Sztuczna Inteligencja

Soma Dutta

Wydział Matematyki i Informatyki, UWM w Olsztynie
soma.dutta@matman.uwm.edu.pl

Wykład - 13: Uczenie się: poszukiwanie najlepszej hipotezy

Semestr letni 2022

Podsumowanie ostatniego wykładu

- Uczenie się możliwej funkcji decyzyjnej na podstawie przykładów treningowych
  - za pomocą sieci neuronowej
  - za pomocą drzew decyzyjnych
- W przypadku prostej sieci neuronowej zakłada się, że mamy informacje o przykładach treningowych w postaci par danych wejściowych i wyjściowych, a zadaniem jest znalezienie wektora wag dla zmiennych wejściowych, aby można było określić funkcję decyzyjną na podstawie ważonej sumy zmiennych wejściowych.
- W przypadku drzewa decyzyjnego przykłady treningowe podane są jako wektory wartości atrybutów i odpowiadających im wartości decyzji. Zadanie polega na znalezieniu drzewa decyzyjnego, które zawiera możliwie najmniejszą liczbę atrybutów i jest spójne (niesprzeczne) wszystkimi przykładami.

Entropia

- Algorytm uczenia się drzewa decyzyjnego polega na wybraniu ważnego podzbioru atrybutów w stosunku do podanego zbioru przykładów.
- Potrzebna jest więc formalna miara ‘dość dobrych’ i ‘bezużytecznych’ atrybutów.
- Tutaj wprowadzono funkcję o nazwie ‘Ważność’ (Importance). W tym celu stosuje się pojęcie wzmocnienia informacji (information gain), które definiuje się przez entropią.
- Warto wiedzieć, że entropia jest podstawową wielkością w teorii informacji (Shanon and Weaver 1949)

Entropia

- Entropia jest miarą niepewności zmiennej losowej; pozyskiwanie informacji odpowiada zmniejszeniu entropii.
- Ogólnie entropia zmiennej losowej V o wartościach v_k, każda z prawdopodobieństwem $p(v_k)$, jest zdefiniowana jako:

  Entropy: $H(V) = \Sigma_k P(v_k) log_2 \frac{1}{P(v_k)} = -\Sigma_k P(v_k) log_2 P(v_k)$.
- Shanon zdefiniował pojęcie zawartości informacji w zmiennej w odniesieniu do jego prawdopodobieśstwa p, oznaczonego jako $I(p)$, przez pewne aksjomaty, a później odkrył, że $I(p) = log_2(\frac{1}{p})$ jest dobrym kandydatem na funkcję I. Relacja między entropią a treścią informacyjną jest podana jako $H(V) = E(I(prob(V)))$.
- Więc entropia rzutu monetą:
  $H(fair)$ = - $(0.5 log_2(0.5) + 0.5 log_2(0.5)) = 1$
- Entropia boolowskiej zmiennej losowej z prawdopodobieństwem q dla 'true' wynosi:
  $B(q) = -(q log_2 q + (1-q) log_2(1-q))$

Entropia atrybutu

- Jeśli zbiór treningowy zawiera p przykładów pozytywnych i n przykładów negatywnych, wówczas entropia atrybutu decyzyjnego dla całego zbioru to: $H(decision) = B(\frac{p}{p+n})$.
- W przykładzie związanym z restauracją $p = n = 6$. Więc odpowiednia entropia jest $B(0,5)$, która jest dokładnie 1 bit.
- Atrybut A z odrębnymi wartościami dzieli zbiór treningowy na podzbiory $E_1, E_2, \ldots E_d$. Każdy podzbiór E_k ma p_k pozytywne przykłady i n_k negatywne przykłady. Tak więc, jeśli pójdziemy wzdłuż gałęzi, potrzebujemy dodatkowych $B(\frac{p_k}{p_k+n_k})$ bitów informacji, aby odpowiedzieć na pytanie.
- Losowo wybrany przykład z zbioru treningowego ma k-tą wartość dla A z prawdopodobieństwem $\frac{p_k+n_k}{p+n}$.
- Tak więc oczekiwana entropia pozostała po przetestowaniu atrybutu A wynosi: $Remainder(A) = \Sigma_{k=1}^{d} \frac{p_k+n_k}{p+n} . B(\frac{p_k}{p_k+n_k})$
- Uzyskanie informacji (information gain) z testu atrybutu A to oczekiwane zmniejszenie entropii: $Gain(A) = B(\frac{p}{p+n}) - Remainder(A)$

