Statističko učenje

Galić, Domagoj-Jure

SVEUČILIŠTE U ZAGREBU

PRIRODOSLOVNO–MATEMATIČKI FAKULTET

MATEMATIČKI ODSJEK

Domagoj-Jure Galić

STATISTIčKO UčENJE

Diplomski rad

Voditelj rada:
prof. dr. sc. Miljenko Huzak

Zagreb, 2018.

Ovaj diplomski rad obranjen je dana _____ pred ispitnim povjerenstvom u sastavu:

1. _____, predsjednik
2. _____, član
3. _____, član

Povjerenstvo je rad ocijenilo ocjenom _____.

Potpisi članova povjerenstva:

1. _____
2. _____
3. _____

Prvenstveno želim izraziti veliku zahvalnost svom mentoru prof. dr. sc. Miljenku Huzaku na silnoj pomoći, ne samo tijekom pisanja ovog rada nego i tijekom čitavog studija. Na svim savjetima i komentarima, na strpljenju i razumijevanju. Na beskrajnoj količini vremena koju je uložio u mene.

Zahvaljujem se svim profesorima i asistentima na Matematičkom odsjeku koji su mi pomogli u obrazovanju.

Zahvaljujem se svim kolegama i dragim prijateljima koji su bili uz mene za vrijeme svih akademskih i privatnih izazova. Ovdje moram izdvojiti Mateju Milinković, Marija Stipčića i Kristijana Kilassu Kvaternika, koji su uvijek bili tu, bez obzira na prirodu problema.

Hvala mojoj obitelji na podršci tokom čitavog studija, što su me trpjeli za vrijeme svih ispita i sa mnom proživljavali sve uspone i padove.

I na kraju, Stjepana, ovaj rad posvećujem tebi. Bez tebe bi sve ovo bilo bezvrijedno.

Sadržaj

Uvod

Današnje vrijeme obilježeno je velikom količinom podataka. Procjenjuje se da na globalnoj razini, u 2017. godini, proizvodimo oko dva i pol trilijuna ($2.5 \cdot 10^{18}$) bajtova dnevno. Od pametnih telefona do pametnih hladnjaka, okruženi smo tehnologijom, a tehnologija generira podatke. Podaci u sebi nose veliku vrijednost, stoga je od vitalne važnosti pronaći metodologije koje iz podataka mogu izvući tu vrijednost.

Teorija statističkog učenja je jedan od odgovora na gornji problem. To je relativno mlada disciplina koja vuče svoje korijene najviše iz statistike, funkcionalne analize te računarske znanosti, a primjene nalazi u mnogim granama znanosti i industrije. Statističko učenje bavi se problematikom procjene statističkog modela iz podataka, u svrhu boljeg razumijevanja mehanizma koji podatke generira ili predikcije.

Razlika između teorije statističkog učenja i metoda klasične statistike možda se najbolje vidi u odgovoru na fundamentalno pitanje svake discipline koja pokušava izvući bilo kakvu vrijednost iz podataka;

Što moramo a priori znati o nepoznatoj funkcijskoj ovisnosti kako bismo ju mogli procijeniti iz podataka?

U klasičnoj statističkoj (frekvecionističkoj, fisherovskoj ili parametarskoj) paradigmi odgovor je: jako puno. Gotovo sve zapravo. Potrebno je znati funkcijsku ovisnost, do na neki (konačan) broj parametara, te se onda ti parametri procjenjuju iz podataka.

Teorija statističkog učenja, kroz općenito paradigmu učenja, daje nešto drugačiji odgovor. Kako bi procijenili funkcijsku ovisnost dovoljno je znati neka generalna svojstva skupa funkcija kojemu prava funkcijska ovisnost pripada.

Osim sustavne teorije, statističko učenje nudi algoritme i rješenja za praktično nošenje sa svakodnevnim problemima u znanosti i industriji.

U ovom radu prednost ćemo dati praktičnim metodama i razrađenim primjerima nad apstraktom teorijom, no kako bi uopće mogli razumjeti prirodu problema, potrebno je razumjeti i nešto teorije.

A kako stara poslovica kaže:
”Ništa nije praktičnije od kvalitetne teorije.”

Poglavlje 1

Osnove statističkog učenja

1.1 Uvod

Teorija statističkog učenja može se podijeliti u mnogo kategorija. Najčešće se radi podjela na nadzirano učenje, nenadzirano učenje te polu-nadzirano učenje, ovisno o tome radimo li s označenim podacima (nadzirano učenje), neoznačenim (nenadzirano učenje) ili radimo s kombinacijom označenih i neoznačenih podataka (polu-nadzirano učenje). Također je česta podjela na probleme regresije te probleme klasifikacije. Ako podaci poprimaju neprekidne vrijednosti, onda je problem regresijske prirode, ako podaci dolaze iz diskretnog skupa vrijednosti, onda je problem klasifikacijski.

1.2 Nadzirano učenje

Podaci koji se koriste u problemima nadziranog učenja imaju pretpostavljenu formu. Postoji skup varijabli koje nazivamo *ulazne varijable*, *nezavisne varijable*, *prediktori* ili *značajke* te drugi skup varijabli koje nazivamo *izlazne*, *zavisne varijable* ili *odgovori*. Podaci obično dolaze u parovima jedne nezavisne i jedne zavisne varijable te je glavni cilj nadziranog učenja, na temelje dobivenih podataka, pronaći pravilo pomoću kojeg se na temalju nezavisne varijable može predvidjeti zavisna. Zavisne varijable se po svojoj strukturi dijele na *kvalitativne* i *kvantitativne*. Kvalitativne varijable se još nazivaju i *kategorijskim*, *diskretnim varijablama* ili *faktorima*. One poprimaju jednu od vrijednosti iz konačnog skupa vrijednosti. Kategorijske varijable dijele se dalje na *kategorijske nominalne*, one čiji skup mogućih vrijednosti nema neku vrstu uređaja, te na *kategorijske ordinalne*, one među kojima postoji definiran uređaj. Kvantitativne varijable, s druge strane, mogu poprimiti vrijednosti iz nekog beskonačnog skupa (najčešće skupa realnih brojeva neke dimenzije).

Ovisno o tipu zavisnih varijabli, problem dijelimo na *klasifikacijske*, za kvalitativne varijable, te *regresijske* za kvantitativne. Ove dvije vrste problema imaju mnogo toga za-

jedničkog te se mogu formulirati kao problem funkcijske aproksimacije.

Nezavisne varijable u ostatku rada uglavnom ćemo označavati simbolom X. Ako je X vektor, njegove komponente označavat ćemo sa X_j. Sve vektore smatrat ćemo vektorima stupcima. Kvantitativne zavisne varijable označavat ćemo simbolom Y, dok ćemo kategorijske zavisne varijable označavati s G. Opservacije ćemo označavati malim slovima, stoga ćemo i-tu opservaciju nezavisne varijable X označiti s x_i. U ovom slučaju, ako je X skalarna veličina, tada je i x_i skalarna veličina. Analogno ako je X vektor, x_i će biti vektor iste dimenzije. Matrice ćemo označavati "masnim" slovima te ćemo $N \times p$ matricu N opservacija p-vektora (vektora dimenzije p) x_i označiti simbolom $\mathbf{X}$. Vektore općenito nećemo označavati masnim slovima, osim kada imaju N komponenti. Tu konvenciju uvodimo kako bi lakše razlikovali p-vektor x_i i-te opservacije p-vektora X od N-vektora $\mathbf{x}_j$ opservacija varijable X_j kao komponente vektora X. Grafički $\mathbf{X}$ ima sljedeću formu:

$$\mathbf{X} = \begin{pmatrix} — & x_1^\tau & — \\ — & x_2^\tau & — \\ & \vdots & \\ — & x_N^\tau & — \end{pmatrix} = \begin{pmatrix} | & | & \dots & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \dots & \mathbf{x}_p \\ | & | & \dots & | \end{pmatrix}.$$

Cilj učenja za regresijski problem sada možemo neformalno opisati na sljedeći način: Za ulaznu varijablu X želimo dati dobru procjenu izlazne varijable Y koju označavamo s $\hat{Y}$. Za klasifikacijski problem formulacija je analogna, samo što procijenjujemo izlaznu varijablu G, te njenu procjenu označavamo s $\hat{G}$.

Podatke na temelju kojih radimo procjenu označavamo s $(x_i,\ y_i)$ ili $(x_i,\ g_i)$ za $i = 1, \dots, N$.

Kako bismo mogli početi razvijati modele, potrebno je prvo formalizirati ideje o kojima smo dosad govorili. Prvo ćemo promatrati slučaj s kvantitativnim izlaznim varijablama. Neka je $X \in \mathbb{R}^p$ nezavisan slučajan vektor, te neka je $Y \in \mathbb{R}$ zavisna slučajna varijabla. Neka je $f_{X,Y}$ njihova zajednička funkcija gustoće (pretpostavljamo da sve slučajne varijable i slučajni vektori s kojima radimo imaju funkcije gustoće). Cilj je pronaći funkciju $f(X)$ koja predviđa vrijednosti varijable Y za dani X. Kako bismo mogli govoriti o kvaliteti procjene, treba nam mjera udaljenosti procijenjene i stvarne vrijednosti izlazne varijable. Iz tog razloga definiramo *funkciju troška* $L(Y, f(X))$. Promatramo kvadratnu grešku, tj. $L(Y, f(X)) = (Y - f(X))^2$. Time f odabiremo kao funkciju koja minimizira srednje kvadratnu grešku (expected prediction error), pa imamo:

$$\begin{aligned} EPE(f) &= \mathbb{E}[(Y - f(X))^2] \\ &= \int (y - f(x))^2 f_{X,\,Y}(x,\ y)\ dy\ dx) \\ &= \int (y - f(x))^2 f_X(x) f_{Y|X}(y|x)\ dy\ dx) \\ &= \int \Big(\int (y - f(x))^2 f_{Y|X}(y|x)\ dy \Big)\ f_X(x)\ dx \\ &= \mathbb{E}[\ \mathbb{E}[(Y - f(X))^2 \mid X]]. \end{aligned} \tag{1.1}$$

Kako bi našli f koji minimizira izraz EPE, dovoljno je minimizirati EPE po točkama:

$$f(x) = \underset{c \in \mathbb{R}}{argmin}\ \mathbb{E}[(Y - c)^2 | X = x]. \tag{1.2}$$

Odavde vidimo da je rješenje regresijska funkcija,

$$f(x) = \mathbb{E}[Y|X = x]. \tag{1.3}$$

Dakle, problem pronalaska najbolje funkcije, zapravo je u ovom smislu problem procjene uvjetnog očekivanja.

Za klasifikacijske probleme postupamo analogno. Skup svih vrijednosti koje varijabla može poprimiti označavamo s $\mathcal{G}$. On se sastoji od pojedinačnih vrijednosti, koje označavamo s $\mathcal{G}_k$. Ako podaci mogu poprimiti K različitih vrijdnosti, odnosno vrijedi $card(\mathcal{G}) = K$, tada imamo:

$$\begin{aligned} EPE &= \mathbb{E}[L(G,\ h(X))] \\ &= \mathbb{E} \sum_{k=1}^{K} L(\mathcal{G}_k,\ h(X))\ f_{G|X}(\mathcal{G}_k|X). \end{aligned} \tag{1.4}$$

Odavde ponovo minimizacijom po točkama, dobijemo:

$$\hat{G}(x) = \underset{g \in \mathcal{G}}{argmin} \sum_{k=1}^{K} L(\mathcal{G}_k,\ g)\ \mathbb{P}(\mathcal{G}_k|X = x). \tag{1.5}$$

Ponekad ćemo radi jednostavnosti identificirati $\mathcal{G}_k$ s k.

U praksi se pokazuje da je problem procjene gornjih očekivanja relativno složen problem. S jedne strane postoje metode koje direktno procjenjuju očekivanje iz podataka, najčešće u nekoj okolini točke x. Alternativa takvim metodama je uvođenje dodatnih pretpostavki, čime zapravo definiramo regresijski model.

Svaka metoda ima svoje prednosti i mane, a razlika u pristupu nas navodi na razmatranje još jednog od ključnih pojmova: *ravnoteža pristranosti i varijance* (eng. bias-variance tradeoff).

1.3 Nenadzirano učenje

Nadzirano učenje funkcionira na principu pitanja i odgovora. Na dostupnom skupu parova nezavisne i zavisne varijable $\mathcal{T} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$, x_i interpretiramo kao pitanje, a y_i kao odgovor. Za svako pitanje x_i model daje odgovor $\hat{y}_i = \hat{f}(x_i)$, na temelju kojega modelu dajemo povratnu informaciju koliko je blizu pravom odgovoru y_i. Tom udaljenošću smatramo vrijednost $L(\hat{y}_i,\ y_i)$.

Neka je $f_{X,Y}(x, y)$ funkcija gustoće nekog slučajnog vektora $(X,\ Y)$, gdje je X nezavisna varijabla, a Y zavisna. Promatrajmo faktorizaciju te funkcije gustoće:

$$f_{X,Y}(x, y) = f_{Y|X}(y|x) \cdot f_X(x). \tag{1.6}$$

U problemima nadziranog učenja uglavnom nas ne zanima marginalna funkcija gustoće nezavisne varijable X, f_X, nego samo funkcija $f_{Y|X}(y|x)$.

U nenadziranom učenju dostupan nam je skup vrijednosti realizacija varijable X, $\mathcal{T} = \{x_1, \ldots, x_N\}$. Cilj nenadziranog učenja je odrediti svojstva distribucije od X, bez pomoći varijable Y. Odnosno, nemamo luksuz spoznaje koliko smo blizu točnom odgovoru. Dakle, zadaća nadziranog učenja je procjena uvjetne funkcije gustoće $f_{Y|X}$, dok je zadaća nenadziranog učenja procjena funkcije gustoće f_X.

U praksi je nezavisna varijabla X vrlo često zapravo $p-$dimenzionalni vektor, gdje je p relativno velik. S druge strane, varijabla Y je često jednodimenzionalna, a kada nije, njena dimenzija je uglavnom dosta manja od p.

Čest problem u nenadziranom učenju je problem *grupiranja podataka* (eng. clustering). Za dani broj grupa K potrebno je za svaku točku procijeniti kojoj grupi ta točka pripada. Primjerice, cilj je grupirati proizvode koji se zajedno kupuju ili korisnike neke usluge koji se slično ponašaju.

U slučajevima kada je dimenzija podataka velika, ponekad je korisno *reducirati dimenzionalnost* na način da projiciramo podatke u prostor manje dimenzije, ali tako da sačuvamo važne karakteristike podataka. Ovaj postupak motiviran je *hipotezom mnogostrukosti* (eng. manifold hypothesis) koja pretpostavlja da stvarni visokodimenzionalni podaci zapravo leže na mnogostrukosti manje dimenzije koja je uložena u originalni prostor.

U praksi podaci često dolaze s nedostajućim vrijednostima. Na primjer, podaci su dobiveni anketiranjem i neki ljudi nisu odgovorili na sva pitanja. Matrica dizajna koju dobijemo imat će "rupe". Metoda *matričnog dopunjavanja* (eng. matrix completion) ima za cilj odrediti vjerojatne vrijednosti nedostajućih podataka.

Ovo su samo neke od primjena nenadziranog učenja te smo ih ovdje naveli zbog potpunosti. Dalje se u radu nećemo baviti metodama nenadziranog učenja.

