3. Linearna regresija

Strojno učenje 1, UNIZG FER, ak. god. 2022./2023.

Jan Šnajder, predavanja, v1.12

Prošli puta uveli smo osnovne koncepte strojnog učenja: **hipotezu**, koja je funkcija koja preslikava iz ulaznih podataka u oznake i definirana je s parametrima theta, i **model**, koji je skup hipoteza indeksiran parametrima theta. Rekli smo da se strojno učenje svodi na pronalaženje najbolje hipoteze u modelu, odnosno nalaženje optimalnih parametara. Pritom kao kriterij uzimamo pogrešku hipoteze, koju mjerimo na skupu označenih primjera, i tu mjeru zovemo **empirijska pogreška**. Zaključili smo da svaki algoritam SU ima **tri komponente**: model, funkciju gubitka (čije je očekivanje funkcija pogreške) i optimizacijski postupak.

Danas ćemo pogledati jedan konkretan algoritam strojnog učenja za **regresiju**. Prisjetimo se: regresija znači predviđanje numeričkih vrijednosti, dakle oznake $\mathcal{Y}$ su brojevi. Npr., predviđanje cijene stanova u Bostonu. Mi ćemo se fokusirati na jedan vrlo važan algoritam, a to je **linearan model regresije**, koji je vrlo važan i u strojnom učenju i u statistici. 1

U statistici, regresija primarno služi za **objašnjavanje odnosa** između dviju skupova varijabli te za statističko zaključivanje (npr., intervali pouzdanosti parametara, testovi statističke značajnosti prediktora). U strojnom učenju, regresija nas zanima prije svega zbog **predviđanja**, 2 dok nas objašnjavanje u pravilu manje zanima. Posljedica toga je da su pogledi na regresiju iz kuta statistike i iz kuta strojnog učenja malo različiti, premda se radi se o istoj stvari.

Današnje predavanje je strukturirano tako da najprije pogledamo najjednostavniji scenarij, a onda ćemo polako ići prema složenijim scenarijima.

1 Jednostavna regresija

Kao motivirajući primjer, zamislimo da želimo napraviti model koji predviđa cijenu nekretnine na temelju njezinih značajki. Očito, to je regresijski problem, jer predviđamo brojčanu vrijednost. Budući da se radi o nadziranom strojnom učenju, trebaju nam i označeni podatci. Općenito, raspolažemo označenim skupom podataka, $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}$, gdje su primjeri n-dimenzijski vektori značajki, $\mathbf{x} \in \mathbb{R}^n$, a oznake su realni brojevi, $y \in \mathbb{R}$. U našem slučaju, oznake y bile bi cijene nekretnina u Bostonu: za svaku nekretninu imamo niz značajki $\mathbf{x}$ koje ju opisuju, i imamo njezinu cijenu y u USD. Želimo napraviti model koji za nove, još neviđene nekretnine može predvidjeti njihovu cijenu.

Kod regresije, hipoteza h preslikava primjere u oznake, dakle $h : \mathbb{R}^n \to \mathbb{R}$, dakle preslikava iz ulaznog prostora (prostora primjera) u brojčane vrijednosti. Npr., hipoteza h za neku nekretninu opisanu značajkama $\mathbf{x}$ izračunava njezinu cijenu, $h(\mathbf{x})$. Kod regresije, varijable koje čine vektor $\mathbf{x}$ nazivamo **ulazne varijable** (ili **nezavisne varijable** ili **prediktorske varijale** ili **kovarijati**), a izlaz y nazivamo **izlazna varijabla** (ili **zavisna varijabla** ili **kriterijska varijabla**).

Danas ćemo se fokusirati na **linearnu regresiju**. Kod linearne regresije, izlaz je linearna kombinacija ulaznih varijabli. Odabirom takvog modela, mi smo implicitno pretpostavili da je izlazna varijabla linearno ovisna o ulaznim varijablama. Naš model (odnosno skup hipoteza definiranih do na parametre) definiran je ovako:

$$h(\mathbf{x}; \mathbf{w}) = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_n x_n$$

Tipično, kod regresije vektor parametara modela umjesto sa $\boldsymbol{\theta}$ označavamo sa $\mathbf{w}$. Parametri $w_0, \ldots, w_n$ određuju utjecaj svake značajke na vrijednost izlazne varijable. U strojnom učenju ti se parametri često nazivaju **težine**, dok se u statistici nazivaju **regresijskim koeficijentima** ili **beta-koeficijentima** i označavaju sa $\beta_0, \ldots, \beta_n$. 3