Zysk informacji definiujący funkcję IMPORTANCE

Example	Alt	Bar	Fri	Hun	Pat	Price	Rain	Res	Type	Est	Decision WillWait
x_1	Yes	No	No	Yes	Some	$$$	No	Yes	French	0-10	Yes
x_2	Yes	No	No	Yes	Full	$	No	No	Thai	30-60	No
x_3	No	Yes	No	No	Some	$	No	No	Burger	0-10	Yes
x_4	Yes	No	Yes	Yes	Full	$	Yes	No	Thai	10-30	Yes
x_5	Yes	No	Yes	No	Full	$$$	No	Yes	French	> 60	No
x_6	No	Yes	No	Yes	Some	$$	Yes	Yes	Italian	0-10	Yes
x_7	No	Yes	No	No	None	$	Yes	No	Burger	0-10	No
x_8	No	No	No	Yes	Some	$$	Yes	Yes	Thai	0-10	Yes
x_9	No	Yes	Yes	No	Full	$	Yes	No	Burger	> 60	No
x_{10}	Yes	Yes	Yes	Yes	Full	$$$	No	Yes	Italian	10-30	No
x_{11}	No	No	No	No	None	$	No	No	Thai	0-10	No
x_{12}	Yes	Yes	Yes	Yes	Full	$	No	No	Burger	30-60	Yes

- $Gain(A)$ jest tym, czego potrzebujemy do realizacji funkcji IMPORTANCE.
- Na przykład:
  $Gain(Patron) = B(\frac{6}{12}) - Remainder(Patron)$
  $= 1 - [\frac{2}{12}B(\frac{0}{2}) + \frac{4}{12}B(\frac{4}{4}) + \frac{6}{12}B(\frac{2}{6})] \approx 0.541$ bits
  $Gain(Type) = 1 - [\frac{2}{12}B(\frac{1}{2}) + \frac{2}{12}B(\frac{1}{2}) + \frac{4}{12}B(\frac{2}{4}) + \frac{4}{12}B(\frac{2}{4})] = 0$ bit
- To potwierdza, że Patron jest lepszym atrybutem do podziału.
- W rzeczywistości w podanym przykładzie 'Patron' ma maksymalne wzmocnienie dowolnego z atrybutów i będzie wybierany przez algorytm uczenia decyzji jako węzeł początkowy.

Ograniczenia

- Brakujące dane: Często w rzeczywistych problemach nie wszystkie wartości atrybutów są znane dla każdego przykładu. Wartości mogły nie zostać zapisane lub koszt ich uzyskania mógł być zbyt duży.
  - Jeden z problemów, który się tu pojawia przy pełnym drzewie decyzyjnym polega na tym, że jak sklasyfikować obiekt, w którym jest brak pewnych wartości atrybutów.
  - Drugi problem polega na tym, jak zmodyfikować definicję wzmocnienia informacji (information gain), gdy w pewnym przykładzie brakuje wartości dla jakiegoś atrybutu.
- Atrybut wielowartościowy: Gdy atrybut ma wiele możliwych wartości, informacja uzyskana za pomocą miary wzmocnienia daje niewłaściwe wskazanie przydatności atrybutu. Na przykład, jeśli atrybutem jest 'Exact-time', wówczas każdy przykład może mieć inną wartość tego atrybutu, a zatem dla każdej gałęzi atrybut ma jeden przykład sklasyfikowany, w wyniku czego atrybut miałby największe znaczenie. Ale wybranie tego podziału jako pierwszego raczej nie da najlepszego drzewa. Jednym z rozwiązań jest zastosowanie współczynnika wzmocnienia (gain ratio).

