Organizační úvod

Domácí úkoly, slidy a spol. budou na webu, šablony na domácí úkoly jsou na GitHubu. Jako komunikační platforma bude využívána Piazza.

Úvod

Definice 0.1 (Klasifikace, regrese)

Klasifikace = Přiřazení jednu z kategorií každému vstupu.

Regrese = Přiřazení nějaké číslo každému vstupu.

Definice 0.2 (Typy učení)

S učitelem = někdo / něco říká, jaká je správná odpověď.

Bez učitele = učení se na datech, kde nejsou žádné odpovědi (např. dělení do skupin).

Zpětnovazebné = nikdo neřekne, zda to děláš správně, ale dozvídáš nějaké výsledky (např. hry, testy, …)

Poznámka
Předpokládejme, že $\mathbf{x} \in \mathbb{R}^D$ je vstup. Dvě základní úlohy jsou pak:

Regrese: Vrátit číslo $t \in \mathbb{R}$.

Klasifikace: Buď vrátit 1 třídu, nebo vrátit pravděpodobnostní distribuci tříd.

Poznámka
Většinou máme trénovací dataset, kde jsou příklady $(\mathbf{x}, t)$ vygenerovány náhodně z data generating distribuce.

Optimalizace se snaží získat co nejlepší výsledek na trénovacích datech, strojové učení na obecných.

Definice 0.3 (Featury)

Featury budeme říkat to, co předhazujeme danému programu (ať je to přímo vstup, nebo už nějak změněný vstup). Česky jsou to „klasifikační rysy“.

Definice 0.4 (Lineární regrese)

Mějme $\mathbf{x} \in \mathbb{R}^D$. Lineární regrese pak vypadá tak, že výstup počítáme jako

$$y(\mathbf{x}, \mathbf{w}, b) = x_1 w_1 + \ldots + x_D w_D + b = \mathbf{x} \cdot \mathbf{w} + b.$$

$\mathbf{w}$ nazýváme váhy a b bias. (Bias se dá často zahrnout do vah tak, že vektory doplníme o b a váhy o 1.)

K určení $\mathbf{w}$ a b se hodí chybová funkce průměrná kvadratická odchylka (dělená 2 místo N). Předpokládejme, že $\mathbf{X} \in \mathbb{R}^{N \times D}$ je matice vstupů a $\mathbf{t} \in \mathbb{R}^N$ je vektor odpovídajících výstupů, pak se dá tato funkce zapsat jako $\frac{1}{2}||\mathbf{Xw} - \mathbf{t}||^2$. Fermatova věta nám říká, že minimum bude nejvýše tam, kde je derivace nulová. To můžeme zobecnit do více rozměrů, tedy máme rovnice

$$\frac{\partial}{\partial w_j} \frac{1}{2} \sum_i^N \left(\mathbf{x}_i^T \mathbf{w} - t_i\right)^2 = \sum_i^N x_{ij} \left(\mathbf{x}_i^T \mathbf{w} - t_i\right) = 0,$$

což lze přepsat na $\mathbf{X}^T (\mathbf{Xw} - \mathbf{t}) = 0$ a to na:

$$\mathbf{X}^t \mathbf{Xw} = \mathbf{X}^T \mathbf{t}.$$

To už lze spočítat (buď metodou nejmenších čtverců, nebo přímo inverzí).

Poznámka
S featurami musíme pracovat opatrně, protože může nastat buď underfitting (model je moc slabí, takže i trénovací chyba je moc velká) nebo overfitting (model je moc naučený na trénovací data a tak už na testovacích dává velkou chybu).

Definice 0.5 (Regularizace)

Chceme zmenšit váhy (protože čekáme, že to zmenší overfitting). To se dělá například tzv. L_2-regularizací, kde upravíme error funkci na

$$\frac{1}{2} \sum_{i=1}^{N} (y(\mathbf{x}_i, \mathbf{w}) - t_i)^2 + \frac{\lambda}{2} ||\mathbf{w}||^2.$$

(y je predikce, t_i ideální odpověď).

Zároveň tato metoda vyřeší, že soustava výše může mít více řešení. (To se vlastně děje proto, že např. jestli se model řídí cm nebo desítkami milimetrů, které jsou obě uvedené ve featurech je jedno).

Poznámka
Abychom určili λ, tak rozdělíme data na trénovací a testovací, jako předtím, a navíc na validační (development). Natrénujeme s různými lambdami na trénovacích datech a pak vybereme nejlepší podle validačních dat.