Perceptronul

Printre primele chestii descoperite in AI. Se utilizeaza pentru probleme de clasificare binara in care clasele sunt linear separabile. Ce inseamna clasificare: am niste puncta intr-un spatiu n-dimensional (date prin coordonatele lor x1, x2, …, xn) si trebe sa zic din ce categorie/clasa fac parte. Ce inseamna linear separabil: pot sa trag o dreapta/plan/hiperplan care sa separe complet cele 2 clase, una sa fie "sub" si una "deasupra" liniei. Un perceptron pentru clasificare binara de puncte in 2D are 2 ponderi (w1 si w2) si un bias. Daca ne uitam la formula de calcul a outputului avem: y=w1*x1+w2*x2+bias (inainte de functia de activare sigmoida; sa nu o uitati ca outputul final este de fapt sigma(y)). Acuma daca mergem in ecuatia parametrica a unei drepte in plan, care este a*x+b*y+c=0, facem corespondenta: a=w1, b=w2, c=bias. Observam ca perceptronul pleaca cu o dreapta arbitrara (w1, w2, bias random) si ajusteaza w1, w2, bias astfel incat dreapta aia sa separe daca nu chiar correct, atunci cat mai bine astfel incat sa fie cat mai putine puncte clasificate incorect. W1 si w2 controleaza "panta" dreptei si bias-ul este translatia stanga-dreapta (sau intercept cum se mai cheama). Mare grija ca exista probleme care nu sunt linear separabile, cum ar fi XOR (sau exclusive). Musai sa stiti ca asta e o forma de invatare supervizata, adica bag date, iese output, compar cu ce trebuia sa dea (supervizat) si calculez o eroare ca fiind ce trebuia sa dea minus ce mi-a dat, si cu asta merg inapoi si „pedepsesc" perceptronul.

ANN

Am zis ca perceptronul nu se stie la chestii care nu sunt liniar separabile. Multa vreme AI-ul a stagnat, pana unii s-or gasit sa puna mai multi perceptroni si sa-i aranjeze in straturi. La un ANN basic, neuronii din stratul superior is conectati cu toti neuronii de strat inferior. Zicem superior si inferior pentru ca informatia „curge" de la input, catre hidden si dupa catre output. Asta se cheama in primul rand feedforward (informatia curge liniar de la input la output) si fully connected ca ii fiecare cu fiecare. La fel ca la perceptron, o sa am o eroare la final si pedepsim prin backpropagation. Pana la urma, ANN-ul zice ca bun nu putem gasi o dreapta in 2D, dar ce-ar fi daca am gasi o „dreapta" in ND, care proiectata in 2D s-ar putea sa fie o „stramba" daca e sa ne luam dupa geometria Euclidiana.

Exemplu concret: avem de clasificat poze alb-negru cu cifre de la 0 la 9, cu dimensiunea patrata de 8*8. Cum traducem asta in termeni de input si output. Pai in primul rand, reteaua are un strat de neuroni de input, noi avem o matrice, deci trebe sa-i facem flatten. A doua chestie, reteaua lucra cel mai bine cu valori in [0, 1] – ca asa ii designul la functia de activare (tangenta hiperbolica sau sigmoida clasica). Cum trecem de la [0, 255] la [0,1] (lucram cu pixeli si astia is tip de data byte). Impartim valorile pixelilor la 255. Formula generala este x=(x-xmin)/(xmax-xmin). Bun, suntem gata cu inputul. La hidden puneti ceva gen numar neuroni input + numar neuroni output. La output e ultima schema ca acolo aveti output categoric, adica ar trebui sa fie un numar intreg de la 0 la 9. Codificam cifrele cu one-hot encoding, adica cifra 4 devine vectorul [0, 0, 0, 0, 1, 0, 0, 0, 0, 0]. Cifra 8 devine [0, 0, 0, 0, 0, 0, 0, 0, 1, 0]. Cam asta e. La final nu o sa va iasa direct din retea asa frumos, ci o sa va iasa [0.12, 0.22, 0.37, ..., blablabla]. Predictia este indexul maximului din vector.