- Atrybuty wejściowe ciągłe i wartości całkowite: Atrybuty ciągłe lub o wartościach całkowitych, takie jak wzrost i waga, mają nieskończony zbiór możliwych wartości. Zamiast generowania nieskończenie wielu gałęzi, algorytmy uczenia drzewa decyzyjnego muszą znajdować punkt podziału, który daje najwyższy przyrost informacji. Na przykład w danym węźle w drzewo, może być tak, że testowanie dla atrybutu waga warunek > 160 daje najwięcej informacji. Algorytm musi zacząć od posortowania wartości atrybutu, a następnie rozważyć tylko punkty podziału między dwoma przykładami w utworzonych porządkach, które mają różne klasyfikacje, jednocześnie śledząc bieżące sumy pozytywnych i negatywnych przykładów po każdej stronie punktu podziału.
- Ponadto, jeśli próbujemy przewidzieć numeryczną wartość wyjścia, na przykład cenę mieszkania, to potrzebne jest drzewo regresji zamiast drzewa decyzyjnego. Drzewo regresji ma przy każdym liściu funkcję liniową jakiegoś podzbioru atrybutów numerycznych, a nie pojedynczą wartość.

Ocena i poszukiwanie najlepszej hipotezy

- Celem uczenia jest wyznaczenie hipotezy, która najlepiej pasuje do przyszłych danych. W związku z tym musimy najpierw zrozumieć wyrażenia 'przyszłe dane' i 'najlepsze dopasowanie'.
- Założenie stacjonarności (Stationarity assumption): Zakładamy, że istnieje rozkład prawdopodobieństwa na przykładach, który pozostaje niezmienny w czasie. Każde dane przykładowe (zanim je zobaczymy) generowane są za pomocą zmiennej losowej E_j, której próbkowanie jest obserwowane za pomocą wartości $e_j = (x_j, y_j)$ z rozkładu tej zmiennej i
  - jest on niezależny od poprzednich przykładów $P(E_j|E_{j-1}, E_{j-2}\ldots) = P(E_j)$
  - a każdy przykład ma identyczny wcześniejszy rozkład prawdopodobieństwa $P(E_j) = P(E_{j-1}) = P(E_{j-2}) = \ldots$
- Przykłady spełniające te założenia są nazywane niezależnymi i z identycznym rozkładem (independent and identically distributed) lub i.i.d.
- To łączy przeszłość z przyszłością. W rzeczywistości z czasem mogą nastąpić powolne zmiany w rozkładzie, co należy wziąć pod uwagę przy bardziej zaawansowanym podejściu do uczenia się.

Aby zdefiniować 'najlepsze dopasowanie'

- Definiujemy poziom błędu hipotezy h jako odsetek popełnianych błędów - odsetek przypadków, dla których $h(x) \neq y$ dla przykładu (x, y).
- Jednak to, że hipoteza h ma niski poziom błędu w zbiorze treningowym nie oznacza tego, że dobrze dokonuje uogólnienia decyzji dla każdego nowego przykladu testowego. Wobec tego, aby uzyskać dokładną ocenę hipotezy, musimy ją przetestować na zbiorze przykładów testowych, dotąd nieznanych.
- Holdout Cross Validation: Najłatwiejszym sposobem jest losowe podzielenie dostępnych danych na zbiór treningowy, na podstawie których algorytm uczenia tworzy h, a pozostałą część jako zbiór testowy, na podstawie którego ocenia się dokładność h. Ta metoda czasami nazywana jest zawieszoną walidacją krzyżową.
- Wadą tej metody jest to, że nie używa wszystkich dostępnych danych; jeśli wykorzystamy połowę danych dla zbioru treningowego, wówczas możemy przyjąć złą hipotezę. Z drugiej strony, jeśli zarezerwujemy tylko 10% danych dla zbioru testowego, to wtedy możemy przypadkowo uzyskać statystycznie niską ocenę rzeczywistej precyzji.

Leave One Out Cross Validation (LOOCV)

- k-fold Cross Validation (k-krotna walidacja krzyżowa): Najpierw podzielimy dane na k równolicznych podzbiorów. Następnie wykonujemy k rund uczenia się; w każdej rundzie jedna częśc danych jest wybierana jako zbiór testowy a pozostałe przykłady są wykorzystywane jako dane treningowe. Średni wynik zbioru testów dla rund powinien dać lepsze oszacowanie błędu niż pojedynczy wynik.
- Leave One Out Cross Validation (walidacja krzyżowa typu minus jeden element): Gdy bierzemy pod uwagę k jako całkowitą liczbę dostępnych przykładów, nazywa się to krzyżową walidacją typu minus jeden element (LOOCV).