Poglavlje 2

Linearne metode za regresiju

2.1 Uvod

Model linearne regresije uvodi relativno jaku pretpostavku: linearnost regresijske funkcije $\mathbb{E}(Y|X = x)$ u varijablama $X_1 = x_1, \ldots, X_p = x_p$. Iako su uglavnom razvijeni u vremenu prije upotrebe računala u statistici, veliku popularnost uživaju i danas. Svoju popularnost duguju činjenici da su relativno jednostavni za interpretirati, te daju jasan odnos između ulaznih i izlaznih varijabli. Zbog svog uglavnom relativno malenog broja parametara, korisni su u slučajevima kada raspolažemo s malim brojem podataka za trening. Linearni modeli mogu se i poopćiti tako da se kao varijable modela promatraju originalne varijable na koje se djeluje nekom (nelinearnom) transformacijom. Time oslobodimo rigidnu strukturu modela, a opet zadržimo jedan dio korisnih svojstava linearnog modela. Takav pristup naziva se *metoda baznih funkcija*.

2.2 Metoda najmanjih kvadrata i linearna regresija

Želimo predvidjeti vrijednost realne izlazne varijable Y, uz danu ulaznu varijablu, vektor $x^\tau = (x_1, \ldots, x_p)$. Tada linearni regresijski model ima formu:

$$f(x) = \beta_0 + \sum_{k=1}^{p} x_k \beta_k. \tag{2.1}$$

Linearan model pretpostavlja linearnost ili približnu linearnost regresijske funkcije $\mathbb{E}(Y|X = x)$. Varijable X_k mogu biti kvantitativne vrijednosti, transformacije kvantitativnih vrijednosti (logaritam, drugi korijen, polinom, spline ...), "dummy" varijable, interakcije među varijablama itd. Neovisno o strukturi i obliku varijabli X_k, bitno je da je model linearan u parametrima β_k.

Cilj je iz skupa podataka $(x_1, y_1), \ldots (x_N, y_N)$ procijeniti parametre modela $(\beta_1, \ldots \beta_p)$, s time da je svaki $x_k^\tau = (x_{k1}, \ldots, x_{kp})$ $p-$dimenzionalni vektor vrijednosti nezavisne vektorske varijable X. Najčešća metoda za procjenu parametra je metoda najmanjih kvadrata, kojom se odabiru parametri $\beta^\tau = (\beta_0, \beta_1, \ldots, \beta_p)$, tako da se minimizira srednje kvadratna greška (eng. residual sum of squares):

$$\begin{aligned} RSS(\beta) &= \sum_{k=1}^{N} (y_k - f(x_k))^2 \\ &= \sum_{k=1}^{N} \left(y_k - \beta_0 - \sum_{j=1}^{p} x_{kj}\beta_j \right)^2. \end{aligned} \tag{2.2}$$

Statistički gledano ova metoda ima smisla ako su podaci $(x_k,\ y_k)$ realizacije slučajnog uzorka distribuiranog kao slučajan vektor $(X,\ Y)$, za $k = 1, \ldots, N$, odnosno, ako su $(Y_k \mid X_k = x_k)$ uvjetno nezavisni za $k = 1, \ldots N$ s time da slučajna varijabla $(Y \mid X = x_0)$ ima funkciju gustoće

$$f_{Y \mid X}(y \mid x_0) = \frac{f_{X,\,Y}(x_0, y)}{f_X(x_0)},$$

za sve x_0, takve da je $f_X(x_0) > 0$.

Sada dolazimo na praktično pitanje, kako minimizirati (2.2)? Neka je $\mathbf{X}$ $N \times (p+1)$ matrica opservacija, čiji je prvi stupac stupac jedinica, a svaki redak je vrijednost nezavisne varijable X (uz jedinicu kao prvi element). Takvu matricu nazivamo *matricom dizajna*. Zavisne varijable predstavljamo $N-$vektorom opaženih vrijednosti $\mathbf{y}$. Sada srednje kvadratnu grešku možemo prikazati vektorski, kao

$$RSS(\beta) = (\mathbf{y} - \mathbf{X}\beta)^\tau (\mathbf{y} - \mathbf{X}\beta). \tag{2.3}$$

Dobivena funkcija je kvadratna u $p+1$ parametru i njenim deriviranjem po parametru β dobijemo

$$\begin{aligned} \frac{\partial RSS}{\partial \beta} &= -2\,\mathbf{X}^\tau(\mathbf{y} - \mathbf{X}\beta) \\ \frac{\partial^2 RSS}{\partial \beta \partial \beta^\tau} &= 2\,\mathbf{X}^\tau\mathbf{X}. \end{aligned} \tag{2.4}$$

Izjednačavanjem (2.4) s nulom tražimo kritične točke te ako uz to pretpostavimo da je $\mathbf{X}$ punog ranga tada jednadžba

$$\mathbf{X}^\tau(\mathbf{y} - \mathbf{X}\beta) = 0 \tag{2.5}$$

ima jedinstveno rješenje

$$\hat{\beta} = (\mathbf{X}^\tau\mathbf{X})^{-1}\mathbf{X}^\tau\mathbf{y}. \tag{2.6}$$

Sada predikciju u točki x_0 dobijemo kao $\hat{f}(x_0) = (1 \;\; x_{01} \;\; x_{02} \;\; \ldots \;\; x_{0p})^\tau \hat{\beta}$. Procijenjenu vrijednost izlazne varijable, za trening skup podataka dobijemo s

$$\begin{aligned} \hat{\mathbf{y}} &= \mathbf{X}\,\hat{\beta} \\ &= \mathbf{X}(\mathbf{X}^\tau\mathbf{X})^{-1}\mathbf{X}^\tau\mathbf{y}. \end{aligned} \tag{2.7}$$

Označimo matricu $\mathbf{X}(\mathbf{X}^\tau\mathbf{X})^{-1}\mathbf{X}^\tau$ sa $\mathbf{H}$. Srednje kvadratnu grešku još se može zapisati kao

$$RSS(\beta) = \|\mathbf{y} - \mathbf{X}\,\beta\|^2. \tag{2.8}$$

Odavde vidimo da možemo minimizirati RSS, tako da odaberemo $\hat{\beta}$ na način da je vektor reziduala, $\mathbf{y} - \hat{\mathbf{y}}$, okomit na prostor razapet vektorima stupcima matrice $\mathbf{X}$. Odnosno, procijenjeni vektor $\hat{\mathbf{y}}$ je ortogonalna projekcija vektora $\mathbf{y}$ na taj prostor. Stoga je matrica $\mathbf{H}$ ortogonalni projektor na prostor razapet stupcima matrice $\mathbf{X}$.

Do sada smo radili pod pretpostavkom da je matrica $\mathbf{X}$ punog ranga, no to ne mora biti slučaj. Ako matrica $\mathbf{X}$ nije punog ranga, tada je matrica $\mathbf{X}^\tau\mathbf{X}$ singularna. U tom slučaju rješenje jednadžbe (2.5) nije jedinstveno. Vektor $\hat{\mathbf{y}} = \mathbf{X}\,\hat{\beta}$ je i dalje jedinstvena projekcija vektora $\mathbf{y}$ na prostor stupaca matrice $\mathbf{X}$, samo prikaz te projekcije preko stupaca matrice $\mathbf{X}$ nije jedinstven. Budući da do nepunog ranga dolazi zbog linearne zavisnosti između stupaca matrice, problem možemo rješiti izbacivanjem jednog ili više redundantnih stupaca.

Kako bi mogli nešto više reći o statističkim obilježjima vektora parametara β, moramo uvesti pretpostavke na distribuciju koja generira podatke. Pretpostavimo da su vrijednosti $(Y_k \mid X_k = x_k)$ međusobno nekorelirane i da imaju konstantu, ne-nul, varijancu σ^2. Sada kovarijacijska matrica parametara procijenjenih metodom najmanjih kvadrata glasi

$$Var(\hat{\beta}) = (\mathbf{X}^\tau\mathbf{X})^{-1}\,\sigma^2. \tag{2.9}$$

Varijanca σ^2 je u pravilu nepoznata pa ju procijenjujemo prema formuli

$$\hat{\sigma}^2 = \frac{1}{N-p-1}\sum_{k=1}^{N}(y_k - \hat{y}_k)^2. \tag{2.10}$$

U nazivniku se nalazi broj $N - p - 1$ umjesto N, kako bi procjenitelj bio nepristran.

Kako bismo mogli dobiti distribuciju parametara i provoditi statističke testove odnosno, određivati pouzdane intervale, potrebne su nam jače pretpostavke. Zato pretpostavljamo da su podaci generirani modelom

$$Y = \beta_0 + \sum_{k=1}^{p} x_k\,\beta_k + \varepsilon, \tag{2.11}$$

gdje je greška slučajna varijabla $\varepsilon \sim N(0,\ \sigma^2)$. Iz (2.11) slijedi

$$\hat{\beta} \sim N(\beta,\ (\mathbf{X}^\tau \mathbf{X})^{-1}\ \sigma^2). \tag{2.12}$$

Također vrijedi

$$(N - p - 1)\ \hat{\sigma}^2 \sim \sigma^2\ \chi^2_{N-p-1}, \tag{2.13}$$

te da su $\hat{\sigma}^2$ i $\hat{\beta}$ nezavisne slučajne varijable.

Sada možemo testirati hipotezu da je neki parametar $\beta_k = 0$ koristeći $z-$test i testnu statistiku

$$Z_k = \frac{\hat{\beta}_k}{\hat{\sigma}\ \sqrt{v_k}}, \tag{2.14}$$

gdje je v_k, k-ti dijagonalni element matrice $(\mathbf{X}^\tau \mathbf{X})^{-1}$. Hipoteze su

$$\begin{aligned} H_0 :&\qquad \beta_k = 0 \\ H_1 :&\qquad \beta_k \neq 0. \end{aligned}$$

Pod nultom hipotezom Z_k ima t distribuciju s $N - (p + 1)$ stupnjeva slobode. Budući da je za velike N t distribucija približno jedinična normalna, često za gornji test koristimo normalnu aproksimaciju kvantila t distribucije.

Ponekad nije smisleno, ili čak nije moguće izbaciti samo jednu varijablu iz modela. U slučaju kada određujemo značajnost cijele grupe varijabli, koristimo $F-$test. Ako imamo originalni model s $p_O + 1$ parametrom i iz njega izveden reducirani model, dobiven izjednačavanjem nekog broja parametara originalnog modela s nulom (te nam ostane $p_R + 1$ ne-nul parametara), tada računamo srednje kvadratnu grešku za originalni model RSS_O i za reducirani model RSS_R te $F-$statistika glasi

$$F = \frac{(RSS_R - RSS_O)/(p_O - p_R)}{RSS_O/(N - p_O - 1)}. \tag{2.15}$$

Pod nultom hipotezom da su parametri koje smo izjednačili s nulom stvarno nula, F statistika ima F distribuciju sa $p_R - p_0$ i $N - p_R - 1$ stupnjeva slobode. $F-$statistike jest omjer razlike varijabilnosti reduciranog i originalnog model koje nisu objašnjene modelom u odnosu na originalnu varijabilnost neobjašnjenu modelom. Odnosno, koliko se neobjašnjena varijabilnost promjeni dodavanjem grupe varijabli od interesa.

2.3 Odabir podskupa varijabli

U problemima s velikim brojem ulaznih varijabli ponekad nam je u interesu odrediti neki manji podskup koji pokazuje najveći utjecaj na varijablu odgovora. Ima različitih razloga

za takvu odluku. S velikim brojem nezavisnih varijabli također imamo i velik parametarski prostor. Ponekad smanjivanjem tog prostora možemo smanjiti i varijancu parametara dobivenih metodom najmanjih kvadrata. Također, kod modela sa stotinama ili tisućama parametara, vrijeme i točnost samog računa treba uzeti u obzir. Osim toga, također dobivamo i na interpretabilnosti modela. S malim brojem parametara koji jako utječu na zavisnu varijablu, iako vjerojatno žrtvujemo preciznost modela, možemo dobiti "širu sliku" kako mehanizam koji generira podatke djeluje. Ponekad je to žrtva koju se isplati podnjeti.

Odabir najboljeg podskupa

Metoda odabira najboljeg podskupa, za svaki $k \leq p$, vraća podskup veličine k, koji ima najmanju predikcijsku grešku. Budući da algoritam treba provjeriti više manje sve mogućnosti, izrazito je nepraktičan za velike p. Bitno je napomenuti da najbolji podskup veličine $k + 1$ ne mora sadržavati najbolji podskup veličine k. Konkretno, najbolji podskup veličine 2 ne mora sadržavati varijablu koja je u najboljem podskupu veličine 1.

Ova metoda nam također ne sugerira način odabira parametra k. Njegov odabir zahtjeva postizanje ravnoteže između pristranosti, varijance i kompleksnosti modela. O metodama koje se koriste za postizanje te ravnoteže bit će riječi kasnije.

Odabir unaprijed i unazad

Umjesto prolaska kroz sve podskupove parametara možemo tražiti sistematičniji način prolaska. Metoda odabira unaprijed je jedan od načina na koji to postižemo. Počinjemo sa slobodnim članom, te modelu dodajemo jednu po jednu nezavisnu varijablu, tako da svaka varijabla koju dodamo, od svih preostalih varijabli, najviše smanji grešku. U ovom slučaju, kao i u slučaju metode odabira najboljeg podskupa, greška ne mora nužno značiti srednje kvadratna greška. Koriste se različite funkcije greške koje vrlo često, osim dijela koji mjeri "udaljenost" modela od podataka, penalizira i kompleksnost modela. Metoda odabira unaprijed pripada klasi pohlepnih algoritama, te on daje familiju ugnježđenih modela, gdje za razliku od metode najboljeg podskupa, model za podskup veličine $k + 1$ sadrži sve varijable modela za podskup veličine k. Iako, u odnosu na metodu odabira najboljeg podskupa, metoda odabira unaprijed daje sub-optimalno rješenje. Ona se može (u realnom vremenu) izračunati za proizvoljno velik p, čak i za p veći od N.

Metoda odabira unatrag pak kreće s druge strane. Ona počinje od punog modela, te izbacuje iz modela nezavisnu varijablu čija je vrijednost testne statistike $z-$testa minimalna. Također kao i kod prethodne dvije metode, može se koristiti i alternativan kriterij za izbacivanje varijabli iz modela.

Također postoje i hibridne metode koje u svakom koraku računaju odabir unaprijed i unatrag te biraju najbolji, a često se koristi i $F-$statistika kako bi se dodale značajne i

maknule beznačajne varijable u modelu.

Bitno je i napomenuti da ponekad nema smisla izbaciti samo jednu varijablu, nego treba promatrati grupu kao cjelinu. To je posebno vrijedi za "dummy" varijable, koje služe za reprezentaciju kategorijskih varijabli s više nivoa.

2.4 Sažimanje parametara

Metoda odabira podskupa predstavlja diskretan postupak. Varijablu ili ostavljamo u modelu, ili izbacujemo iz modela. Takav postupak značajno doprinosi povećanju varijabilnosti te samim time ne smanjuje predikcijsku grešku. Metode sažimanja parametara, s druge strane, pomalo smanjuju vrijednosti parametara prema nuli. Na taj način metode imaju jednu vrstu neprekidnosti zbog čega im je varijabilnost u pravilu manja od metoda odabira podskupa.