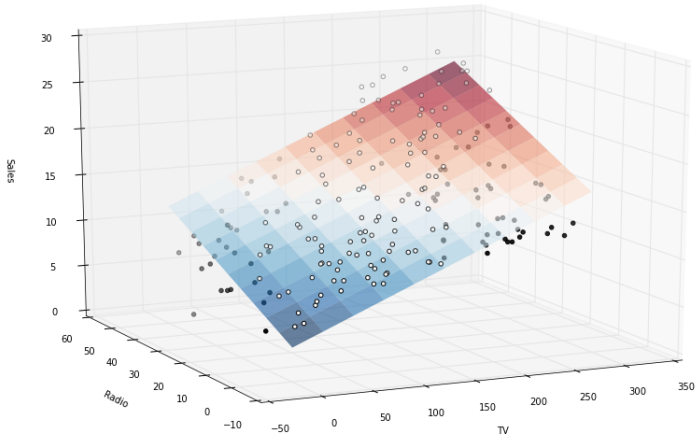
Za jednodimenzijski ulazni prostor, $n = 1$, hipoteza će biti **pravac**. To je takozvana **jednostavna regresija** (engl. *simple regression*). To je najjednostavniji linearni model regresije, kod kojega imamo samo jednu ulaznu varijablu:

$$h(x; w_0, w_1) = w_0 + w_1 x$$

▸ **PRIMJER**

- (1) Ušteđevina s obzirom na dob (raste više-manje linearno)
- (2) Cijena automobila s obzirom na kilometražu (naglo padajuća)

Ako je $n = 2$, onda hipoteza opisuje **ravninu** u vektorskom prostoru koji je definiran domenom $\mathcal{X} = \mathbb{R}^2$ i kodomenom $\mathcal{Y} = \mathbb{R}$ (preciznije, vektorski prostor dobivamo kao kartezijev umnožak tih dvaju skupova). To izgleda ovako:



Ako $n > 2$, hipoteza h opisuje **hiperravninu** u prostoru $\mathbb{R}^n \times \mathbb{R}$. Položaj te hiperravnine određen je vektorom težina $\mathbf{w}$.

Pogledajmo sada malo detaljnije **algoritam linearne regresije**. Prisjetimo se najprije da je svaki algoritam strojnog učenja definiran trima komponentama: modelom, funkcijom gubitka i postupkom optimizacije. Radi jednostavnosti, ograničimo se na jednostavnu regresiju, $n = 1$. Model smo već definirali. Trebaju nam još funkcija gubitka i optimizacijski postupak.

Krenimo od **funkcije gubitka**. Prisjetimo se: to je pogreška koju hipoteza čini na svakom pojedinačnom primjeru x, ako je točna oznaka primjera y. Kod regresije, za funkciju gubitka (oznaka L) tipično se koristi **kvadratno odstupanje** između ciljne i predviđene vrijednosti:

$$L\big(y, h(x)\big) = (y - h(x))^2$$

U statistici, odstupanje ciljne vrijednosti od predviđene vrijednosti, $y - h(x)$, naziva se **rezidual**. Možemo onda reći da je funkcija gubitka jednaka **kvadratu reziduala**.

Ovdje se legitimno možemo zapitati: zašto bismo koristili baš kvadratno odstupanje kao funkciju gubitka? Zašto ne samo razliku između ciljne i predviđene vrijednosti? Zato što je

gubitak gubitak, neovisno o smjeru. Ako bismo računali samo razlike, pozitivni i negativni gubitci bi se međusobno poništavali, a to ne želimo. No, zašto onda ne koristimo apsolutnu vrijednost te razlike? Odgovor se krije u potrebi da optimiziramo funkciju pogreške. Naime, funkcija apsolutne vrijednosti nije derivabilna, dok funkcija kvadratnog odstupanja jest. Ako funkcija gubitka nije derivabilna, onda to sigurno neće biti i funkcija pogreške, koja je definirana kao njezino očekivanje. Zato dajemo prednost kvadratnom odstupanju. No, vidjet ćemo kasnije da ima i dubljih razloga zašto koristiti kvadratno odstupanje.

Što je s **funkcijom pogreške**? Prisjetimo se: gubitak i pogreška nisu isto. Rekli smo na prethodnom predavanju da je pogreška hipoteze zapravo **očekivana vrijednost gubitka hipoteze**, koju empirijski izračunavamo kao prosječnu vrijednost funkcije gubitka na skupu označenih primjera. Slično ćemo definirati i pogrešku hipoteze regresije:

$$E(h|\mathcal{D}) = \frac{1}{2}\sum_{i=1}^{N} L(y^{(i)}, h(x^{(i)})) = \frac{1}{2}\sum_{i=1}^{N} \left(y^{(i)} - h(x^{(i)})\right)^2$$

s tom razlikom da umjesto prosjeka, dakle umjesto $1/N$, koristimo $1/2$ zbog kasnije matematičke jednostavnosti. Ta sitna izmjena, međutim, nema baš nikakvog utjecaja na optimizaciju: minimizator (odnosno vrijednost za koju funkcija poprima minimum) bit će isti, neovisno o konstanti kojom množimo funkciju gubitka. Ovu funkciju pogreške nazivamo **funkcija kvadratne pogreške** (engl. *quadratic error function*). 4

Treća komponenta algoritma regresije jest **optimizacijski postupak**: potraga za onom hipotezom u modelu, $h \in \mathcal{H}$, koja minimizira empirijsku pogrešku. To jest:

$$h^* = \operatorname*{argmin}_{h\in\mathcal{H}} E(h|\mathcal{D}) = \operatorname*{argmin}_{h\in\mathcal{H}} \frac{1}{2}\sum_{i=1}^{N} \left(y^{(i)} - h(x^{(i)})\right)^2$$

