

Kompletteringar till föreläsning i generalisering, val av modellstruktur

T. Olofsson

1 Uppdelningen av J i $Bias^2$ och $Variance$.

Vi antar följande modell för sambandet mellan x och t :

$$t(x) = h(x) + e \quad (1)$$

där e är en brusterm med väntevärde noll och varians σ^2 och där $h(x)$ är någon, för oss okänd, funktion som vi är intresserade att skatta. Vi vill alltså kunna skapa en funktion som imiterar $h(x)$ baserat på träningsdata $\mathcal{X} = \{x_1, t_1, \dots, x_N, t_N\}$. För detta ändamål ansätter vi en modellfunktion $g(x, \mathbf{w})$ där \mathbf{w} står för en samling modellparametrar. Vi vet att det bästa valet av $g(x, \mathbf{w})$ är $g(x, \mathbf{w}) = E[t|x]$, dvs det betingade väntevärdet av t . Detta är den s.k. *regressionsfunktionen*.

Det vi vill minimera är det totala förväntade kvadratfelet $J = E[(g(x, \hat{\mathbf{w}}) - t)^2]$ där vi ska ta väntevärdet över både t och x . Vi kan göra följande uppdelning av J :

$$J = E[(g(x, \hat{\mathbf{w}}) - t)^2] = E[(g(x, \hat{\mathbf{w}}) - h(x))^2] + \sigma^2 = J_{modell} + \sigma^2, \quad (2)$$

Den sista termen är konstant och därför oberoende av träningsmängder och eventuella inställningar av parametrar. Uppenbarligen sätter den en gräns för hur bra vi kan skatta t , baserat på x . Detta bör stämma med intuitionen eftersom det den bästa skattningen av t vi rimligtvis kan göra är $\hat{t}(x) = h(x)$. I detta fall får vi ett fel som är $\hat{t}(x) - h(x) = e$ och i medeltal får vi förstås kvadratfelet $E[e^2] = \sigma^2$.

Det vi *kan* påverka, och som därför är mer intressant för vår del, är den första termen, J_{modell} . Notera att de skattade parametrarna hela tiden erhålls m.h.a. av träningsdata vilka vi betraktar som stokastiska (vi "drar" våra träningsexempel från en fördelning med PDF $p(t, x)$). Det fel mellan $g(x, \hat{\mathbf{w}})$ och $h(x)$ vi får beror alltså dels på bruset, e , och dels på vilken realisering av träningsmängden vi får.

På liknande sätt som ovan kan vi dela upp den första termen ytterligare.

$$\begin{aligned} J_{modell} &= E[(g(x, \hat{\mathbf{w}}) - h(x))^2] = \{ \text{lägg-till-och-dra-ifrån-tricket} \} = \\ &E[(g(x, \hat{\mathbf{w}}) - E_{\mathcal{X}}[g(x, \hat{\mathbf{w}})] + E_{\mathcal{X}}[g(x, \hat{\mathbf{w}})] - h(x))^2] = \\ &E[(E_{\mathcal{X}}[g(x, \hat{\mathbf{w}})] - g(x, \hat{\mathbf{w}}))^2 + 2(E_{\mathcal{X}}[g(x, \hat{\mathbf{w}})] - g(x, \hat{\mathbf{w}}))(E_{\mathcal{X}}[g(x, \hat{\mathbf{w}})] - h(x)) + (E_{\mathcal{X}}[g(x, \hat{\mathbf{w}})] - h(x))^2] \end{aligned} \quad (3)$$

Väntevärdet $E_{\mathcal{X}}[g(x, \hat{\mathbf{w}})]$ är ett väntevärde m.a.p träningsdatamängden, \mathcal{X} . Vi tänker oss ett scenario där vi upprepade gånger får utföra vår parameterinställning för nya realiseringar av träningsdata. Varje sådan träningsmängd ger upphov till en parametervektorskattning, $\hat{\mathbf{w}}$ och vårt skattade funktionsvärde för ett visst x blir för just denna träningsmängd $g(x, \hat{\mathbf{w}})$. Tar vi medelvärdet över många sådan tänkbara träningsmängder så får vi till slut en medelapproximation $E_{\mathcal{X}}[g(x, \hat{\mathbf{w}})]$.