Hrbat-regresija

Hrbat-regresija (eng. ridge regression) sažima parametre tako što penalizira veličinu parametara, efektivno ih smanjujući po apslutnoj vrijednosti. Funkcija troška koju minimiziramo u hrbat-regresiji ima sljedeću formu:

$$RSS(\beta,\ \lambda) = \sum_{k=1}^{N} \left(y_k - \beta_0 - \sum_{j=1}^{p} x_{kj}\,\beta_j\right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2\ . \tag{2.16}$$

Budući je ideja hrbat-regresije sažimanje parametara, ne bi nas trebala iznenaditi činjenica, da metoda nije invarijantna na skaliranje. Stoga je uobičajena praksa standardizacija podataka (nezavisnih varijabli) prije procjene parametara. Važno je primijetiti da smo u regularizacijskom pribrojniku izostavili koeficijent β_0. Regularizacijom slobodnog člana dobili bi da metoda ovisi o odabiru ishodišta za prikaz Y. Odnosno, kada bi svakoj realizaciji zavisne varijable dodali konstantan član c, to ne bi rezultiralo pomakom predikcija za c.

Prvo procjenjujemo β_0 i to sa $\bar{y} = \frac{1}{N}\sum_{k=1}^{N} y_k$. Ostale koeficjente procjenjujemo bez slobodnog člana. Pretpostavimo da su ulazni podaci standardizirani, stavimo ih u $N \times p$ matricu $\mathbf{X}$ (za razliku od matrice dizajna od prije koja je imala $p+1$ stupac), vektor $\mathbf{y}$ definiramo kao centrirani vektor varijabli odgovora, $\mathbf{y}_i = y_i - \bar{y}$, a vektor $\beta = (\beta_1,\ \ldots,\ \beta_p)^\tau$ (bez koeficijenta β_0 koji smo već procijenili).

Sada funkciju troška možemo zapisati u vektorskom obliku

$$RSS(\beta,\ \lambda) = (\mathbf{y} - \mathbf{X}\,\beta)^\tau\,(\mathbf{y} - \mathbf{X}\,\beta) + \lambda\,\beta^\tau\,\beta. \tag{2.17}$$

Deriviranjem po β, te izjednačavanjem dobivene derivacije s nulom, vidimo da se minimum postiže za

$$\hat{\beta}^{rg} = (\mathbf{X}^\tau\,\mathbf{X} + \lambda I)^{-1}\,\mathbf{X}^\tau\mathbf{y}, \tag{2.18}$$

gdje je I $p \times p$ matrica identiteta. Dijagonalan član λI čini gornju matricu regularnom čak i kada je $\mathbf{X}^\tau \mathbf{X}$ singularna, stoga je $\hat{\beta}^{rg}$ uvijek jedinstven.

Promatranjem *dekompozicije singularnih vrijednsoti* (SVD, eng. singular value decomposition) matrice standardiziranih ulaza $\mathbf{X}$, možemo dobiti bolju intuiciju što se događa s parametrima u hrbat-regresiji. SVD $N \times p$ matrice $\mathbf{X}$ ima oblik

$$\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^\tau, \tag{2.19}$$

gdje su $\mathbf{U}$ i $\mathbf{V}$, $N \times p$ i $p \times p$, matrice s ortonormiranim stupcima. $\mathbf{D}$ je $p \times p$ dijagonalna matirca s padajućim elementima na dijagonali $d_1 \geq d_2 \geq \cdots \geq d_p \geq 0$, koje zovemo *singularnim vrijednostima* matrice $\mathbf{X}$.

Sada za projicirane vrijednosti $\hat{\mathbf{Y}}^{ls}$, za koje su parametri procijenjeni običnom linearnom regresijom, imamo

$$\begin{aligned} \mathbf{X}\hat{\beta}^{ls} &= \mathbf{X}(\mathbf{X}^\tau \mathbf{X})^{-1}\mathbf{X}^\tau \mathbf{y} \\ &= \mathbf{U}\mathbf{U}^\tau \mathbf{y} \\ &= \sum_{j=1}^{p} \mathbf{u}_j \mathbf{u}_j^\tau \mathbf{y}, \end{aligned} \tag{2.20}$$

gdje su $\mathbf{u}_j$ stupci matrice $\mathbf{U}$. Budući je $\mathbf{U}$ matrica s ortogonalnim stupcima, vrijedi da je $\mathbf{U}^\tau = \mathbf{U}^{-1}$, pa iz (2.20) vidimo da je $\mathbf{X}\hat{\beta}^{ls}$ zapravo $\mathbf{y}$ raspisan u bazi koju čine stupci matrice $\mathbf{U}$.

Za hrbat-regresiju, rješenje ima sljedeću formu:

$$\begin{aligned} \mathbf{X}\hat{\beta}^{rg} &= \mathbf{X}\,(\mathbf{X}^\tau \mathbf{X} + \lambda I)^{-1}\,\mathbf{X}^\tau\,\mathbf{y} \\ &= \mathbf{U}\,\mathbf{D}\,(\mathbf{D}^2 + \lambda I)^{-1}\,\mathbf{D}\,\mathbf{U}^\tau\,\mathbf{y} \\ &= \sum_{j=1}^{p} \left(\frac{d_j^2}{d_j^2 + \lambda}\right) \mathbf{u}_j\,\mathbf{u}_j^\tau\,\mathbf{y}. \end{aligned} \tag{2.21}$$

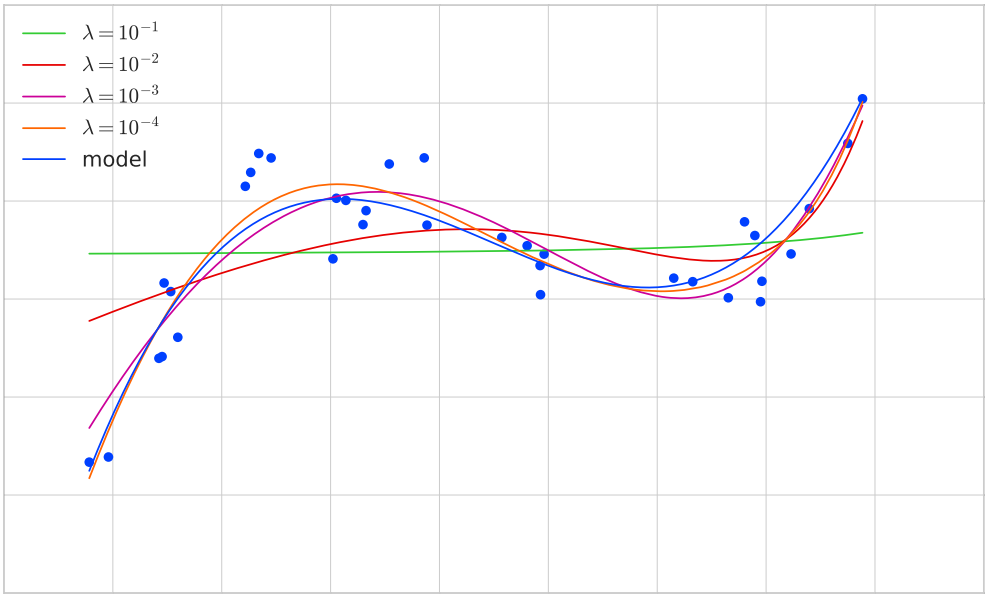
Budući da je $\lambda > 0$, vidimo da u hrbat regresiji koeficijente dobijemo kao prikaz vektora $\mathbf{y}$ u bazi koju čine stupci matrice $\mathbf{U}$, skaliranu s faktorom $d_j^2/(d_j^2+\lambda) \leq 1$, gdje je d_j varijabilnost u smjeru $\mathbf{v}_j$, $j-$tog stupca matrice $\mathbf{V}$. Budući da je $(d_j)_j$ padajući niz nenegativnih brojeva, vidimo da će se kasnije komponente sve više reducirati.

Za hrbat-regresiju definiramo *efektivan broj stupnjeva slobode* s

$$\begin{aligned} df(\lambda) &= tr[\mathbf{X}(\mathbf{X}^\tau \mathbf{X} + \lambda I)^{-1}\,\mathbf{X}^\tau] \\ &= \sum_{j=1}^{p} \frac{d_j^2}{d_j^2 + \lambda}. \end{aligned} \tag{2.22}$$

Sada umjesto u terminima parametra λ možemo govoriti u terminima efektivnog broja stupnjeva slobode. Vidimo da za parametre procijenjene običnom metodom najmanjih kvadrata, efektivni broj stupnjeva slobode je jednak p, odnosno efektivni broj stupnjeva slobode i broj stupnjeva slobode se podudaraju.

Efektivni broj stupnjeva slobode će biti važan kasnije, kada ćemo uspoređivati kompleksnosti modela.



Slika 2.1: Regularizacija polinoma reda deset hrbat-regresijom s različitim regularizacijskim parametrima

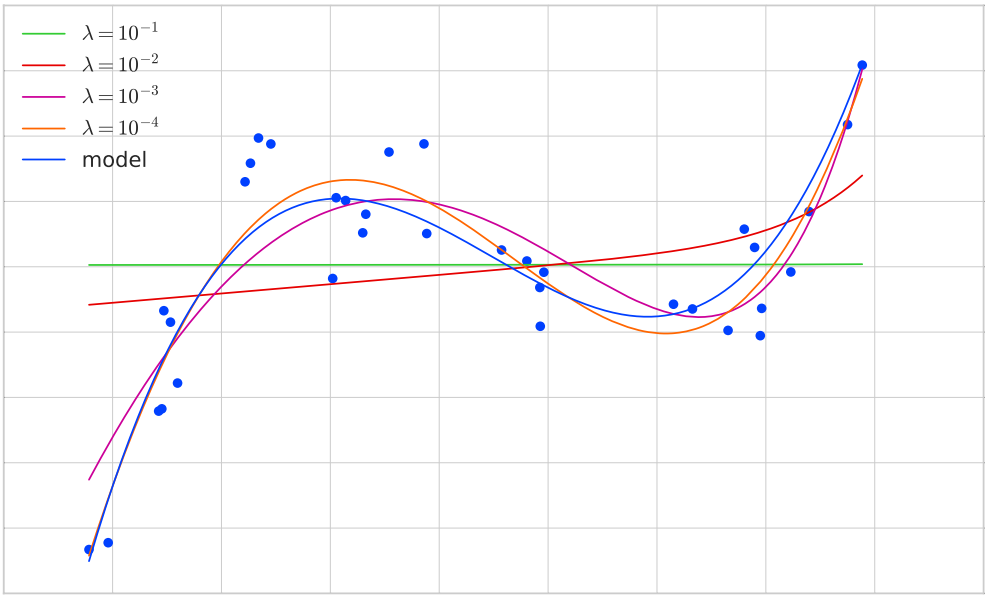
LASSO regresija

LASSO (eng. Least Absolute Shrinkage and Selection Operator) kao i hrbat-regresija spada u kategoriju metoda sažimanja. Od hrbat-regresije razlikuje se u formi regulari-

zacijskog člana. Za LASSO funkcija greške ima sljedeću formu:

$$RSS(\beta,\ \lambda) = \frac{1}{2}\sum_{k=1}^{N}\left(y_k - \beta_0 - \sum_{j=1}^{p} x_{kj}\,\beta_j\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j|. \tag{2.23}$$

Vidimo da u se u LASSO-u parametar β u regularizacijskom članu pojavljuje kroz svoju L^1 normu, dok u hrbat-regresiji kroz svoju L^2 normu. Kao posljedica načina na koji se provodi regularizacija u LASSO-u, problem je nelinearan u y_k i zbog toga ne postoji zatvorena forma za parametre u LASSO-u. Kao posljedica promjene norme preko koje se vrši sažimanje parametara, mijenjaju se i karakteristike parametara koje dobijemo u ovisnosti o parametru λ. Za razliku od hrbat-regresije, ako u LASSO-u vrijednost parametra λ postavimo dovoljno visoko, neki koeficijenti će ispasti nula. Zato se LASSO može smatrati neprekidnom varijantom odabira najboljeg podskupa. Kao i u svim ostalim regularizacijskim metodama parametar λ treba odabrati kako bi minimizirali, unaprijed definiranu, funkciju troška.



Slika 2.2: Regularizacija polinoma reda deset LASSO regresijom s različitim regularizacijskim parametrima

Poglavlje 3

Linearne metode za klasifikaciju

3.1 Uvod

Klasifikacijske metode uglavnom rade na način da particioniraju prostor s kojeg dolaze ulazne varijable. Ovisno o predikcijskoj funkciji te granice mogu biti različitog oblika. Skupina metoda za koje su te *granice odluke* (eng. decision boundaries) linearne, nazivamo *linearnim metodama za klasifikaciju*. Metode se sastoje od modeliranja *diskriminacijske funkcije* $\delta_k(x)$, za svaku klasu k te klasifikaciju vrijednosti x u klasu s najvećom vrijednosti diskriminacijske funkcije. Metode koje modeliraju vjerojatnosti pripadnosti pojedinoj klasi $\mathbb{P}(G = k \mid X = x)$, također spadaju u ovu skupinu metoda (jer klasificiramo točku u onu klasu s najvećom vjerojatnosti). U tom slučaju vrijedi $\delta_k(x) = \mathbb{P}(G = k \mid X = x)$.

Da bi granica odluke bila linearna, nije nužno zahtijevati da je diskriminacijska funkcija linearna. Dovoljno je zahtijevati da je neka njena monotona transformacija linearna.

3.2 Klasifikacija pomoću regresije

Za reprezentaciju pripadnosti u pojedinu kategoriju koristimo *indikatorske varijable*. Dakle ako varijabla može poprimiti K različitih vrijednosti, koristiti ćemo K indikatora $Y_1, \ldots, Y_k$, takvih da vrijedi

$$Y_k = \begin{cases} 1, & G = k, \\ 0, & \text{inače.} \end{cases} \tag{3.1}$$

Za svaku vrijednost izlazne varijable pripadne indikatorske funkcije složimo u vektor $Y = (Y_1, \ldots, Y_k)^\tau$[1]. Za N izlaznih vrijednosti složimo sve vektore u $N \times K$ matricu koju nazivamo *indikatorskom matricom odgovora* $\mathbf{Y}$. Sada postavljamo problem kao regresijski, odnosno

[1] U engleskoj literaturi to se često naziva *one-hot encoding*.

tražimo matricu $\hat{B}$ koja minimizira srednje kvadratnu grešku. Rješenje je dano s

$$\hat{\mathbf{B}} = (\mathbf{X}^\tau \mathbf{X})^{-1} \mathbf{X}^\tau \mathbf{Y}. \tag{3.2}$$

Važno je primijetiti kako ovdje imamo $p+1$ parametar za svaku varijablu $\mathbf{y}_k$, dakle matrica $\hat{B}$ je matrica reda $(p+1) \times K$.