Budući da je hipoteza h indeksirana parametrima (w_0, w_1), mi zapravo tražimo parametre modela koji minimiziraju pogrešku, tj.:

$$(w_0, w_1)^* = \operatorname*{argmin}_{w_0,w_1} \frac{1}{2}\sum_{i=1}^{N} \left(y^{(i)} - (w_1 x^{(i)} + w_0)\right)^2$$

Razmotrimo ovaj optimizacijski problem s matematičkog stajališta. Imamo funkciju dviju varijabli i tražimo njezin minimum. Što bi nam prvo trebalo pasti na pamet? Pa, trebali bismo pronaći nultočke derivacije ove funkcije: derivirati funkciju, izjednačiti je s nulom te riješiti sustav jednadžbi. (Primijetimo i da je ovo konveksna funkcija, pa će derivacija prvog reda imati samo jednu nultočku, i ta je nultočka onda sigurno minimizator funkcije.) Pokažimo kako bi to izlgedalo. Deriviramo funkciju pogreške i izjednačavamo s nulom:

$$\nabla_{w_0,w_1} E(h|\mathcal{D}) = 0$$

Budući da se radi o funkciji dvije varijable, računamo parcijalne derivacije po prvom parametru (w_0) i po drugom parametru (w_1) i izjednačavamo ih s nulom:

$$\frac{\partial}{\partial w_0}\left[\frac{1}{2}\sum_{i}^{N} \left(y^{(i)} - (w_1 x^{(i)} + w_0)\right)^2\right] = 0$$

$$\frac{\partial}{\partial w_1}\left[\frac{1}{2}\sum_{i}^{N} \left(y^{(i)} - (w_1 x^{(i)} + w_0)\right)^2\right] = 0$$

Nakon nekoliko koraka, koje ćemo ovdje preskočiti jer nam nisu naročito interesantni, došli bismo do sljedećeg rješenja: 5

$$w_0 = \bar{y} - w_1\bar{x}$$

$$w_1 = \frac{\sum_i^N x^{(i)} y^{(i)} - N\bar{x}\bar{y}}{\sum_i^N (x^{(i)})^2 - N\bar{x}^2}$$

gdje su $\bar{x}$ i $\bar{y}$ srednje vrijednosti značajke x odnosno oznake y kroz sve primjere u skupu $\mathcal{D}$.

Uočimo ovdje dvije stvari. Prva je da se naša optimizacija svela na jednostavno uvrštavanje brojeva u formulu: želim li pronaći parametre (w_0, w_1) koji minimiziraju funkciju kvadratne pogreške, sve što trebam napraviti jest primijeniti gornje formule. Za ovakve probleme, gdje je minimizator dan formulom i optimizacija se, umjesto nekog heurističkog postupka, svodi na uvrštavanje brojeva u tu formulu, kažemo da imaju **rješenje u zatvorenoj formi** (engl. *closed form solution*). To znači da je rješenje neki matematički izraz koji koristi "široko prihvaćene" funkcije i operatore (logaritam, korijen, trigonometrijske funkcije i sl.) te se može izračunati u konačnome broju koraka. U strojnom učenju jako volimo kada naš optimizacijski problem ima rješenje u zatvorenoj formi jer to u pravilu znači da je optimizacija jednostavna i egzaktna. 6

Drugu stvar koju želim da uočite jest da su formule za w_0 i w_1 zapravo različite, i da je formula za w_1 zapravo poprilično ružna. Ono što ovdje doduše ne možete uočiti, ali biste se u to lako mogli uvjeriti da to pokušate izvesti, jest da bismo za model s tri parametara, (w_0, w_1, w_2), dobili složenije formule, za model sa četiri parametara još složenije, itd. To nije idealno, jer želimo općenito rješenje optimizacijskog postupka, u smislu da jednadžba rješenja ne ovisi o broju parametara (konkretno rješenje, naravno, mora ovisiti o konkretnom broju parametara kao što ovisi i o konkretnom skupu označenih primjera). Kao što (možda?) slutite, način da postignemo takvu općenitost bit će da rješenje optimizacijskog problema izvedemo u matričnoj formi (jer će to rješenje onda vrijediti za matrice dizajna i vektore oznaka proizvoljnih dimenzija). I, doista, to ćemo i napraviti (nekoliko stranica ispod).

2 Vrste regresije

Nakon što smo pogledali ovaj jednostavan slučaj regresije, pokušajmo poopćiti naš pristup. Bavili smo se jednostavnom regresijom, koja ima samo jednu ulaznu varijablu. To je ograničeno korisno: stvarno zanimljivi fenomeni iz stvarnog života uglavnom se ne mogu dobro opisati samo jednom nezavisnom varijablom. Srećom, jednostavna regresija je samo jedna vrsta regresije. Postoje i druge vrste regresije. Konkretno, prema broju **ulaznih (nezavisnih, prediktorskih)** varijabli, regresija može biti **jednostruka regresija** (ili **jednostavna regresija**), kod koje $n = 1$ (na ulazu je samo jedna značajka) ili **višestruka regresija** (ili **multipla regresija**), kod koje $n > 1$ (primjer je opisan s više značajki). S druge strane, prema broju **izlaznih (zavisnih, kriterijskih)** varijabli, regresija može biti **univarijatna regresija** (ili **jednoizlazna regresija**), kod koje $h(\mathbf{x}) = y$, ili **multivarijatna regresija** (ili **višeizlazna regresija**), kod koje $h(\mathbf{x}) = \mathbf{y}$ (izlazna varijabla također je vektor). Kada ukrižimo ove dvije podjele, dobivamo četiri vrste regresije, koje tablično možemo prikazati ovako:

	Jedan izlaz (y)	**Više izlaza** ($\mathbf{y}$)
Jedan ulaz (x)	(Univarijatna) jednostavna	Multivarijatna jednostavna
Više ulaza ($\mathbf{x}$)	(Univarijatna) višestruka	Multivarijatna višestruka

Tipično se podrazumijeva da je regresija univarijatna; ako to nije slučaj, onda je dobro da se eksplicitno kaže da je multivarijatna. Zbog toga se univarijatna višestruka regresija u literaturi tipično jednostavno naziva **višestruka regresija** (engl. *multiple regression*).

Mi ćemo se na ovom predmetu fokusirati upravo na **višestruku regresiju**, budući da se ona najčešće koristi, a i najbolje korespondira s klasifikacijskim modelima kojima ćemo se baviti kasnije. Dakle, fokusirat ćemo se na regresiju s više ulaza i s jednim izlazom, tj. za svaki primjer, opisan nizom značajki, predviđamo jedan broj. Npr., cijena nekretnina u Bostonu, gdje je nekretnina opisana nizom značajki (broj soba, udaljenost od glavne prometnice, ima li lift, jesu li susjedi podnošljivi, itd.). Univarijatna jednostavna regresija je, naravno, samo poseban slučaj univarijatne višestruke regresije gdje je $n = 1$. 7

3 Tri komponente algoritma linearne regresije

Pogledajmo sada opet tri osnovne komponente algoritma linearne regresije, i to višestruke, tj. one za koju $n > 1$.

1. **Model**

$$h(\mathbf{x}; \mathbf{w}) = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_n x_n = \sum_{j=1}^{n} w_j x_j + w_0$$

Kao što smo već sugerirali gore, izračun će biti mnogo jednostavniji ako pređemo na matrični račun. U tu svrhu svaki vektor primjera $\mathbf{x}^{(i)}$ proširit ćemo jednom tzv. **dummy značajkom**, $x_0^{(i)} = 1$. Vrijednost te značajke uvijek je jednaka 1, i služi tome da se pomnoži težinom w_0, pa da sve težine možemo tretirati jednako. Model je onda: [8]

$$h(\mathbf{x}; \mathbf{w}) = \mathbf{w}^{\mathrm{T}} \mathbf{x}$$

gdje je $\mathbf{w}^{\mathrm{T}}\mathbf{x}$ skalarni produkt vektora $\mathbf{w}$ i $\mathbf{x}$, zapisan kao skalarni produkt vektor-retka $\mathbf{w}^{\mathrm{T}}$ (transponirani vektor-stupac) i vektor-stupca $\mathbf{x}$.

2. **Funkcija gubitka** (i pripadna **funkcija pogreške**)

Algoritam regresije kao gubitak koristi kvadratno odstupanje, koje nazivamo i **kvadratni gubitak** (engl. *quadratic loss*):

$$L(y^{(i)}, h(\mathbf{x}^{(i)})) = \left(y^{(i)} - h(\mathbf{x}^{(i)})\right)^2$$

Funkcija pogreške definirana je kao zbroj gubitaka kroz sve primjere iz označenog skupa $\mathcal{D}$, podijeljena s dva (i to je proporcionalno empirijskom očekivanju gubitka):

$$E(h|\mathcal{D}) = \frac{1}{2} \sum_{i=1}^{N} \left(y^{(i)} - h(\mathbf{x}^{(i)})\right)^2$$

3. **Optimizacijski postupak**

Kao i prošli tjedan, optimizacijski postupak definirat ćemo najprije vrlo apstraktno:

$$\mathbf{w}^* = \operatorname*{argmin}_{\mathbf{w}} E(\mathbf{w}|\mathcal{D})$$

U kontekstu regresije, gdje je funkcija pogreške definirana preko kvadratnog gubitka, ovaj se postupak naziva postupak **najmanjih kvadrata** (engl. *least squares*) ili metoda **obični najmanji kvadrati** (engl. *ordinary least squares, OLS*). Naziv reflektira činjenicu [9] da tražimo parametre $\mathbf{w}$ za koje je zbroj kvadrata odstupanja (kvadrata reziduala) najmanji.

Dobra vijest je da, kod linearnog modela regresije, neovisno o broju značajki n, rješenje ovog optimizacijskog problema uvijek postoji u **zatvorenoj formi**. Dakle, rješenje ćemo moći napisati kao matematički izraz koji će biti izračunljiv u konačnome broju koraka. (Doduše, vidjet ćemo da taj izračun neće biti posve trivijalan, jer će uključivati izračun inverza matrice, a to kod velikih matrica nije trivijalno.)