Wybór modelu

- Zadanie znalezienia najlepszej hipotezy ma dwa podzadania: wybór przestrzeni modelu definiowany przez przestrzeń hipotez, a następnie optymalizacja znajduje najlepszą hipotezę w tej przestrzeni.
- Na przykład, jeśli funkcja decyzyjna jest reprezentowana przez wielomian, wówczas zazwyczaj wielomian wyższego stopnia może być lepiej dopasowany do danych; ale gdy stopień jest zbyt wysoki, będą one prowadziły do przeuczenia i będą słabo sprawdzały się na danych testowych. Wybór stopnia wielomianu jest przykładem problemu wyboru przestrzeni modelu.
- Wybór modelu można sparametryzować ze względu na jego rozmiar. W przypadku hipotezy wielomianowej możemy obsłużyć rozmiar poprzez ustawienie stopień wielomianu. W przypadku drzew decyzyjnych rozmiarem może być liczbą węzłów w drzewie. Ogólnie chcemy znaleźć wartość tego parametru, czyli rozmiar, tak, aby najlepiej równoważyła niedopasowanie i nadmierne dopasowanie aby zapewnić najlepszą dokładność na zbiorze testowym.

Od poziomu błędu do straty

- Jest coś więcej niż minimalizowanie poziomu błędu, którym musimy się zająć.
- Rozważmy problem klasyfikowania wiadomości e-mail jako spam lub nie spam. Gorzej jest gdy sklasyfikujemy nie spam jako spam (a zatem potencjalnie pomijamy ważną wiadomość), niż gdy sklasyfikujemy spam jako nie spam. Tak więc klasyfikator z 1% poziomem błędu, w którym sklasyfikowano prawie wszystkie błędy spam jako nie spam byłby lepszy niż klasyfikator z poziomem błędu 0,5%, jeśli większość błędów polega na zaklasyfikowaniu nie spam jako spam.
- Poza poziomem błędu potrzebujemy również koncepcji użyteczności (utility) modelu. Musimy zminimalizować poziom błędu, a jednocześnie zmaksymalizować użyteczność.

Funkcja straty

- W uczeniu maszynowym zwykle użytność wyraża się za pomocą funkcji straty (loss function).
- Loss function: Funkcja straty $L(x, y, \hat{y})$ jest zdefiniowana jako ilość użyteczności utraconej przez przewidywanie $h(x) = \hat{y}$, gdy poprawną odpowiedzią jest $f(x) = y$.
  $L(x, y, \hat{y}) =$
  Utility(result of using y given x) - Utility(result of using $\hat{y}$ given x)
- Jest to najbardziej ogólne sformułowanie funkcji straty. Często używana jest wersja uproszczona, $L(y, \hat{y})$, czyli niezależne od x.
- Na przykład, chcemy takiej funkcji straty, że $L(spam, non - spam) = 1$ ale $L(non - spam, spam) = 10$.

Przykłady funkcji strat

- Absolute value loss (Bezwzględna strata wartości):
  $L_1(y, \hat{y}) = |y - \hat{y}|$.
- Squared error loss (Kwadratowa funkcja straty):
  $L_2(y, \hat{y}) = (y - \hat{y})^2$
- 0/1 loss (0/1 strata):
  $L_{0/1}(y, \hat{y}) = 0$, jeśli $y = \hat{y}$
  $= 1$, inaczej
  Funkcja straty $L_{0/1}$ ma stratę 1 dla niepoprawnej odpowiedzi i jest odpowiednia dla wyników o wartościach dyskretnych.

