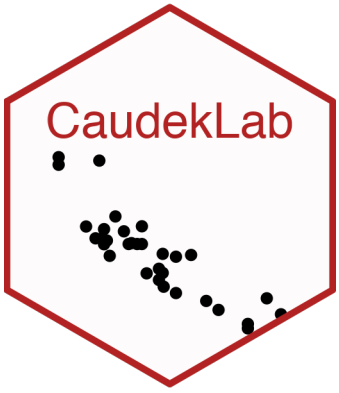
Psicometria

Corrado Caudek

2022-01-13

Questo documento è stato realizzato con:

- LaTeX e la classe memoir (http://www.ctan.org/pkg/memoir);
- R (http://www.r-project.org/) e RStudio (http://www.rstudio.com/);
- bookdown (http://bookdown.org/) e memoiR (https://ericmarcon.github.io/memoiR/).



Nel blog della mia pagina personale sono forniti alcuni approfondimenti degli argomenti qui trattati. https://ccaudek.github.io/caudeklab/

Indice

Data della versione presente: Gennaio 13, 2022.

Prefazione

Data Science per psicologi contiene il materiale delle lezioni dell'insegnamento di *Psicometria B000286* (A.A. 2021/2022) rivolto agli studenti del primo anno del Corso di Laurea in Scienze e Tecniche Psicologiche dell'Università degli Studi di Firenze. *Psicometria* si propone di fornire agli studenti un'introduzione all'analisi dei dati in psicologia. Le conoscenze/competenze che verranno sviluppate in questo insegnamento sono quelle della Data science, ovvero un insieme di conoscenze/competenze che si pongono all'intersezione tra statistica (ovvero, richiedono la capacità di comprendere teoremi statistici) e informatica (ovvero, richiedono la capacità di sapere utilizzare un software).

La psicologia e la Data science

> *It's worth noting, before getting started, that this material is hard. If you find yourself confused at any point, you are normal. Any sense of confusion you feel is just your brain correctly calibrating to the subject matter. Over time, confusion is replaced by comprehension […]*
>
> — Richard McElreath

Sembra sensato spendere due parole su un tema che è importante per gli studenti: quello indicato dal titolo di questo Capitolo. È ovvio che agli studenti di psicologia la statistica non piace. Se piacesse, forse studierebbero Data science e non psicologia; ma non lo fanno. Di conseguenza, gli studenti di psicologia si chiedono: "perché dobbiamo perdere tanto tempo a studiare queste cose quando in realtà quello che ci interessa è tutt'altro?" Questa è una bella domanda.

C'è una ragione molto semplice che dovrebbe farci capire perché la Data science è così importante per la psicologia. Infatti, a ben pensarci, la psicologia è una disciplina intrinsecamente statistica, se per statistica intendiamo quella disciplina che studia la variazione delle caratteristiche degli individui nella popolazione. La psicologia studia *gli individui* ed è proprio la variabilità inter- e intra-individuale ciò che vogliamo descrivere e, in certi casi, predire. In questo senso, la psicologia è molto diversa dall'ingegneria, per esempio. Le proprietà di un determinato ponte sotto certe condizioni, ad esempio, sono molto simili a quelle di un altro ponte, sotto le medesime condizioni. Quindi, per un ingegnere la statistica è poco importante: le proprietà dei materiali sono unicamente dipendenti dalla loro composizione e restano costanti. Ma lo stesso non può dirsi degli individui: ogni individuo è unico e cambia nel tempo. E le variazioni tra gli individui, e di un individuo nel tempo, sono l'oggetto di studio proprio della psicologia: è dunque chiaro che i problemi che la psicologia si pone sono molto diversi da quelli affrontati, per esempio, dagli ingegneri. Questa è la ragione per cui abbiamo tanto bisogno della Data science in psicologia: perché la Data science ci consente di descrivere la variazione e il cambiamento. E queste sono appunto le caratteristiche di base dei fenomeni psicologici.

Sono sicuro che, leggendo queste righe, a molti studenti sarà venuta in mente la seguente domanda: perché non chiediamo a qualche esperto di fare il "lavoro sporco" (ovvero le analisi statistiche) per noi, mentre noi (gli psicologi) ci occupiamo solo di ciò che ci interessa, ovvero dei problemi psicologici slegati dai dettagli "tecnici" della Data

science? La risposta a questa domanda è che non è possibile progettare uno studio psicologico sensato senza avere almeno una comprensione rudimentale della Data science. Le tematiche della Data science non possono essere ignorate né dai ricercatori in psicologia né da coloro che svolgono la professione di psicologo al di fuori dell'Università. Infatti, anche i professionisti al di fuori dall'università non possono fare a meno di leggere la letteratura psicologica più recente: il continuo aggiornamento delle conoscenze è infatti richiesto dalla deontologia della professione. Ma per potere fare questo è necessario conoscere un bel po' di Data science! Basta aprire a caso una rivista specialistica di psicologia per rendersi conto di quanto ciò sia vero: gli articoli che riportano i risultati delle ricerche psicologiche sono zeppi di analisi statistiche e di modelli formali. E la comprensione della letteratura psicologica rappresenta un requisito minimo nel bagaglio professionale dello psicologo.

Le considerazioni precedenti cercano di chiarire il seguente punto: la Data science non è qualcosa da studiare a malincuore, in un singolo insegnamento universitario, per poi poterla tranquillamente dimenticare. Nel bene e nel male, gli psicologi usano gli strumenti della Data science in tantissimi ambiti della loro attività professionale: in particolare quando costruiscono, somministrano e interpretano i test psicometrici. È dunque chiaro che possedere delle solide basi di Data science è un tassello imprescindibile del bagaglio professionale dello psicologo. In questo insegnamento verrano trattati i temi base della Data science e verrà adottato un punto di vista bayesiano, che corrisponde all'approccio più recente e sempre più diffuso in psicologia.

Come studiare

> *I know quite certainly that I myself have no special talent. Curiosity, obsession and dogged endurance, combined with self-criticism, have brought me to my ideas.*
>
> — Albert Einstein

Il giusto metodo di studio per prepararsi all'esame di Psicometria è quello di seguire attivamente le lezioni, assimilare i concetti via via che essi vengono presentati e verificare in autonomia le procedure presentate a lezione. Incoraggio gli studenti a farmi domande per chiarire ciò che non è stato capito appieno. Incoraggio gli studenti a utilizzare i forum attivi su Moodle e, soprattutto, a svolgere gli esercizi proposti su Moodle. I problemi forniti su Moodle rappresentano il livello di difficoltà richiesto per superare l'esame e consentono allo studente di comprendere se le competenze sviluppate fino a quel punto sono sufficienti rispetto alle richieste dell'esame.

La prima fase dello studio, che è sicuramente individuale, è quella in cui è necessario acquisire le conoscenze teoriche relative ai problemi che saranno presentati all'esame. La seconda fase di studio, che può essere facilitata da scambi con altri e da incontri di gruppo, porta ad acquisire la capacità di applicare le conoscenze: è necessario capire come usare un software (R) per applicare i concetti statistici alla specifica situazione del problema che si vuole risolvere. Le due fasi non sono però separate: il saper fare molto spesso ci aiuta a capire meglio.

Sviluppare un metodo di studio efficace

> *Memorization is not learning.*
>
> — Richard Phillips Feynman

Avendo insegnato molte volte in passato un corso introduttivo di analisi dei dati ho notato nel corso degli anni che gli studenti con l'atteggiamento mentale che descriverò qui sotto generalmente ottengono ottimi risultati. Alcuni studenti sviluppano naturalmente questo approccio allo studio, ma altri hanno bisogno di fare uno sforzo per maturarlo.

Fornisco qui sotto una breve descrizione del "metodo di studio" che, nella mia esperienza, è il più efficace per affrontare le richieste di questo insegnamento (Burger & Starbird, 2012).

- Dedicate un tempo sufficiente al materiale di base, apparentemente facile; assicuratevi di averlo capito bene. Cercate le lacune nella vostra comprensione. Leggere presentazioni diverse dello stesso materiale (in libri o articoli diversi) può fornire nuove intuizioni.
- Gli errori che facciamo sono i nostri migliori maestri. Istintivamente cerchiamo di dimenticare subito i nostri errori. Ma il miglior modo di imparare è apprendere dagli errori che commettiamo. In questo senso, una soluzione corretta è meno utile di una soluzione sbagliata. Quando commettiamo un errore questo ci fornisce un'informazione importante: ci fa capire qual è il materiale di studio sul quale dobbiamo ritornare e che dobbiamo capire meglio.
- C'è ovviamente un aspetto "psicologico" nello studio. Quando un esercizio o problema ci sembra incomprensibile, la cosa migliore da fare è dire: "mi arrendo", "non ho idea di cosa fare!". Questo ci rilassa: ci siamo già arresi, quindi non abbiamo niente da perdere, non dobbiamo più preoccuparci. Ma non dobbiamo fermarci qui. Le cose "migliori" che faccio (se ci sono) le faccio quando non ho voglia di lavorare. Alle volte, quando c'è qualcosa che non so fare e non ho idea di come affontare, mi dico: "oggi non ho proprio voglia di fare fatica", non ho voglia di mettermi nello stato mentale per cui "in 10 minuti devo risolvere il problema perché dopo devo fare altre cose". Però ho voglia di *divertirmi* con quel problema e allora mi dedico a qualche aspetto "marginale" del problema, che so come affrontare, oppure considero l'aspetto più difficile del problema, quello che non so come risolvere, ma invece di cercare di risolverlo, guardo come altre persone hanno affrontato problemi simili, opppure lo stesso problema in un altro contesto. Non mi pongo l'obiettivo "risolvi il problema in 10 minuti", ma invece quello di farmi un'idea "generale" del problema, o quello di capire un caso più specifico e più semplice del problema. Senza nessuna pressione. Infatti, in quel momento ho deciso di non lavorare (ovvero, di non fare fatica). Va benissimo se "parto per la tangente", ovvero se mi metto a leggere del materiale che sembra avere poco a che fare con il problema centrale (le nostre intuizioni e la nostra curiosità solitamente ci indirizzano sulla strada giusta). Quando faccio così, molto spesso trovo la soluzione del problema che mi ero posto e, paradossalmente, la trovo in un tempo minore di quello che, in precedenza, avevo dedicato a "lavorare" al problema. Allora perché non faccio sempre così? C'è ovviamente l'aspetto dei "10 minuti" che non è sempre facile da dimenticare. Sotto pressione, possiamo solo agire in maniera automatica, ovvero possiamo solo applicare qualcosa che già sappiamo fare. Ma se dobbiamo imparare qualcosa di nuovo, la pressione è un impedimento.
- È utile farsi da soli delle domande sugli argomenti trattati, senza limitarsi a cercare di risolvere gli esercizi che vengono assegnati. Quando studio qualcosa mi viene in mente: "se questo è vero, allora deve succedere quest'altra cosa". Allora verifico se questo è vero, di solito con una simulazione. Se i risultati della simulazione sono quelli che mi aspetto, allora vuol dire che ho capito. Se i risultati sono diversi da quelli che mi aspettavo, allora mi rendo conto di non avere capito e ritorno indietro a studiare con più attenzione la teoria che pensavo di avere capito – e ovviamente mi rendo conto che c'era un aspetto che avevo frainteso. Questo tipo di verifica è qualcosa che dobbiamo fare da soli, in prima persona: nessun altro può fare questo al posto nostro.
- Non aspettatevi di capire tutto la prima volta che incontrate un argomento nuovo.[1] È utile farsi una nota mentalmente delle lacune nella vostra comprensione e tornare su di esse in seguito per carcare di colmarle. L'atteggiamento naturale, quando

[1] Ricordatevi inoltre che gli individui tendono a sottostimare la propria capacità di apprendere (Horn & Loewenstein, 2021).

non capiamo i dettagli di qualcosa, è quello di pensare: “non importa, ho capito in maniera approssimativa questo punto, non devo preoccuparmi del resto”. Ma in realtà non è vero: se la nostra comprensione è superficiale, quando il problema verrà presentato in una nuova forma, non riusciremo a risolverlo. Per cui i dubbi che ci vengono quando studiamo qualcosa sono il nostro alleato più prezioso: ci dicono esattamente quali sono gli aspetti che dobbiamo approfondire per potere migliorare la nostra preparazione.

- È utile sviluppare una visione d’insieme degli argomenti trattati, capire l’obiettivo generale che si vuole raggiungere e avere chiaro il contributo che i vari pezzi di informazione forniscono al raggiungimento di tale obiettivo. Questa organizzazione mentale del materiale di studio facilita la comprensione. È estremamente utile creare degli schemi di ciò che si sta studiando. Non aspettate che sia io a fornirvi un riepilogo di ciò che dovete imparare: sviluppate da soli tali schemi e tali riassunti.
- Tutti noi dobbiamo imparare l’arte di trovare le informazioni, non solo nel caso di questo insegnamento. Quando vi trovate di fronte a qualcosa che non capite, o ottenete un oscuro messaggio di errore da un software, ricordatevi: “Google is your friend”.