4 Postupak najmanjih kvadrata

Razmotrimo sada detaljnije postupak najmanjih kvadrata, odnosno kako možemo izračunati optimalne parametre $\mathbf{w}$ u smislu zbroja kvadratnih odstupanja, i to za višestruku linearnu

regresiju, gdje $n > 1$. Raspolažemo skupom primjera (kod regresije to su ulazne varijable odnosno "nezavisne varijable"):

$$\mathbf{X} = \begin{pmatrix} 1 & x_1^{(1)} & x_2^{(1)} \dots & x_n^{(1)} \\ 1 & x_1^{(2)} & x_2^{(2)} \dots & x_n^{(2)} \\ \vdots & & & \\ 1 & x_1^{(N)} & x_2^{(N)} \dots & x_n^{(N)} \end{pmatrix}_{N \times (n+1)}$$

Prisjetimo se, matricu primjera $\mathbf{X}$ nazivamo **matrica dizajna**. Imamo također i vektor izlaznih vrijednosti (tj. oznake primjera odnosno "zavisne varijable"):

$$\mathbf{y} = \begin{pmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{pmatrix}_{N \times 1}$$

4.1 Krivi put: egzaktno rješenje

Pokušajmo prvo s nečim što nije postupak najmanjih kvadata, a što će se pokazati kao loša ideja. Naime, na naš optimizacijski problem možemo gledati kao na rješavanje sustava jednadžbi. Idealno, mi bismo željeli pronaći rješenje za koje vrijedi

$$(\mathbf{x}^{(i)}, y^{(i)}) \in \mathcal{D}.\ h(\mathbf{x}^{(i)}) = y^{(i)}$$

odnosno

$$(\mathbf{x}^{(i)}, y^{(i)}) \in \mathcal{D}.\ \mathbf{w}^{\mathrm{T}}\mathbf{x}^{(i)} = y^{(i)}$$

To bi bilo **egzaktno rješenje** našeg problema: rješenje kod kojega regresijski model za svaki ulazni primjer savršeno predviđa njegovu izlaznu vrijednost. Tako definiran problem zapravo je sustav od N jednadžbi s $(n+1)$ nepoznanicom. Takav sustav jednadžbi možemo elegantno napisati kao **matričnu jednadžbu**:

$$\begin{pmatrix} 1 & x_1^{(1)} & x_2^{(1)} \dots & x_n^{(1)} \\ 1 & x_1^{(2)} & x_2^{(2)} \dots & x_n^{(2)} \\ \vdots & & & \\ 1 & x_1^{(N)} & x_2^{(N)} \dots & x_n^{(N)} \end{pmatrix} \cdot \begin{pmatrix} w_0 \\ w_1 \\ \vdots \\ w_n \end{pmatrix} = \begin{pmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{pmatrix}$$

to jest

$$\mathbf{X}\mathbf{w} = \mathbf{y}$$

(uvjerite se da je to isto kao i sustav jednadžbi koji smo napisali ranije). Egzaktno rješenje ovog sustava jednadžbi dobivamo tako da jednadžbu slijeva pomnožimo sa $\mathbf{X}^{-1}$:

$$\mathbf{w} = \mathbf{X}^{-1}\mathbf{y}$$

Ovo je lijepo, međutim, neće uvijek biti rješivo. Naime, problem će često biti **loše definiran** (engl. *ill-defined*), što znači da gornji sustav linearnih jednadžbi nema rješenja (**nekonzistenatan je**) ili da pak **nema jedinstvenog rješenja** (ima beskonačno mnogo rješenja). Prvi i najbanalniji problem jest taj što matrica dizajna $\mathbf{X}$, da bi imala inverz, nužno mora biti **kvadratna**. No, broj primjera općenito može biti veći od broja parametara, $N > n+1$, ili može biti manji, $N < n+1$. U prvom slučaju kažemo da je sustav jednadžbi **preodređen** (engl. *overdetermined*), a u drugom da je **pododređen** (engl. *underdetermined*). U oba slučaja matrica $\mathbf{X}$ neće biti kvadratna i neće imati inverz. Istini za volju, zahtjev da je matrica $\mathbf{X}$ kvadratna igra ulogu

samo ako rješenje sustava jednadžbi želimo riješiti pomoću inverza matrice. Možemo koristiti neku drugu metodu, npr., Gaussovu metodu eliminacije, koja je primjenjiva i na pravokutne matrice. No to, naravno, i dalje ne znači da će sustav jednadžbi imati rješenje niti da će ono biti jedinstveno. Konkretno, ako je **rang** matrice $\mathbf{X}$ manji od ranga proširene matrice $\mathbf{X}|\mathbf{y}$, onda je sustav jednadžbi nekonzistentan. Nadalje, ako rang matrice $\mathbf{X}$ nije jednak broju stupaca (odnosno broju parametara, koji je jednak $n+1$), onda sustav ima beskonačno mnogo rješenja. Drugim riječima, da bi sustav imao jedno i jedinstveno rješenje, rangovi obaju matrica $\mathbf{X}$ i $\mathbf{X}|\mathbf{y}$ moraju biti jednaki $n+1$.

▸ **PRIMJER**

Evo nekoliko trivijalnih primjera koji demonstriraju da nas egzaktno rješenje sustava linearnih jednadžbi u praksi neće zadovoljiti.