Uogólnienie oczekiwanych strat

- Uczący się może teoretycznie zmaksymalizować swoją oczekiwaną użyteczność, wybierając hipotezę istotną, która minimalizuje oczekiwaną stratę na wszystkich parach wejścia-wyjścia, które zobaczy.
- W tym celu musimy wziąć pod uwagę wcześniejszy rozkład prawdopodobieństwa na przykładach, to znaczy $P(x, y)$ dla par wejściowych i wyjściowych.
- Niech E będzie zbiorem wszystkich możliwych przykładów wejścia-wyjścia. Zatem oczekiwana uogólniona strata dla hipotezy h (względem funkcji straty L) wynosi $GenLoss_L(h) = \Sigma_{(x,y)\in E} L(y, h(x))P(x, y)$
- Najlepsza hipoteza h^* to ta o minimalnej oczekiwanej stracie uogólniającej $h^* = argmin_{h\in\mathcal{H}} GenLoss_L(h)$

Strata empiryczna

- Często nie znamy rozkładu prawdopodobieństwa par danych wejściowych i wyjściowych w przykładach. Agent uczący się może oszacować tylko utratę uogólnienia na podstawie straty empirycznej na zbiorze N przykładów treningowych. Ten zbiór oznaczamy przez E.
- Empirical loss: $EmpLoss_{L,E}(h) = \frac{1}{N}\Sigma_{(x,y)\in E} L(y, h(x))$
- Oszacowana najlepsza hipoteza $\hat{h}^*$ jest wtedy tą o minimalnej stracie empirycznej $\hat{h}^* = argmin_{h\in\mathcal{H}} EmpLoss_{L,E}(h)$

Empiryczna miara jakości hipotezy

Inny sposób oceny funkcji hipotezy.

- Dane dzielone są na zbiór treningowy U_{trn} i zbiór testowy U_{tst}.
- Hipoteza $h : X \mapsto V_{dec}$ jest indukowana na podstawie zbioru treningowego U_{trn}.
- Skuteczność hipotezy $Accuracy(h)$ jest mierzona proporcją poprawnie sklasyfikowanych obiektów ze zbioru testowego.

$$Accuracy(h) = \frac{|\{x \in U_{tst} : h(x) = dec(x)\}|}{|U_{tst}|}$$

Metody przypisywania decyzji dla przykładów testowych

- Teraz pojawia się pytanie, jak przypisać wartości wyjściowe lub wartości decyzji do przykładów testowych na podstawie przykładów treningowych.
- Nearest neighbour model (Model najbliższych sąsiadów):
  - Biorąc pod uwagę przykładowy obiekt testowy x_q, ten model wybiera k przykładów ze zbioru treningowego które są najbliższe x_q. Nazywamy go k-NN modelem. Ten zbiór k najbliższych sąsiadów x_q jest oznaczany jako $NN(k, x_q)$.
  - Aby sklasyfikować x_q, najpierw znajdujemy $NN(k, x_q)$, a następnie liczymy najpopularniejsze decyzje wśród sąsiadów (co stanowi większość głosów w przypadku klasyfikacji binarnej).
  - Aby uniknąć remisu, zwykle k jest wybrana jako liczba nieparzysta. Możemy przyjąć średnią lub medianę wartości k sąsiadów.

Potrzeba pomiaru odległości

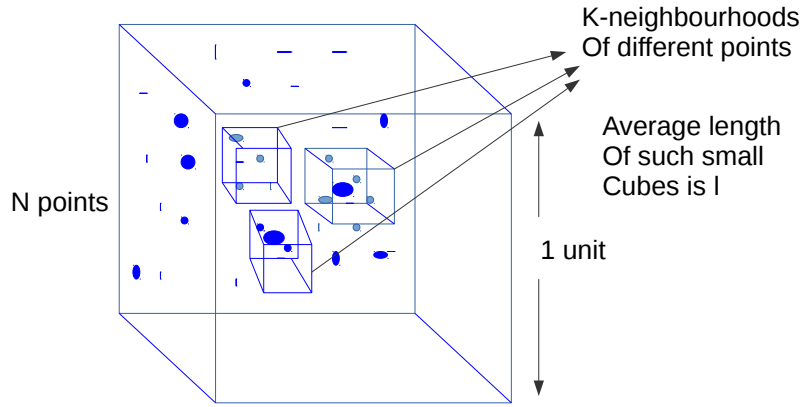
- Słowo 'najbliższy' oznacza potrzebę pomiaru odległości.
- Odległość między przykładem treningowym x_j $(= \langle x_{j1}, x_{j2}, \ldots x_{jm} \rangle)$ a przykładem testowym x_q $(= \langle x_{q1}, x_{q2}, \ldots x_{qm} \rangle)$
  - Euclidean distance: $D(x_j, x_q) = \sqrt{\Sigma_i |x_{ji} - x_{qi}|^2}$ gdzie x_{ji} reprezentuje i-tą wartość w wektorze wartości reprezentujących wejście x_j.
  - Manhattan distance: $D(x_j, x_q) = \Sigma_i |x_{ji} - x_{qi}|$ gdzie x_{ji} reprezentuje i-tą wartość w wektorze wartości reprezentujących wejście x_j.
  - Hamming distance: Z boolowskimi wartościami atrybutów; liczbę atrybutów, na których x_j i x_q róznią się, nazywa się odległością Hamminga.
- Często odległość euklidesowa jest stosowana, jeśli w wymiarach mierzone są podobne właściwości, takie jak jako szerokość, wysokość i głębokość pudełka, a odległość Manhattan jest używana, jeśli różnią się między sobą, np. są wiekiem, wagą i płcią pacjenta.