Corrado Caudek

Capitolo 1

Valutare e confrontare i modelli

In breve

Il principio base del metodo scientifico è la *replicabilità* delle osservazioni: le osservazioni che non possono essere replicate sono poco interessanti. Parallelamente, una caratteristica fondamentale di un modello scientifico è la *generalizzabilità*: se un modello è capace di descrivere soltanto le proprietà di uno specifico campione di osservazioni, allora è poco utile. Ma come è possibile valutare la generalizzabilità di un modello statistico? Questa è la domanda a cui cercheremo di rispondere in questo Capitolo.

Secondo Johnson et al. (2022), nel valutare un modello, il ricercatore deve porsi tre domande critiche.

- Quali conseguenze più ampie derivano dall'inferenza? Come e chi ha raccolto i dati? Colui che svolge la ricerca otterrebbe di benefici manipolando i dati (escludendo delle osservazioni; selezionando il campione)? Che impatto hanno inferenze che vengono tratte dai dati sugli individui e sulla società? Quali pregiudizi o strutture di potere possono essere coinvolti in questa analisi?
- Che tipo di distorsioni sistematiche potrebbero essere presenti nell'analisi statistica? Ricordiamo la famosa citazione di George Box: "Tutti i modelli sono sbagliati, ma alcuni sono utili". È dunque importante sapere quanto è sbagliato il modello. Le assunzioni che stanno alla base del modello sono ragionevoli? Il meccanismo generatore dei dati che è stato ipotizzato è adeguato per il fenomeno in esame?
- Quanto è accurato il modello? Quanto sono lontane dalla realtà le previsioni del modello?

Per approfondire questi temi, si rinvia al testo di Johnson et al. (2022). Qui ci concentreremo su uno dei temi critici relativa alla validità di un modello, ovvero sul tema della generalizzabilità del modello.

Nella scienza l'utilità di una teoria viene verificata esaminando la corrispondenza tra predizioni teoriche e osservazioni. Se vi sono discrepanze significative tra predizioni e osservazioni ciò suggerisce che la teoria, o nella nostra visione più ristretta, il modello statistico, è poco utile. Il problema della capacità predittiva del modello non riguarda soltanto l'adeguatezza del modello in riferimento ad uno specifico campione di dati, ma riguarda anche la capacità di un modello statistico sviluppato in un campione di dati di ben adattarsi ad altri campioni della stessa popolazione.

In generale, i modelli statistici tendono a non generalizzarsi bene a un nuovo campione; questo perché sfruttano le caratteristiche specifiche dei dati del campione e tendono a produrre risultati eccessivamente ottimistici (cioè le dimensioni dell'effetto) che sovrastimano la dimensione dell'effetto atteso sia nella popolazione che in nuovi campioni.

Benché i problemi della generalizzabilità dei modelli e il metodo chiave per valutarli – ovvero, la convalida incrociata (*cross-validation*) – siano stati discussi sin dagli esordi della letteratura psicometrica (Lord, 1950), tali temi sono stati sottovalutati nella formazione psicologica contemporanea e nella ricerca. Tuttavia, questi concetti diventeranno sempre più importanti considerata l'enfasi corrente sulla necessità di condurre ricerche replicabili. Un'introduzione a questi temi è fornita, da esempio, da Song et al. (2021). Nello specifico, Song et al. (2021) mostrano che un modello che viene adattato a un campione (*campione di calibrazione*) non si generalizza bene a un altro campione (*campione di convalida*): la capacità predittiva del modello è minore quando il modello viene applicato al campione di convalida piuttosto che al campione di calibrazione. Questo problema è detto *sovra-adattamento* (*overfitting*). In generale, Song et al. (2021) mostrano come la capacità di generalizzazione del modello diminuisce (a) all'aumentare della complessità del modello, (b) al diminuire dell'ampiezza del campione di calibrazione, e (c) al diminuire della dimensione dell'effetto nella popolazione.

Sebbene i modelli statistici producono comunemente un sovra-adattamento, è anche possibile che essi producano un *sotto-adattamento* (*underfitting*) dei dati. Tale mancanza di adattamento è dovuta dalla variabilità campionaria e dalla complessità del modello. Il sotto-adattamento porta ad un R^2 basso e ad un *MSE* alto, sia nei campioni di calibrazione che in quelli di convalida. Per questo motivo, la scarsa generalizzabilità del modello può essere dovuta sia al sovra-adattamento che al sotto-adattamento del modello.

Per aumentarne la capacità di generalizzazione del modello devono essere soddisfatte tre condizioni: (a) campioni di calibrazione grandi, (b) dimensioni dell'effetto non piccole nella popolazione, e (c) modelli che non siano inutilmente complessi. Tuttavia, nella ricerca psicologica queste tre condizioni sono difficili da soddisfare: l'aumento della dimensione del campione spesso richiede l'utilizzo di maggiori risorse, la dimensione di un dato effetto nella popolazione non è soggetta alla discrezione dei ricercatori e la complessità del modello è spesso guidata da motivazioni teoriche. Pertanto, negli studi psicologici la generalizzabilità dei modelli è spesso problematica. Ciò rende necessario che il ricercatore fornisca informazioni aggiuntive relative alla capacità del modello di generalizzarsi a nuovi campioni. L'obiettivo di questo capitolo è di descrivere come questo possa essere fatto utilizzando l'approccio bayesiano.

1.1 Capacità predittiva

Nel framework bayesiano il problema della generalizzabilità di un modello viene affrontato valutando la capacità predittiva del modello, laddove per capacità predittiva si intende la capacità di un modello, i cui parametri sono stati stimati usando le informazioni di un campione, di ben adattarsi ad un campione di osservazioni future. In questo Capitolo cercheremo di rispondere a tre domande.

1. Quali criteri consentono di valutare la capacità predittiva di un modello?
2. Come quantificare la capacità predittiva di un modello usando solo un campione di osservazioni?
3. Come confrontare le capacità predittive di modelli diversi?

1.2 Il rasoio di Ockham

Il problema di scegliere il modello più adatto a spiegare un fenomeno di interesse è uno dei più importanti problemi in campo scientifico. I ricercatori si chiedono: il modello è completo? È necessario aggiungere un nuovo parametro al modello? Come può essere migliorato il modello? Se ci sono modelli diversi, qual'è il modello migliore?

Per rispondere a queste domande è possibile usare il rasoio di Ockham: *frustra fit per plura quod potest fieri per pauciora* ("si fa inutilmente con molte cose ciò che si può fare

con poche cose"). Parafrasando la massima si potrebbe dire: se due modelli descrivono i dati egualmente bene, viene sempre preferito il modello più semplice. Questo è il principio che sta alla base della ricerca scientifica.

Il rasoio di Ockham, però, non consente sempre di scegliere tra modelli alternativi. Se due modelli fanno le stesse predizioni ma differiscono in termini di complessità — per esempio, relativamente al numero di parametri di cui sono costituiti — allora è facile decidere: viene preferito il modello più semplice, anche perché, pragmaticamente, è il più facile da usare. Tuttavia, in generale, i modelli differiscono sia per complessità (ovvero, per il numero di parametri) che per accuratezza (ovvero, per la grandezza degli errori di predizione). In tali circostanze il rasoio di Ockham non è sufficiente: non consente infatti di trovare un equilibrio tra accuratezza e semplicità.

In questo Capitolo ci chiederemo come sia possibile misurare l'accuratezza predittiva di un modello. Ciò ci consentirà, in seguito, di usare il rasoio di Ockham: a parità di accuratezza, sarà possibile scegliere il modello più semplice. Ma nella pratica scientifica non si sacrifica mai l'accuratezza per la semplicità: il criterio prioritario è sempre l'accuratezza.

Secondo McElreath (2020), la selezione tra modelli deve evitare due opposti errori: il sovra-adattamento e il sotto-adattamento. Tale problema va sotto il nome di *bias-variance trade-off*: il sotto-adattamento, infatti, porta a distorsioni (*bias*) nella stima dei parametri, mentre il sovra-adattamento porta a previsioni scadenti in campioni futuri. Spesso l'incertezza relativa alla scelta del modello (sotto-adattamento versus sovra-adattamento) passa inosservata ma il suo impatto può essere drammatico.

> Standard statistical practice ignores model uncertainty. Data analysts typically select a model from some class of models and then proceed as if the selected model had generated the data. This approach ignores the uncertainty in model selection, leading to over-confident inferences and decisions that are more risky than one thinks they are. (Hoeting et al., 1999)

In questo Capitolo esamineremo alcune tecniche bayesiane che possono essere utilizzate per operare una selezione tra modelli alternativi, tenendo sotto controllo i pericoli del sovra-adattamento e del sotto-adattamento. In particolare, ci chiederemo quale, tra due o più modelli, sia quello da preferire in base al criterio della capacità predittiva.

Stargazing

Nella pratica concreta della ricerca, il metodo più comune per la selezione tra modelli alternativi utilizza i test di ipotesi statistiche di stampo frequentista. Questo metodo viene chiamato *stargazing*, poiché richiede soltanto l'esame degli asterischi (∗∗) che si trovano nell'output di un software statistico (gli asterischi marcano i coefficienti del modello che sono "statisticamente significativi"): alcuni ricercatori ritengono che il modello con più stelline sia anche il modello migliore. Questo però non è vero. Al di là dei problemi legati ai test dell'ipotesi nulla, è sicuramente un errore usare i test di significatività per la selezione di modelli: i valori-p non consentono di trovare un equilibrio tra *underfitting* e *overfitting*. Infatti, le variabili che migliorano la capacità predittiva di un modello non sono sempre statisticamente significative; d'altra parte, le variabili statisticamente significative non sempre migliorano la capacità predittiva di un modello.

Quando ci chiediamo quale, tra modelli alternativi, è il modello che meglio rappresenta il "vero" processo di generazione dei dati, ci troviamo di fronte al problema di quantificare il grado di "vicinanza" di un modello al "vero" processo di generazione dei dati. Si noti che, in tale confronto, facciamo riferimento sia alla famiglia distributiva così come ai valori dei parametri. Ad esempio, il modello $y_i \sim \mathcal{N}(5, 3)$ è diverso dal modello $y_i \sim \mathcal{N}(5, 6)$, ed è anche diverso dal modello $y_i \sim \Gamma(2, 2)$. I primi due modelli appartengono alla stessa famiglia distributiva ma differiscono nei termini dei valori dei parametri; gli ultimi due modelli appartengono a famiglie distributive diverse (gaussiano

vs. Gamma). Per misurare il grado di "vicinanza" tra due modelli, $\mathcal{M}_1$ e $\mathcal{M}_2$, la metrica di gran lunga più popolare è la *divergenza di Kullback-Leibler*. Per chiarire questo concetto è però prima necessario introdurre la nozione di entropia.

1.3 Entropia

Se vogliamo ottenere una comprensione intuitiva del concetto di entropia[1] possiamo pensare a quant'è informativa una distribuzione. Maggiore è l'entropia di una distribuzione, meno informativa sarà quella distribuzione e più uniformemente verranno assegnate le probabilità agli eventi. In altri termini, ottenere la risposta di "42" è più informativo della risposta "42 $\pm$ 5", che a sua volta è più informativo della risposta "un numero qualsiasi". L'entropia quantifica questa osservazione qualitativa.

Il concetto di entropia si applica sia alle distribuzioni continue sia a quelle discrete, ma è più facile da capire usando le distribuzioni discrete. Negli esempi successivi vedremo alcuni esempi applicati al caso discreto, ma gli stessi concetti si applicano al caso continuo.

Entropia di un singolo evento

Il concetto di entropia può essere usato per descrivere la quantità di informazione fornita da un evento. L'intuizione che sta alla base del concetto di entropia è che l'informazione fornita da un evento descrive la sorpresa suscitata dall'evento: gli eventi rari (a bassa probabilità) sono più sorprendenti – e quindi forniscono più informazione – degli eventi comuni (ad alta probabilità). In altre parole,

- un evento a bassa probabilità è sorprendente e fornisce molta informazione;
- un evento ad alta probabilità è poco o per niente sorprendente e fornisce poca (o nessuna) informazione.

È possibile quantificare l'informazione fornita dal verificarsi di un evento mediante la probabilità di quell'evento. Una tale quantità di informazione è chiamata "informazione di Shannon", "auto-informazione" o semplicemente "informazione" e, per un evento discreto x, può essere calcolata come:

$$\text{informazione}(x) = -\log_2 p(x),$$

dove $\log_2$ è il logaritmo in base 2 e $p(x)$ è la probabilità dell'evento x.

La scelta del logaritmo in base 2 significa che l'unità di misura dell'informazione è il bit (cifre binarie). Questo può essere interpretato dicendo che l'informazione misura il numero di bit richiesti per rappresentare un evento.[2] Solitamente, si denota la quantità di informazione con $h()$:

$$h(x) = -\log p(x).$$

[1] La nozione di entropia fu introdotta agli inizi del XIX secolo nel campo della termodinamica classica; il secondo principio della termodinamica è infatti basato sul concetto di entropia che, in generale, è assunto come una misura del disordine di un sistema fisico. Successivamente Boltzmann fornì una definizione statistica di entropia. Nel 1948 Shannon impiegò la nozione di entropia nell'ambito della teoria delle comunicazioni.

