Jednowarstwowe Sieci Neuronowe jako klasyfikatory do wielu klas

(c) Marcin Sydow

Plan

- dyskretny perceptron i jego ograniczenia
- inne funkcje aktywacji
- wielo-klasyfikacja przy pomocy jedno-warstwowe sieci neuronowej
- ograniczenia jedno-warstwowej sieci neuronowej
- miary ewaluacyjne dla klasyfikacji

Ograniczenia pojedynczego perceptronu

Pojedyczny perceptron może być użyty jako klasyfikator w przypadku najwyżej 2 klas

Nawet dla 2 klas są przypadki, które nie mogą być rozwiązane przez pojedynczy perceptron.

Ma też pewne oczywiste ograniczenia: może rozróżniać tylko rejony **liniowo-separowalne** w przestrzeni atrybutów

Funkcja aktywacji neuronu

Wartość "aktywacji" perceptronu: $\sum_i w_i x_i - \Theta$ (zwana także "net") jest następnie użyta jako argument w tzw. **funkcji aktywacji**, która ostatecznie zwraca wyjście neuronu.

Jest wiele rodzajów funkcji aktywacji.

Ze względu na typ numeryczny wartości wyjścia:

- dyskretny (liczba całkowita): może być użyty do klasyfikacji
- ciągły (liczba zmiennoprzecinkowa): może być użyty do regresji (lub również klasyfikacji)

Ze względu na maksymalną (aktywacja) i minimalną (brak aktywacji) zwracaną wartość:

- unipolarny (dyskretny: {0,1}, ciągły: [0,1])
- bipolarny (dyskretny: {-1,1}, ciągły: [-1,1])

Ze względu na "kształt" funkcji aktywacji (progowa, liniowa, sigmoidalna, etc.)

Przykłady najważniejszych funkcji aktywacji

Niech x oznacza wektor wejściowy, $net = \sum_i w_i x_i - \Theta$,
(y oznacza wyjście neuronu)

Najczęściej używane funkcje aktywacji:

- funkcja "signum" (znak): $y = signum(net)$
- funkcja progowa: $y = \lfloor x > 0 \rceil$
- funkcja sigmoidalna: $y = \frac{1}{1+e^{-net}}$
- funkcja liniowa ("surowe" wyjście): $y = net$

mini-test: które funkcje odpowiadają:

Przykłady najważniejszych funkcji aktywacji

Niech x oznacza wektor wejściowy, $net = \sum_i w_i x_i - \Theta$,
(y oznacza wyjście neuronu)

Najczęściej używane funkcje aktywacji:

- funkcja "signum" (znak): $y = signum(net)$
- funkcja progowa: $y = \lfloor x > 0 \rceil$
- funkcja sigmoidalna: $y = \frac{1}{1+e^{-net}}$
- funkcja liniowa ("surowe" wyjście): $y = net$

mini-test: które funkcje odpowiadają:
ciągłemu/dyskretnemu neuronowi?, unipolarnemu/bipolarnemu?

Przykłady najważniejszych funkcji aktywacji

Niech x oznacza wektor wejściowy, $net = \sum_i w_i x_i - \Theta$,
(y oznacza wyjście neuronu)

Najczęściej używane funkcje aktywacji:

- funkcja "signum" (znak): $y = signum(net)$
- funkcja progowa: $y = \lfloor x > 0 \rceil$
- funkcja sigmoidalna: $y = \frac{1}{1+e^{-net}}$
- funkcja liniowa ("surowe" wyjście): $y = net$

mini-test: które funkcje odpowiadają: ciągłemu/dyskretnemu neuronowi?, unipolarnemu/bipolarnemu?

która funkcja aktywacji nadaje się do klasyfikacji/regresji?

