

**Komitety**

# Komitet?

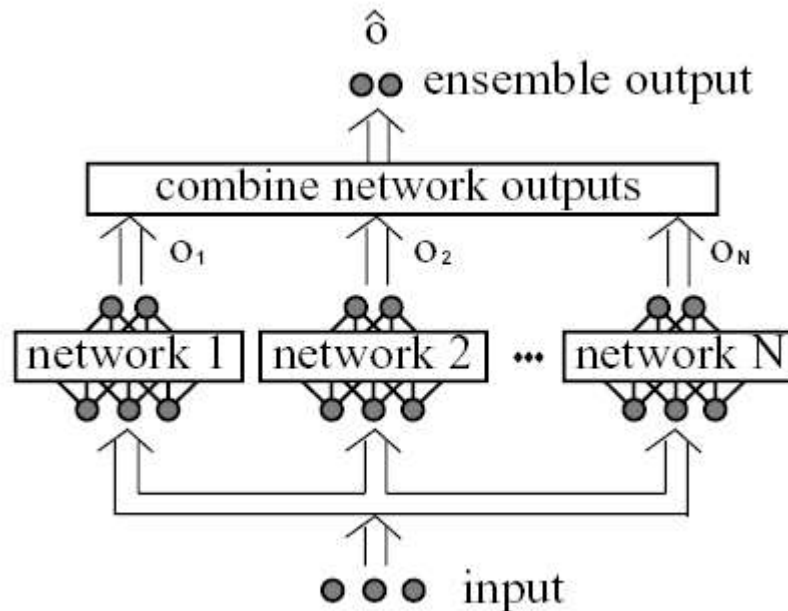
- anglojęzyczne odpowiedniki: ensemble, committee
- jest czymś w rodzaju głosowania ekspertów
- daje możliwość optymalnego połączenia wyników otrzymanych przez różne modele
- każdy taki model jest owym ekspertem, czyli jest wyspecjalizowany w jakimś kierunku – jest to wymagane, aby komitet wносił coś nowego

# Zalety i wady

- ogólniejsze (w sensie mniej stronnicze) rozwiązywanie problemów – zmniejszenie błędu wynikającego z dopasowania do danych
- polepszenie i stabilizacja otrzymywanych wyników
- podział na moduły – ważne przy złożonych problemach
- zmniejszenie złożoności obliczeniowej
- niejasne powody podejmowanych decyzji (mamy do czynienia z różnymi modelami)

# Ogólna architektura

Przykład ilustruje schemat komitetu klasyfikującego, do którego należy  $N$  sieci neuronowych. Wyjście komitetu ( $\hat{o}$ ) jest tworzone na podstawie wyjść ( $o_i$ ) poszczególnych sieci.



# Modele w komitetach

- Komitety homogeniczne – składają się z jednego typu modeli np. drzew decyzji – różnice takich modeli uzyskujemy poprzez trenowanie ich na różnych podzbiorach danych treningowych (cross-walidacja) lub z różnymi parametrami
- Komitety heterogeniczne – zbudowane z różnych typów modeli np. połączenie drzew decyzji i sieci neuronowych w jednym Komitecie

# Głosowania

- Głosowanie proste – większością głosów; wygrywa klasa, którą wybrało większość modeli
- Głosowanie ważone – każdemu modelowi przypisujemy wagę jego głosu, która z biegiem czasu może się zmieniać lub taką, która jest adekwatna do danego zadania
- Głosowanie wybranych – decydują jedynie istotne głosy np. o wysokiej (większej od ustalonej) wartości estymacji (prawdopodobieństwa)

# Metody redukcji błędów

Metody redukujące sprzeczności (wariancje):

- bagging – Breiman (1998)
- boosting – Freund and Schapire (1996)

Metody redukujące dopasowywanie do danych:

- stacking – Wolpert (1992)

# Bagging

- metodą bootstrap generujemy m różnych zbiorów (próbek) wybierając k pozycji z k-elementowego zbioru danych treningowych z zastępowaniem – część pozycji może pojawić się więcej niż raz, innych może nie być w ogóle
- trenujemy każdy model na innej takiej próbce zbioru treningowego (imitacja uczenia na różnych zbiorach treningowych)
- wyjście komitetu otrzymujemy przez proste uśrednienie lub głosowanie