[2] È possibile pensare all'entropia nei termini del numero di domande sì/no che devono essere poste per ridurre l'incertezza. Per esempio, se in un certo giorno ci può essere solo sole o pioggia, per ridurre l'incertezza, a fine giornata chiediamo: "ha piovuto?" La risposta (sì/no) ad una singola domanda elimina l'incertezza, e quindi l'informazione ottenuta (ovvero, la riduzione dell'incertezza) è uguale ad 1 bit. Se in una certa giornata ci potrebbero essere sole, pioggia o neve, per ridurre l'incertezza sono necessarie due domande: "c'era sole?"; "ha piovuto?" In questo secondo caso, l'informazione ottenuta (ovvero, la riduzione dell'incertezza) è uguale ad 2 bit. Usando un logaritmo in base 2, dunque, l'entropia può essere interpretata come il numero minimo di bit necessari per codificare la quantità di informazione nei dati.

Il segno negativo garantisce che il risultato sia sempre positivo o zero. L'informazione è zero quando la probabilità dell'evento è 1.0, ovvero quando l'evento è certo (assenza di sorpresa).

Esempio 1.1. Consideriamo il lancio di una moneta equilibrata. La probabilità di testa (e croce) è 0.5. La quantità di informazione di ottenere "testa" è dunque

```
-log2(0.5)
#> [1] 1
```

Per rappresentare questo evento abbiamo bisogno di 1 bit di informazione. Se la stessa moneta venisse lanciata n volte, la quantità di informazione necessaria per rappresentare questo evento (ovvero, questa sequenza di lanci) sarebbe pari a n bit. Se la moneta non è equilibrata e la probabilità di testa è 0.1, allora l'evento "testa" è più raro e richiede più di 3 bit di informazione:

```
-log2(0.1)
#> [1] 3.32
```

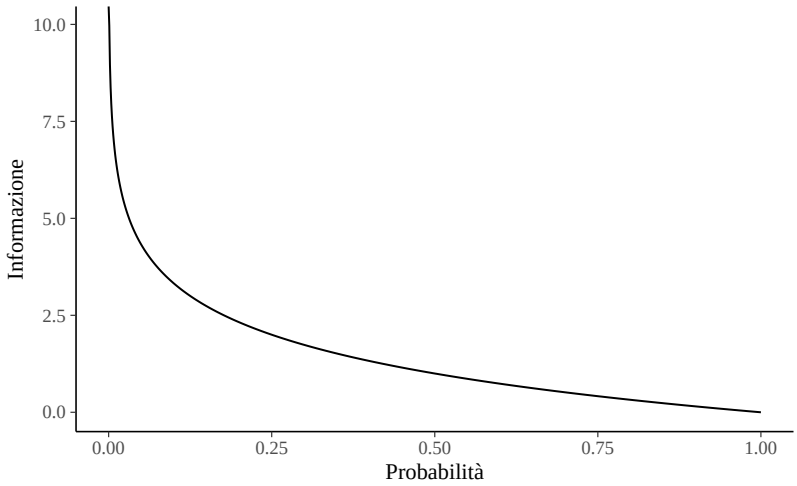
Consideriamo ora il lancio di un dado. Ci possiamo chiedere quanta informazione sia fornita, ad esempio, dall'evento "esce il valore 6". Dato che la probabilità di ottenere un 6 è più piccola della probabilità di ottenere "testa" nel lancio di una moneta, ci possiamo aspettare, nel lancio del dado, una maggiore sorpresa, ovvero una maggiore quantità di informazione. La quantità di informazione dell'evento "esce un 6" nel lancio di un dado

```
-log2(1/6)
#> [1] 2.58
```

è infatti più del doppio della quantità di informazione dell'evento "esce testa" nel lancio di una moneta.

Esempio 1.2. Nella figura successiva viene esaminata la relazione tra probabilità e informazione, per valori di probabilità nell'intervallo tra 0 e 1.

```
p <- seq(0, 1, length.out = 1000)
h <- -log2(p)
ggplot(tibble(p, h), aes(p, h)) +
  geom_line() +
  labs(
    x = "Probabilità",
    y = "Informazione"
  )
```



La figura mostra che questa relazione non è lineare, è infatti leggermente sublineare. Questo ha senso dato che abbiamo usato una funzione logaritmica.

Entropia di una variabile casuale

Possiamo estendere questa discussione pensando ad un insieme di eventi, ovvero ad una distribuzione. Nella teoria della probabilità, per fare riferimento ad un insieme di eventi e alle associate probabilità, usiamo la nozione di variabile casuale. L'entropia quantifica l'informazione che viene fornita da una variabile casuale.

Definizione 1.1. Sia $Y = y_1, \dots, y_n$ una variabile casuale e $p_t(y)$ una distribuzione di probabilità su Y. Si definisce la sua entropia (detta di Shannon) come:

$$H(Y) = -\sum_{i=1}^{n} p_t(y_i) \cdot \log_2 p_t(y_i). \tag{1.1}$$

Per interpretare la (1.1), consideriamo un esempio discusso da Martin et al. (2022).

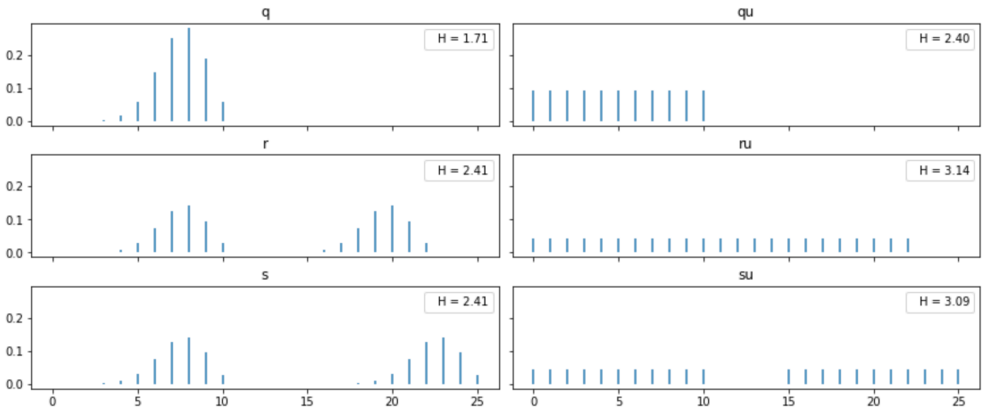


Figura 1.1: *Funzioni di massa di probabilità e associata entropia.*

Nella figura 1.1 sono rappresentate sei distribuzioni. viene anche riportato il valore di entropia di ciascuna distribuzione. La distribuzione con il picco più pronunciato o con la dispersione minore è `q`, e questa è la distribuzione con il valore di entropia più basso tra le sei distribuzioni considerate. La distribuzione `q ~ binom(n = 10, p = 0.75)`, e quindi ci sono 11 possibili eventi. `qu` è una distribuzione uniforme con gli stessi 11 possibili eventi. Possiamo vedere che l'entropia di `qu` è maggiore di quella di `q`. Infatti,

se calcoliamo l'entropia di distribuzioni binomiali con $n = 10$ e valori diversi di p ci possiamo rendere conto che nessuno di tali valori ha un'entropia maggiore di `qu`. Abbiamo bisogno di aumentare $n \approx 3$ volte per trovare la prima distribuzione binomiale con entropia maggiore di `qu`. Passiamo alla riga successiva. Generiamo la distribuzione `r` prendendo `q` e spostandolo a destra e quindi normalizzando (per garantire che la somma di tutte le probabilità sia 1). Poiché `r` ha una dispersione maggiore di `q`, la sua entropia è maggiore. `ru` è la distribuzione uniforme con lo stesso numero di possibili eventi di `r` (22) – si noti che sono stati inclusi come possibili valori anche quelli nella valle tra i due picchi. Ancora una volta l'entropia della versione uniforme è quella con l'entropia più grande.

Gli esempi discussi finora sembrano suggerire che l'entropia sia proporzionale alla varianza di una distribuzione. Verifichiamo questa intuizione esaminiamo le ultime due distribuzioni della figura 1.1. La distribuzione `s` è simile a `r` ma è presente una maggiore separazione tra i due picchi della distribuzione – dunque, la dispersione aumenta. Ciò nonostante, l'entropia resta invariata. Quindi la relazione tra entropia e varianza non è così semplice come sembrava. Il risultato che abbiamo trovato può essere spiegato dicendo che, nel calcolo dell'entropia, non vengono considerati gli eventi con probabilità nulla (quindi, nell'esempio, è stato possibile aumentare la varianza della distribuzione senza cambiare l'entropia). La distribuzione `su` è stata costruita sostituendo i due picchi in `s` con `qu` (e normalizzando). Possiamo vedere che `su` ha un'entropia minore di `ru`, anche se `su` ha una dispersione maggiore di `ru`. Questo è dovuto al fatto che `su` distribuisce la probabilità totale tra un numero minore di eventi (22) di `ru` (che ne conta 23); quindi è sensato che `su` abbia un'entropia minore di `ru`.

Per chi fosse interessato, il codice Python usato per generare la figura 1.1 è riportato qui sotto.

```
import scipy
import numpy as np

x = range(0, 26)
q_pmf = scipy.stats.binom(10, 0.75).pmf(x)
qu_pmf = scipy.stats.randint(0, np.max(np.nonzero(q_pmf))+1).pmf(x)
r_pmf = (q_pmf + np.roll(q_pmf, 12)) / 2
ru_pmf = scipy.stats.randint(0, np.max(np.nonzero(r_pmf))+1).pmf(x)
s_pmf = (q_pmf + np.roll(q_pmf, 15)) / 2
su_pmf = (qu_pmf + np.roll(qu_pmf, 15)) / 2

_, ax = plt.subplots(3, 2, figsize=(12, 5), sharex=True, sharey=True,
  constrained_layout=True)
ax = np.ravel(ax)
zipped = zip([q_pmf, qu_pmf, r_pmf, ru_pmf, s_pmf, su_pmf],
  ["q", "qu", "r", "ru", "s", "su"])
for idx, (dist, label) in enumerate(zipped):
    ax[idx].vlines(x, 0, dist, label=f"H = {scipy.stats.entropy(dist):.2f}")
    ax[idx].set_title(label)
    ax[idx].legend(loc=1, handlelength=0)
```

Esempio 1.3. Consideriamo un esempio riguardante le previsioni del tempo. Supponiamo che le probabilità di pioggia e sole siano, rispettivamente, $p_1 = 0.3$ e $p_2 = 0.7$. Quindi

$$H(p) = -[p(y_1)\log_2 p(y_1) + p(y_2)\log_2 p(y_2)] \approx 0.61.$$

Svolgendo i calcoli in R abbiamo:

```
p <- c(0.3 , 0.7)
-sum(p*log(p))
#> [1] 0.611
```

Se però viviamo a Las Vegas, allora le probabilità di pioggia e sole saranno qualcosa come $p(y_1) = 0.01$ e $p(y_2) = 0.99$. In questo secondo caso, l'entropia è 0.06, ovvero, molto minore di prima. Infatti, a Las Vegas non piove quasi mai, per cui quando abbiamo imparato che, in un certo giorno, non ha piovuto, abbiamo imparato molto poco rispetto a quello che già sapevamo in precedenza.

Esempio 1.4. Abbiamo visto in precedenza che, se gli esiti possibili sono pioggia o sole con $p(y_1) = 0.7$, $p(y_2) = 0.3$, allora l'entropia è

```
-(0.7 * log(0.7) + 0.3 * log(0.3))
#> [1] 0.611
```

Se gli esiti possibili sono pioggia, neve o sole con $p(y_1) = 0.7$, $p(y_2) = 0.15$ e $p(y_3) = 0.15$, rispettivamente, allora l'entropia sarà maggiore, ovvero pari a 0.82.

```
-(0.7 * log(0.7) + 0.15 * log(0.15) + 0.15 * log(0.15))
#> [1] 0.819
```

1.4 La divergenza di Kullback-Leibler

È comune in statistica utilizzare una distribuzione di probabilità q per approssimare un'altra distribuzione p – generalmente, questo viene fatto se p non è conosciuta o è troppo complessa. In questi casi possiamo chiederci quanta informazione viene perduta usando q al posto di p, o equivalentemente quanta ulteriore incertezza stiamo introducendo nell'analisi statistica. Intuitivamente, una risposta alla nostra domanda viene fornita da una quantità che ha valore zero quando q è uguale a p, e un valore positivo altrimenti. Seguendo la definizione (1.1) di entropia, possiamo quantificare questa perdita di informazione calcolando il valore atteso della differenza tra $\log(p)$ e $\log(q)$. Una tale quantità viene chiamata *entropia relativa* o *divergenza di Kullback-Leibler*:

$$\mathbb{KL}(p \,||\, q) = \mathbb{E}(\log p - \log q) \tag{1.2}$$

Quindi $\mathbb{KL}(p \,||\, q)$ ci fornisce la differenza media nelle probabilità logaritmiche quando si usa q per approssimare p. Poiché gli eventi si manifestano secondo p, dobbiamo calcolare il valore atteso rispetto a p. Per le distribuzioni discrete abbiamo:

$$\mathbb{KL}(p \,||\, q) = \sum_i^n p_i(\log p_i - \log q_i) = \sum_i^n p_i \frac{p_i}{q_i} \tag{1.3}$$

Riarrangiando i termini otteniamo:

$$\mathbb{KL}(p \,||\, q) = -\sum_i^n p_i(\log q_i - \log p_i), \tag{1.4}$$

ovvero,

$$\mathbb{KL}(p \,||\, q) = \underbrace{-\sum_i^n p_i \log q_i}_{H(p,q)} - \underbrace{\left(-\sum_i^n p_i \log p_i\right)}_{H(p)}, \tag{1.5}$$

laddove $H(p)$ è l'entropia di p e $H(p, q) = -\mathbb{E}[\log q]$ può essere intesa come l'entropia di q, ma valutata secondo i valori di p.