Funkcja sigmoidalna

Wariant unipolarny: $y = \frac{1}{1+e^{-net}}$

Wariant bipolarny: $y = \frac{2}{1+e^{-net}} - 1$

Funkcja może być wyposażona w parametr "stromości" $\lambda \in (0, \infty)$:

$$y = \frac{1}{1 + e^{-\lambda \cdot net}}$$

(im wyższa jego wartość tym bardziej stromy jest wykres funkcji)

Funkcja sigmoidalna ma kilka ważnych własności:

- jest ciągła i rosnąca
- własność "wzmacniania" (amplifikacji)
- ma pochodną i jej pochodna ma prostą formę podobną do tej samej funkcji (jest to ważna matematycznie własność dla metody wstecznej propagacji błędu w wielowarstwowych sieciach neuronowych)

Jednowarstwowa sieć neuronowa jako wielo-klasyfikator

Każdy pojedyczny perceptron może klasyfikować do 2 klas.

Gdy mamy więcej niż 2 klasy, możemy użyć całej **warstwy** perceptronów aby dokonywać klasyfikacji.

Typowa architektura jest następująca:

- każde wejście jest podłączone do każdego perceptrona
- wyjścia poszczególnych perceptronów są agregowane aby wyznaczyć wyjście całej takiej 1-warstwowej sieci

Interpretowanie wyjścia sieci neuronowej

W przypadku 2 klas, wyjście perceptronu stanowiącego klasyfikator jest naturalnie interpretowane:

- maximum aktywacji: “klasa 1”
- minimum aktywacji: “klasa 0”

W przypadku wielu klas, klasyfikator w formie 1-warstwowej sieci neuronów ma wiele wyjść. Istnieją 2 główne podejścia do architektury i reprezentacji wyjścia sieci:

- “lokalne”
- “globalne”

"Lokalna" architektura i reprezentacja wyjścia

Liczba perceptronów jest dokładnie taka sama jak liczba klas.

Każdy perceptron jest trenowany do aktywacji dla dokładnie jedej klasy

Prawidłowe wyjście takiej architektury jest następujące: dokładnie jeden perceptron jest aktywny (i wyznacza decyzję klasyfikatora) a pozostałe są niekatywne.

"Globalna" architektura i reprezentacja wyjścia

W tym przypadku liczba perceptronów nie jest dokładnie określona (ale może być mniejsza niż w lokalnej)

Decyzja klasyfikacyjna wyznaczana jest na podstawie kombinacji wyjść wszystkich perceptronów.

Uwaga: skoro każdy perceptron ma 2 możliwe wyjścia to dla K klas potrzeba nie mniej niż $log_2 K$ perceptronów (ale często więcej).

Lokalna reprezentacja ma tę zaletę, że jeśli jest możliwa to łatwiej ją wytrenować. Z drugiej strony, potrzebuje więcej perceptronów i nie zawsze można ją stosować.

Użycie ciągłych neuronów do klasyfikacji

Zamiast dyskretnych perceptronów w warstwie można użyć też ciągłych (o ciągłej funkcji aktywacji).

Wtedy podejście może być następujące:

- każdy ciągły neuron jest trenowany aby maksymalnie się aktywować tylko dla "swojej" klasy
- decyzja klasyfikacyjna podjęta jest na podstawie tego neurona, który się **maksymalnie aktywuje**

Takie podejście jest bardziej odporne na niepożądane sytuacje niż klasyczna dyskretna reprezentacja "lokalna", ponieważ praktycznie każde wyjście może być interpretowalne. (zredukowany jest problem jednoczesnej aktywacji wielu neuronów)

Ewaluacja klasyfikatorów

Używane są następujące miary ewaluacji klasyfikatorów:

- Dokładność (ang. accuracy)
- Precyzja i Pełność (Precision, Recall) (tylko 2 klasy)
- F-miara (tylko 2 klasy)
- Macierz omyłek (ang. Confusion Matrix) (dowolna liczba klas)

Dokładność (Accuracy) i wady tej miary

Najprosztszą miarą jakości klasyfikatora jest dokładność, czyli procentowy udział przypadków prawidłowo zaklasyfikowanych w zbiorze testowym

Problem: wyobraźmy sobie 2 klasy A i B, przy czym 99% przypadków klasyfikowanych jest do klasy A. W takim przypadku, "oszukany" klasyfikator, który zawsze "na ślepo" przyporządkowuje do klasy A osiągałby aż 99% dokładności! (w istocie jest bezużyteczny, rozważmy np. detektor pożaru, etc.)