Sada novu vrijednost x klasificiramo tako da prvo izračunamo vrijednost

$$\hat{f}(x)^\tau = (1,\ x^\tau) \cdot \hat{\mathbf{B}}. \tag{3.3}$$

Zatim vrijednost x dodijelimo klasi $\hat{G}(x)$ tako da vrijedi:

$$\hat{G}(x) = \underset{k \in \{1,\ldots,K\}}{argmax} \left(\hat{f}(x)\right)_k. \tag{3.4}$$

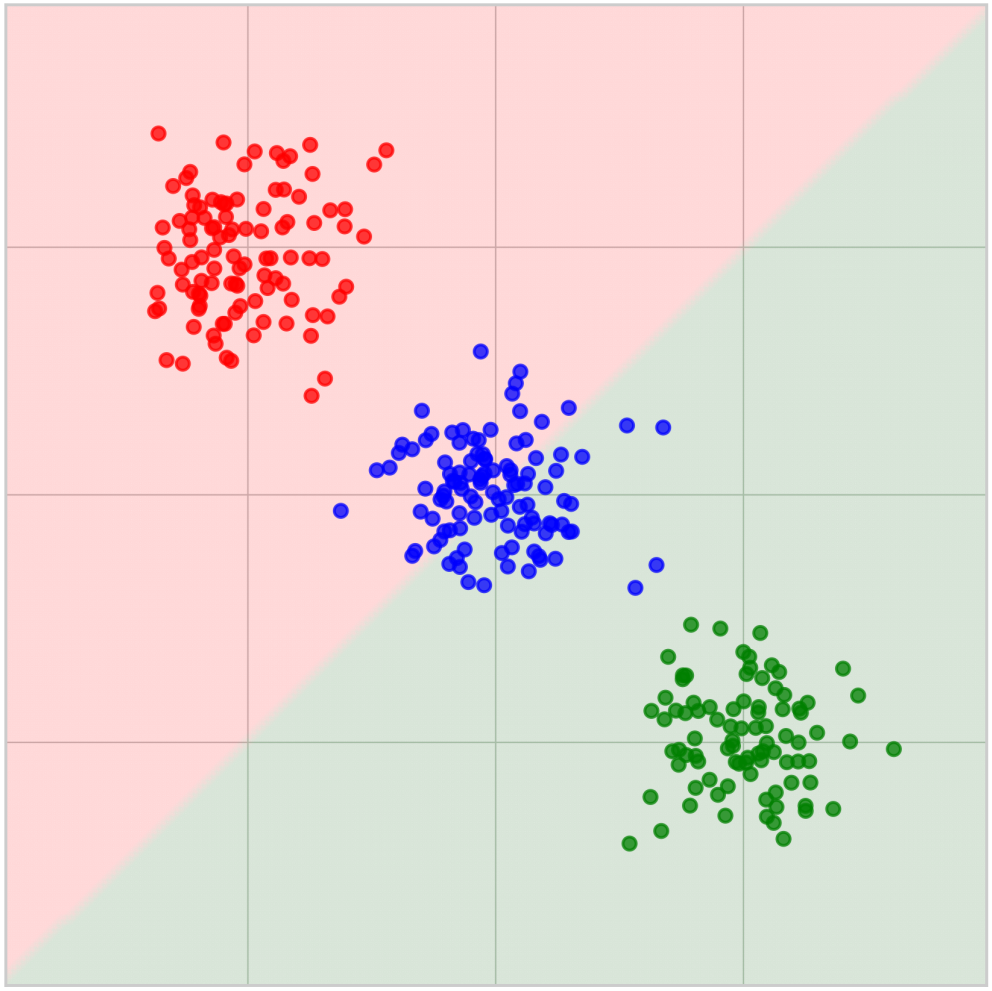
Ovdje sa $\left(\hat{f}(x)\right)_k$ označavamo $k-$tu koordinatu vektora $\hat{f}(x)$. Odnosno, x klasificiramo u klasu k ako je k koordinata vektora $\hat{f}(x)$ koja ima najveću vrijednost među svim koordinatama.

Intuicija iza ovog postupka je sljedeća: Za slučajnu varijablu Y_k vrijedi:

$$\mathbb{E}[Y_k \mid X = x] = \mathbb{P}(G = k \mid X = x). \tag{3.5}$$

Postavlja se pitanje koliko dobro možemo aproksimirati vjerojatnost linearnom funkcijom. Vrijednosti $\left(\hat{f}(x)\right)_k$ mogu biti negativne ili veće od 1, što znači da nismo dobili vjerojatnost. No to ne znači da model nije upotrebljiv. Ako koristimo transformirane ulazne podatke $h(x)$, možemo značajno poboljšati kvalitetu procjene.

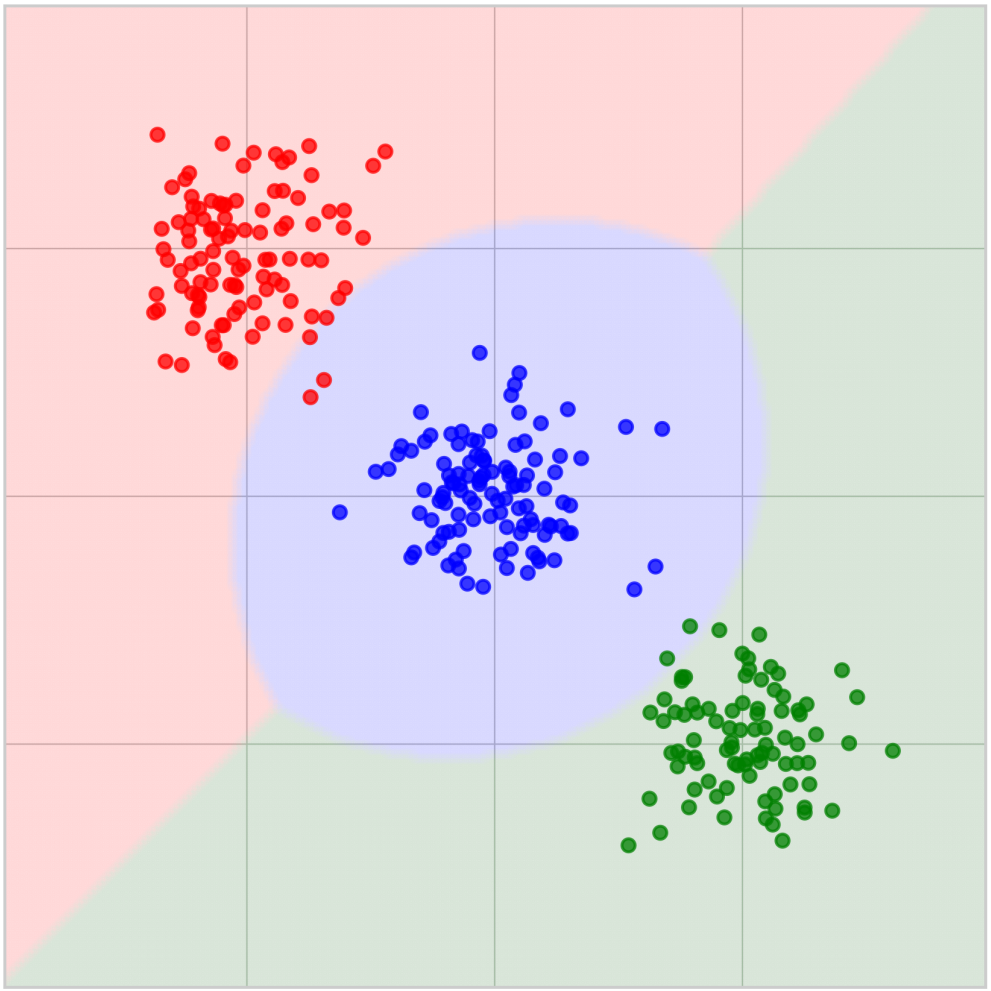
Za $K \geq 3$ treba biti oprezan jer, zbog rigidne strukture linearnog modela, može doći do *maskiranja* podataka. Odnosno, slučaj kada imamo linearno separabilne klase (klase koje se mogu odijeliti hiperravninom), ali model ne prepozna razliku između dvije ili više klasa.



Slika 3.1: Klasa označena plavom bojom je maskirana. Vrijednost diskriminacijske funkcije plave klase nikada ne dominira nad onom crvene ili zelene klase.

Slika 3.1 pokazuje maskiranje podataka upotrebom linearne regresijske funkcije. Do-

davanjem kvadratnog člana dobijemo sliku 3.2 na kojoj vidimo relativno dobru separaciju klasa.



Slika 3.2: Dodavanjem kvadratnog člana dobivamo relativno dobru separaciju, više nema maskiranja.

3.3 Linearna diskriminacijska analiza

Označimo s f_k funkciju gustoće varijable X uvjetovane na $G = k$ (znači, u slučaju da je X diskretna slučajna varijabla vrijedi $f_k(x) = \mathbb{P}(X = x \mid G = k)$) i neka je $\pi_k := \mathbb{P}(G = k)$. Uz pomoć Bayesovog teorema imamo:

$$\mathbb{P}(G = k \mid X = x) = \frac{f_k(x) \cdot \pi_k}{\sum_{l=1}^{K} f_l(x) \cdot \pi_l}. \tag{3.6}$$

Vidimo da je znati funkcije f_k gotovo ekvivalentno poznavanju veličine $\mathbb{P}(G = k \mid X = x)$.

Linearnu diskriminacijsku analizu (LDA) dobijemo kada pretpostavimo da je svaka klasa distribuirana normalno pri čemu su im kovarijacijske matrice jednake:

$$f_k(x) = \frac{1}{(2\pi)^{p/2} \cdot \det(\Sigma)^{1/2}} e^{-\frac{1}{2}(x-\mu_k)^\tau \, \Sigma^{-1} \, (x-\mu_k)}. \tag{3.7}$$

Kada uspoređujemo dvije klase dovoljno je gledati log-omjer njihovih uvjetnih vjerojatnosti. Imamo:

$$\log \frac{\mathbb{P}(G = k \mid X = x)}{\mathbb{P}(G = l \mid X = x)} = \log \frac{f_k(x)}{f_l(x)} + \log \frac{\pi_k}{\pi_l} \tag{3.8}$$

$$\begin{aligned} &= \log \frac{\pi_k}{\pi_l} - \frac{1}{2}(\mu_k + \mu_l)^\tau \, \Sigma^{-1} \, (\mu_k - \mu_l) \\ &\quad + x^\tau \, \Sigma^{-1} (\mu_k - \mu_l). \end{aligned} \tag{3.9}$$

Pretpostavka jednakosti kovarijacijskih matrica dovela je do toga da se dosta članova pokratilo, te nam je ostala funkcija linearna u x. Za svaki k definiramo *linearnu diskriminacijsku funkciju* kao:

$$\delta_k(x) = x^\tau \, \Sigma^{-1} \, \mu_k - \frac{1}{2} \mu_k^\tau \, \Sigma^{-1} \, \mu_k + \log \pi_k. \tag{3.10}$$

Sada iz (3.9) slijedi:

$$\log\frac{\pi_k}{\pi_l} - \frac{1}{2}(\mu_k+\mu_l)^\tau\,\Sigma^{-1}\,(\mu_k-\mu_l) + x^\tau\,\Sigma^{-1}(\mu_k-\mu_l) = \tag{3.11}$$

$$= \log\pi_k - \log\pi_l - \frac{1}{2}\Big[\mu_k^\tau\,\Sigma^{-1}\,(\mu_k-\mu_l) + \mu_l^\tau\,\Sigma^{-1}(\mu_k-\mu_l)\Big] + x^\tau\,\Sigma^{-1}\,\mu_k - x^\tau\,\Sigma^{-1}\,\mu_l \tag{3.12}$$

$$= \Big(\log\pi_k + x^\tau\,\Sigma^{-1}\,\mu_k\Big) - \Big(\log\pi_l + x^\tau\,\Sigma^{-1}\,\mu_l\Big) - \frac{1}{2}\Big[\mu_k^\tau\,\Sigma^{-1}\,\mu_k^\tau - \cancel{\mu_k^\tau\,\Sigma^{-1}\,\mu_l^\tau} + \cancel{\mu_l^\tau\,\Sigma^{-1}\,\mu_k^\tau} - \mu_l^\tau\,\Sigma^{-1}\,\mu_l^\tau\Big] \tag{3.13}$$

$$= \log\pi_k + x^\tau\,\Sigma^{-1}\,\mu_k - \frac{1}{2}\mu_k^\tau\,\Sigma^{-1}\,\mu_l + \log\pi_l + x^\tau\,\Sigma^{-1}\,\mu_l - \frac{1}{2}\mu_l^\tau\,\Sigma^{-1}\,\mu_l \tag{3.14}$$

$$= \delta_k(x) - \delta_l(x). \tag{3.15}$$

Vidimo da maksimizacija uvjetne vjerojatnosti pripadnosti klasi daje isti rezultat kao i odabrati k za koji je $\delta_k(x)$ maksimalna.

U praksi ne znamo parametre normalne razdiobe iz koje podaci dolaze, stoga ih moramo procijeniti iz trening-skupa podataka:

- $\hat{\pi}_k = N_k/N$, gdje je N_k broj trening-podataka iz klase k,
- $\hat{\mu}_k = \sum_{g_j=k} x_j/N_k$,
- $\hat{\Sigma} = \sum_{k=1}^{K}\sum_{g_j=k}(x_j-\hat{\mu}_k)(x_j-\hat{\mu}_k)^\tau/(N-K)$.

Ako ne pretpostavimo jednakost svih kovarijacijskih matrica normalnih razdioba, ne dođe do takvog kraćenja kao u (3.9) te nam ostaje kvadratni član od x. U tom slučaju dobijemo *kvadratnu diskriminacijsku funkciju* (QDA)

$$\delta_k(x) = -\frac{1}{2}\log\det(\Sigma_k) - \frac{1}{2}(x-\mu_k)^\tau\,\Sigma^{-1}\,(x-\mu_k) + \log\pi_k. \tag{3.16}$$

Procjene parametara u QDA su slične kao u LDA, ali u QDA se svaka kovarijacijska matrica mora procjenjivati posebno čime se značajno povećava broj parametara.

3.4 Logistička regresija

Ideja modela logističke regresije je modelirati vjerojatnosti $\mathbb{P}(G=k \mid X=x)$ kao glatko transformirane linearne funkcije, ali za razliku od klasifikacije pomoću regresije u cjelini 3.2 želimo da su vrijednosti uvijek pozitivne te da je suma svih vjerojatnosti jednaka 1.