Bidraget från korstermen, $2(E_{\mathcal{X}}[g(x, \hat{\mathbf{w}})] - g(x, \hat{\mathbf{w}}))(E_{\mathcal{X}}[g(x, \hat{\mathbf{w}})] - h(x))$ visar sig försvinna¹ och kvar blir endast

$$\begin{aligned} J_{modell} &= E[(g(x, \hat{\mathbf{w}}) - h(x))^2] = E[(E_{\mathcal{X}}[g(x, \hat{\mathbf{w}})] - g(x, \hat{\mathbf{w}}))^2] + E[(E_{\mathcal{X}}[g(x, \hat{\mathbf{w}})] - h(x))^2] \\ &= Bias^2 + Variance \end{aligned} \quad (4)$$

Den första termen ska vi tolka som den del av det förväntade totala kvadratfelet som kommer sig av att vi har en modell som inte klarar av att realisera funktionen $h(x)$. Exempelvis kan det bero på att vi försöker

¹Analysen är ganska omständlig och utelämnas därför.

approximera ett polynom, $h(x)$, som i praktiken har ett högt ordningstal m.h.a ett polynom, $g(\cdot)$, med ett lägre ordningstal. Ett annat exempel kan vara att $g(x, \mathbf{w})$ står för en flerlagerperceptron.² Ju färre neuroner desto mindre möjligheter att konstruera komplicerade funktioner.

Den andra termen illustrerar det faktum att även *variationer* kring funktionsmedelvärdet $E_{\mathcal{X}}[g(x, \hat{\mathbf{w}})]$ ger ett bidrag till det totala felet. Alltså, vi kan mycket väl tänka oss att vi har lyckats hitta en modellstruktur för $g(x, \mathbf{w})$ sådan att vi för ett givet antal träningsexempel kan förvänta oss att göra en perfekt skattning av $h(x)$ i *medeltal*, men att variationerna kring detta medelvärde är alltför stora för att vi ska kunna lita på en enstaka skattning.

På samma sätt kan man tänka sig att vi har valt en modellordning (antal neuroner, antal polynomkoefficienter) som vid upprepade träningar med ett givet antal träningsexempel tenderar att ge oss väldigt stabila parametervärden, och därför även låg varians i vår skattning $g(x, \hat{\mathbf{w}})$, dvs *Variance* är liten. Självklart måste även *Bias*² vara låg om det totala felet J ska kunna vara låg. Extremfallet har vi då vi helt bortser ifrån data vid vår skattning av parametrarna och sätter dessa till samma värden hela tiden. Detta innebär att *Variance* = 0!. Vi har dock ingen anledning att tro att detta skulle vara en bra metod. *Bias*² kommer med största sannolikhet att vara mycket stor i detta fall.

2 Regularisering

Det sistnämnda extremfallet som visade hur man enkelt kan få termen *Variance* till att vara noll ger vissa ide'er om hur man kan få ner denna term, fast på ett kanske något mer nyanserat och genomtänkt sätt. Vi har tidigare i kursen tagit upp skillnaden mellan ML-skattningar och MAP-skattningar. En MAP-skattning tar i viss mån hänsyn till information som inte direkt finns i träningssmängden. Exemplet med att sätta alla parametrar till en förutbestämd konstant skulle här alltså vara ett extremfall av MAP-skattning där vi låter vår *a priori*-information dominera totalt över våra observationer. Fördelen med detta angreppssätt var att få ned *Variance*.

Låt oss vara lite mindre hårdnackade och låta träningsdata spela in något mer i vår skattning. Rimligtvis bör vi då kunna minska *Bias*² på viss bekostnad av ökad *Variance*. Helst skulle vi vilja finna den prior som ger den bästa avvägningen mellan *Bias*² och *Variance*. Hur detta går till är definitivt ett svårt problem. Än så länge kanske det kan räcka med att inse att det bör existera en sådan prior.

3 Validering

Vårt problem är att, baserat på våra träningsdata, finna en modellstruktur med goda generaliseringsegenskaper, dvs en modellstruktur som ger ett minimalt värde på $J = E[(g(x, \hat{\mathbf{w}}) - t)^2] = Bias^2 + Variance + \sigma^2$. Om vi kunde, på nåt sätt, estimerar J så vore vi i hamn. Då kunde vi helt sonika beräkna J för våra olika alternativa modellstrukturer (polynom av olika ordningar, MLP:er med olika antal noder etc) och välja den modellordning som ger lägsta värdet på J .