Normalizacja

- Jeśli korzystamy bezpośrednio z liczb w każdym wymiarze w wektora wartości, wtedy zmiana skali w dowolnym wymiarze może wpływać na całkowitą odległość.
- Oznacza to, że jeśli zmienimy skalę dla wymiaru x_j w i-tym miejscu z centymetrów na mile, zachowując te same skale dla innych atrybutów, możemy uzyskać różne zbiory najbliższych sąsiadów.
- Aby temu zapobiec, musimy zastosować normalizację pomiarów w każdym wymiarze.
- Normalization: Prostym podejściem jest obliczenie średniej μ_i i odchylenia standardowego σ_i wartości w każdym i-tym wymiar x_j i przeskalowanie ich, aby x_{ji} stało się $\frac{(x_{ji}-\mu_i)}{\sigma_i}$.

Wady: przekleństwo wymiarowości

- W mało wymiarowych przestrzeniach z dużą ilością danych metoda najbliższych sąsiadów działają bardzo dobrze. Jednak jako wymiary rosną, napotykamy problem.



- Rozważmy k-najbliższych sąsiadów na zbiorze danych N obiektów równomiernie rozmieszczonych we wnętrzu hipersześcianu n-wymiarowego.
- Definiujemy k-sąsiedztwo obiektu jako najmniejszy hipersześcian zawierający k-najbliższych sąsiadów.
- Niech l będzie średnią długością boku hipersześcianek reprezentujących sąsiedztwa punktów. Następnie objętość sasiedztwa (która zawiera k punktów) to l^n, a dla pełnej kostki objętość wynosi 1. A więc średnio $l^n = \frac{k}{N}$; tzn. $l = (\frac{k}{N})^{\frac{1}{n}}$
- Niech $k = 10$, $N = 1{,}000{,}000$. W dwóch wymiarach (n = 2; jednostka kwadrat), średnie sąsiedztwo ma $l = 0{,}003$, niewielki ułamek kwadratu jednostkowego, i w 3 wymiarach to zaledwie 2% długości krawędzi kostki jednostkowej. Ale zanim dotrzemy do wymiaru 17, jest połówką długości krawędzi hipersześcianu jednostkego, a przy wymiarze 200 jest to 94%.

