag-of-words**: zakładamy, że położenie słowa nie ma znaczenia, a słowo ma taki sam wpływ na klasyfikację, niezależnie od tego czy występuje jako pierwsze, dwudzieste, czy ostatnie słowo w dokumencie. Zakładamy więc, że cechy $f_1, f_2, \ldots, f_n$ kodują tylko nazwę słowa a nie pozycję.
- Drugie jest nazywane założeniem warunkowej **niezależności Naïve Bayes**: jest to założenie, że prawdopodobieństwa $P(f_i|c)$ są niezależne, biorąc pod uwagę klasę c, a zatem :

$$P\left(f_1, f_2, \ldots, f_n \mid c\right) = P\left(f_1 \mid c\right) \cdot P\left(f_2 \mid c\right) \cdot \ldots \cdot P\left(f_n \mid c\right)$$

Stąd:

$$c_{NB} = \underset{c \in C}{\operatorname{argmax}} P(c) \prod_{f \in F} P(f \mid c)$$

Słowo jako cecha dokumentu

Jeżeli założymy, że cechą opisującą dokument jest słowo, to uzyskamy następujący wzór, przy założeniu, że w_i to i-te słowo występujące w dokumencie.

$$c_{NB} = \underset{c \in C}{\operatorname{argmax}} P(c) \prod_{i=1,\ldots,n} P(w_i \mid c)$$

Jest to naturalna konsekwencja zastosowania Bag of Words.

Przestrzeń log

Wykonanie obliczeń w przestrzeni log, zwiększa tempo uczenia i zapobiega zmniejszeniu się wartości poprzez mnożenie prawdopodobieństw.

Po przejściu do przestrzeni log uzyskujemy następują wzór na MAP

$$c_{NB} = \underset{c \in C}{\operatorname{argmax}} \log(P(c)) \sum_{i=1,\ldots,n} \log(P(w_i \mid c)) \tag{3}$$

Uczenie NBC $P(c)$

Uczenie polega na wyznaczeniu wartości prawdopodobieństw $P(c)$ i $P(w_i|c)$.

Obliczanie $P(c)$ i $P(w_i|c)$

Użyte zostanie podejście częstotliwościowe. Prawdopodobieństwo klasy to określenie, jaki procent dokumentów w zbiorze treningowym znajduje się w każdej klasie c. Czyli:

Niech N_c to liczba dokumentów w zbiorze treningowym z przypisaną klasą c, a N_{doc} to całkowita liczba dokumentów. Wówczas:

$$P(c) = \frac{N_c}{N_{doc}}$$

Uczenie NBC $P(w_i|c)$

Prawdopodobieństwo wystąpienia słowa w klasie $P(w_i|c)$, oznacza istnienie słowa w worku słów danej klasy. Zatem $P(w_i|c)$ obliczymy jako iloraz pojawień się słowa w_i we wszystkich dokumentach tematu/klasy c i słów występujących w klasie.
Zakładamy, że:

1. Łączymy wszystkie dokumenty z kategorii c w jeden duży tekst "kategorii c".
2. Tworzymy zbiór wszystkich słów we wszystkich klasach i nazywamy go słownikiem V.
3. Następnie obliczamy częstotliwości w_i w połączonym dokumencie.

$$P\left(w_i \mid c\right) = \frac{\text{count}\left(w_i, c\right)}{\sum_{w \in V} \text{count}(w, c)}$$

Problem z wartościami zerowymi

Example

Rozważamy przykład: Chcemy oszacować prawdopodobieństwo wystąpienia słowa „fantastic" w danej klasie, ale załóżmy, że nie ma dokumentów w zbiorze uczącym, które zawierają słowo „fantastic" i są sklasyfikowane jako pozytywne. Słowo „fantastic"pojawiło się użyte sarkastycznie w klasie negatywnej. W takim przypadku prawdopodobieństwo pojawienia się tej cechy w klasie pozytywnej będzie zerowe:

$$P\left(fantastic \mid positive\right) = \frac{\text{count}\left(fantastic, positive\right)}{\sum_{w \in V} \text{count}(w, positive)} = 0$$

W NCB mnoży się wszystkie prawdopodobieństwa warunkowe i jedna wartość zerowa w klasie spowoduje, że prawdopodobieństwo klasy będzie zerowe, bez względu na inne dowody (słowa)!

Rozwiązanie problemu z wartościami zerowymi

Wygładzanie Laplace'a jest powszechnie stosowane w klasyfikacji tekstu:

$$P\left(w_i \mid c\right) = \frac{\text{count}\left(w_i, c\right) + 1}{\sum_{w \in V}(\text{count}(w, c) + 1)} = \frac{\text{count}\left(w_i, c\right) + 1}{\left(\sum_{w \in V} \text{count}(w, c)\right) + |V|}$$

gdzie $|V|$ liczność słownika, czyli wszystkich słów z dokumentów ze wszystkich klas.