Neka je skup primjera za učenje za jednostavnu regresiju $\mathcal{D} = \{(x^{(i)}, y^{(i)})\} = \{(1,1),(1,2)\}$. Matrica dizajna $\mathbf{X}$ doduše jest kvadratna, ali je sustav nekonzistentan (matrica $\mathbf{X}$ je ranga 1, dok je proširena matrica $\mathbf{X}|\mathbf{y}$ ranga 2). To je zato što za primjer $x = 1$ imamo dvije različite oznake. Suprotno, za označeni skup primjera $\mathcal{D} = \{(1,1),(1,1)\}$ matrica $\mathbf{X}$ je i dalje kvadratna, ali matrica $\mathbf{X}|\mathbf{y}$ ima rang 1, dok $n+1 = 2$, pa sustav ima beskonačno mnogo rješenja. Intuitivno, to je zato što imamo samo jedan primjer u skupu za učenje, a kroz jednu točku možemo provući beskonačno mnogo pravaca.

Razmotrimo sada skup primjera $\mathcal{D} = \{(1,1),(2,2),(3,3)\}$. Ove točke leže na pravcu i očekujemo naći egzaktno rješenje za regresijski pravac. Doduše, to rješenje ne možemo naći gornjom jednadžbom (inverzom matrice), jer matrica dizajna $\mathbf{X}$ nije kvadratna (nego je dimenzija 3×2), odnosno sustav je preodređen. Međutim, sustav jest konzistentan (rang matrice $\mathbf{X}$ jednak je 2 i to je jednako rangu matrice $\mathbf{X}|\mathbf{y}$) i ima jedinstveno rješenje (rang je jednak $n+1 = 2$), pa rješenje $(w_0 = 0, w_1 = 1)$ možemo dobiti, na primjer, Gaussovom metodom eliminacije. No, u stvarnosti podatci neće biti tako idealni. Prisjetimo se **šuma** o kojemu smo pričali prošli prošli put. Dovoljno je da se, zbog šuma, jedna od ovih triju točaka malo pomakne s pravca, i cijela stvar pada u vodu. Na primjer, ako je skup primjera za učenje $\mathcal{D} = \{(1,1),(2,2.1),(3,3)\}$ (zbog šuma je oznaka drugog primjera 2.1 umjesto 2), već imamo nekonzistentan sustav jednadžbi (rang od $\mathbf{X}$ je 2, a rang od $\mathbf{X}|\mathbf{y}$ je 3). Očito, to je zato što kroz ove tri točke ne možemo provući pravac, budući da one ne leže na istom pravcu.

Slično, razmotrimo skup primjera $\mathcal{D} = \{((1,1),1),((2,2),2)\}$ za dvoulaznu ($n = 2$) višestruku regresiju. Odgovarajuća proširena matrica $\mathbf{X}|\mathbf{y}$ izgleda ovako:

$$\left(\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 1 & 2 & 2 & 2 \end{array}\right)$$

Rang matrice $\mathbf{X}$ i rang matrice $\mathbf{X}|\mathbf{y}$ jednaki su 2, što je manje od broja parametara ($n+1 = 3$), pa zaključujemo da sustav jednadžbi ima beskonačno mnogo rješenja. To je zato jer raspolažemo samo dvama primjerima: svaki primjer je jedna točka u 3d-prostoru definirana koordinatama (x_1, x_2, y), a kroz dvije točke u 3d-prostoru možemo provući beskonačno mnogo ravnina.

Ovi primjeri nam ukazuju da, iz perspektive strojnog učenja, egzaktno rješenje nije dovoljno robusno. Čak i ako su podatci nastali nekom linearnom funkcijom, u skupu primjera za učenje postojat će šum, pa ne možemo očekivati da će sve točke savršeno ležati na pravcu. Nadalje, čak ako imamo manje primjera nego parametara, ipak bismo voljeli imati neko rješenje. Sve u svemu, egzaktno rješenje pretpostavlja da je svijet savršen, ali on to nije. U nesavršenom svijetu, približno rješenje je savršeno dobro.

4.2 Pravi put: rješenje najmanjih kvadrata

Zbog ograničenja koja smo upravo ustanovili, umjesto da tražimo egzaktno rješenje sustava $\mathbf{X}\mathbf{w} = \mathbf{y}$, tražit ćemo **približno** rješenje sustava. Što mislimo pod "približno"? Mogli bismo to definirati na razne načine, no kod postupka najmanjih kvadrata "približno" je definirano u smislu najmanjih kvadratnih odstupanja. To smo zapravo već i napravili, kada smo gledali onaj prvi primjer, s jednostavnom regresijom. Tamo smo funkciju gubitka definirali preko kvadratne funkcije gubitka, pa dakle ta funkcija upravo mjeri kvadratno odstupanje rješenja od oznaka u

skupu podataka. Sada bismo takav pristup želi poopćiti na višestruku regresiju, dakle na slučaj s više značajki, $n > 1$.