# Boosting

- podzbiory treningowe oraz klasyfikatory tworzone są sekwencyjnie w przeciwieństwie do baggingu, w którym tworzone są one losowo i niezależnie (równolegle) już od pierwszego kroku algorytmu
- przechowywane są wagi dla każdej instancji ze zbioru danych treningowych – im wyższa waga, tym mocniej instancja wpływa na uczenie klasyfikatora (początkowo wagi inicjujemy równymi wartościami)

# Boosting c.d.

- w każdym kroku zmieniane są wagi klasyfikatorów w taki sposób, iż niepoprawnie sklasyfikowane obiekty otrzymują większe wagi w nowym, zmodyfikowanym zbiorze treningowym
- końcowa decyzja jest ustalana w wyniku głosowania, w którym głos każdego klasyfikatora jest funkcją jego dokładności (precyzji)
- istnieją różne odmiany boostingu – najbardziej popularną jest AdaBoost.M1

# AdaBoost.M1

Niech  $\omega_x^t$  oznacza wagę instancji  $x$  w próbie o numerze  $t$

- zainicjuj wszystkie wagi  $\omega_x^1$  na  $1/n$ , gdzie  $n$  jest ilością wektorów treningowych (instancji)
- dla prób o numerze  $t = 1, 2, \dots, T$  wykonaj:
  - utwórz klasyfikator  $C^t$  z danych instancji przy rozkładzie wag  $\omega^t$
  - oblicz błąd  $\varepsilon^t$  tego klasyfikatora, jako sumę wag niepoprawnie sklasyfikowanych instancji
  - jeśli  $\varepsilon^t > 0.5$ , to przerwij procedurę oraz zamień  $T$  na  $t - 1$
  - jeśli  $\varepsilon^t = 0$ , to przerwij procedurę oraz przyjmij  $T = t$
  - w pozostałych przypadkach utwórz wektor wag  $\omega^{t+1}$  dla kolejnej próby poprzez przemnożenie wag instancji, które  $C^t$  poprawnie sklasyfikował przez czynnik  $\beta^t = \varepsilon^t / (1 - \varepsilon^t)$  a następnie ich normalizację
- końcowy wynik jest sumą głosów klasyfikatorów  $C^1, \dots, C^T$ , przy czym każdy głos ma wartość  $\log(1/\beta^t)$

# Bagging i boosting

- bagging wymaga aby system uczący nie był trwały (stały) – każda mała zmiana na zbiorze treningowym powinna prowadzić do utworzenia nowego klasyfikatora
- boosting nie chroni przed używaniem systemów uczących, które słabo przewidują (ich błąd na danym przedziale może utrzymywać się poniżej 50%)
- 10 iteracji (10 klasyfikatorów) może zmniejszyć poziom błędu nawet o 10% do 19% (Quinlan, 1996)
- obie metody polepszają dokładność algorytmów, które są niestabilne (Breiman, 1998)

# AdaBoost vs bagging

Wyniki eksperymentu przeprowadzonego przez Breimana.

Komitet złożony z drzew decyzji typu CART.

Podane wartości to błędy (w procentach) na danym zbiorze.

Data Set	AdaBoost	Bagging
heart	1.1	2.8
breast cancer	3.2	3.7
ionosphere	6.4	7.9
diabetes	26.6	23.9
glass	22.0	23.2
soybean	5.8	6.8
letters	3.4	6.4
satellite	8.8	10.3
shuttle	0.007	0.014
DNA	4.2	5.0
digit	6.2	10.5

*Source:* Breiman, L., Combining predictors, in *Combining Artificial Neural Nets*, Sharkey, A.J.C., Ed., Springer-Verlag, New York, NY, 1999.

# Stacking

- technika wykorzystująca wielopoziomowe uczenie zamiast głosowania
- wejściowy zbiór danych tworzy poziom zerowy, na którym uczą się wszystkie bazowe klasyfikatory
- na dane poziomu pierwszego składają się wyjścia bazowych klasyfikatorów
- w kolejnym etapie jako wejście traktujemy dane pierwszego poziomu, a jako wyjście końcową klasyfikację
- i podobnie dla kolejnych poziomów...

# Stacking c.d.

- stacking jest bardziej wyrafinowaną metodą cross-walidacji, która może zmniejszyć błąd spowodowany dopasowaniem do danych
- zalecane jest używanie jak najmniej podobnych typów klasyfikatorów, w celu zmniejszenia ilości błędów korelacji danych
- okoliczności, w jakich stacking „działa” są ciągle nieznane (Wolpert, 1992)

**Koniec**