Riarrangiando l'equazione precedente otteniamo:

$$H(p, q) = H(p) + \mathbb{KL}(p \mid\mid q), \tag{1.6}$$

il che mostra come la divergenza KL possa essere interpretata come l'incremento di entropia rispetto a $H(p)$, quando si usa q per rappresentare p.

Definizione 1.2. Per due distribuzioni discrete p_t e $p_{\mathcal{M}}$, la divergenza KL di $p_{\mathcal{M}}$ da p_t è definita come:

$$D_{KL}(p_t \mid\mid p_{\mathcal{M}}) = \sum_{i=1}^{n} p_t(y_i) \cdot [\log p_t(y_i) - \log p_{\mathcal{M}}(y_i)]\,. \tag{1.7}$$

La D_{KL} introduce un piccolo cambiamento alla (1.1): anziché considerare una sola distribuzione di probabilità, p_t, consideriaamo anche un'approssimazione a tale distribuzione, ovvero $p_{\mathcal{M}}$. Calcolando la differenza dei logaritmi dei valori delle due distribuzioni giungiamo alla (1.7).

Se c'è una perfetta corrispondenza tra le due distribuzioni, $p(\cdot) = q(\cdot)$, allora

$$D_{KL}(p \mid\mid q) = \mathbb{E}(\log p - \log q). \tag{1.8}$$

Quindi,

$$D_{KL}(p \mid\mid q) \sum_{i=1}^{n} p_t(y_i) \cdot [\log p_t(y_i) - \log p_t(y_i)] = 0, \tag{1.9}$$

ovvvero: nessuna incertezza aggiuntiva viene introdotta se una distribuzione viene usata per rappresentare se stessa. Altrimenti, cioè se $p_t \neq p_{\mathcal{M}}$, la D_{KL} assume valori nell'intervallo $[0, \infty]$: all'aumentare della differenza tra $p_{\mathcal{M}}$ e p_t aumenta il valore $D_{KL}(p_t \mid\mid p_{\mathcal{M}})$. Il modello con la misura D_{KL} più bassa è ritenuto il migliore, nel senso che l'informazione persa quando si approssima la distribuzione del meccanismo generatore dei dati con la distribuzione prevista dal modello è la più bassa.

Esempio 1.5. (da McElreath, 2020) Sia la distribuzione target $p = \{0.3, 0.7\}$. Supponiamo che la distribuzione approssimata q possa assumere valori da $q = \{0.01, 0.99\}$ a $q = \{0.99, 0.01\}$. Calcoliamo la divergenza KL.

Le istruzioni R sono le seguenti:

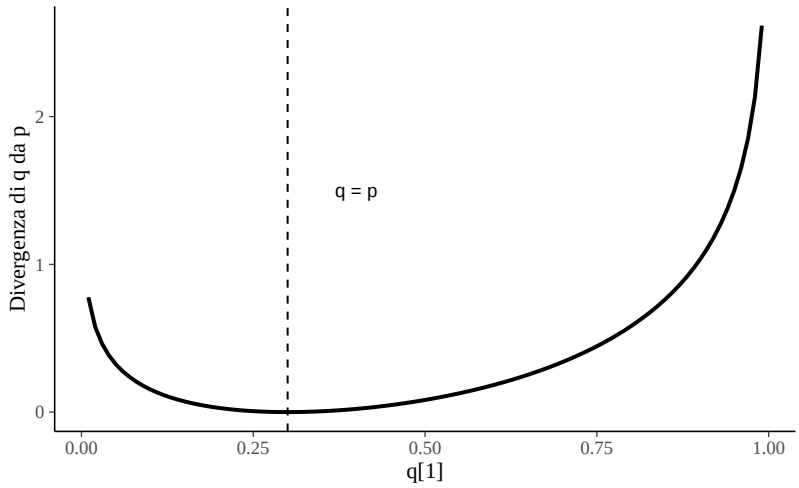
```
t <-
  tibble(
    p_1 = .3,
    p_2 = .7,
    q_1 = seq(from = .01, to = .99, by = .01)
  ) %>%
  mutate(
    q_2 = 1 - q_1
  ) %>%
  mutate(
    d_kl = (p_1 * log(p_1 / q_1)) + (p_2 * log(p_2 / q_2))
  )

head(t)
#> # A tibble: 6 × 5
#>     p_1   p_2   q_1   q_2  d_kl
#>   <dbl> <dbl> <dbl> <dbl> <dbl>
#> 1   0.3   0.7  0.01  0.99 0.778
#> 2   0.3   0.7  0.02  0.98 0.577
```

```
#> 3   0.3   0.7  0.03  0.97 0.462
#> 4   0.3   0.7  0.04  0.96 0.383
#> 5   0.3   0.7  0.05  0.95 0.324
#> 6   0.3   0.7  0.06  0.94 0.276
```

Nella figura seguente sull'asse delle ascisse sono rappresentati i valori q e sull'asse delle ordinate sono riportati i corrispondenti valori D_{KL}.

```
t %>%
  ggplot(aes(x = q_1, y = d_kl)) +
  geom_vline(xintercept = .3, linetype = 2) +
  geom_line(size = 1) +
  annotate(geom = "text", x = .4, y = 1.5, label = "q = p",
           size = 3.5) +
  labs(x = "q[1]",
       y = "Divergenza di q da p")
```



Tanto meglio la distribuzione q approssima la distribuzione target tanto più piccolo è il valore di divergenza KL.

Esempio 1.6. Sia p una distribuzione binomiale di parametri $\theta = 0.2$ e $n = 5$

```
n <- 4
p <- 0.2
true_py <- dbinom(0:n, n, 0.2)
true_py
#> [1] 0.4096 0.4096 0.1536 0.0256 0.0016
```

Sia q_1 una approssimazione a p:

```
q1 <- c(0.46, 0.42, 0.10, 0.01, 0.01)
q1
#> [1] 0.46 0.42 0.10 0.01 0.01
```

Sia q_2 una distribuzione uniforme:

```
q2 <- rep(0.2, 5)
q2
#> [1] 0.2 0.2 0.2 0.2 0.2
```

La divergenza KL di q_1 da p è

```
sum(true_py * log(true_py / q1))
#> [1] 0.0293
```

La divergenza KL di q_2 da p è:

```
sum(true_py * log(true_py / q2))
#> [1] 0.486
```

È chiaro che perdiamo una quantità maggiore di informazioni se, per descrivere la distribuzione binomiale p, usiamo la distribuzione uniforme q_2 anziché q_1.

La divergenza dipende dalla direzione

La divergenza KL non è una vera e propria metrica: per esempio, non è simmetrica. In generale, $D_{KL}(p_t \,||\, p_{\mathcal{M}}) \neq D_{KL}(p_{\mathcal{M}} \,||\, p_t)$, ovvero la D_{KL} da p_t a $p_{\mathcal{M}}$ è diversa dalla D_{KL} da $p_{\mathcal{M}}$ a p_t.

Esempio 1.7. Usando le seguenti istruzioni R otteniamo:

```
tibble(direction = c("Da q a p", "Da p a q"),
       p_1 = c(.01, .7),
       q_1 = c(.7, .01)) %>%
  mutate(p_2 = 1 - p_1,
         q_2 = 1 - q_1) %>%
  mutate(d_kl = (p_1 * log(p_1 / q_1)) + (p_2 * log(p_2 / q_2)))
#> # A tibble: 2 × 6
#>   direction   p_1   q_1   p_2   q_2  d_kl
#>   <chr>     <dbl> <dbl> <dbl> <dbl> <dbl>
#> 1 Da q a p   0.01  0.7   0.99  0.3   1.14
#> 2 Da p a q   0.7   0.01  0.3   0.99  2.62
```

1.5 Expected log predictive density

Nel caso continuo, la divergenza KL diventa:

$$D_{KL}(p_t \,||\, p_{\mathcal{M}}) = \int_{-\infty}^{+\infty} p_t(y) \log p_t(y) \, \mathrm{d}y - \int_{-\infty}^{+\infty} p_t(y) \log p_{\mathcal{M}}(y) \, \mathrm{d}y. \tag{1.10}$$

Se vengono confrontati due modelli, il primo termine della (1.10) resta costante e il confronto si riduce al secondo termine della (1.10), ovvero

$$\int_{-\infty}^{+\infty} p_t(y) \log p_{\mathcal{M}}(y) \, \mathrm{d}y. \tag{1.11}$$

Riscriviamo ora la (1.11) facendo riferimento alla distribuzione predittiva a posteriori, $p(\tilde{y} \mid y)$, perché ciò a cui siamo interessati è la divergenza di $p(\tilde{y} \mid y)$ da $p_t(y)$:

$$\text{elpd} = \int_{\tilde{y}} p_t(\tilde{y}) \log p(\tilde{y} \mid y) \, \mathrm{d}\tilde{y}. \tag{1.12}$$

La (1.12) è chiamata *expected log predictive density* (elpd) e fornisce la risposta al problema che ci eravamo posti all'inizio di questo Capitolo, ovvero il problema di definire un criterio per valutare la capacità predittiva di un modello. Possiamo pensare alla (1.12) dicendo che essa descrive la distribuzione predittiva a posteriori del modello ponderando

la verosimiglianza dei possibili dati futuri con la vera distribuzione p_t. Di conseguenza, valori elpd più grandi corrispondono ad una maggiore capacità predittiva del modello.

Non dobbiamo preoccuparci di trovare una formulazione analitica della distribuzione predittiva a posteriori $p(\tilde{y} \mid y)$ perché, come abbiamo visto nel Capitolo **??**, è possibile approssimare tale distribuzione mediante simulazione. Notiamo però che la (1.12) è formulata nei termini del vero modello generatore dei dati, p_t, il quale, ovviamente, è ignoto.[3] Di conseguenza, la quantità elpd non può mai essere calcolata in maniera esatta, ma può essere solo stimata. Il secondo problema di questo Capitolo è capire come la (1.12) possa essere stimata utilizzando un campione di osservazioni.

Log pointwise predictive density

Ingenuamente, potremmo pensare di stimare la (1.12) ipotizzando che la distribuzione del campione coincida con p_t. Usare la distribuzione del campione come proxy del vero modello generatore dei dati (ovvero, ipotizzare che la distribuzione del campione rappresenti fedelmente p_t) comporta due conseguenze:

- dato che il campione è finito, anziché eseguire un'operazione di integrazione, possiamo semplicemente sommare la densità predittiva a posteriori delle osservazioni;
- non è necessario ponderare per p_t, in quanto assumiamo che la distribuzione empirica del campione corrisponde a p_t (ciò significa assumere che i valori più comunemente osservati nel campione siano anche quelli più verosimili nella vera distribuzione p_t).

Questo conduce alla seguente equazione:[4]

$$\frac{1}{n}\sum_{i=1}^{n}\log p(y_i^{rep} \mid y). \tag{1.13}$$

La quantità (1.13), senza il passaggio finale della divisione per il numero di osservazioni, è chiamata *log pointwise predictive density* (lppd)

$$\text{lppd} = \sum_{i=1}^{n}\log p(y_i^{rep} \mid y) \tag{1.14}$$

e corrisponde alla somma delle densità predittive logaritmiche delle n osservazioni. Valori più grandi della (1.14) sono da preferire perché indicano una maggiore accuratezza media. È anche comune vedere espressa la quantità precedente nei termini della *devianza*, ovvero alla lppd moltiplicata per -2. In questo secondo caso sono da preferire valori piccoli.

È importante notare che lppd fornisce una *sovrastima* della (1.12). Tale sovrastima è dovuta al fatto che, nel calcolo della (1.14), abbiamo usato $p(y^{rep} \mid y)$ al posto di $p(\tilde{y} \mid y)$: in altri termini, abbiamo considerato le osservazioni del campione come se fossero un nuovo campione di dati. In una serie di simulazioni, McElreath (2020) esamina il significato di questa sovrastima. Nelle simulazioni la devianza viene calcolata come funzione della complessità (ovvero, il numero di parametri) del modello. La simulazione mostra che lppd aumenta al crescere del numero di parametri del modello. Ciò significa che lppd mostra lo stesso limite del coefficiente di determinazione: aumenta all'aumentare della complessità del modello.

[3] Se il modello sottostante i dati fosse noto non avremmo bisogno di cercare il modello migliore, perché p_t è il modello migliore.

[4] In riferimento alla notazione, ricordiamo che Gelman et al. (2014) distinguono tra y^{rep} e $\tilde{y}$. I valori y^{rep} corrispondono ad un'altra possibile realizzazione del medesimo modello statistico che ha prodotto y mediante determinati valori dei parametri θ (repliche sotto lo stesso modello statistico). I valori $\tilde{y}$ corrispondono invece ad un campione empirico di dati osservato in qualche futura occasione.