Testowanie NBC nieistniejącym słowem

Co zrobić ze słowami, które pojawiają się w danych testowych, ale nie ma ich w słoniku V, ponieważ nie pojawiły się w żadnym dokumencie trenującym w żadnej z klas?

Rozwiązaniem dla nieznanych słów jest zignorowanie ich - usunięcie ich z dokumentu testowego i nie uwzględnianie dla nich wiarygodności.

Stop words

Usprawnieniem modelu klasyfikacyjnego może być ominięcie słów neutralnych. Z tego powodu niektóre systemy decydują się całkowicie zignorować pewną klasę słów tzw. „stop words", np. w j. ang „the" i „a" itp.

Wykrycie stop-words

- Można posortować słowa według częstotliwości słów/wpisów do słownika i zdefiniowanie pierwszych od 10 do 100 jako „stop words".
- Alternatywnie można użyć jednej z wstępnie zdefiniowanych list „stop words" dostępnych online.

Następnie wszystkie są usuwane z dokumentów treningowych i testowych, tak jakby nigdy nie ich tam nie było.
Uwaga W większości programów do klasyfikacji tekstu, użycie listy „stop words" nie poprawia wydajności, dlatego też częściej korzysta się z całego V .

Algorytm NBC

function TRAIN NAIVE BAYES(D, C) **returns** log $P(c)$ and log $P(w|c)$

for each class $c \in C$ # Calculate $P(c)$ terms
  N_{doc} = number of documents in D
  N_c = number of documents from D in class c
  $logprior[c] \leftarrow \log \frac{N_c}{N_{doc}}$
  $V \leftarrow$ vocabulary of D
  $bigdoc[c] \leftarrow$ **append**(d) **for** d $\in$ D **with** class c
  for each word w in V # Calculate $P(w|c)$ terms
    $count(w,c) \leftarrow$ # of occurrences of w in $bigdoc[c]$
    $loglikelihood[w,c] \leftarrow \log \frac{count(w,c) + 1}{\sum_{w' \text{ in } V} (count(w',c) + 1)}$
return *logprior*, *loglikelihood*, V

function TEST NAIVE BAYES(*testdoc*, *logprior*, *loglikelihood*, C, V) **returns** best c

for each class $c \in C$
  $sum[c] \leftarrow logprior[c]$
  for each position i in *testdoc*
    $word \leftarrow testdoc[i]$
    if $word \in V$
      $sum[c] \leftarrow sum[c] + loglikelihood[word,c]$
return $\text{argmax}_c\ sum[c]$

[3]

[3] Speech and Language Processing by D Jurafsky, J.H. Martin

Przykład - zbiór danych

Dany jest zbiór uczący i testujący.

	C	Doc
Zbiór uczący	-	just plain boring
	-	entirely predictable and lacks energy
	-	no surprises and very few laughs
	+	very powerful
	+	the most fun film of the summer
Zbiór testowy	?	predictable with no fun

- Liczba dokumentów w klasie pozytywnej - 2
- Liczba dokumentów w klasie negatywnej - 3
- Liczba słów w słowniku - 20
- Liczba słów w klasie pozytywnej - 9
- Liczba słów w klasie negatywnej - 14

Przykład – uczenie (cząstkowe)

Dla klas liczmy $P(c)$

$$P(-) = \frac{3}{5} \quad P(+) = \frac{2}{5}$$

Następnie obliczamy tylko prawdopodobieństwa słów ze zdania testowego „predictable with no fun”

Słowo „with” nie występuje w zbiorze uczącym, więc zostaje odrzucone. Prawdopodobieństwa warunkowe dla pozostałych trzech słów „predictable”, „no” i „fun” są następujące:

$$\begin{array}{c|c} P(\text{ predictable } \mid -) = \frac{1+1}{14+20} & P(\text{ predictable } \mid +) = \frac{0+1}{9+20} \\ P(\text{ no } \mid -) = \frac{1+1}{14+20} & P(\text{ no } \mid +) = \frac{0+1}{9+20} \\ P(\text{ fun } \mid -) = \frac{0+1}{14+20} & P(\text{ fun } \mid +) = \frac{1+1}{9+20} \end{array}$$

Przykład – klasyfikacja

Zdanie „predictable with no fun" po usunięciu słowa „with" ma szacowane prawdopodobieństwo przynależności do klasy „-" i „+" odpowiednio:

$$P(-) * P(S|-) = \frac{3}{5} \cdot \frac{2 \cdot 2 \cdot 1}{34^3} = 6.1 \cdot 10^{-5}$$

$$P(+) * P(S|+) = \frac{2}{5} \cdot \frac{1 \cdot 1 \cdot 2}{29^3} = 3.2 \cdot 10^{-5}$$

Model przewiduje klasę negatywną dla zdania testowego gdyż

$$P(-) * P(S|-) > P(+) * P(S|+)$$

Możliwe poprawki wprowadzone do algorytmu

Standardowy NBC może się dobrze sprawdzać w klasyfikacji tekstów, można jednak zastosować niewielkie zmiany, które poprawiają wydajność.