Iz naših zahtjeva jasno je da ne možemo modelirati direktno linearnim funkcijama. Zato log-omjer vjerojatnosti modeliramo linearnim funkcijama. Model ima sljedeći oblik:

$$\begin{aligned}
\log \frac{\mathbb{P}(G = 1 \mid X = x)}{\mathbb{P}(G = K \mid X = x)} &= \beta_{10} + \beta_1^\tau \, x \\
\log \frac{\mathbb{P}(G = 2 \mid X = x)}{\mathbb{P}(G = K \mid X = x)} &= \beta_{20} + \beta_2^\tau \, x \\
&\vdots \\
\log \frac{\mathbb{P}(G = K-1 \mid X = x)}{\mathbb{P}(G = K \mid X = x)} &= \beta_{(K-1)0} + \beta_{K-1}^\tau \, x.
\end{aligned} \tag{3.17}$$

Izbor nazivnika u (3.17) je proizvoljan, vrijedi:

$$\log \frac{\mathbb{P}(G = k \mid X = x)}{\mathbb{P}(G = K \mid X = x)} = \beta_{k0} + \beta_k^\tau \, x \iff \tag{3.18}$$

$$\log \mathbb{P}(G = k \mid X = x) = \log \mathbb{P}(G = K \mid X = x) + \beta_{k0} + \beta_k^\tau \, x. \tag{3.19}$$

Odavde vidimo da vrijedi:

$$\log \frac{\mathbb{P}(G = k \mid X = x)}{\mathbb{P}(G = l \mid X = x)} = \log \mathbb{P}(G = k \mid X = x) - \log \mathbb{P}(G = l \mid X = x) \tag{3.20}$$

$$\begin{aligned}
&= \cancel{\log \mathbb{P}(G = K \mid X = x)} + \beta_{k0} + \beta_k^\tau \, x \\
&\quad - \cancel{\log \mathbb{P}(G = K \mid X = x)} - \beta_{l0} - \beta_l^\tau \, x
\end{aligned} \tag{3.21}$$

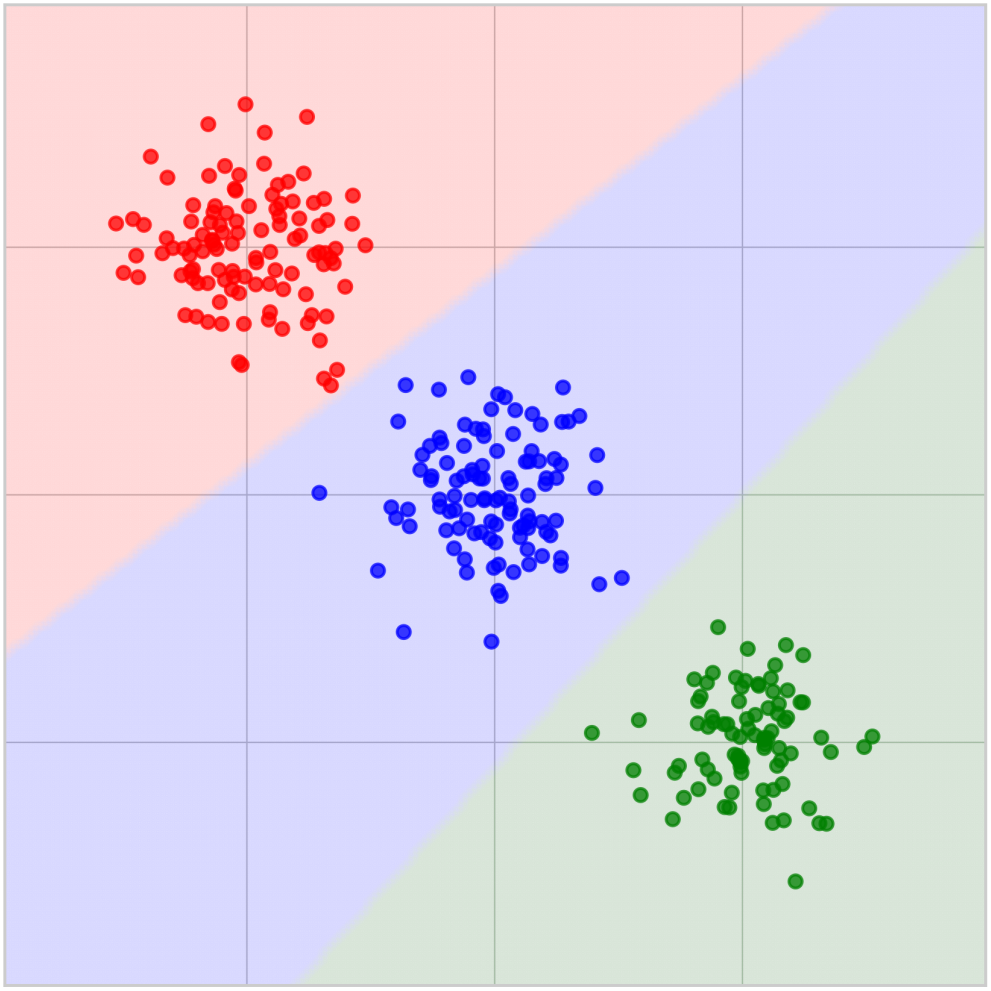
$$= (\beta_{k0} - \beta_{l0}) + (\beta_k - \beta_l)^\tau \, x. \tag{3.22}$$

Dakle dobili smo isti model. Da bismo vidjeli kako je suma svih vjerojatnosti jednaka 1, dovoljno je primijetiti da vrijedi:

$$\begin{aligned}
\mathbb{P}(G = k \mid X = x) &= \frac{e^{\beta_{k0} + \beta_k^\tau \, x}}{1 + \sum_{l=1}^{K-1} e^{\beta_{l0} + \beta_l^\tau \, x}}, \quad k = 1, \ldots, K-1 \\
\mathbb{P}(G = K \mid X = x) &= \frac{1}{1 + \sum_{l=1}^{K-1} e^{\beta_{l0} + \beta_l^\tau \, x}}.
\end{aligned} \tag{3.23}$$

Tvrdnja je sada očita.

Označimo s $\theta = (\beta_{10}, \beta_1, \ldots, \beta_{(K-1)0}, \beta_{K-1})$ vektor parametara logističkog modela, te označimo $p_k(x; \theta) = \mathbb{P}(G = k \mid X = x)$. Za slučaj $K = 2$ model je posebno jednostavan jer tada se sastoji od jedne linearne funkcije.



Slika 3.3: Logistička regresija primijenjena na isti skup podataka kao i na slikama 3.1 i 3.2. Vidimo separaciju linearnim granicama, bez maskiranja.

Parametri logističke regresije obično se procjenjuju metodom maksimalne vjerodostoj-

nosti. Dakle, tražimo parametre θ koji maksimiziraju izraz

$$l(\theta) = \sum_{j=1}^{N} \log p_{g_j}(x_j;\ \theta), \tag{3.24}$$

gdje je $p_k(x_j;\ \theta) = \mathbb{P}(G = k \mid X = x_j;\ \theta)$. Stavimo $\phi_k^\tau = (\beta_{k0},\ \beta_k^\tau)$ i dodajmo 1 na prvu koordinatu vektora x_j.

Sada vrijedi:

$$l(\theta) = \sum_{j=1}^{N} \Big(\sum_{k=1}^{K} \mathbb{1}_{\{g_j=k\}} \log p_k(x_j;\ \theta) \Big) \tag{3.25}$$

$$\begin{aligned} &= \sum_{j=1}^{N} \Big[\sum_{k=1}^{K-1} \mathbb{1}_{\{g_j=k\}} \Big(\phi_k^\tau\, x_j - \log\big(1 + \sum_{l=1}^{K-1} e^{\phi_l^\tau\, x_j}\big) \Big) \\ &\quad - \mathbb{1}_{\{g_j=K\}} \log\big(1 + \sum_{l=1}^{K-1} e^{\phi_l^\tau\, x_j}\big) \Big] \end{aligned} \tag{3.26}$$

$$\begin{aligned} &= \sum_{j=1}^{N} \Big[\sum_{k=1}^{K-1} \mathbb{1}_{\{g_j=k\}} \cdot \phi_k^\tau\, x_j - \sum_{k=1}^{K-1} \mathbb{1}_{\{g_j=k\}} \log\big(1 + \sum_{l=1}^{K-1} e^{\phi_l^\tau x_j}\big) \\ &\quad - \mathbb{1}_{\{g_j=K\}} \log\big(1 + \sum_{l=1}^{K-1} e^{\phi_l^\tau x_j}\big) \Big] \end{aligned} \tag{3.27}$$

$$= \sum_{j=1}^{N} \Big[\sum_{k=1}^{K-1} \mathbb{1}_{\{g_j=k\}} \cdot \phi_k^\tau\, x_j - \log\big(1 + \sum_{l=1}^{K-1} e^{\phi_l^\tau x_j}\big) \cdot \Big(\sum_{s=1}^{K} \mathbb{1}_{\{g_j=s\}} \Big) \Big] \tag{3.28}$$

$$= \sum_{j=1}^{N} \Big[\sum_{k=1}^{K-1} \mathbb{1}_{\{g_j=k\}} \cdot \phi_k^\tau\, x_j - \log\big(1 + \sum_{l=1}^{K-1} e^{\phi_l^\tau x_j}\big) \Big]. \tag{3.29}$$

Ovdje treba primijetiti da u (3.28) vrijedi

$$\sum_{s=1}^{K} \mathbb{1}_{\{g_j=s\}} = \mathbb{1}_{\{g_j \in \{1,\ldots,\,K\}\}} = 1. \tag{3.30}$$

U slučaju $K = 2$ stvari se dosta pojednostavljuju: imamo $y_j = 1$ kada je $g_j = 1$ i $y_j = 0$, kada je $g_j = 2$. Dakle, vrijedi $y_j = \mathbb{1}_{\{g_j=1\}}$. Sada imamo:

$$l(\theta) = \sum_{j=1}^{N} \Big[\mathbb{1}_{g_j=1} \cdot \phi^\tau\, x_j - \log\big(1 + e^{\phi^\tau\, x_j}\big) \Big] \tag{3.31}$$

$$= \sum_{j=1}^{N} \Big[y_j \cdot \phi^\tau\, x_j - \log\big(1 + e^{\phi^\tau\, x_j}\big) \Big]. \tag{3.32}$$

Da bi maksimizirali funkciju vjerodostojnosti, tražimo njenu derivaciju po ϕ i izjednačavamo je s nulom.

$$\frac{\partial l(\phi)}{\partial \phi} = \sum_{j=1}^{N} x_j(y_j - p(x_j;\ \phi)) = 0, \tag{3.33}$$

gdje je $p(x;\ \phi) = p_1(x;\ \phi)$.

Vidimo da je jednadžba (3.33) nelinearna u ϕ. Jednadžba nema analitičko rješenje te je potrebna neka numerička metoda za njeno rješavanje. Mi se ovdje time nećemo baviti.

Spomenimo još da se, kao i linearna regresija, logistička regresija može regularizirati i to na gotovo analogan način. Razlika je u toliko što se ne minimizira funkcija troška nego se maksimizira funkcija cilja. Tada umjesto dodavanja regularizacijske funkcije, ovdje ju oduzimamo. Tražimo parametre β_0, β koji zadovoljavaju sljedeći izraz:

$$\max_{\beta_0,\beta} \left\{ \sum_{j=1}^{N} \left[y_j(\beta_0 + \beta^\tau\ x_j) - \log(1 + e^{\beta_0+\beta^\tau\ x_j}) \right] - \lambda \sum_{j=1}^{p} |\beta_j| \right\}. \tag{3.34}$$

Ili možemo problem zapisati u terminima funkcije troška:

$$\min_{\beta_0,\beta} \left\{ -\sum_{j=1}^{N} \left[y_j(\beta_0 + \beta^\tau\ x_j) - \log(1 + e^{\beta_0+\beta^\tau\ x_j}) \right] + \lambda \sum_{j=1}^{p} |\beta_j| \right\}. \tag{3.35}$$

Za traženje optimuma (3.34) i (3.35) regularizirane logističke regresije također su potrebne numeričke metode.

Poglavlje 4

Ocjena i odabir modela

4.1 Uvod

Zanima nas koliko je kvalitetna predikcija prethodno treniranog modela na novom skupu podataka. Kada govorimo o toj kvaliteti, govorimo o sposobnosti *generalizacije* modela. Kada je predikcija dobra kažemo da model dobro generalizira.

Generalizacija modela je jako bitna stavka u odabiru modela, jer nam govori kakve rezultate možemo očekivati od odabranog modela. Kada odabiremo model iz neke familije modela, bitno je odrediti kriterij, metriku kojom mjerimo uspješnost generalizacije modela kako bi mogli uspoređivati više modela.

4.2 Pristranost, varijanca i kompleksnost

Neka je Y zavisna varijabla, X nezavisna varijabla (ili vektor) te neka je $\hat{f}(X)$ model, procijenjen na nekom skupu $\mathcal{T}$ podatka za trening. Greška između predikcije dane modelom i opažene vrijednosti dana je s $L(Y,\ \hat{f}(X))$.

Definiramo *testnu grešku* ili *grešku generalizacije* kao predikcijsku grešku modela nad novim podacima, nezavisnim od skupa za trening

$$Err_{\mathcal{T}} = \mathbb{E}[L(Y,\ \hat{f}(X)) \mid \mathcal{T}]. \tag{4.1}$$

Skup za trening $\mathcal{T}$ je fiksiran i greška generalizacije ovisi o njemu.

S druge strane možemo promatrati i *očekivanu predikcijsku grešku*

$$Err = \mathbb{E}[\mathbb{E}[L(Y,\ \hat{f}(X)) \mid \mathcal{T}]]. \tag{4.2}$$

Očekivanu predikcijsku grešku možemo smatrati očekivanom greškom generalizacije ako skup za trening odabiremo slučajno.

Cilj nam je procijeniti grešku generalizacije $Err_{\mathcal{T}}$, no u praksi je jednostavnije efikasno procijeniti očekivanu predikcijsku grešku Err.

Trening-greška je prosjek troška na trening-skupu,

$$\overline{err} = \frac{1}{N}\sum_{k=1}^{N} L(y_k,\ \hat{f}(x_k)), \tag{4.3}$$

gdje je skup za trening dan sa $\mathcal{T} = \{(x_k, y_k)\ :\ k = 1, \ldots, N\}$. Želimo odrediti očekivanu testnu grešku procijenjenog modela $\hat{f}$, no kako kompleksnost modela raste, tako raste i njegova sposobnost prilagođavanja kompleksnijoj strukturi podataka (bila ta struktura vjerna stvarnoj funkcijskoj ovisnosti ili nastala djelovanjem nekog slučajnog efekta). Zato rastom kompleksnosti modela smanjujemo pristranost, ali povećavamo varijancu. Cilj nam je naći model "srednje" kompleksnosti koji minimizira očekivanu testnu grešku. Nažalost, trening-greška nije dobar procjenitelj testne greške, kao ni sposobnosti modela da generalizira. Kod modela veće kompleksnosti javlja se *preprilagođenost* (eng. overfitting) ili *pretreniranost* (eng. overtraining) modela što je situacija u kojoj je model naučio dio slučajnog šuma kao funkcijsku ovisnost[1]. Bitno je napomenuti kako na umu možemo imati dva različita cilja:

> **Odabir modela** - procjena kvalitete više modela kako bismo među njima odabrali najbolji
>
> **Ocjena modela** - kada smo već odabrali model procjenjujemo njegovu grešku generalizacije na novim podacima.

Ovisno o volumenu podataka kojim raspolažemo različito postupamo. U slučaju da raspolažemo (dovoljno) velikim volumenom podataka, najbolje rješenje za oba problema je podijeliti podatke na tri dijela: skup za trening, skup za provjeru (eng. validation) i testni skup. Na trening-skupu treniramo modele, na skupu za provjeru procjenujemo predikcijske greške modela te na testnom skupu procjenujemo grešku generalizacije odabranog modela. Ako nemamo dovoljno podataka, uglavnom se koriste metode koje aproksimiraju korak provjere, bilo analitičkim metodama, bilo višestrukim korištenjem istih uzoraka.

4.3 Ravnoteža pristranosti i varijance

I dalje pretpostavljamo oblik $Y = f(X) + \varepsilon$ gdje je $\mathbb{E}[\varepsilon] = 0$ i $Var(\varepsilon) = \sigma_\varepsilon^2$. Zanima nas očekivana testna greška regresijskog modela u točki $X = x_0$, s kvadratnom funkcijom

[1] Većina autora koristi pojam preprilagođenosti modela za fenomen koji smo opisali, no neki autori taj fenomen nazivaju pretreniranošću modela, a model nazivaju preprilagođenim ako postoji model manje kompleksnosti jednake testne greške. Mi nećemo raditi tu distinkciju.

troška.
Vrijedi:

$$Err(x_0) = \mathbb{E}[(Y - \hat{f}(x_0))^2 \mid X = x_0] \tag{4.4}$$

$$= \mathbb{E}[((f(X) - \hat{f}(x_0)) + \varepsilon)^2 \mid X = x_0] \tag{4.5}$$

$$\begin{aligned} &= \mathbb{E}[\varepsilon^2 \mid X = x_0] + 2 \cdot \mathbb{E}[\varepsilon \cdot (f(X) - \hat{f}(x_0)) \mid X = x_0] \\ &+ \mathbb{E}[(f(X) - \hat{f}(x_0))^2 \mid X = x_0] \end{aligned} \tag{4.6}$$

$$\begin{aligned} &= \sigma_\varepsilon^2 + \mathbb{E}[f^2(X) \mid X = x_0] - 2 \cdot \mathbb{E}[f(X) \cdot \hat{f}(x_0) \mid X = x_0] \\ &+ \mathbb{E}[\hat{f}^2(x_0) \mid X = x_0] \end{aligned} \tag{4.7}$$

$$= \sigma_\varepsilon^2 + f^2(x_0) - 2 \cdot f(x_0) \cdot \mathbb{E}[\hat{f}(x_0)] + \mathbb{E}[\hat{f}^2(x_0)] \tag{4.8}$$

$$= \sigma_\varepsilon^2 + (\mathbb{E}[\hat{f}(x_0) - f(x_0)])^2 + Var(\hat{f}(x_0)) \tag{4.9}$$

gdje je $\mathbb{E}[\varepsilon \cdot (f(X) - \hat{f}(x_0)) \mid X = x_0] = 0$ u retku (4.6) zbog nezavisnosti između slučajne varijable ε i $\hat{f}(x_0)$.