Esempio 1.8. Esaminiamo un esempio tratto da Bayesian Data Analysis for Cognitive Science nel quale la elpd viene calcolata in forma esatta oppure mediante approssimazione. Supponiamo di disporre di un campione di n osservazioni. Supponiamo inoltre di conoscere il vero processo generativo dei dati (qualcosa che in pratica non è mai possibile), ovvero:

$$p_t(y) = \mathrm{B}(1, 3).$$

I dati sono

```
set.seed(75)
n <- 10000
y_data <- rbeta(n, 1, 3)
head(y_data)
#> [1] 0.5506 0.1335 0.8025 0.2143 0.0191 0.0868
```

Supponiamo inoltre di avere adattato ai dati un modello bayesiano $\mathcal{M}$ e di avere ottenuto la distribuzione a posteriori per i parametri del modello. Inoltre, supponiamo di avere derivato la forma analitica della distribuzione predittiva a posteriori per il modello:

$$p(y^{rep} \mid y) \sim \mathrm{B}(2, 2).$$

Questa distribuzione ci dice quanto sono credibili i possibili dati futuri.

Conoscendo la vera distribuzione dei dati $p_t(y)$ possiamo calcolare in forma esatta la quantità elpd, ovvero

$$\text{elpd} = \int_{y^{rep}} p_t(y^{rep}) \log p(y^{rep} \mid y) \, \mathrm{d}y^{rep}.$$

Svolgiamo i calcoli in R otteniamo:

```
# True distribution
p_t <- function(y) dbeta(y, 1, 3)
# Predictive distribution
p <- function(y) dbeta(y, 2, 2)
# Integration
integrand <- function(y) p_t(y) * log(p(y))
integrate(f = integrand, lower = 0, upper = 1)
#> -0.375 with absolute error < 6.8e-07
```

Tuttavia, in pratica non conosciamo mai $p_t(y)$. Quindi approssimiamo elpd usando la (1.12):

$$\frac{1}{n} \sum_{i=1}^{n} \log p(y_i \mid y).$$

Così facendo, e svolgendo i calcoli in R, otteniamo

```
1/n * sum(log(p(y_data)))
#> [1] -0.364
```

un valore diverso da quello trovato in precedenza.

1.6 Criterio di informazione e convalida incrociata K-fold

Nel Paragrafo precedente abbiamo visto che la (1.14) fornisce una sovrastima della elpd. Il modo migliore per stimare elpd è raccogliere un nuovo campione indipendente

di dati, che si ritiene condivida lo stesso processo di generazione dei dati del campione corrente, e stimare elpd sul nuovo campione. Questa procedura è chiamata *out-of-sample validation*. Il problema, ovviamente, è che di solito non abbiamo le risorse per raccogliere un nuovo campione. Di conseguenza, gli statistici hanno messo a punto vari metodi per evitare la sovrastima della elpd che deriva dal solo utizzo del campione corrente. Ci sono due approcci generali:

- l'introduzione di un fattore di correzione;
- la convalida incrociata cosiddetta K-fold.

AIC, DIC e WAIC

Allo scopo di evitare la sovrastima della (1.14), le statistiche *Akaike Information Criterion* (AIC), *Deviance Information Criterion* (DIC) e *Widely Applicable Information Criterion* (WAIC) introducono un fattore di correzione. Le statistiche DIC e WAIC sono più complesse di AIC, ma producono un'approssimazione migliore. Tuttavia, i valori AIC, DIC e WAIC sono spesso molto simili tra loro. Per convenienza, dunque, qui ci accontenteremo di esaminare da vicino la statistica più semplice, ovvero AIC.

Criterio d'informazione di Akaike

Il criterio d'informazione di Akaike (in inglese *Akaike information criterion*, indicato come AIC) fornisce un metodo molto semplice per stimare la devianza media *out-of-sample*.

Definizione 1.3. Il criterio d'informazione di Akaike è definito come

$$AIC = -2 \log p(y \mid \hat{\theta}_{MLE}) + 2k, \tag{1.15}$$

dove k è il numero di parametri stimati nel modello e $p(y \mid \hat{\theta}_{MLE})$ è il valore massimizzato della funzione di verosimiglianza del modello stimato.

Dividendo per -2, otteniamo elpd_{AIC}:

$$\widehat{\text{elpd}}_{AIC} = \log p(y \mid \hat{\theta}_{MLE}) - k, \tag{1.16}$$

dove k è il fattore di correzione introdotto per evitare la sovrastima discussa in precedenza.

AIC è di interesse principalmente storico e produce una approssimazione attendibile di elpd quando:

1. le distribuzioni a priori sono non informative;
2. la distribuzione a posteriori è approssimativamente gaussiana multivariata;
3. la dimensione n del campione è molto maggiore del numero k dei parametri.

Esempio 1.9. Per meglio comprendere la statistica $\widehat{\text{elpd}}_{AIC}$, esaminiamo un esempio discusso da Gelman et al. (2014). Sia $y_1, \dots, y_n \sim \mathcal{N}(\theta, 1)$ un campione di osservazioni. Nel caso di una distribuzione a priori non-informativa $p(\theta) \propto 1$, la stima di massima verosimiglianza è $\bar{y}$. La log-verosimiglianza è

$$\begin{aligned} \log p(y \mid \hat{\theta}_{MLE}) &= -\frac{n}{2} \log(2\pi) - \frac{1}{2} \sum_{i=1}^{n} (y_i - \bar{y})^2 \\ &= -\frac{n}{2} \log(2\pi) - \frac{1}{2}(n-1)s_y^2, \end{aligned} \tag{1.17}$$

dove s_y^2 è la varianza campionaria.

Nel caso di un modello Normale con con varianza nota e una distribuzione a priori uniforme viene stimato un solo parametro, per cui

$$\begin{aligned}\widehat{\text{elpd}}_{AIC} &= \log p(y \mid \hat{\theta}_{MLE}) - k \\ &= -\frac{n}{2}\log(2\pi) - \frac{1}{2}(n-1)s_y^2 - 1. \end{aligned} \tag{1.18}$$

Convalida incrociata K-fold

La sovrastima della (1.14) può anche essere evitata usando una tecnica chiamata *K-fold cross-validation*. Mediante questo metodo vengono stimati i parametri del modello tralasciando una porzione di osservazioni (chiamata *fold*) dal campione per poi valutare il modello sulle osservazioni che sono state escluse. Una stima complessiva dell'accuratezza si ottiene poi calcolando la media del punteggio di accuratezza ottenuto in ogni fold. Il numero minimo di fold è 2; all'altro estremo, è possibile impiegare una singola osservazione in ciascun fold e adattare il modello tante volte (n) quante sono le singole osservazioni. Questa strategia è chiamata *leave-one-out cross-validation* (LOO-CV).

Importance sampling

La strategia LOO-CV è computazionalmente onerosa (ovvero, richiede un tempo di esecuzione molto lungo). È però possibile approssimare LOO-CV mediante un metodo chiamato *Pareto-smoothed importance sampling cross-validation* [PSIS; Vehtari et al. (2017)]. Tralasciando qui i dettagli matematici, l'intuizione di base è che PSIS fa leva sul punteggio di "importanza" posseduto da ciascuna osservazione all'interno della distribuzione a posteriori. Per "importanza" si intende il fatto che alcune osservazioni hanno un impatto maggiore sulle proprietà della distribuzione a posteriori di altre: se viene rimossa un'osservazione importante, le proprietà della distribuzione a posteriori cambiano molto; se viene rimossa un'osservazione poco importante, la distribuzione a posteriori cambia poco. L'"importanza" così intesa viene chiamata "peso" (*weight*) e tali pesi vengono utilizzati per stimare l'accuratezza *out-of-sample* del modello. PSIS-LOO-CV richiede che il modello venga adattato una volta soltanto ai dati e fornisce una stima della devianza *out-of-sample* che evita la sovrastima della (1.14). Inoltre, PSIS-LOO-CV fornisce un feedback sulla propria affidabilità identificando le osservazioni i cui pesi molto elevati potrebbero rendere imprecisa la predizione.

Valori $\widehat{\text{elpd}}_{\text{LOO}}$ più grandi indicano una maggiore accuratezza predittiva. In alternativa, anziché considerare $\widehat{\text{elpd}}$, è possibile usare la quantità $-2 \cdot \widehat{\text{elpd}}$, la quale è chiamata *LOO Information Criterion* (LOOIC). In questo secondo caso, valori LOOIC più piccoli sono da preferire.

La quantità $\widehat{\text{elpd}}_{\text{LOO}}$ viene calcolata dai pacchetti `loo` e `brms` ed è chiamata `elpd_loo` o `elpd_kfold`. È anche possibile calcolare la differenza della quantità `elpd_loo` per modelli alternativi, insieme alla deviazione standard della distribuzione campionaria di tale differenza.

Confronto tra AIC e LOO-CV

Per fare un esempio, faremo qui un confronto tra $\widehat{\text{elpd}}_{AIC}$ e $\widehat{\text{elpd}}_{LOO-CV}$. Esaminiamo nuovamente l'associazione tra il QI dei figli e il QI delle madri nel campione di dati discusso da Gelman et al. (2020). Una tale relazione può essere descritta da un modello di regressione nel quale la y corrisponde al QI dei figli e la x al QI delle madri.

Leggiamo i dati in R:

```
library("foreign")
df <- read.dta(here("data", "kidiq.dta"))
df$y <- scale(df$kid_score)[, 1]
df$x1 <- scale(df$mom_iq)[, 1]
```

```
head(df)
#>   kid_score mom_hs mom_iq mom_work mom_age       y      x1
#> 1        65      1  121.1        4      27 -1.0679  1.4078
#> 2        98      1   89.4        4      25  0.5489 -0.7092
#> 3        85      1  115.4        4      27 -0.0881  1.0295
#> 4        83      1   99.4        3      25 -0.1860 -0.0367
#> 5       115      1   92.7        4      27  1.3818 -0.4836
#> 6        98      0  107.9        1      18  0.5489  0.5268
```

Dato che AIC non è una statistica bayesiana, può essere calcolata mediante strumenti frequentisti:

```
m1_freq <- lm(y ~ x1, data = df)
AIC(m1_freq) / -2
#> [1] -570
```

Per ottenere LOO-CV adattiamo ai dati un modello di regressione bayesiano:

```
modelString = "
data {
  int<lower=0> N;
  vector[N] x1;
  vector[N] y;
}
parameters {
  real alpha;
  real beta1;
  real<lower=0> sigma;
}
transformed parameters {
  vector[N] mu;
  for (n in 1:N){
    mu[n] = alpha + beta1*x1[n];
  }
}
model {
  alpha ~ normal(0, 1);
  beta1 ~ normal(0, 1);
  sigma ~ cauchy(0, 1);
  y ~ normal(mu, sigma);
}
generated quantities {
  vector[N] y_rep;
  vector[N] log_lik;
  for (n in 1:N){
    y_rep[n] = normal_rng(mu[n], sigma);
    log_lik[n] = normal_lpdf(y[n] | x1[n] * beta1, sigma);
  }
}
"
writeLines(modelString, con = "code/simplereg.stan")
```

```
data1_list <- list(
  N = length(df$kid_score),
  y = df$y,
```

```
  x1 = df$x1
)
```

```
file1 <- file.path("code", "simplereg.stan")
```

```
mod1 <- cmdstan_model(file1)
```

Eseguiamo il campionamento MCMC:

```
fit1 <- mod1$sample(
  data = data1_list,
  iter_sampling = 4000L,
  iter_warmup = 2000L,
  seed = SEED,
  chains = 4L,
  parallel_chains = 2L,
  cores = 4L,
  refresh = 0,
  thin = 1
)
```

Calcoliamo infine la quantità $\widehat{\text{elpd}}_{LOO-CV}$:

```
loo1_result <- fit1$loo(cores = 4)
print(loo1_result)
#>
#> Computed from 16000 by 434 log-likelihood matrix
#>
#>          Estimate   SE
#> elpd_loo   -568.6 14.5
#> p_loo         1.9  0.2
#> looic      1137.2 28.9
#> ------
#> Monte Carlo SE of elpd_loo is 0.0.
#>
#> All Pareto k estimates are good (k < 0.5).
#> See help('pareto-k-diagnostic') for details.
```

Si noti la somiglianza tra $\widehat{\text{elpd}}_{LOO-CV}$ e $\widehat{\text{elpd}}_{AIC}$. In conclusione, possiamo dunque dire che $\widehat{\text{elpd}}_{LOO-CV}$ è la risposta bayesiana allo stesso problema che trova una soluzione frequentista nella statistica $\widehat{\text{elpd}}_{AIC}$.

Confronto tra modelli mediante LOO-CV

Come menzionato in precedenza, l'obiettivo centrale della misurazione dell'accuratezza predittiva è il confronto di modelli. Una volta capito come calcolare LOO-CV con un condice scritto in linguaggio Stan, svolgeremo ora un confronto di modelli.[5]

[5] A questo proposito, è necessario aggiungere una nota di cautela. Come fa notare McElreath (2020), fare previsioni e inferire i rapporti causali sono due cose molto diverse. Statistiche quali AIC, WAIC e LOO-CV consentono di individuare modelli con buone capacità predittive. Tali modelli, tuttavia, non riflettono necessariamente la struttura causale del fenomeno considerato: la selezione di modelli basata unicamente sull'accuratezza predittiva non garantisce che venga selezionato il modello che riflette la struttura causale del fenomeno (si veda anche Navarro, 2019).