Binarnym NB - opis

Dla klasyfikacji tekstu, bardzo często informacja czy słowo występuje czy nie, wydaje się mieć większe znaczenie niż jego częstotliwość. Można zatem słowo liczyć w każdym dokumencie tylko raz. W ten sposób poprawia się wydajność. Ten wariant jest nazywany binarnym NB.

Wariant algorytmu używa tego samego równania, z tą różnicą, że dla każdego dokumentu usuwamy wszystkie duplikaty słów przed połączeniem ich w jeden duży dokument.

Binarny NB - przykład

Cztery dokumenty (skrócone i znormalizowane) są zamienione na binarne. Liczba wystąpień nie musi być 1; słowo „great" ma liczbę 2 nawet dla binarnego NB, ponieważ pojawia się w wielu dokumentach.

Four original documents:

- − it was pathetic the worst part was the boxing scenes
- − no plot twists or great scenes
- \+ and satire and great plot twists
- \+ great scenes great film

After per-document binarization:

- − it was pathetic the worst part boxing scenes
- − no plot twists or great scenes
- \+ and satire great plot twists
- \+ great scenes film

	NB Counts		Binary Counts	
	+	−	+	−
and	2	0	1	0
boxing	0	1	0	1
film	1	0	1	0
great	3	1	2	1
it	0	1	0	1
no	0	1	0	1
or	0	1	0	1
part	0	1	0	1
pathetic	0	1	0	1
plot	1	1	1	1
satire	1	0	1	0
scenes	1	2	1	2
the	0	2	0	1
twists	1	1	1	1
was	0	2	0	1
worst	0	1	0	1

[4]

[4] Speech and Language Processing by D Jurafsky, J.H. Martin

Klasyfikacja sentymentów – negacje

Example

Weź pod uwagę różnicę między

- „I really like this movie" (pozytywny) a
- „I didn't like this movie" (negatywny).

Negacja wyrażona przez „didn't" zmieniła całkowicie znaczeni słowa „like".

Podobnie, negacja może zmodyfikować negatywne słowo, aby stworzyć pozytywną recenzję („don't dismiss this film, doesn't let us get bored").

Klasyfikacja sentymentów – negacje cd.

Dodanie przedrostka

Powszechnie stosowaną metodą radzenia sobie z negacją: podczas normalizacji tekstu dodaj przedrostek „NOT” do każdego słowa po znaku logicznej negacji (n’t, not, no, never). Tak więc zdanie:

Example

„didn’t like this movie , but I”

staje się

„didn’t NOT_like NOT_this NOT_movie , but I”

Leksykony

W niektórych sytuacjach możemy mieć niewystarczające dane uczące (mała liczba etykiet), aby nauczyć NBC używających wszystkich dostępnych słów w zbiorze uczącym. W takich przypadkach możemy wykorzystać leksykony – listy słów zetykietyzowanych.
Dla rozpoznania semntymentu mogą to być:

1. General Inquirer
2. LIWC
3. MPQA zawiera 6885 słów, 2718 pozytywnych i 4912 negatywnych, z których każde jest zaznaczone, czy jest mocno czy słabo stronnicze.
4. opinion lexicon

Leksykony – sposoby wykorzystania

Sposobem na użycie leksykonów w NBC jest dodanie funkcji, która jest liczona za każdym razem, gdy pojawia się słowo z leksykonu.

Leksykon gęsty

Tworzy się dwie cechy: „to słowo występuje w pozytywnym leksykonie” i „to słowo występuje w leksykonie negatywnym”. Traktujemy wszystkie instancje słów w leksykonie jako zliczające się dla jednej z dwóch cech. (Nie liczymy każdego słowa osobno).

Cechy:

- Jeśli mamy dużo danych w zbiorze uczącym i jeśli dane testowe odpowiadają danym w zbiorze uczącym, użycie tylko dwóch cech leksykalnych nie będzie działać tak dobrze, jak użycie wszystkich słów.
- Natomiast, gdy dane treningowe są skąpe lub nie są reprezentatywne dla zestawu testowego, używanie gęstego leksykonu zamiast skąpych pojedynczych słów może prowadzić do lepszej generalizacji.

Przetwarzanie tekstu - część 1

Lematyzacja

to sprowadzanie danego słowa do jego formy podstawowej (hasłowej), która reprezentuje dany wyraz.

Example

wiórkami —> Lematyzacja: **wiórek**
jeżdżący —> Lematyzacja: **jeździć**
uległa —> Lematyzacja: **ulec**

Przetwarzanie tekstu - część 2

Stemming

Analiza morfologiczna zwracająca dla zadanego słowa jego znaczenionośny rdzeń (morfem/stemm) słowa.

Example

zbudowany —> Morfem: **budow**

Źródła

- **Probability Theory** https://www.cs.huji.ac.il/course/2005/cbio/Handouts/probability.pdf
- **NLP texbook** https://web.stanford.edu/~jurafsky/slp3/

Dziękuję za uwagę!