Inne miary są potrzebne szczególnie w przypadku, gdy mamy wiele klas i błędy mają bardziej złożoną strukturę.

Macierz omyłek

Kwadratowa macierz $K \times K$, gdzie K jest liczbą klas.

Każdy wiersz odpowiada faktycznej klasie obiektów.

Każda kolumna odpowiada klasie wskazanej przez klasyfikator (być może nieprawidłowo)

Każda komórka (i, j) zawiera liczbę przypadków (lub procent) obiektów klasy i zaklasyfikowanych jako j.

Przykład:

zaklasyfikowano jako ->	a	b	c
a = Iris-setosa	50	0	0
b = Iris-versicolor	0	44	6
c = Iris-virginica	0	5	45

Pytanie: Jak wyglądałaby macierz idealnego klasfikatora?

Macierz omyłek

Kwadratowa macierz $K \times K$, gdzie K jest liczbą klas.

Każdy wiersz odpowiada faktycznej klasie obiektów.

Każda kolumna odpowiada klasie wskazanej przez klasyfikator (być może nieprawidłowo)

Każda komórka (i, j) zawiera liczbę przypadków (lub procent) obiektów klasy i zaklasyfikowanych jako j.

Przykład:

zaklasyfikowano jako ->	a	b	c
a = Iris-setosa	50	0	0
b = Iris-versicolor	0	44	6
c = Iris-virginica	0	5	45

Pytanie: Jak wyglądałaby macierz idealnego klasfikatora? (byłaby to macierz diagonalna)

Ewaluacja klasyfikatora gdy są tylko 2 klasy: Precyzja i Pełność

Gdy mamy tylko 2 klasy (nazwijmy je "pozytywną" i "negatywną") możemy użyć klasycznych miar precyzji i pełności (Precision i Recall) (oznaczane jako P oraz R)
Miary te pochodzą z dziedziny wyszukiwania informacji (ang. Information Retrieval, IR)

Definition

Precyzja to proporcja przypadków zaklasyfikowanych jako pozytywne i faktycznie pozytywnych do wszystkich zaklasyfikowanych jako pozytywne

Definition

Pełność to proporcja przypadków zaklasyfikowanych jako pozytywne i faktycznie pozytywnych do wszystkich faktycznie pozytywnych

Wartości P i R są pomiędzy 0 a 1 (im wyższe tym lepiej)

F-miara

W praktyce P i R są w pewnym sensie sprzeczne i zwykle poprawianie jednej z nich pogarsza drugą.

Ponieważ trudno jest w praktyce zbudować klasyfikator maksymalizujący równocześnie P i R, wprowadzona inną miarę, która zbiorczo reprezentuje te 2 miary za pomocą jendej liczby: **F-miara**:

Definition

$F = \frac{2 \cdot P \cdot R}{P+R}$ (jest to średnia harmoniczna P i R)

Intuicyjnie, jeśli F-miara jest wysoka, to obie miary P i R muszą być wysokie.

Przykład

Rozważmy następującą macierz omyłek:

zaklasyfikowano jako $\rightarrow$	pozytywne	negatywne
pozytywne	40	5
negatywne	10	45

Precyzja: $P = \frac{40}{(40+10)} = \frac{4}{5}$

Pełność: $R = \frac{40}{(40+5)} = \frac{8}{9}$

F-miara: $F = \frac{2 \cdot \frac{4}{5} \cdot \frac{8}{9}}{\frac{4}{5} + \frac{8}{9}} = \frac{64}{76} = \frac{16}{19}$

Zagadnienia do przyswojenia:

- dyskretny perceptron i jego ograniczenia
- inne funkcje aktywacji
- wielo-klasyfikacja przy pomocy jedno-warstwowe sieci neuronowej
- ograniczenia jedno-warstwowej sieci neuronowej
- miary ewaluacyjne dla klasyfikacji

Dziękuję za uwagę.