Considereremo qui un confronto di modelli di regressione. Il modello di regressione discusso nel Paragrafo precedente prevede il QI dei bambini dal QI delle madri. Aggiungiamo a tale modello un secondo predittore che corrisponde all'età della madre. L'aggiunta di tale predittore migliori l'accuratezza predittiva del modello?

```
modelString = "
data {
  int<lower=0> N;
  vector[N] x1;
  vector[N] x2;
  vector[N] y;
}
parameters {
  real alpha;
  real beta1;
  real beta2;
  real<lower=0> sigma;
}
transformed parameters {
  vector[N] mu;
  for (n in 1:N){
    mu[n] = alpha + beta1*x1[n] + beta2*x2[n];
  }
}
model {
  alpha ~ normal(0, 1);
  beta1 ~ normal(0, 1);
  beta2 ~ normal(0, 1);
  sigma ~ cauchy(0, 1);
  y ~ normal(mu, sigma);
}
generated quantities {
  vector[N] y_rep;
  vector[N] log_lik;
  for (n in 1:N){
    y_rep[n] = normal_rng(mu[n], sigma);
    log_lik[n] = normal_lpdf(y[n] | x1[n] * beta1 + x2[n] * beta2, sigma);
  }
}
"
writeLines(modelString, con = "code/mreg2.stan")
```

```
df$x2 <- scale(df$mom_age)[, 1]
```

```
data2_list <- list(
  N = length(df$kid_score),
  y = df$y,
  x1 = df$x1,
  x2 = df$x2
)
```

```
file2 <- file.path("code", "mreg2.stan")
```

```
# compile model
mod2 <- cmdstan_model(file2)
```

```
# Running MCMC
fit2 <- mod2$sample(
  data = data2_list,
  iter_sampling = 4000L,
  iter_warmup = 2000L,
  seed = SEED,
  chains = 4L,
  parallel_chains = 2L,
  cores = 4L,
  refresh = 0,
  thin = 1
)
```

```
fit2$summary(c("alpha", "beta1", "beta2", "sigma"))
#> # A tibble: 4 × 10
#>   variable     mean   median     sd    mad      q5    q95  rhat
#>   <chr>       <dbl>    <dbl>  <dbl>  <dbl>   <dbl>  <dbl> <dbl>
#> 1 alpha    0.000387 0.000570 0.0431 0.0427 -0.0706 0.0709  1.00
#> 2 beta1    0.442    0.442    0.0434 0.0428  0.372  0.514   1.00
#> 3 beta2    0.0510   0.0511   0.0431 0.0431 -0.0192 0.122   1.00
#> 4 sigma    0.896    0.896    0.0306 0.0303  0.847  0.947   1.00
#> # … with 2 more variables: ess_bulk <dbl>, ess_tail <dbl>
```

```
loo2_result <- fit2$loo(cores = 4)
print(loo2_result)
#>
#> Computed from 16000 by 434 log-likelihood matrix
#>
#>          Estimate   SE
#> elpd_loo   -569.0 14.5
#> p_loo         3.0  0.3
#> looic      1137.9 29.0
#> ------
#> Monte Carlo SE of elpd_loo is 0.0.
#>
#> All Pareto k estimates are good (k < 0.5).
#> See help('pareto-k-diagnostic') for details.
```

Consideriamo infine un terzo modello che utilizza come predittori, oltre al QI della madre, una variabile dicotomica (codificata 0 o 1) che distingue madri che hanno completato le scuole superiori da quelle che non le hanno completate. Nuovamente, la domanda è se l'aggiunta di tale predittore migliori la capacità predittiva del modello.

```
modelString = "
data {
  int<lower=0> N;
  vector[N] x1;
  vector[N] x3;
  vector[N] y;
}
parameters {
```

```
  real alpha;
  real beta1;
  real beta3;
  real<lower=0> sigma;
}
transformed parameters {
  vector[N] mu;
  for (n in 1:N){
    mu[n] = alpha + beta1*x1[n] + beta3*x3[n];
  }
}
model {
  alpha ~ normal(0, 1);
  beta1 ~ normal(0, 1);
  beta3 ~ normal(0, 1);
  sigma ~ cauchy(0, 1);
  y ~ normal(mu, sigma);
}
generated quantities {
  vector[N] y_rep;
  vector[N] log_lik;
  for (n in 1:N){
    y_rep[n] = normal_rng(mu[n], sigma);
    log_lik[n] = normal_lpdf(y[n] | x1[n] * beta1 + x3[n] * beta3, sigma);
  }
}
"
writeLines(modelString, con = "code/mreg3.stan")
```

```
df$x3 <- df$mom_hs
```

```
data3_list <- list(
  N = length(df$kid_score),
  y = df$y,
  x1 = df$x1,
  x3 = df$x3
)
```

```
file3 <- file.path("code", "mreg3.stan")
```

```
mod3 <- cmdstan_model(file3)
```

```
fit3 <- mod3$sample(
  data = data3_list,
  iter_sampling = 4000L,
  iter_warmup = 2000L,
  seed = SEED,
  chains = 4L,
  parallel_chains = 2L,
  cores = 4L,
  refresh = 0,
  thin = 1
)
```

```
fit3$summary(c("alpha", "beta1", "beta3", "sigma"))
#> # A tibble: 4 × 10
#>   variable   mean median     sd    mad     q5     q95  rhat ess_bulk
#>   <chr>     <dbl>  <dbl>  <dbl>  <dbl>  <dbl>   <dbl> <dbl>    <dbl>
#> 1 alpha    -0.225 -0.225 0.0951 0.0939 -0.380 -0.0673  1.00    7808.
#> 2 beta1     0.414  0.414 0.0445 0.0440  0.340  0.487   1.00   10200.
#> 3 beta3     0.287  0.288 0.108  0.106   0.108  0.463   1.00    7832.
#> 4 sigma     0.890  0.889 0.0300 0.0295  0.842  0.941   1.00   11733.
#> # … with 1 more variable: ess_tail <dbl>
```

```
loo3_result <- fit3$loo(cores = 4)
print(loo3_result)
#>
#> Computed from 16000 by 434 log-likelihood matrix
#>
#>          Estimate   SE
#> elpd_loo   -584.2 16.4
#> p_loo         7.4  0.6
#> looic      1168.4 32.8
#> ------
#> Monte Carlo SE of elpd_loo is 0.0.
#>
#> All Pareto k estimates are good (k < 0.5).
#> See help('pareto-k-diagnostic') for details.
```

Per eseguire un confronto tra modelli in termini della loro capacità predittiva esaminiamo la differenza di LOO-CV tra coppie di modelli. Le seguenti istruzioni R producono la quantità `elpd_diff`, ovvero la differenza tra stime della elpd fornite da due modelli. Il primo argomento della funzione `loo_compare()` specifica il modello che viene usato come confronto. Nella prima riga dell'output, il valore `elpd_diff` è 0 (cioè, $x - x = 0$). Nelle righe successive sono riportate le differenze rispetto al modello di confronto (in questo caso, il modello 1). La colonna `se_diff` riporta l'errore standard di tali differenze.

L'incertezza della stima dell'accuratezza *out-of-sample* si distribuisce in maniera approssimativamente normale con media uguale al valore riportato dal software e deviazione standard uguale a ciò che è indicato nell'output come errore standard. Quando il campione è piccolo, questa approssimazione produce una forte sottostima dell'incertezza, ma fornisce comunque una stima migliore di AIC, DIC e WAIC.

```
w <- loo_compare(loo1_result, loo2_result, loo3_result)
print(w)
#>        elpd_diff se_diff
#> model1   0.0       0.0
#> model2  -0.4       1.3
#> model3 -15.6       6.0
```

Per interpretare l'output, usiamo il criterio suggerito da Gelman et al. (1995): consideriamo "credibile" una differenza se `elpd_diff` è almeno due volte maggiore di `se_diff`. Nel caso presente, dunque, il confronto tra il modello 2 e il modello 1 indica che la quantità `elpd_diff` è molto piccola rispetto al suo errore standard. Questo accade se un predittore è associato in modo trascurabile con la variabile dipendente. I dati presenti, dunque, non offrono alcuna evidenza che aggiungere dell'età della madre come predittore migliori la capacità predittiva del modello. Nel confronto tra modello 3 e modello 1, invece, la quantità `elpd_diff` è maggiore di due volte il valore dell'errore standard. Questo suggerisce un incremento della capacità predittiva del modello quiando il livello di istruzione della madre viene incluso tra i predittori.

È anche possibile calcolare l'intervallo di credibilità per `elpd_diff`:

```
15.5 + c(-1, 1) * qnorm(.95, 0, 1) * 6.0
#> [1]  5.63 25.37
```

Outlier

Si è soliti pensare che la maggior parte delle osservazioni del campione sia prodotta da un unico meccanismo generatore dei dati, mentre le rimanenti osservazioni sono la realizzazione di un diverso processo stocastico. Le osservazioni che appartengono a questo secondo gruppo si chiamano *outlier*. È dunque necessario identificare gli outlier e limitare la loro influenza sull'inferenza.[6]

Poniamoci ora il problema di identificare gli outlier con la tecnica PSIS-LOO-CV. Quando PSIS-LOO-CV viene calcolato con il pacchetto `loo`, l'output riporta il parametro di forma della distribuzione di Pareto (valore `k`). Tale valore può essere utilizzato per identificare gli outlier. Infatti, il valore `k` valuta, per ciascun punto del campione, l'approssimazione usata da PSIS-LOO-CV. Se $k < 0.5$, i pesi di importanza vengono stimati in modo accurato; se il valore k di Pareto di un punto è > 0.7, i pesi di importanza possono essere inaccurati. Le osservazioni con $k > 0.7$ sono dunque osservazioni outlier.

Per fare un esempio concreto, introduciamo nel campione dell'esempio precedente una singola osservazione outlier.

```
df1 <- df
dim(df1)
#> [1] 434   9
df1$x1[434] <- 10
df1$y[434] <- 10
```

Sistemiamo i dati nel formato appropriato per Stan:

```
data1a_list <- list(
  N = length(df1$kid_score),
  y = df1$y,
  x1 = df1$x1
)
```

Adattiamo nuovamente il modello 1 ad un campione di dati che contiene un outlier.

```
fit1a <- mod1$sample(
  data = data1a_list,
  iter_sampling = 4000L,
  iter_warmup = 2000L,
  seed = SEED,
  chains = 4L,
  parallel_chains = 2L,
  cores = 4L,
  refresh = 0,
  thin = 1
)
```

[6] McElreath (2020) nota che, spesso, i ricercatori eliminano i valori anomali prima di adattare un modello ai dati, basandosi solo sulla distanza dal valore medio della variabile dipendente misurata in termini di unità di deviazione standard. Secondo McElreath (2020) questo non dovrebbe mai essere fatto: un'osservazione può essere considerata come un valore anomalo o un valore influente solo alla luce delle predizioni di un modello (mai prima di avere adattato il modello ai dati). Se ci sono solo pochi valori anomali una strategia possibile è quella di riportare i risultati delle analisi statistiche svolte su tutto il campione dei dati oppure dopo avere eliminato le osservazioni anomale e influenti.

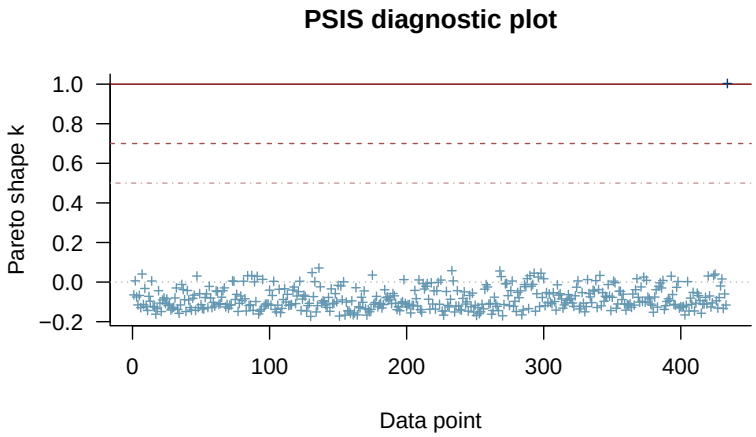
```
loo1a_result <- fit1a$loo(cores = 4)
```

Una tabella diagnostica che riassume le stime dei parametri di forma della distribuzione di Pareto si ottiene nel modo seguente:

```
print(loo1a_result)
#>
#> Computed from 16000 by 434 log-likelihood matrix
#>
#>          Estimate   SE
#> elpd_loo   -586.6 20.1
#> p_loo         7.1  5.4
#> looic      1173.2 40.3
#> ------
#> Monte Carlo SE of elpd_loo is NA.
#>
#> Pareto k diagnostic values:
#>                          Count Pct.    Min. n_eff
#> (-Inf, 0.5]   (good)     433   99.8%   9998
#>  (0.5, 0.7]   (ok)         0    0.0%   <NA>
#>    (0.7, 1]   (bad)        0    0.0%   <NA>
#>    (1, Inf)   (very bad)   1    0.2%   13
#> See help('pareto-k-diagnostic') for details.
```