Ovdje na scenu nastupa **matrični račun**. Naime, sada kada imamo više značajki, puno je jednostavnije da kod optimizacije pređemo na zapis gdje ćemo koristiti matrice, i da onda pokušamo izvesti rješenje s operacijama nad matricama. 13

Prisjetimo se, naša funkcija pogreške za regresiju je:

$$E(\mathbf{w}|\mathcal{D}) = \frac{1}{2}\sum_{i=1}^{N}\left(\mathbf{w}^{\mathrm{T}}\mathbf{x}^{(i)} - y^{(i)}\right)^2$$

Matrični oblik ove funkcije je:

$$E(\mathbf{w}|\mathcal{D}) = \frac{1}{2}(\mathbf{X}\mathbf{w} - \mathbf{y})^{\mathrm{T}}(\mathbf{X}\mathbf{w} - \mathbf{y})$$

Ovdje si uzmite malo vremena da se uvjerite da je to doista tako. (Shvatili ste kada znate objasniti kamo je nestao kvadrat i kamo je nestala suma.) Sada idemo dalje raspisivati desnu stranu izraza. Transpoziciju primjenjujemo na pojedinačne pribrojnike (jer $(\mathbf{A}+\mathbf{B})^{\mathrm{T}} = \mathbf{A}^{\mathrm{T}} + \mathbf{B}^{\mathrm{T}}$), zatim primjenjujemo distributivnost i nakon toga sljedeće dvije jednakosti iz matričnog računa:

$$(\mathbf{A}^{\mathrm{T}})^{\mathrm{T}} = \mathbf{A}$$
$$(\mathbf{A}\mathbf{B})^{\mathrm{T}} = \mathbf{B}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}$$

I tako dobivamo:

$$E(\mathbf{w}|\mathcal{D}) = \frac{1}{2}(\mathbf{w}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}}\mathbf{X}\mathbf{w} - \mathbf{w}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}}\mathbf{y} - \mathbf{y}^{\mathrm{T}}\mathbf{X}\mathbf{w} + \mathbf{y}^{\mathrm{T}}\mathbf{y})$$

Ovdje se sada trebamo prisjetiti da je $E(\mathbf{w}|\mathcal{D})$ zapravo skalar, što znači da svi izrazi koje ovdje zbrajamo zapravo moraju biti dimenzija 1×1. Doista, na primjer za $\mathbf{w}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}}\mathbf{y}$:

$$\underbrace{1 \times (n+1)}_{\mathbf{w}^{\mathrm{T}}} \cdot \underbrace{(n+1) \times N}_{\mathbf{X}^{\mathrm{T}}} \cdot \underbrace{N \times 1}_{\mathbf{y}} = 1 \times 1$$

Ako su to skalari, onda ih možemo transponirati a da time ništa ne promijenimo, jer transpozicija skalara daje isti taj skalar. A to onda znači da su srednja dva pribrojnika u gornjem izrazu jednaka, jer

$$\mathbf{w}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}}\mathbf{y} = (\mathbf{w}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}}\mathbf{y})^{\mathrm{T}} = \left((\mathbf{w}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}})\mathbf{y}\right)^{\mathrm{T}} = \mathbf{y}^{\mathrm{T}}(\mathbf{w}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}})^{\mathrm{T}} = \mathbf{y}^{\mathrm{T}}(\mathbf{X}\mathbf{w})$$

gdje smo iskoristili gornje dvije jednakosti iz matričnog računa. Tako u konačnici dobivamo:

$$E(\mathbf{w}|\mathcal{D}) = \frac{1}{2}(\mathbf{w}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}}\mathbf{X}\mathbf{w} - 2\mathbf{y}^{\mathrm{T}}\mathbf{X}\mathbf{w} + \mathbf{y}^{\mathrm{T}}\mathbf{y})$$

Sada želimo naći minimizator ove funkcije. Za to nam trebaju pravila za **deriviranje matrica**. Postoji niz pravila, ali srećom nama će biti dovoljna samo ova dva:

$$\frac{\mathrm{d}}{\mathrm{d}\mathbf{x}}\mathbf{A}\mathbf{x} = \mathbf{A}$$
$$\frac{\mathrm{d}}{\mathrm{d}\mathbf{x}}\mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{x} = \mathbf{x}^{\mathrm{T}}(\mathbf{A} + \mathbf{A}^{\mathrm{T}})$$

Prije nego što nastavimo, da vidimo što mi sad zapravo želimo. Mi funkciju pogreške $E(\mathbf{w}|\mathcal{D})$ želimo derivirati po vektoru $\mathbf{w}$, jer tražimo minimum po parametrima $\mathbf{w}$. Funkcija E je funkcija vektora $\mathbf{w}$, dakle to je funkcija od $(n+1)$ varijabli. Što će biti derivacija funkcije

E po vektoru $\mathbf{w}$, tj. što je $\nabla_{\mathbf{w}} E$? To je **gradijent** funkcije E. Gradijent je opet funkcija, i to vektorska funkcija, koja za neku točku $\mathbf{w}$ daje smjer (vektor) najbržeg rasta funkcije E. Dakle $\nabla_{\mathbf{w}} E$ je $(n+1)$-dimenzijski vektor (tj. gradijent). Stacionarna točka funkcije je ona u kojoj je gradijent jednak nuli (preciznije, nul-vektoru), a za funkciju E stacionarna točka će biti točka minimuma (prisjetimo se, funkcija E je konveksna, dakle njezina stacionarna točka sigurno je njezin minimum). U konačnici nas, dakle, zanima koja je to točka gdje je gradijent funkcije E jednak nuli. Eto, sada kada smo se svega ovoga prisjetili, možemo dalje...

