Lab1: Klasyfikacja tekstów

Tokenizacja, Bag of words, Lematyzacja, Stemming, TF-IDF

---

mgr inż. Dawid Wiśniewski

Pokój: 2.7.2 BT (konsultacje 15:10 - 16:40, wtorki)

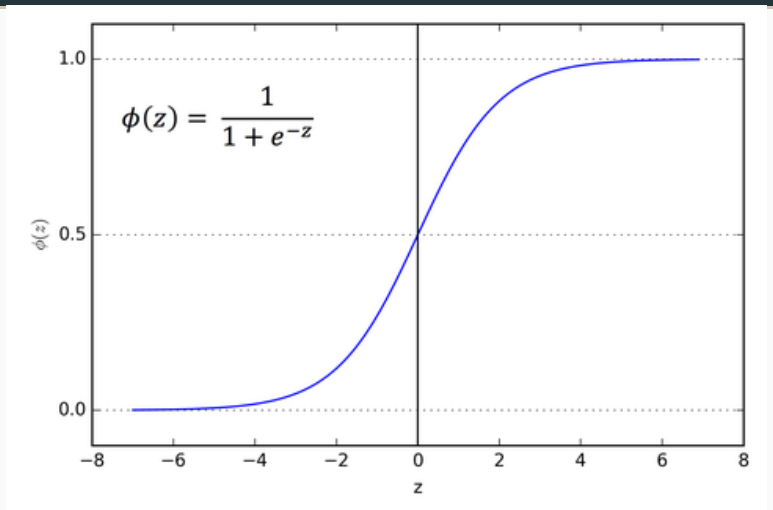
Klasyfikacja - regresja logistyczna

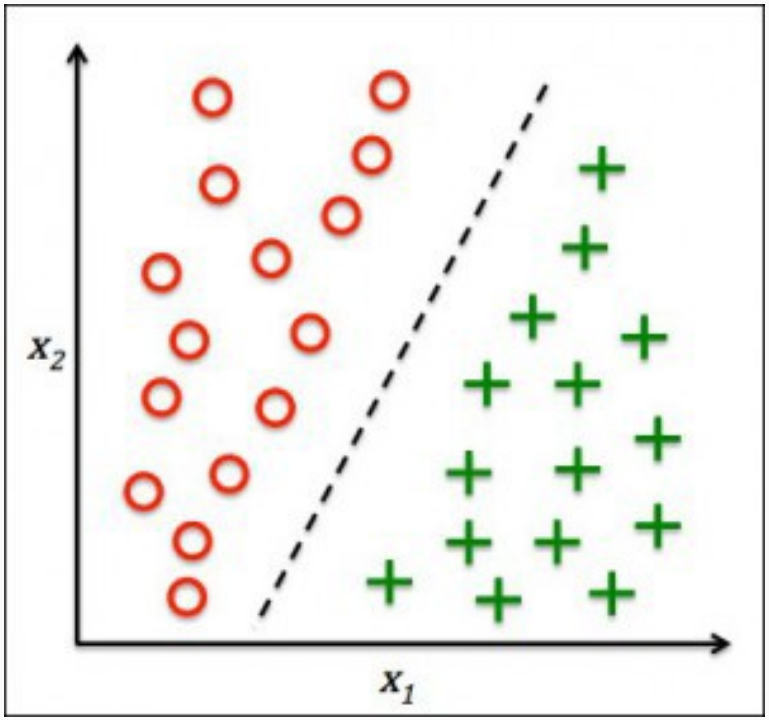
Metraż {x1}	Cena {x2}	Warto?
52	240	TAK
32	200	NIE
52	250	TAK
55	270	TAK

$z = \vec{w}\vec{x} + b = w_1 \cdot x_1 + w_2 \cdot x_2 + b$

$y = \frac{1}{1+e^{-z}}$

y - prawdopodobieństwo, że warto





Klasyfikacja w NLP

- detekcja spamu (spam/ham)
- automatyczna moderacja (wykrywanie hejtu) (hejt/nie-hejt)
- fact checking (fakt/bzdura)
- wykrywanie sentymentu (pozytywny, negatywny, neutralny)
- detekcja języka (polski, angielski, hiszpański)

. . . ale tekst to nie liczby

Większość algorytmów uczenia maszynowego oczekuje cech w postaci liczbowej. Tekst to nie liczby :(.