U jednadžbi (4.9) imamo tri člana. Prvi, σ_ε^2, je ireducibilna greška. Ona je intrinzična u problemu, predstavlja varijancu oko očekivanja $f(x_0)$ i ne ovisi o odabiru modela. Drugi, $(\mathbb{E}[\hat{f}(x_0) - f(x_0)])^2$, je kvadrat pristranosti modela. Pristranost modela je veličina koja govori za koliko se očekivanje modela razlikuje od stvarnog očekivanja. Zadnji član predstavlja varijancu modela u danoj točki. U pravilu kako raste kompleksnost modela, tako pristranost pada, no varijanca modela također raste. Razlog toge je što model uči slučajni šum, koji ne predstavlja dio stvarne funkcijske ovisnosti.

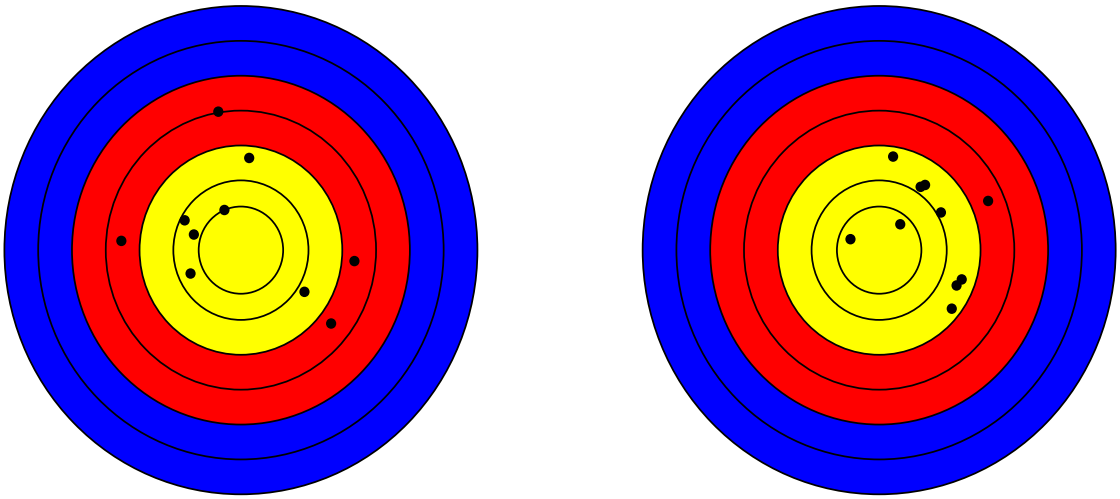
Omjer pristranosti i varijance možda je najlakše shvatiti kroz primjer. Zamislimo da imamo dva strijelca. Svaki strijelac ispaliti će deset strijela na metu. Prvi strijelac je nepreciznij, ali drugi ima krivo podešen ciljnik (za potrebe primjera pretpostavimo da to ne zna i da obojica gađaju centar). Prvi strijelac ima malu pristranost, ali veliku varijancu, a drugi ima veliku pristranost i malu varijancu.

Koji od njih dvojice je bolji ovisi o odnosu pristranosti i varijance svakoga.

4.4 Optimizam trening greške

Neka je dan trening-skup $\mathcal{T} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$. Označimo sa (X^0, Y^0) nove podatke (ne nalaze se u trening-skupu). Tada je testna greška modela $\hat{f}_\mathcal{T}$ treniranog na $\mathcal{T}$ dana sa:

$$Err_\mathcal{T} = \mathbb{E}[L(Y^0, \hat{f}_\mathcal{T}(X^0)) \mid \mathcal{T}] \tag{4.10}$$



Prvi strijelac Drugi strijelac

Slika 4.1: Prikaz realizacije gađanja strijelaca.

Trening greška

$$\overline{err} = \frac{1}{N}\sum_{k=1}^{N} L(y_k,\ \hat{f}(x_k)) \tag{4.11}$$

će u pravilu biti manja od testne greške $Err_{\mathcal{T}}$ zato što koristimo iste podatke za trening i procjenu greške. Budući da su parametri odabrani tako da minimiziraju (trening) grešku, procjena greške s podacima iz istog trening-skupa bit će preoptimistična.

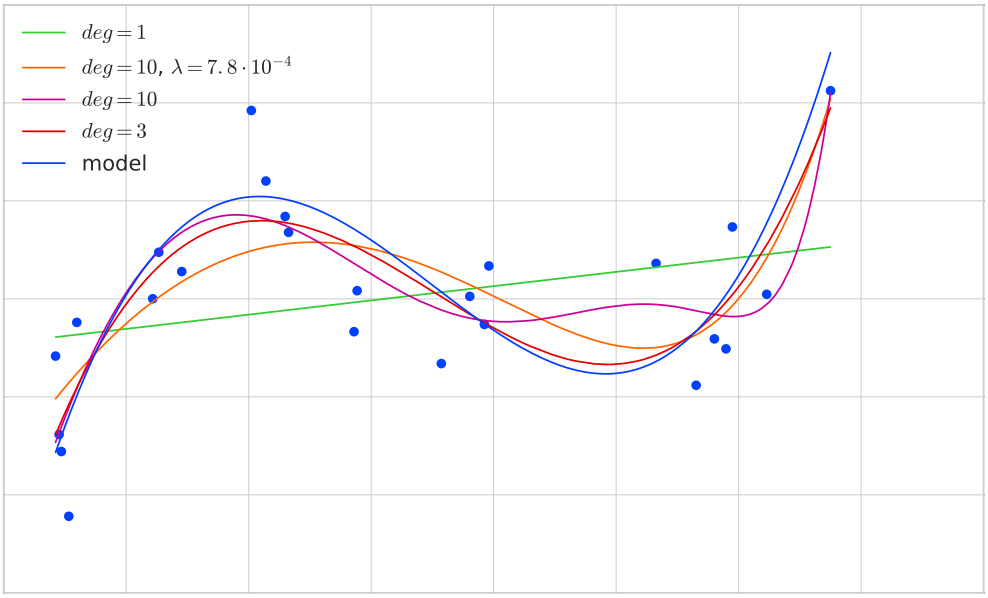
Testnu grešku možemo smatrati *izvan-uzoračkom* greškom, utoliko što se ulazne varijable generalizacijske greške ne moraju podudarati s ulaznim varijablama podataka u trening skupu. Zato definiramo *unutar-uzoračku* grešku (eng. in-sample error) kao:

$$Err_{in} = \frac{1}{N}\sum_{k=1}^{N} \mathbb{E}[L(Y_k^0,\ \hat{f}(x_k)) \mid \mathcal{T}] \tag{4.12}$$

Ovdje Y_k^0 znači da je varijabla Y nezavisna od trening-skupa (ne nalazi se u njemu), ali promatramo njenu realizaciju u točki x_k, trening-skupa $\mathcal{T}$.

Sada definiramo *optimizam* kao razliku unutar-uzoračke greške i trening-greške, odnosno:

$$op = Err_{in} - \overline{err}. \tag{4.13}$$



Slika 4.2: Prikaz modela različitih stupnjeva i regularizacija.

Optimizam je najčešće pozitivan jer je trening greška manja od unutar-uzoračke greške.

Način na koji sada možemo procjenjivati testnu grešku je tako da procijenimo optimizam te ga dodamo trening grešci $\overline{err}$. Alternativno postoje metode koje direktno procjenjuju izvan-uzoračku (očekivanu testnu grešku) Err.

4.5 Unakrsna provjera

Unakrsna provjera (eng. cross-validation) vjerojatno je najjednostavnija i najupotrebljivanija metoda za procjenu testne greške. Metoda direktno procjenjuje testnu grešku $Err = \mathbb{E}[L(Y, \hat{f}(X))]$.

Unakrsna provjera se koristi u slučajevima kada ne raspolažemo dovoljnom količinom podataka da bi dio ostavili sa strane, za naknadnu procjenu testne greške, pa podatke trebamo reciklirati. Na taj način svaki podatak se koristi za treniranje i za testiranje modela. Podijelimo podatke na K (otprilike) jednakih dijelova, odaberemo jedan od tih podskupova te treniramo model na svim osim tog podskupa.

Neka je $\kappa : \{1, \ldots, N\} \mapsto \{1, \ldots, K\}$ indeksna funkcija koja za svaki indeks j pripadne točke (x_j, y_j) vraća indeks podskupa u koji smo ju stavili, npr. $\kappa(j) = k$. Neka je $\hat{f}^{-k}$ model treniran na svim osim k-tom podskupu podataka. Tada je procjena predikcijske greške metodom unakrsne provjere:

$$CV(\hat{f}) = \frac{1}{N} \sum_{j=1}^{N} L(y_j,\ \hat{f}^{-\kappa(j)}(x_j)). \tag{4.14}$$

Metoda unakrsne provjere jako je korisna za određivanje *hiperparametara* modela. Hiperparametri modela su parametri koji se moraju definirati prije početka samog procesa učenja. Na primjer, parametri regularizacije su hiperparametri modela. Ako je $\hat{f}(\cdot,\ \lambda)$ familija modela indeksirana hiperparametrom λ, za tu familiju definiramo funkciju koja procjenjuje predikcijsku grešku u ovisnosti o parametru sa:

$$CV(\hat{f},\ \lambda) = \frac{1}{N} \sum_{j=1}^{N} L(y_j,\ \hat{f}^{-\kappa(j)}(x_j,\ \lambda)). \tag{4.15}$$

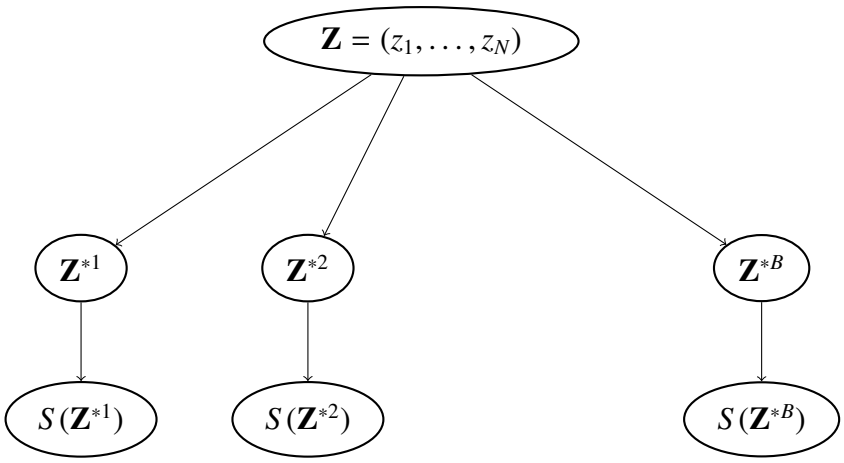
Naposljetku odabiremo vrijednost hiperparametra $\hat{\lambda}$, tako da minimizira vrijednost funkcije $CV(\hat{f},\ \lambda)$.

Ostaje otvoreno pitanje odabira K particija skupa podataka. Unakrsna provjera može biti računski skupa za velik K. Zahtijeva K treniranja modela. Također, zbog sličnosti skupova za trening, varijabilnost metode može biti velika, no s malim K, metoda može imati pristranost.

4.6 Metode bootstrapa

Označimo skup podataka za trening sa $\mathbf{Z} = (z_1,\ z_2,\ \ldots,\ z_N)$, gdje su $z_j = (x_j,\ y_j)$. Sada generiramo *bootstrap* tako da iz $\mathbf{Z}$ na slučajan način, s jednakom vjerojatnošću da izaberemo svaki, odaberemo N podataka. Dakle, bootstrap uzorak ima istu duljinu N kao i originalni uzorak. Postupak ponovimo B puta i dobijemo B bootstrap uzoraka $\mathbf{Z}^{*1},\ \ldots, \mathbf{Z}^{*B}$. Ako je $S(\mathbf{Z})$ veličina izračunata iz podataka, tada pomoću bootstrapa možemo procijeniti distribuciju od $S(\mathbf{Z})$.

No i dalje nije jasno kako upotrijebiti bootstrap metodu na procjenu predikcijske greške. Jedna od metoda je trenirati model na bootstrap uzorku te izračunati grešku na originalnom



Slika 4.3: Shematski prikaz procesa izvlačenja bootstrap uzoraka.

trening-skupu podataka. Označimo li s $\hat{f}^{*b}(x_j)$ predikciju u točki x_j modela treniranog na bootstrap uzorku b, tada procjena predikcijske greške iznosi:

$$\widehat{Err}_{boot} = \frac{1}{B}\frac{1}{N}\sum_{b=1}^{B}\sum_{j=1}^{N} L(y_j,\ \hat{f}^{*b}(x_j)). \tag{4.16}$$

Iz formule (4.16) vidimo da predikcijsku grešku procjenjujemo kao prosjek testnih grešaka modela treniranih na bootstrap uzorcima i evaluiranih na originalnom trening-skupu.

U ovom slučaju bootstrap uzorke smatramo trening-skupovima, a originalni trening-skup tretiramo kao testni skup. Problem nam predstavlja činjenica da su ti skupovi vjerojatno jako slični, zbog čega bi nam procjena greške mogla ispasti manja od stvarne vrijednosti. To možemo izbjeći tako da koristimo isti trik kao i kod unakrsne provjere, to jest radimo predikciju samo na onim vrijednostima koje se ne pojavljuju u bootstrap uzorku na kojemu je model treniran. Neka je C^{-j} skup svih indeksa svih bootstrap uzoraka koji ne sadrže točku $(x_j,\ y_j)$. Tada revidiranu procjenu računamo kao:

$$\widehat{Err}^{(1)} = \frac{1}{N}\sum_{j=1}^{N}\frac{1}{card(C^{-j})}\sum_{b\in C_{-j}} L(y_j,\ \hat{f}^{*b}(x_j)). \tag{4.17}$$

Ovdje $card(C^{-j})$ predstavlja kardinalni broj skupa C^{-j}. Formula (4.17) govori da testnu grešku procjenjujemo kao prosjek svih prosjeka predikcijskih grešaka modela treniranih na bootstrap uzorcima i testiranim na točkama koje se u tim uzorcima pojavljuju. Dakle točno ono što smo i htjeli postići.