Ett sätt³ att mäta $J = E[(g(x, \hat{\mathbf{w}}) - t)^2] = Bias^2 + Variance + \sigma^2$ vore att generera en mycket stor mängd par av nya datamängder. Varje par innehåller dels en träningsmängd och en s.k. valideringsmängd. Den senare betecknar vi nedan med hjälp av tilde-tecknet.

$$\begin{aligned}
 \mathcal{X}_1 &= \{x_1^1, t_1^1, x_2^1, t_2^1, \dots, x_N^1, t_N^1\} & \text{och} & & \tilde{\mathcal{X}}_1 &= \{\tilde{x}_1^1, \tilde{t}_1^1, \tilde{x}_2^1, \tilde{t}_2^1, \dots, \tilde{x}_N^1, \tilde{t}_N^1\} \\
 \mathcal{X}_2 &= \{x_1^2, t_1^2, x_2^2, t_2^2, \dots, x_N^2, t_N^2\} & \text{och} & & \tilde{\mathcal{X}}_2 &= \{\tilde{x}_1^2, \tilde{t}_1^2, \tilde{x}_2^2, \tilde{t}_2^2, \dots, \tilde{x}_N^2, \tilde{t}_N^2\} \\
 & \vdots & & & & \\
 \mathcal{X}_K &= \{x_1^K, t_1^K, x_2^K, t_2^K, \dots, x_N^K, t_N^K\} & \text{och} & & \tilde{\mathcal{X}}_K &= \{\tilde{x}_1^K, \tilde{t}_1^K, \tilde{x}_2^K, \tilde{t}_2^K, \dots, \tilde{x}_N^K, \tilde{t}_N^K\}
 \end{aligned}
 \tag{5}$$

Varje träningsmängd, \mathcal{X}_i ger en skattning $\hat{\mathbf{w}}_i$. För att skatta J kan vi nu medelvärdesbilda över både $\hat{\mathbf{w}}_i$

² x står som vanligt för insignal och \mathbf{w} representerar vikterna. Antalet vikter beror på vilken struktur vi har valt hos vår MLP. Ju fler neuroner desto fler vikter.

³Som tyvärr i de flesta situationer är orealistisk och t.o.m. meningslöst

och exemplen i $\tilde{\mathcal{X}}_j$:

$$J = \frac{1}{K} \sum_{i=1}^K \frac{1}{N} \sum_{n=1}^N (g(\tilde{x}_n^i, \hat{\mathbf{w}}_i) - \tilde{t}_n^i)^2 \quad (6)$$

Tyvärr är denna teknik meningslös annat än möjligtvis illustrationsändamål (som i labben). Vi kan dock göra en (grov) skattning av J , baserat på endast en datamängd \mathcal{X} . Denna teknik, som med ett samlingsnamn kallas *validering*, kan utföras på nåt av följande sätt:

- “*Hold out*”. Vi avdelar en del av träningsmängden för validering. Dela alltså upp \mathcal{X} i två delar: \mathcal{X}_{train} och \mathcal{X}_{val} . Skatta \mathbf{w} baserat på exempel i \mathcal{X}_{train} och estimerar J mha \mathcal{X}_{val} via:

$$\hat{J} = \frac{1}{N_{val}} \sum_{x_n \in \mathcal{X}_{val}} (g(x_n, \hat{\mathbf{w}}) - t_n)^2 \quad (7)$$

- *Korsvalidering*. Dela in \mathcal{X} i S st (icke överlappande) delmängder $\mathcal{X}_1, \dots, \mathcal{X}_S$. Låt $\mathcal{X}_{j'} = \{\mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_{j-1}, \mathcal{X}_{j+1}, \dots, \mathcal{X}_S\}$, dvs den datamängd vi får genom att ta bort \mathcal{X}_j från \mathcal{X} . Beräkna $\hat{\mathbf{w}}_j$ med hjälp av $\mathcal{X}_{j'}$. Beräkna

$$J_j = \frac{1}{N_j} \sum_{x_n \in \mathcal{X}_{j'}} (g(x_n, \hat{\mathbf{w}}_j) - t_n)^2 \quad (8)$$

och skatta till slut J med

$$\hat{J} = \frac{1}{S} \sum_{j=1}^S J_j \quad (9)$$

Fördelen med den sista metoden är att vi kan få ner variansen i skattningen av J och därigenom få ett bättre beslutsunderlag vid val av optimal modellstruktur. Anta, rent hypotetiskt, att J_j och J_i för $i \neq j$ vore statistiskt oberoende⁴. Då skulle gälla att variansen $var(\hat{J}) = var(J_j)/S$. Vi skulle alltså kunna få en stabilare skattning av J genom att “tröska igenom” data på detta sätt. Specialfallet då $S = N$ där N är antalet träningsdata i \mathcal{X} kallas för “leave-one-out”⁵. Den uppenbara nackdelen är förstås att antalet beräkningar ökar. I stället för en skattning av \mathbf{w} , beräknar vi S st skattningar.

⁴Detta kan de rimligtvis inte vara eftersom de har räknats fram till stor del baserat på samma data.

⁵Går även under namnet “Jack knife estimation”