. . . ale tekst to nie liczby

Większość algorytmów uczenia maszynowego oczekuje cech w postaci liczbowej. Tekst to nie liczby :(.

Zatem zróbmy z tekstów liczby:

Dok1: 'Ala ma kota i Ala ma psa.'

Dok2: 'Ala ma poprawkę z MP.'

. . . ale tekst to nie liczby

Większość algorytmów uczenia maszynowego oczekuje cech w postaci liczbowej. Tekst to nie liczby :(.

Zatem zróbmy z tekstów liczby:

Dok1: 'Ala ma kota i Ala ma psa.'

Dok2: 'Ala ma poprawkę z MP.'

1. Podzielmy tekst na tokeny (tokenizacja):
   Dok1: ['Ala', 'ma', 'kota', 'i', 'Ala', 'ma', 'psa', '.']

. . . ale tekst to nie liczby

Większość algorytmów uczenia maszynowego oczekuje cech w postaci liczbowej. Tekst to nie liczby :(.

Zatem zróbmy z tekstów liczby:

Dok1: 'Ala ma kota i Ala ma psa.'

Dok2: 'Ala ma poprawkę z MP.'

1. Podzielmy tekst na tokeny (tokenizacja):
   Dok1: ['Ala', 'ma', 'kota', 'i', 'Ala', 'ma', 'psa', '.']
2. Ze stokenizowanego zbioru wszystkich dokumentów stwórzmy listę unikalnych tokenów w nich zaobserwowanych (słownik/vocabulary):
   ['Ala', 'ma', 'kota', 'i', 'psa', '.', 'poprawkę', 'z', 'MP']

. . . ale tekst to nie liczby

Większość algorytmów uczenia maszynowego oczekuje cech w postaci liczbowej. Tekst to nie liczby :(.

Zatem zróbmy z tekstów liczby:

Dok1: 'Ala ma kota i Ala ma psa.'

Dok2: 'Ala ma poprawkę z MP.'

1. Podzielmy tekst na tokeny (tokenizacja):
   Dok1: ['Ala', 'ma', 'kota', 'i', 'Ala', 'ma', 'psa', '.']
2. Ze stokenizowanego zbioru wszystkich dokumentów stwórzmy listę unikalnych tokenów w nich zaobserwowanych (słownik/vocabulary):
   ['Ala', 'ma', 'kota', 'i', 'psa', '.', 'poprawkę', 'z', 'MP']
3. Reprezentujmy dokumenty jako wektory, na i-tej pozycji liczba wystąpień i-tego słowa ze słownika w naszym dokumencie
   Dok1: [2, 2, 1, 1, 1, 1, 0, 0, 0]
   Dok2: [1, 1, 0, 0, 0, 1, 1, 1, 1]

Bag of Words

Tak skonstruowana reprezentacja to **Bag of Words**.

Działa całkiem nieźle mimo oczywistych wad:

['Mike', 'likes', 'dog', '.', 'cat']

- Mike likes dog. [1, 1, 1, 1, 0]
- Dogs likes Mike. [1, 1, 1, 1, 0]

Przekleństwo wymiarowości / curse of dimensionality

Rozmiar słownika to często dziesiątki / setki tysięcy tokenów.

Mając nawet tysiące przykładów treningowych - zaobserwujemy tylko nieznaczny ułamek możliwych zestawów cech.

Ryzyko przeuczenia - możemy wyuczyć się, że bardzo rzadkie tokeny (liczby, literówki) dobrze separują klasy.

Selekcja cech - feature selection

Ograniczmy rozmiar słownika bazując na liczności słów(tokenów) - intuicyjnie - słowa(tokeny) częste są ważniejsze, więc ograniczmy słownik do n najczęstszych . . . ale czy na pewno jest to dobry pomysł?

Selekcja cech - feature selection

Ograniczmy rozmiar słownika bazując na liczności słów(tokenów) - intuicyjnie - słowa(tokeny) częste są ważniejsze, więc ograniczmy słownik do *n* najczęstszych . . . ale czy na pewno jest to dobry pomysł?