Un grafico che riporta le stime dei parametri di forma della distribuzione di Pareto per ciascuna osservazione è dato da:

```
plot(loo1a_result)
```



Il valore `k` stimato da PSIS-LOO-CV mette chiaramente in luce il fatto che il valore introdotto nel campione è un outlier. L'indice dell'osservazione outlier è identificato con:

```
pareto_k_ids(loo1a_result, threshold = 0.7)
#> [1] 434
```

1.7 Selezione di variabili

I concetti che sono stati introdotti in questo Capitolo, tra le altre cose, risultano utili per affrontare un problema importante in psicologia, ovvero quello della semplificazione di un modello di regressione che contiene molti predittori. Il problema è quello di selezionare un insieme di variabili indipendenti così che tale selezione non comporti una apprezzabile perdita nella capacità predittiva del modello ristretto rispetto al modello completo. Un modo per identificare le variabili rilevanti per prevedere una determinata variabile risposta è quello di utilizzare il metodo basato sulla proiezione, come discusso nel seguente link e in Piironen e Vehtari (2017). Per descrivere questa procedura, adatto qui un esempio discusso da Mark Lai in Course Handouts for Bayesian Data Analysis Class. Iniziamo a leggere i dati.

```
kidiq <- rio::import(here::here("data", "kidiq.dta"))
kidiq <- kidiq %>%
  mutate(
    mom_hs = factor(mom_hs, labels = c("no", "yes"))
  )
```

Per potere usare delle distribuzione a priori sensate per i parametri, standardizzo le variabili numeriche.

```
scale_this <- function(x) as.vector(scale(x))
kidiq_scaled <- kidiq %>%
  as_tibble() %>%
  mutate(across(where(is.numeric), scale_this))
kidiq_scaled <- kidiq_scaled %>%
  mutate(
    mom_hs = kidiq$mom_hs
  )
glimpse(kidiq_scaled)
#> Rows: 434
#> Columns: 5
#> $ kid_score <dbl> -1.0679, 0.5489, -0.0881, -0.1860, 1.3818, 0.548…
#> $ mom_hs    <fct> yes, yes, yes, yes, yes, no, yes, yes, yes, yes,…
#> $ mom_iq    <dbl> 1.4078, -0.7092, 1.0295, -0.0367, -0.4836, 0.526…
#> $ mom_work  <dbl> 0.9342, 0.9342, 0.9342, 0.0878, 0.9342, -1.6051,…
#> $ mom_age   <dbl> 1.5602, 0.8198, 1.5602, 0.8198, 1.5602, -1.7718,…
```

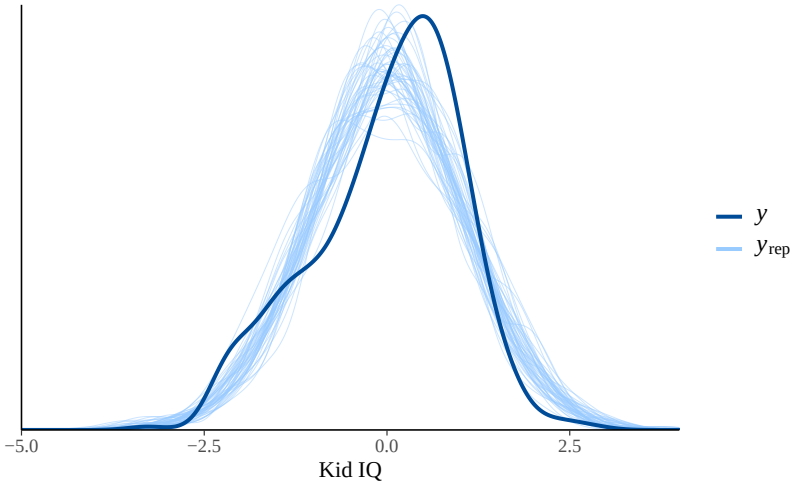
Il seguente modello di regressione utilizza `kid_score` quale variabile dipendente e, quali predittori, include tutte le altre variabili disponibili e le loro interazioni a due vie.

```
m1 <- brm(
  kid_score ~ (mom_iq + mom_hs + mom_work + mom_age)^2,
  data = kidiq_scaled,
  prior = c(
    prior(normal(0, 1), class = "Intercept"),
    prior(normal(0, 1), class = "b"),
    prior(student_t(4, 0, 1), class = "sigma")
  ),
  seed = 2302,
  chains = 4L,
  cores = 4L,
  refresh = 0,
  backend = "cmdstan"
```

```
)
#> Running MCMC with 4 parallel chains...
#>
#> Chain 1 finished in 0.2 seconds.
#> Chain 2 finished in 0.2 seconds.
#> Chain 3 finished in 0.2 seconds.
#> Chain 4 finished in 0.2 seconds.
#>
#> All 4 chains finished successfully.
#> Mean chain execution time: 0.2 seconds.
#> Total execution time: 0.3 seconds.
```

Un grafico che riporta un posterior predictive check si ottiene con l'istruzione seguente:

```
pp_check(m1, ndraws = 50, alpha = 0.5) +
  xlim(-5, 4) +
  labs(x = "Kid IQ")
```

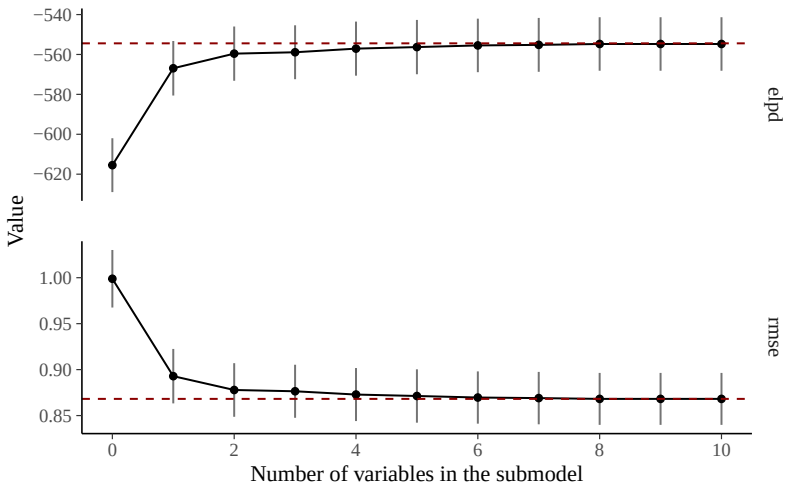


Identifichiamo ora l'importanza relativa delle variabili indipendenti nei termini della loro importanza per la previsione:

```
# Variable selection
vs <- projpred::varsel(m1)
```

Un grafico dell'importanza relativa di ciascuna variable per la previsione di `kid_score` si ottiene nel modo seguente:

```
# plot predictive performance on training data
plot(vs, stats = c("elpd", "rmse"))
```



Troviamo ora il numero di variabili da mantenere, in base al modello completo:

```
projpred::suggest_size(vs)
#> [1] 5
```

Usiamo quindi il metodo `cv_varsel()` per eseguire la convalida incrociata per vedere quante variabili dovrebbero essere incluse nel modello:

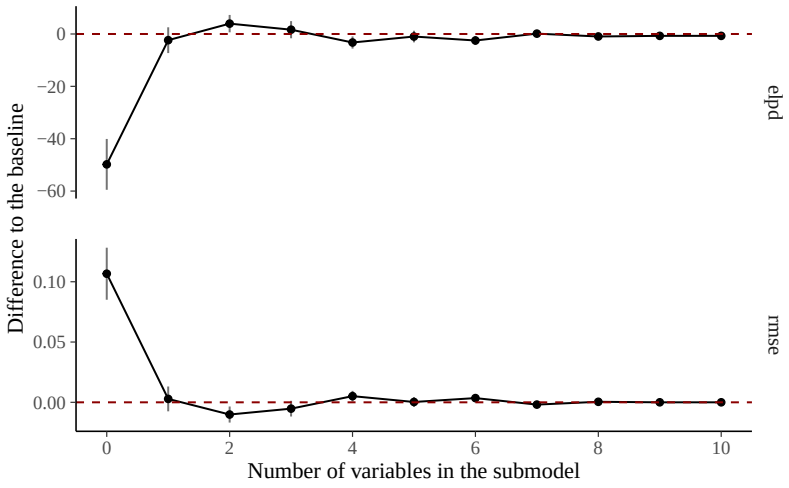
```
# With cross-validation
cvs <- projpred::cv_varsel(m1, verbose = FALSE)
```

In base al metodo della convalida incrociata, il numero di variabili da mantenere è

```
projpred::suggest_size(cvs)
#> [1] 1
```

Generiamo il grafico dei risultati della convalida incrociata, questa volta relativi al modello completo:

```
plot(cvs, stats = c("elpd", "rmse"), deltas = TRUE)
```



Stampiamo l'elenco delle variabili ordinate in base alla loro importanza relativa, secondo il metodo della convalida incrociata:

```
summary(cvs, stats=c('mse'), type = c('mean','se'))
#>    size   solution_terms    mse mse.se
#> 2     0             <NA> 1.001 0.0651
#> 3     1           mom_iq 0.804 0.0536
#> 4     2    mom_iq:mom_hs 0.781 0.0519
#> 5     3           mom_hs 0.790 0.0530
#> 6     4   mom_hs:mom_age 0.808 0.0539
#> 7     5  mom_hs:mom_work 0.800 0.0541
#> 8     6 mom_work:mom_age 0.805 0.0539
#> 9     7  mom_iq:mom_work 0.796 0.0529
#> 10    8   mom_iq:mom_age 0.800 0.0530
#> 11    9         mom_work 0.799 0.0528
#> 12   10          mom_age 0.799 0.0528
```

Il metodo basato sulla proiezione produce le distribuzioni a posteriori basate su una proiezione dal modello completo sul modello semplificato. In altre parole, si pone la domanda: "Se vogliamo un modello con solo `mom_iq` nel modello, quali coefficienti dovrebbero essere usati per fare in modo che l'accuratezza della previsione risultante sia la più vicina possibile a quella del modello completo?". I coefficienti ottenuti con il metodo basato sulla proiezione saranno dunque diversi da quelli che si avrebbero se si stimasse direttamente il modello utilizzando il solo predittore `mom_iq` (ad es. `m2`). I risultati ottenuti da studi basati sulla simulazione hanno mostrato che il metodo basato sulla proiezione produce un modello con prestazioni predittive migliori.

```
proj1 <- projpred::project(
  cvs,
  nv = suggest_size(cvs),
  seed = 123,
  ns = 1000
)
posterior_summary(proj1) %>%
  round(3)
#>           Estimate Est.Error   Q2.5 Q97.5
#> Intercept    0.002     0.037 -0.064 0.075
#> mom_iq       0.445     0.037  0.374 0.516
#> sigma        0.916     0.015  0.891 0.948
```

Per fare un confronto, stimiamo i coefficienti del modello di regressione che include unicamente la variabile `mom_iq`:

```
m2 <- brm(kid_score ~ mom_iq,
  data = kidiq_scaled,
  prior = c(
    prior(normal(0, 1), class = "Intercept"),
    prior(normal(0, 1), class = "b"),
    prior(student_t(4, 0, 1), class = "sigma")
  ),
  seed = 2302,
  chains = 4L,
  cores = 4L,
  refresh = 0,
```

```
  backend = "cmdstan"
)
#> Running MCMC with 4 parallel chains...
#>
#> Chain 1 finished in 0.0 seconds.
#> Chain 2 finished in 0.0 seconds.
#> Chain 3 finished in 0.0 seconds.
#> Chain 4 finished in 0.0 seconds.
#>
#> All 4 chains finished successfully.
#> Mean chain execution time: 0.0 seconds.
#> Total execution time: 0.3 seconds.
```

```
summary(m2)
#>  Family: gaussian
#>   Links: mu = identity; sigma = identity
#> Formula: kid_score ~ mom_iq
#>    Data: kidiq_scaled (Number of observations: 434)
#>   Draws: 4 chains, each with iter = 1000; warmup = 0; thin = 1;
#>          total post-warmup draws = 4000
#>
#> Population-Level Effects:
#>           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
#> Intercept    -0.00      0.04    -0.08     0.08 1.00     4154
#> mom_iq        0.45      0.04     0.36     0.53 1.00     4078
#>           Tail_ESS
#> Intercept     3062
#> mom_iq        2888
#>
#> Family Specific Parameters:
#>       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
#> sigma     0.90      0.03     0.84     0.96 1.00     4067     3113
#>
#> Draws were sampled using sample(hmc). For each parameter, Bulk_ESS
#> and Tail_ESS are effective sample size measures, and Rhat is the potential
#> scale reduction factor on split chains (at convergence, Rhat = 1).
```

Nel caso presente, le differenze sono minime, ma questo non è sempre vero.