No, time nismo u potpunosti riješili problem, samo smo ga potencijalno zamijenili drugim problemom. Riječ je o tome što smo u svakom od uprosječenih testnih grešaka koje

računamo smanjili broj točaka na kojima računamo grešku. Označimo li s P vjerojatnost da se točka $(x_j,\ y_j)$ nalazi u bootstrap uzorku $\mathbf{Z}^{*b}$, vrijedi:

$$P = 1 - \left(1 - \frac{1}{N}\right)^N \tag{4.18}$$

$$= 1 - \left(1 + \frac{(-1)}{N}\right)^N \tag{4.19}$$

$$\approx 1 - \frac{1}{e} \tag{4.20}$$

$$\approx 0.632. \tag{4.21}$$

Ovdje (4.18) vrijedi jer je vjerojatnost da točku $(x_j,\ y_j)$ ne odaberemo na bilo koje mjesto u uzorku $\mathbf{Z}^{*b}$ jednaka $1 - \frac{1}{N}$ te je odabir točaka u bootstrap uzorku nezavisan. Približna jednakost (4.20) vrijedi na limesu jer je

$$\lim_{n\to\infty}\left(1 + \frac{x}{n}\right)^n = e^x.$$

Dakle, vjerojatnost svake točke da se pojavi u bilo kojem bootstrap uzorku je otprilike 0.632. U bootstrap uzorcima se u prosjeku pojavljuje $N \cdot 0.632$ različitih vrijednosti (u ovom slučaju dvije točke $(x_i,\ y_i)$ i $(x_j,\ y_j)$ smatramo različitima ako je $i \neq j$). Stoga će procjenitelj $\widehat{Err}^{(1)}$ biti pristran otprilike kao i metoda unakrsne provjere za $K = 2$.

Tome se može doskočiti *".632 procjeniteljem"* definiranim kao:

$$\widehat{Err}^{(.632)} = .368 \cdot \overline{err} + .632 \cdot \widehat{Err}^{(1)}. \tag{4.22}$$

Međutim niti .632 procjenitelj nije bez mana, te se zna dogoditi da daje jako pogrešne procjene u slučaju kada je model pretreniran. Moguće je poboljšati .632 procjenitelj na način da uzmemo u obzir stupanj pretreniranosti modela, no mi se time ovdje nećemo baviti.

Poglavlje 5

Umjetne neuronske mreže

5.1 Uvod

Umjetne neuronske mreže su statistički modeli koji su inspirirani načinom na koji funkcionira mozak. Uživaju veliku popularnost u zadnjih nekoliko godina, a glavni razlog tome je njihova uspješna primjena u širokom spektru područja i povećanje performansi računala do razine gdje je trening modela postao moguć u realnom vremenu. Od klasifikacije slika i prepoznavanja lica do modeliranja kreditnog rizika, umjetne neuronske mreže spadaju među najrasprostranjenije modele statističkog učenja danas.

5.2 Sastavni dijelovi

Osnovna gradivna jedinica neuronske mreže je model *neurona*. Neuron se sastoji od *vektora težina*, *konstantnog člana* i *aktivacijske funkcije*. Neka je x vektor ulaznih podataka, $x = (x_1, \ldots, x_P)^\tau$. Izlaznu vrijednost računamo tako da prvo skalarno pomožimo x s vektorom težina $w = (w_1, \ldots, w_P)^\tau$ te pribrojimo konstantan član b. Označimo privremenu vrijednost sa z, dakle vrijedi:

$$z = \langle w|x\rangle + b = w^\tau x + b. \tag{5.1}$$

Nakon toga izlaznu vrijednost a dobijemo tako da na privremenu vrijednost z djelujemo (nelinearnom) aktivacijskom funkcijom g.

$$a = g(z) = g(w^\tau x + b). \tag{5.2}$$

Povijesno, aktivacijska funkcija je bila step funkcija

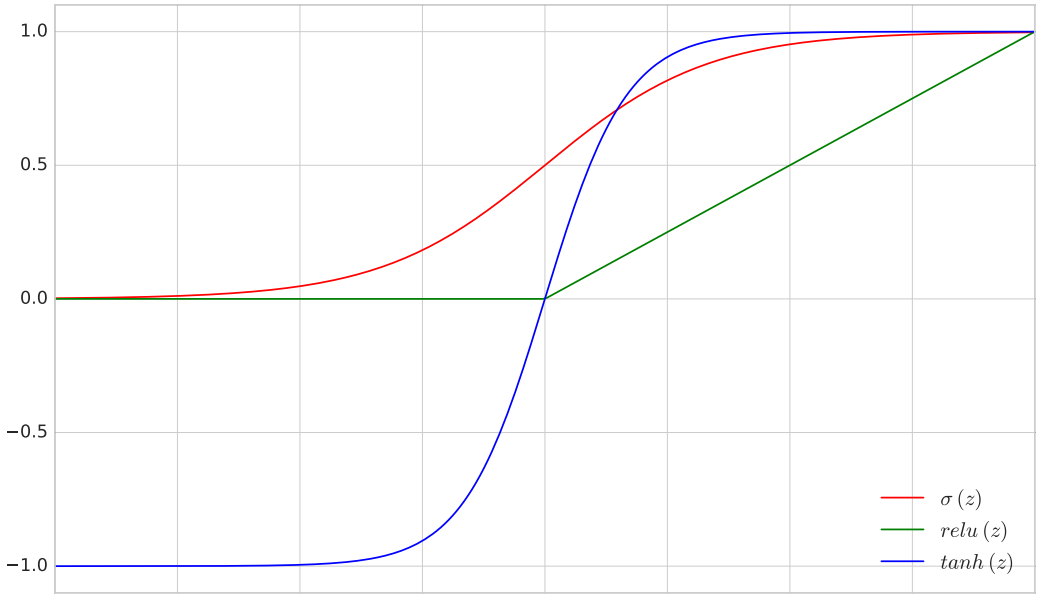
$$g(z) = \begin{cases} 0, & z \leq 0 \\ 1, & z > 0, \end{cases} \tag{5.3}$$

no ona se danas rijetko koristi. Danas se kao aktivacijske funkcije uglavnom koriste sigmoida ($\sigma(z)$), tangens hiperbolni ($\tanh(z)$) ili ReLU ($relu(z)$). Vrijedi:

$$\sigma(z) = \frac{1}{1+e^{-z}} \tag{5.4}$$

$$\tanh(z) = \frac{1-e^{-2z}}{1+e^{-2z}} \tag{5.5}$$

$$relu(z) = \max(0,\ z). \tag{5.6}$$



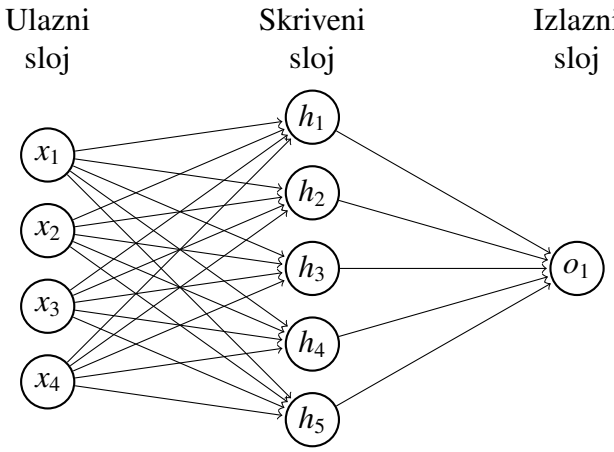
Slika 5.1: Grafovi aktivacijskih funkcija. Vrijednosti na x-osi su skalirane kako bi na istom grafu dobili bolji dojam oblika funkcija.

Primijetimo da ako odaberemo sigmoidu kao aktivacijsku funkciju, tada je model neurona zapravo model logističke regresije.

5.3 Arhitektura neuronske mreže

U umjetnim neuronskim mrežama neuroni su raspoređeni po slojevima. Svaki sloj može se sastojati od jednog ili više neurona. Najčešći tip mreže je *unaprijedna* neuronska mreža (eng. feedforward), gdje su izlazi jednog sloja ulazi drugog. U toj arhitekturi nema petlji i nema "preskakanja" slojeva.

Slojevi su podijeljeni u tri grupe: *ulazni sloj*, *skriveni sloj* te *izlazni sloj*. Neuronska mreža se tada sastoji od jednog ulaznog, nula ili više skrivenih slojeva te jednog izlaznog sloja.



Slika 5.2: Shematski prikaz unaprijedne neuronske mreže s jednim ulaznim, jednim skrivenim te jednim izlaznim slojem. Na slici nije prikazan slobodan član.

U teoriji je moguće da svaki sloj, ili čak svaki neuron, ima drugačiju aktivacijsku funkciju, no u praksi se to ne događa. Eventualna razlika u aktivacijskim funkcijama nastupa između aktivacijske funkcije skrivenih slojeva i aktivacijske funkcije izlaznog sloja. Izbor aktivacijske funkcije izlaznog sloja ovisi o problemu.

Primjerice, u regresijskom problemu možemo odabrati ReLU, ili čak odabrati identitetu, dok u klasifikacijskom problemu s dvije klase ima smisla odabrati sigmoidu.

5.4 Regularizacija

Kao i u drugim modelima koje smo promatrali, i općenito u modelima statističkog učenja, regularizacijom povećavamo pristranost modela i smanjujemo njegovu varijabilnost s ci-

ljem postizanja ravnoteže koja konačno ima manju ukupnu grešku. Ovdje ćemo ukratko opisati neke od često korištenih metoda regularizacije neuronskih mreža.

Kao i kod linearnih modela, regularizacija dodavanjem člana koji ovisi o normi parametara modela funkciji troška, efikasna je metoda regularizacije i kod neuronskih mreža. Neka je $\boldsymbol{\theta}$ vektor parametara modela. Onda funkciju troška možemo iskazati u terminima matrice dizajna $\mathbf{X}$, vektora (matrice) odgovora $\mathbf{Y}$ te vektora parametara $\boldsymbol{\theta}$: $J(\boldsymbol{\theta};\ \mathbf{X},\ \mathbf{Y})$. Regulariziranu funkciju troška $\tilde{J}$ sada definiramo s:

$$\tilde{J}(\boldsymbol{\theta};\ \mathbf{X},\ \mathbf{Y}) = J(\boldsymbol{\theta};\ \mathbf{X},\ \mathbf{Y}) + \lambda\Omega(\boldsymbol{\theta}), \tag{5.7}$$

gdje je λ regularizacijski koeficijent, a Ω funkcija koja penalizira veličinu parametara modela. Uglavnom se Ω odabire tako da ne penalizira konstantne članove, kao i kod linearnih metoda. Ako za Ω odaberemo L^2 normu težina, postižemo isti efekt kao i kod hrbat-regresije, to jest neprekidno sažimanje parametara prema nuli. Ako se odlučimo za Ω odabrati L^1 normu težina, dobit ćemo efekt kao i kod LASSO-a. Moguć je odabir proizvoljne norme težina modela, ali se odabire neka od gornje dvije.

Budući da se neuronske mreže treniraju iterativnim metodama možemo promatrati grešku nakon svake iteracije. Ako nacrtamo grešku u ovisnosti o broju iteracija dobit ćemo funkciju padajućeg trenda. Ako podijelimo podatke na trening-skup i test-skup, vidjet ćemo da iako na trening-skupu greška pada, u nekom trenutku na test-skupu greška počinje rasti. To je trenutak u kojem model postaje preprilagođen. *Metoda ranog zaustavljanja* (eng. early stopping) prekida treniranje modela u tom trenutku. Svaki put kada se nakon iteracije greška na testnom skupu smanji, spremimo dobivene parametre i vrijednost greške na testnom skupu. Treniranje prekidamo kada se testna greška nije smanjila određen, unaprijed definiran, broj iteracija.

Metoda napuštanja (eng. dropout) je metoda regularizacije kojom se na početku iteracije svaki neuron izbacuje s unaprijed definiranom vjerojatnošću $0 < p < 1$. Izračunaju se novi parametri za tako prorijeđenu mrežu te se postupak ponavlja u idućoj iteraciji.

Poglavlje 6

Primjeri

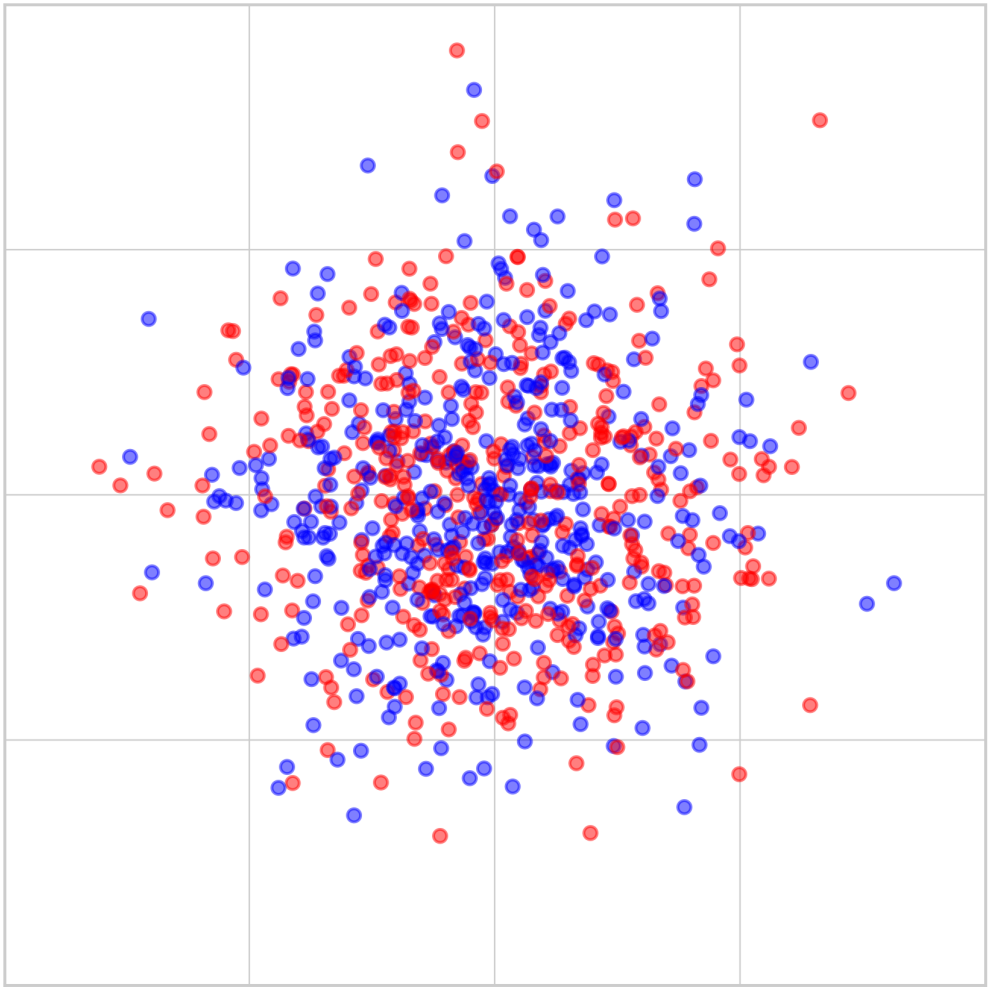
6.1 Uvod

Niti najbolji model ne može nam pomoći ako imamo loše podatke. Postoji fraza koja se često koristi u modeliranju: ”smeće unutra, smeće van”, odnosno GIGO (eng. Garbage In, Garbage Out). U ovom slučaju loši podaci mogu značiti mnogo stvari, bilo da je riječ o nekvalitetnim mjerenjima, nereprezentativnom uzorku, nedostajućim vrijednostima ili jednostavno podaci nemaju strukturu koju možemo iskoristi. Uzmimo za primjer dvije klase sa slike 6.1. Ne postoji algoritam koji bi efikasno mogao odijeliti te dvije klase. Svaki algoritam koji smo spomenuli koristi geometriju prostora u kojemu se podaci nalaze. Neovisno koristimo li podatke u njihovoj izvornoj formi, ili koristimo podatke s nekim njihovim trasformacijama, tako povećavajući dimenziju problema.

U ovom poglavlju dat ćemo nekoliko primjera i osvrt na razliku u geometriji dobivenih rješenja.