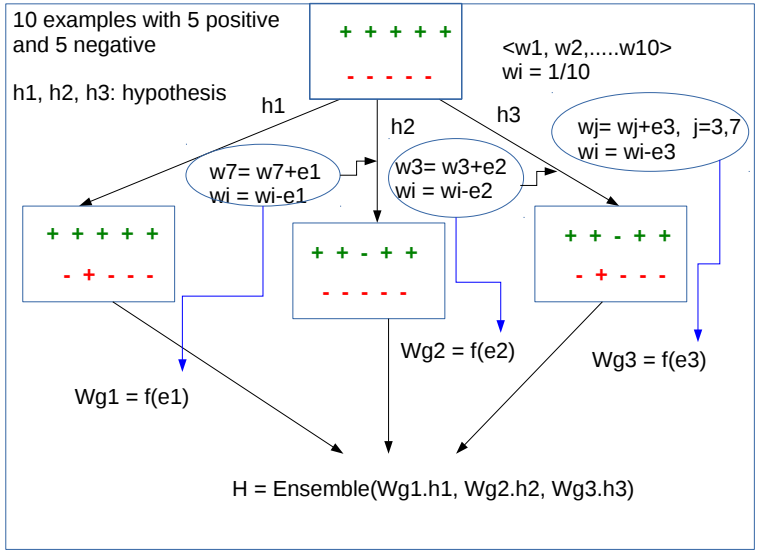
Ensemble learning

- Do tej pory analizowaliśmy metody uczenia się, w których wybierana jedna hipoteza z przestrzeni hipotez, służy do przewidywania. Ideą metod uczenia zespołowego jest wybiór zbioru lub zestawu hipotez z przestrzeni hipotez i agregowanie ich prognoz.
- Motywacja do nauki zespołowej jest prosta. Rozważmy zespół 5 hipotez i przypuśćmy, że łączymy ich przewidywania zwykłą większością głosów. Dla tego zespołu, aby błędnie sklasyfikować nowy przykład, co najmniej trzy z pięciu hipotez muszą błędnie sklasyfikować ten przykład. Jest to o wiele mniej prawdopodobne niż błędna klasyfikacja według jednej hipotezy.
- Zakładamy, że każda hipoteza h_k w zespole ma błąd p - to znaczy prawdopodobieństwo, że losowo wybrany przykład jest błędnie sklasyfikowany przez h_k wynosi p. Ponadto zakładamy, że błędy popełniane przez każdą hipotezę są niezależne. W takim przypadku, jeśli p jest małe, wówczas prawdopodobieństwo wystąpienia dużej liczby błędnych klasyfikacji jest niewielkie agregując prawdopodobieństwa błędów wszystkich hipotez.

Wzmocnienie jest najczęściej stosowaną metodą zespołową

- Weighted training set: Zbiór treningowy, w którym każdy przykład ma przypisaną wagę $w_j \geq 0$. Im wyższa jest waga przykładu, tym wyższe jest znaczenie, jakie przypisuje się mu podczas uczenia się hipotezy.
- Wzmocnienie (boosting) zaczyna się od $w_j = 1$ dla wszystkich przykładów (tj. normalny zbiór treningowy). Poczynając od tego zbioru generowana jest pierwsza hipoteza h_1. Ta hipoteza klasyfikuje niektóre przykłady poprawnie, a niektóre niepoprawnie. Chcielibyśmy, aby kolejna hipoteza była lepsza. Więc dla błędnie sklasyfikowanych przykładów zwiększamy ich wagę, jednocześnie zmniejszamy wagę dla poprawnie sklasyfikowanych przykładów.
- Na podstawie tego nowego ważonego zbioru treningowego generujemy hipotezę h_2. Proces jest kontynuowany, dopóki nie wygenerujemy K hipotez, gdzie K jest wejściem do algorytmu wzmocnienia.
- Ostateczna hipoteza złożona jest kombinacją większości ważonej wszystkich hipotez K, każda ważona zgodnie z tym, jak dobrze spisała się na zbiorze treningowym.

ADA-boosting



- Na przykład: Dane wejściowe: (i) 10 przykłady treningowe, (ii) algorytm uczenia się L (iii) 3 hipotez wygenerowanych przez L
- W hipotezie h_1 siódmy przykład jest błędnie sklasyfikowany, więc błąd jest obliczany na podstawie pewnego wzoru błędu, a waga siǫdmego przykładu jest zwiększana przez dodanie błędu, a waga pozostałych przykładów jest zmniejszana o ten sam błąd.
- Teraz druga hipoteza h_2 została wygenerowana na tym samym zestawie przykładów z tą poprawioną wagą. Proces trwa, aż zostaną wygenerowane wszystkie trzy hipotezy.
- Wreszcie na podstawie błędu generowanego przez każdą hipotezę obliczana jest odpowiednia waga dla każdej hipotezy. Tę wagę można obliczyć za pomocą funkcji zastosowanej do poziomu błędu każdej hipotezy.
- Na koniec algorytm zwraca nową hipotezę utworzoną, agregując trzy hipotezy z ich odpowiednimi wagami.

Dziękuję za uwagę