DNT pratybos 3 paskaita
(gradientinio nusileidimo metodas)

input = 0.5 Norime išmokyti tikslo
goal = 0.8 teisingai prognozuoti
rezultatą goal.

Prognozės algoritmas

predict = input * weight

Prognozės tikslumą vertinsime atsižvelgiant į paklaidą

<u>error</u> = $(\text{predict} - \text{goal})^2$
(python ** 2)

weight = 0.5 (pradinis artinys).

MM atveju atitiktu <u>tiesinės regresijos</u> metodą.

Šiuo metodu surandame

$\boxed{\text{weight} = 1.6}$

Apibendrinimas:
turime poras $(input_j, goal_j)$
$j = 1, \ldots, M.$

$$error = \sum_{j=1}^{M} (input_j \times weight - goal_j)^2$$

Išvedame tiesinės regresijos kreivę geriausiai tinkančią visiems apmokymo duomenims.

DNT apmokymo metu ir ieškome parametro weight geriausios reikšmės.

Bet - naudojame iteracinį algoritmą, kurį galėsime apibendrinti daug sudėtingesnėms DNT architektūroms ir svarbiems taikymams atvejams.

Pirmame pateiktame algoritme (žr. psl. 9) – Paskaita 2

varijuojame svorio weight reikšmę, radę geresnę reikšmę nuspendžiame ar didiname, ar mažiname svorio reikšmę.

- kadangi nežinome, kuria kryptimi keisti parametrą, weight, tai jį didiname ir mažiname (bandymų)
- toks algoritmas smulkiai aprašomas naujas, kai turime daugiau svorio parametrų (o jų gali būti daug!).

Todėl nagrinėsime gradientinio nusileidimo metodą. Jį be paaiškinimo (motyvacijos) pateikti ir mokomosios knygos autorius

Kol kas nagrinėjome vieno kintamojo f-ją $f(x)$

Norime rasti $\min f(x) = f(\bar{x}_0)$

$$\text{arba } \min f(x) = \bar{x}_0.$$

Iš analizės kurso žinome, kad f-jos gradientas (vieno kintamojo atveju $\frac{df}{dx}$ - išvestinės reikšmė taške $\tilde{x}_0$) apibrėžia f-jos greičiausio didėjimo kryptį. Eisime antigradiento kryptimi

$$x_{n+1} = x_n - \gamma \frac{df}{dx}(x_n)$$

γ - parenkame tokį, kad $f(x_{n+1}) < f(x_n)$

$$error(weight) = \sum_{j=1}^{M} (in_j \times weight - g_j)^2$$

$$\boxed{\frac{d\,error}{d\,weight} = 2 \sum_{j=1}^{M} (in_j \times weight - g_j) \times in_j}$$

Parametrą minimizuojame antigradiento kryptimi (automatiškai randame mažėjimo kryptį!)

$$w_{n+1} = w_n - \gamma_n \cdot 2 \sum_{j=1}^{M} (in_j \times w_n - g_j)\, in_j$$

Parametrą γ_n parenkame tokį, kad

$$error(w_{n+1}) < error(w_n)$$

γ_n parinkimo algoritmai gali būti įvairūs, paprasčiausias iš jų patikrinti kelias γ reikšmes (1.5, 1, 0.5, 0.25 ir imti tokią γ reikšmę, kada yra mažiausia paklaida $error(w_{n+1})$.

(Mokymo pavyzdyje $\gamma_n = 1/2$).

Realiuose DNT taikymuose svorių parametrų $w_1, w_2, \dots, w_k$ yra daug ir reikia rasti jų tokias reikšmes, kad DNT prognozių tikslumas būtų kiek galima geresnis

$$E = E(w_1, w_2, \dots, w_k).$$

Tokios funkcijos gradientas

$$\nabla E = \left(\frac{\partial E}{\partial w_1}, \frac{\partial E}{\partial w_2}, \dots, \frac{\partial E}{\partial w_k} \right)$$

Naują tinklo svorių rinkinį galime ieškoti tokiu algoritmu

$$W^{n+1} = W^n - \gamma_n \nabla E(W^n),$$

čia

$$W^n = (w_1^n, w_2^n, \dots, w_K^n).$$