Najczęściej występujące słowa w dokumentach spamowych: ['the', 'of', 'a' 'an', 'in'] Najczęściej występujące słowa w dokumentach niespamowych: ['the', 'a', 'an', 'of', 'in']

Selekcja cech - feature selection

Ograniczmy rozmiar słownika bazując na liczności słów(tokenów) - intuicyjnie - słowa(tokeny) częste są ważniejsze, więc ograniczmy słownik do *n* najczęstszych . . . ale czy na pewno jest to dobry pomysł?

Najczęściej występujące słowa w dokumentach spamowych: ['the', 'of', 'a' 'an', 'in'] Najczęściej występujące słowa w dokumentach niespamowych: ['the', 'a', 'an', 'of', 'in']

Można usunąć listę najpopularniejszych słów w języku (stoplistę), ale na dłuższą metę to również nie jest idealne rozwiązanie.

TF-IDF

Miara TF-IDF - słowa ważne to takie, które dobrze różnicują klasy - występują często w jednej klasie, ale rzadko (najlepiej wcale) w innych.

TF-IDF

Miara TF-IDF - słowa ważne to takie, które dobrze różnicują klasy - występują często w jednej klasie, ale rzadko (najlepiej wcale) w innych.

$$TFIDF_{i,j} = tf_{i,j} \cdot idf_i$$

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}}$$

tf - częstość występowania tokenu w danym dokumencie

$$idf_i = log \frac{|D|}{|\{d : t_i \in d\}|}$$

idf - logarytm z ilości wszystkich dokumentów, przez ilość dokumentów gdzie dany token występuje co najmniej raz

TF-IDF

Miara TF-IDF - słowa ważne to takie, które dobrze różnicują klasy - występują często w jednej klasie, ale rzadko (najlepiej wcale) w innych.

$$TFIDF_{i,j} = tf_{i,j} \cdot idf_i$$

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}}$$

tf - częstość występowania tokenu w danym dokumencie

$$idf_i = log \frac{|D|}{|\{d : t_i \in d\}|}$$

idf - logarytm z ilości wszystkich dokumentów, przez ilość dokumentów gdzie dany token występuje co najmniej raz

Po zastosowaniu TF-IDF najważniejsze słowa: ['free', 'won', 'call', 'subscribe']

Dalsze usprawnienia - stemming i lematyzacja

Fleksja jest przydatna, ale często w problemie klasyfikacji zbędna (ważniejsza jest obecność słowa niż jego odmiana).

Stemming - pozostawienie tematu wyrazu poprzez usunięcie końcówek fleksyjnych: studies -> studi, studied->studi

Lematyzacja - zamiana słowa na jego wersję podstawową studies: -> study, poszedł -> iść

Ewaluacja klasyfikatorów

Przed uruchomieniem treningu dzielimy zbiór danych na osobne zbiory: **treningowy** i **testowy**.

Uczymy algorytm na zbiorze treningowym, na zbiorze testowym zaś oceniamy jego jakość:

testset: [dok1: SPAM, dok2: SPAM, dok3, HAM, dok4: HAM]
predykcje: [dok1: HAM , dok2: SPAM, dok3, HAM, dok4: HAM]

Accuracy (trafność) - jaki procent decyzji podjętych przez klasyfikator można uznać za poprawny?

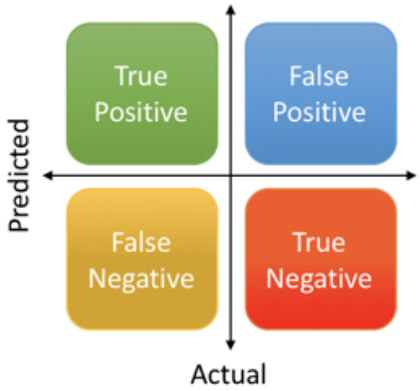
W przypadku powyżej: 3/4, zatem accuracy = 0.75

F1, precyzja, recall

Precision = $\frac{\text{True Positive}}{\text{Actual Results}}$ or $\frac{\text{True Positive}}{\text{True Positive + False Positive}}$

Recall = $\frac{\text{True Positive}}{\text{Predicted Results}}$ or $\frac{\text{True Positive}}{\text{True Positive + False Negative}}$

Accuracy = $\frac{\text{True Positive + True Negative}}{\text{Total}}$



$$F1 = \frac{2PR}{P+R}$$