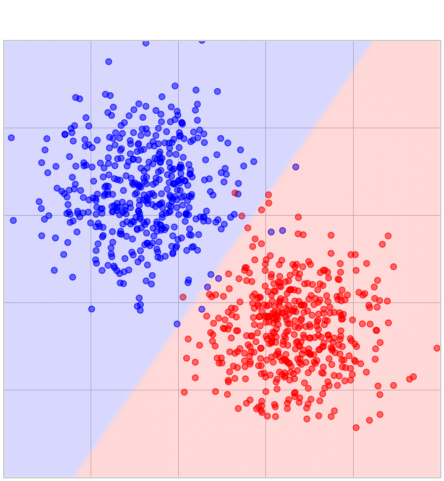
6.2 Linearno separabilne klase

Princip parsimonije tvrdi da je najprihvatljivije objašnjenje ono najjednostavnije. U ovom kontekstu to znači da ako imamo više modela koji daju otprilike istu predikciju, biramo onaj jednostavniji to jest računski manje kompleksan, s manjim brojem parametara ili jednostavno interpretabilniji.

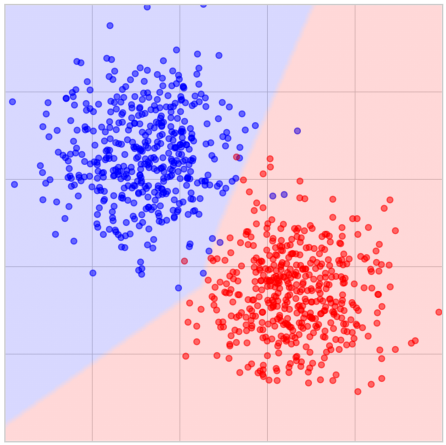


Slika 6.1: Prikaz dvije međusobno neodjeljive klase.

Promotrimo sliku 6.2. Obje metode postižu skoro savršenu separaciju, no neuronska mreža je kompliciraniji model, koji je skuplji za trenirati. U ovom slučaju nema smisla odabrati neuronsku mrežu.



Separacija dobivena logističkom regresijom.



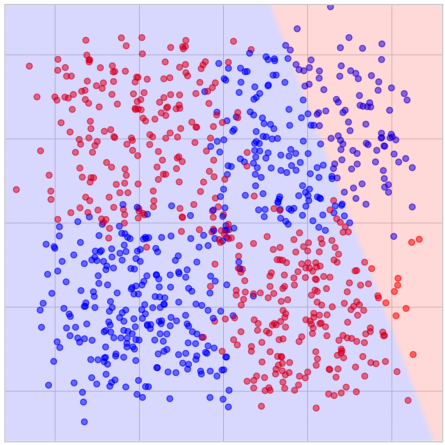
Separacija dobivena neuronskom mrežom s 5 skrivenih slojeva, s 3, 5, 5, 5, 3 neurona po slojevima.

Slika 6.2: Usporedba granica odluke dobivnih za različite algoritme na istom skupu podataka.

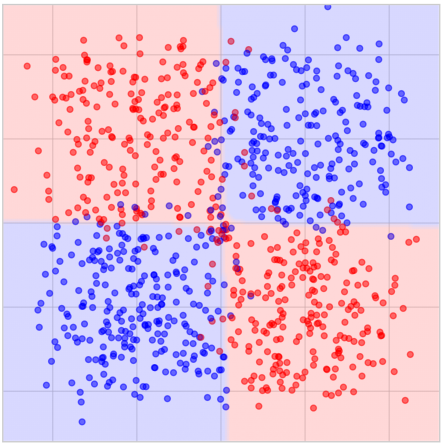
Uzmimo za primjer drugi slučaj gdje su podaci linearno separabilni, ali tek kada dodamo još dimenzija.

Na lijevoj strani slike 6.3 vidimo da je granica dosta loše postavljena, ali kako god da ju postavimo ne možemo dobiti dobru separaciju. Na desnoj slici vidimo dobru separaciju, međutim to više nije logistička regresija u koordinatama x i y, kao na lijevoj slici, već za ulazne varijable koristimo x, y, x^2, y^2 i $x \cdot y$. S druge strane koristimo li neuralnu mrežu samo na koordinatama x i y, dobijemo rezultat na desnoj strani slike 6.4

Vidimo da neuronska mreža postiže jako dobru separaciju, ali kada promotrimo izgled granice odluke vidimo da ona dosta krivudava. To bi mogla biti naznaka preprilagođenosti modela podacima. Također granica na densoj slici je vjerojatno robusnija, pogotovo s obzirom na točke daleko od središta.



Separacija dobivena logističkom regresijom.



Separacija dobivena logističkom regresijom uz upotrebu baznih funkcija: kvadrata koordinata i interakcije.

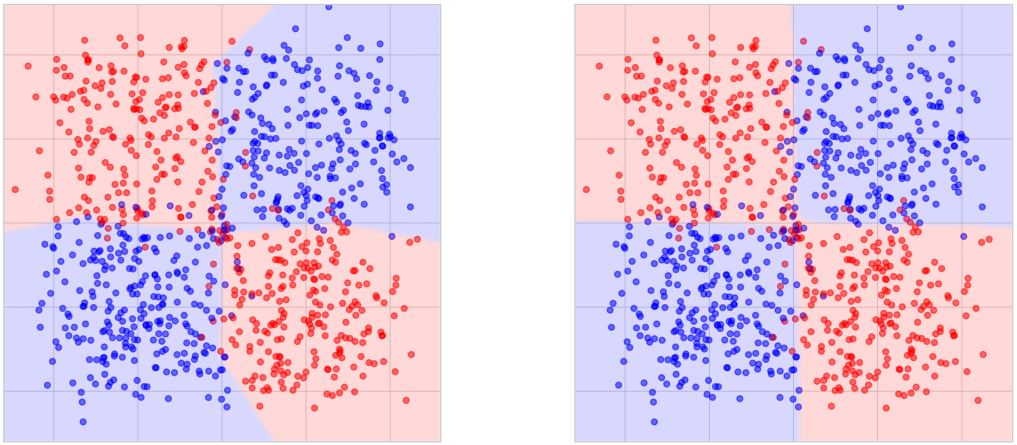
Slika 6.3: Usporedba granica odluke dobivnih za različite algoritme na istom skupu podataka.

6.3 Linearno neseparabilne klase

Problemi koje ovdje rješavamo su dvodimenzionalni zato što ih je lako vizualizirati, međutim u praksi problemi s kojima se susrećemo znaju imati jako puno dimenzija. Stoga je korištenje geometrijske intuicije za odabir baznih funkcija gotovo nemoguće. Ovdje koristimo vrlo jednostavne (polinomijalne) bazne funkcije, no za D-dimenzionalni problem postoji D^p polinomijalnih članova reda p. Dakle dimenzija problema raste eksponencijalno u redu polinoma. Ako znamo nešto dodatno o prirodi problema, možemo svoje predznanje uključiti u model odabirom specijalne bazne funkcije. Međutim, ako ne znamo ništa dodatno, onda je teško napraviti išta smisleno.

Neuronske mreže su u takvim slučajevima iznimno korisne. One mogu napraviti jako komplicirane granice odluke te nije potrebno nikakvo predznanje vezano o strukturi problema za njihovo korištenje.

Na lijevoj strani slike 6.5 vidimo da smo korištenjem bazne funkcije uspjeli postići da je granica odluke kružnica, ali crveni krug u sredini pogrešno klasificiramo. S druge strane,



Separacija dobivena neuronskom mrežom identične arhitekture kao i ona na slici 6.2.

Separacija dobivena logističkom regresijom uz upotrebu baznih funkcija.

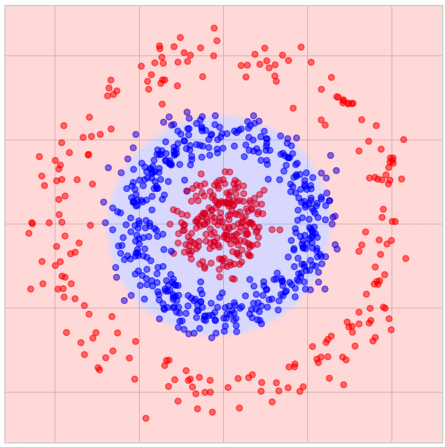
Slika 6.4: Usporedba granica odluke dobivnih za različite algoritme na istom skupu podataka. S lijeve strane imamo neuronskom mrežom na originalnom skupu podataka, a na desnoj dobivenu logističkom regresijom na skupu transformiranom baznim funkcijama.

na desnoj slici, korištenjem neuronske mreže dobili smo kružni vjenac kao područje klasificirano u plavu klasu. Dakle, crveno područje unutar plavog područja točno klasificiramo.

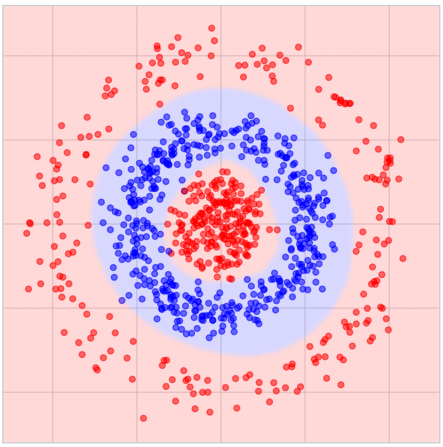
Ponekada nije dovoljno korištenje neuralnih mreža na originalnim podacima. Ponekad je prvo potrebno transformirati podatke. Ponekad i kada nije neophodno zna jako poboljšati rezultate i ubrzati proces. Iako u teoriji neuronska mreža s dovoljnim brojem skrivenih slojeva može aproksimirati bilo koju funkciju proizvoljno dobro, u praksi je neke funkcije relativno teško aproksimirati.

Na lijevoj strani slike 6.6 vidimo da bez upotrebe baznih funkcija model sve klasificira kao da dolazi iz crvene klase, dok na desnoj strani čini iznenađujuće dobar posao u separaciji crvene i plave spirale.

Teško je dati dobar recept za rješavanje ovakvih problema. Zato neki smatraju da je modeliranje više umjetnost nego znanost. Potrebno je puno vremena i truda, nešto iskustva i trikova. I kao sa svime ostalim, ponešto sreće.

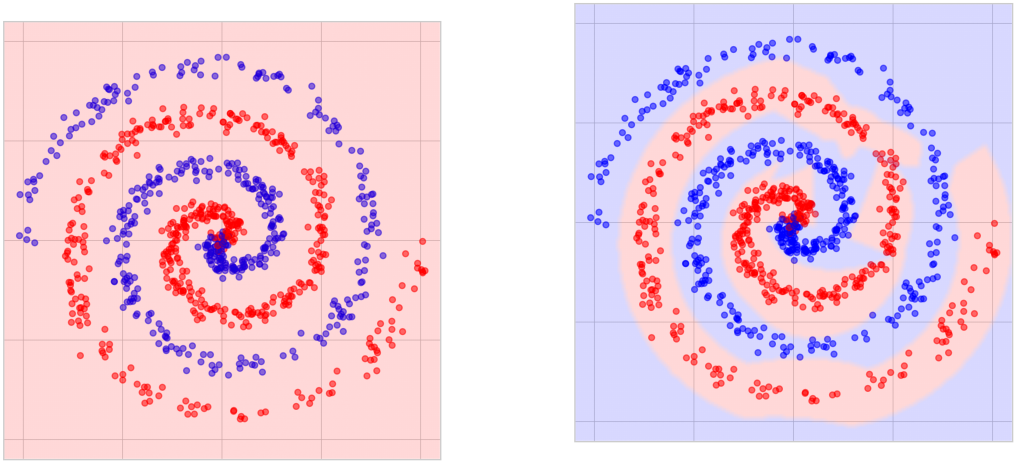


Separacija dobivena logističkom regresijom koristeći bazne funkcije kvadrata i interakcije.



Separacija dobivena neuronskom mrežom arhitekture kao i na slici 6.2.

Slika 6.5: Usporedba granica odluke dobivnih za različite algoritme na istom skupu podataka. S lijeve strane imamo separaciju dobivenu logističkom regresijom na skupu transformiranom baznim funkcijama, a sa desne strane separaciju neuronskom mrežom na originalnom skupu podataka.



Separacija dobivena neuronskom mrežom na originalnim podacima.

Separacija dobivena neuronskom mrežom na podacima transformiranim upotrebom bazne funkcije kvadrata i interakcije.

Slika 6.6: Usporedba granica odluke dobivnih korištenjem neuronske mreže na originalnim podacima i uz upotrebu baznih funkcija.

Bibliografija

[1] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer-Verlag, 2006.

[2] K. P. Burnham i D. R. Anderson, *Model Selection and Multimodal Inference*, Springer-Verlag, 1998.

[3] T. M. Cover i J. A. Thomas, *Elements of information theory*, John Wiley & Sons, Inc., 2006.

[4] R. Durrett, *Probability: theory and examples*, Cambridge University Press, 2010.

[5] I. Goodfellow, Y. Bengio i A. Courville, *Deep Learning*, MIT Press, 2016.

[6] T. Hastie, R. Tibshirani i J. Friedman, *The Elements Of Statistical Learning*, Springer-Verlag, 2009.

[7] G. James, D. Witten, T. Hastie i R. Tibshirani, *An Introduction to Statistical Learning*, Springer-Verlag, 2013.

[8] K. P. Murphy, *Machine learning: a probabilistic perspective*, MIT Press, 2012.

[9] N. V. Vapnik, *The nature of statistical learning theory*, Springer-Verlag, 1995.

[10] N. V. Vapnik, *Statistical learning theory*, John Wiley & Sons, Inc., 1998.

Sažetak

U ovom radu dali smo je kratki pregled područja statističkog učenja, osnovnu terminologiju te smo se usredotočili na problem nadziranog učenja. Pojasnili smo razliku između regresijskih i klasifikacijskih problema i dali njihovu matematičku formulaciju u terminima različitih funkcija troška. Uveli smo linearne regresijske i klasifikacijske modela te naveli često korištene pretpostavke te bitne rezultate. Diskutirali smo regularizacijske metode te dali interpretacije različitim odabirima regularizacijskog člana. Uveli smo osnovne metode ocjene i odabira modela, definirali pojam ravnoteže pristranosti i varijance, diskutirali razliku između trening greške i testne greške te pokazali neke metode za procjenu testne greške. Također smo uveli model umjetne neuronske mreže, definirali strukturu neurona, naveli često korištene aktivacijske funkcije te ukratko opisali popularne regularizacijske metode. Završili smo s nekoliko primjera koji imaju svrhu ilustrirati neke od metoda o kojima smo diskutirali.

Summary

In this work we gave a brief overview of statistical learning theory. We introduced basic terminology and focused on supervised learning. We explained the difference between regression and classification problems and introduced their mathematical formulation regarding different cost functions. We introduced linear regression and classification models and discused usual assumptions as well as some important results. We also discused some regularization methods and interpreted different choices for regularization terms. We introduced basic model assessment and selection methods, defined bias-variance tradeoff, discused difference between training and test error and showed some methods for test error estimation. We introduced artificial neural networks, defined a model of a neuron, listed common activation functions and described some popular regularization methods. In the end we gave some examples with the goal of illustrating methods discused in this work.

Životopis

Rođen sam 1993. godine u Zagrebu. Završio sam osnovnu školu Josipa Jurja Strossmayera te potom i XV gimnaziju. Upisujem preddiplomski sveučilišni studij matematike na PMF-u u Zagrebu, a po završetku diplomski sveučilišni studij matematičke statistike.