1.8 Confronto di modelli tramite elpd

Confrontiamo ora la capacità predittiva a posteriori dei due modelli rispetto alla loro elpd. Ricordiamo che tanto maggiore è elpd rispetto ad un nuovo insieme di dati futuri $\tilde{y}$, $\log p(\tilde{y} \mid y)$, tanto maggiore è l'accuratezza predittiva del modello. Iniziamo a calcolare elpd per i due modelli:

```
loo1 <- loo::loo(m1)
loo2 <- loo::loo(m2)
c(loo1$estimates[1], loo2$estimates[1])
#> [1] -567 -570
```

La quantità elpd non fornisce una metrica interpretabile per l'accuratezza predittiva di un singolo modello. Risulta invece utile per il confronto tra modelli alternativi. Un confronto tra il modello completo e il modello semplificato si ottiene mediante la funzione `loo_compare()`:

```
loo::loo_compare(loo1, loo2)
#>    elpd_diff se_diff
#> m1  0.0       0.0
#> m2 -2.2       5.0
```

I risultati di tale confronto indicano che il Modello `m1` ha il valore elpd più basso e, dunque, sarebbe quello da preferire. Tuttavia, se si considera la differenza in elpd in riferimento all'errore standard corrispondente (nella colonna `se_diff`), ne risulta una differenza relativamente piccola. Per il Modello `m1` elpd è uguale a -567.3 e per `m2` è -569.5. La differenza è pari a (-567.3 - -569.5) = 2.2, con un errore standard stimato di 5.0. I dati dunque suggeriscono che la vera differenza in elpd tra `m1` e `m2` sia compresa tra $\pm$ 2 errori standard (ovvero nel caso presente, 10 unità) dalla differenza stimata di -2.2 unità, ovvero sia inclusa nell'intervallo $-2.2 \pm 2 \cdot 5 = (-12.2, 7.8)$. Dato il valore elpd $= 0$ è compreso nell'intervallo di $\pm$ due standard error dalla differenza stimata, i dati non forniscono evidenze convincenti che l'accuratezza predittiva a posteriori di `m1` sia superiore a quella di `m2`. Inoltre, dato che il Modello `m2` è più semplice di `m1`, concludiamo che `m2` sia il modello migliore tra i due considerati (rasoio di Ockham).

1.9 Coefficiente di determinazione bayesiano

Gelman et al. (2019) definiscono il coefficiente di determinazione bayesiano come

$$R^2 = \frac{\mathbb{V}_\mu}{\mathbb{V}_\mu + \mathbb{V}_{\text{res}}}, \tag{1.19}$$

dove $\mathbb{V}_\mu$ è la varianza del valore atteso predetto dal modello e $\mathbb{V}_{\text{res}}$ è la varianza dei residui. Entrambe queste quantità sono stimate considerando gli indici a posteriori del modello adattato.

Di seguito vengono calcolati i coefficienti di determinazione bayesiani dei due modelli discussi sopra:

```
loo_R2(m1, robust = TRUE) %>%
  round(3)
#>    Estimate Est.Error  Q2.5 Q97.5
#> R2    0.201     0.036 0.125  0.27
loo_R2(m2, robust = TRUE) %>%
  round(3)
#>    Estimate Est.Error  Q2.5 Q97.5
#> R2    0.196     0.033 0.128 0.255
```

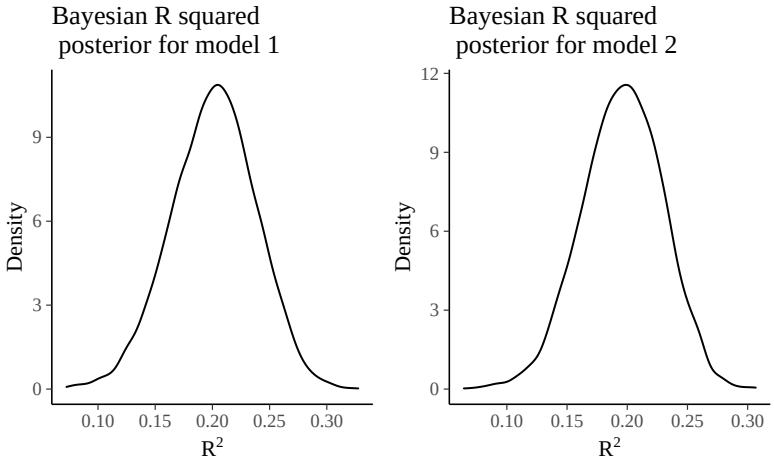
Una rappresentazione grafica della distribuzione a posteriori dei due coefficienti di determinazione bayesiani si ottiene com le seguenti istruzioni:

```
library("patchwork")
library("latex2exp")

m1_fit_r2 <- loo_R2(m1, summary = FALSE)
foo <- tibble(R2 = as.numeric(m1_fit_r2))
h1 <- foo %>%
  ggplot(aes(x = R2)) +
  geom_density(alpha = 0.5) +
  xlab(TeX("$R^2$")) +
  ylab("Density") +
  ggtitle("Bayesian R squared\n posterior for model 1")
```

```
m2_fit_r2 <- loo_R2(m2, summary = FALSE)
foo2 <- tibble(R2 = as.numeric(m2_fit_r2))
h2 <- foo2 %>%
  ggplot(aes(x = R2)) +
  geom_density(alpha = 0.5) +
  xlab(TeX("$R^2$")) +
  ylab("Density") +
  ggtitle("Bayesian R squared\n posterior for model 2")

h1 | h2
```



Considerato l'intervallo a posteriori del 95%, anche in questo caso non abbiamo evidenze convincenti che l'uso di un solo predittore faccia diminuire la capacità predittiva del modello.

Considerazioni conclusive

Dati due modelli computazionali che forniscono resoconti diversi di un set di dati, come possiamo decidere quale modello è maggiormente supportato dai dati? Nel presente Capitolo abbiamo visto come il problema del confronto di modelli possa essere formulato nei termini di un problema di inferenza statistica.

Abbiamo visto come la divergenza KL possa essere usata per confrontare una "vera" distribuzione di probabilità con una sua approssimazione. Abbiamo anche visto come che, da un punto di vista Bayesiano, il problema del confronto tra modelli viene presentato nei termini della capacità predittiva di un modello per nuove osservazioni future.

> The central question is then how one should decide among a set of competing models. A short answer is that a model should be selected based on its generalizability, which is defined as a model's ability to fit current data but also to predict future data. (Myung, 2003)

Se un modello non si generalizza bene a nuovi dati, si può sostenere che il modello è inappropriato o almeno manca di alcune caratteristiche importanti dato che non cattura la natura del vero processo di generazione dei dati sottostante $p_t(\tilde{y})$. La capacità predittiva di un modello viene comunemente descritta in termini della sua densità predittiva logaritmica attesa (ELPD):

$$\overline{\text{ELPD}} = \int \log p_{\mathcal{M}}(\tilde{y} \mid y) p_t(\tilde{y}) d\tilde{y}.$$

L'ELPD per il modello $\mathcal{M}$ può essere interpretata come la media pesata della densità predittiva logaritmica $\log p_{\mathcal{M}}(\tilde{y}_i \mid y)$ per una nuova osservazione per il modello $\mathcal{M}$, dove i pesi derivano dal vero processo di generazione dei dati $p_t(\tilde{y})$. Grandi valori di $\overline{\text{ELPD}}(\mathcal{M})$ indicano che il modello prevede bene nuove osservazioni $\tilde{y}$, mentre piccoli valori $\overline{\text{ELPD}}(\mathcal{M})$ mostrano che il modello non si generalizza bene a nuovi dati. In pratica, però, la vera densità $p_t(\tilde{y})$ è incognita. Una *stima* di $\overline{\text{ELPD}}(\mathcal{M})$ può essere ottenuta con il metodo di validazione incrociata leave-one-out (LOO) in cui il modello tante volte (n) quante sono le singole osservazioni (*leave-one-out cross-validation*, LOO-CV). La strategia LOO-CV è computazionalmente troppo onerosa per qualunque scopo pratico e viene quindi approssimata mediante un metodo chiamato *Pareto-smoothed importance sampling cross-validation* [PSIS; Vehtari et al. (2017)] – che non richiede di adattare il modello n volte. Tale stima della densità predittiva logaritmica viene chiamata ELPD-LOO. Maggiore è il punteggio ELPD-LOO di un modello, migliore è l'accuratezza predittiva out-of-sample del modello. L'errore standard di ELPD-LOO fornisce una descrizione dell'incertezza sulle prestazioni predittive per dati futuri sconosciuti. Nel confronto dei modelli, quando la differenza in ELPD-LOO è maggiore di 4, il numero di osservazioni è maggiore di 100, e in assenza di un errore di specificazione del modello, la differenza dei valori ELPD-LOO di due modelli segue la distribuzione normale. Nel confronto di modelli, un valore $|\text{elpd}_{\text{diff}}/SE_{\text{diff}}|$ maggiore di 2 può dunque essere considerato degno di menzione ("noteworthy") (Gelman et al., 2020).

Anche se la procedura descritta sopra viene correntemente usata dai ricercatori, è però necessaria una nota di cautela. Navarro (2019) ci fa notare che il problema statistico del confronto di modelli non risolve il problema scientifico della selezione di teorie. A questo proposito usa una citazione di George Box:

> Since all models are wrong the scientist must be alert to what is importantly wrong. It is inappropriate to be concerned about mice when there are tigers abroad.

La metafora delle tigri di George Box fa riferimento evidentemente all'assunzione che sta alla base delle procedure discusse in questo Capitolo, ovvero all'ipotesi che il vero meccanismo generatore dei dati sia noto e che l'unica incognita corrisponda ai parametri. Tuttavia le cose non sono così semplici: nei casi di interesse scientifico è lo stesso meccanismo generatore dei dati ad essere sconosciuto. I ricercatori non comprendono appieno i fenomeni che stanno studiando (altrimenti perché studiarli?) e qualunque descrizione formale di un fenomeno (modello) è sbagliata in un modo sconosciuto e sistematico. Di conseguenza, è "facile" fare inferenza sulla capacità predittiva del modello, ma è molto difficile fare inferenza sulla struttura causale dei fenomeni. In altre parole, se le analisi statistiche ci dicono che un modello ha una buona accuratezza predittiva, con ciò non abbiamo imparato nulla sulla struttura causale del fenomeno. Ma è anche vera l'affermazione opposta: un modello che non ha *neppure* una buona accuratezza predittiva è sicuramente inutile — non è in grado né di fare previsioni accurate né di catturare la struttura causale.

Bibliografia

Burger, E. B., & Starbird, M. (2012). *The 5 elements of effective thinking*. Princeton University Press. (Cit. a p. ix).

Gelman, A., Carlin, J. B., Stern, H. S., & Rubin, D. B. (1995). *Bayesian data analysis*. Chapman; Hall/CRC. (Cit. a p. 21).

Gelman, A., Goodrich, B., Gabry, J., & Vehtari, A. (2019). R-squared for Bayesian regression models. *The American Statistician*, 307–309 (cit. a p. 29).

Gelman, A., Hill, J., & Vehtari, A. (2020). *Regression and other stories*. Cambridge University Press. (Cit. alle pp. 15, 31).

Gelman, A., Hwang, J., & Vehtari, A. (2014). Understanding predictive information criteria for Bayesian models. *Statistics and Computing*, *24*(6), 997–1016 (cit. alle pp. 12, 14).

Hoeting, J. A., Madigan, D., Raftery, A. E., & Volinsky, C. T. (1999). Bayesian model averaging: a tutorial (with comments by M. Clyde, David Draper and EI George, and a rejoinder by the authors. *Statistical science*, *14*(4), 382–417 (cit. a p. 3).

Horn, S., & Loewenstein, G. (2021). Underestimating Learning by Doing. *Available at SSRN 3941441* (cit. a p. ix).

Johnson, A. A., Ott, M., & Dogucu, M. (2022). *Bayes Rules! An Introduction to Bayesian Modeling with R*. CRC Press. (Cit. a p. 1).

Lord, F. M. (1950). Efficiency of prediction when a regression equation from one sample is used in a new sample. *ETS Research Bulletin Series*, *1950*(2), 1–6 (cit. a p. 2).

Martin, O. A., Kumar, R., & Lao, J. (2022). *Bayesian Modeling and Computation in Python*. CRC Press. (Cit. a p. 6).

McElreath, R. (2020). *Statistical rethinking: A Bayesian course with examples in R and Stan* (2nd Edition). CRC Press. (Cit. alle pp. 3, 9, 12, 17, 22).

Myung, I. J. (2003). Tutorial on maximum likelihood estimation. *Journal of mathematical Psychology*, *47*(1), 90–100 (cit. a p. 30).

Navarro, D. J. (2019). Between the devil and the deep blue sea: Tensions between scientific judgement and statistical model selection. *Computational Brain & Behavior*, *2*(1), 28–34 (cit. alle pp. 17, 31).

Piironen, J., & Vehtari, A. (2017). Comparison of Bayesian predictive methods for model selection. *Statistics and Computing*, *27*(3), 711–735 (cit. a p. 24).

Song, Q. C., Tang, C., & Wee, S. (2021). Making sense of model generalizability: A tutorial on cross-validation in R and Shiny. *Advances in Methods and Practices in Psychological Science*, *4*(1), 2515245920947067 (cit. a p. 2).

Vehtari, A., Gelman, A., & Gabry, J. (2017). Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC. *Statistics and computing*, *27*(5), 1413–1432 (cit. alle pp. 15, 31).

Elenco delle figure

Abstract This document contains the material of the lessons of Psicometria B000286 (2021/2022) aimed at students of the first year of the Degree Course in Psychological Sciences and Techniques of the University of Florence, Italy.

Keywords Data science, Bayesian statistics.