Derivirajmo funkciju pogreške i izjednačimo je s nul-vektorom:

$$\nabla_{\mathbf{w}} E = \frac{1}{2}\Big(\mathbf{w}^{\mathrm{T}}\big(\mathbf{X}^{\mathrm{T}}\mathbf{X} + (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{\mathrm{T}}\big) - 2\mathbf{y}^{\mathrm{T}}\mathbf{X}\Big) = \mathbf{w}^{\mathrm{T}}(\mathbf{X}^{\mathrm{T}}\mathbf{X}) - \mathbf{y}^{\mathrm{T}}\mathbf{X} = \mathbf{0}$$

iz čega slijedi:

$$\mathbf{w}^{\mathrm{T}}(\mathbf{X}^{\mathrm{T}}\mathbf{X}) = \mathbf{y}^{\mathrm{T}}\mathbf{X}$$

odnosno, nakon transpozicije (uz primjenu gornjih pravila za transpoziciju umnoška matrica):

$$\mathbf{X}^{\mathrm{T}}\mathbf{X}\mathbf{w} = \mathbf{X}^{\mathrm{T}}\mathbf{y}$$

Ovo je sustav tzv. **normalnih jednadžbi**. Rješenje sustava dobivamo množenjem s $(\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}$ slijeva:

$$\mathbf{w} = (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y} = \mathbf{X}^{+}\mathbf{y}$$

i to je naše rješenje za vektor parametara $\mathbf{w}$ koji minimizira kvadratno odstupanje.

Ovo zaslužuje nekoliko komentara. Prvo, vidimo da smo rješenje opet dobili u zatvorenoj formi, što je vrlo lijepo. Drugo, matrica koju smo označili sa $\mathbf{X}^{+}$, definirana kao $\mathbf{X}^{+} = (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}$, je takozvani **pseudoinverz** (točnije: Moore-Penroseov inverz) matrice $\mathbf{X}$. Što je pseudoinverz? Pseudoinverz je poopćenje koncepta inverza matrice. Za razliku od običnog inverza, koji postoji samo za kvadratne matrice punog ranga, pseudoinverz **postoji za baš svaku matricu**. Kao poseban slučaj, ako je matrica $\mathbf{X}$ kvadratna i punog ranga, onda je pseudoinverz jednak običnom inverzu, $\mathbf{X}^{+} = \mathbf{X}^{-1}$. 14

Ovdje je sada važno primijetiti da je pseudoinverz sam po sebi rješenje problema najmanjih kvadrata. Naime, pogreška kvadratnog odstupanja bila je otpočetka ugrađena u ovaj postupak, jer smo od nje krenuli. To znači da, kada izračunamo pseudinverz, dobit ćemo parametre $\mathbf{w}$ koji nam daju hiperravninu $h(\mathbf{x}; \mathbf{w})$ koja je optimalna u smislu najmanjih kvadratnih odstupanja od označenih podataka.

Preciznije, rješenje $\mathbf{w}$ dano ovom jednadžbom je takvo da je vektor $\mathbf{X}\mathbf{w}$ u smislu kvadratnih odstupanja minimalno udaljen od vektora $\mathbf{y}$, odnosno to je rješenje koje minimizira L_2-normu $\|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2$. To je ujedno i razlog zašto gornje jednadžbe nazivamo "normalnima". 15

Ako je sustav jednadžbi pododređen i ima više rješenja, onda pseudoinverz daje rješenje s najmanjom normom $\|\mathbf{w}\|_2$. Ako je sustav preodređen, pseudoinverz daje rješenje koje minimizira $\|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2$, no ako je i takvih više, opet daje ono s najmanjom normom $\|\mathbf{w}\|_2$.

4.3 Računalni aspekti postupka najmanjih kvadrata

Dakle, ono što je ovdje dobro je da naš optimizacijski postupak daje rješenje u zatvorenoj formi. Međutim, u praksi taj izračun, koji je u domeni **numeričke matematike**, ipak može biti poprilično računalno zahtjevan.

Naime, primijetite da mi ovdje računamo **inverz** umnoška matrica dizajna sa samom sobom: $\mathbf{X}^{\mathrm{T}}\mathbf{X}$. Tu matricu nazivamo **Gramova matrica**. Gramova matrica može biti poprilično velika, pa izračun inverza može biti poprilično skup – tipično $\mathcal{O}(n^3)$ (npr., metodom LU-dekompozicije). Stoga je ključno pitanje: kojih je dimenzija matrica koju invertiramo, $(\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}$? Odgovor je: dimenzije su $(n+1) \times (n+1)$). Dakle, na složenost izračuna inverza matrice utječe samo broj značajki n, a ne i broj primjera N. To je dobro imati na umu. 16

Druga stvar koju valja primijetiti jest da, premda je Gramova matrica očigledno uvijek kvadratna (dimenzija $(n+1)\times(n+1)$), ona ne mora biti punog ranga. Naime, može se pokazati da je rang Gramove matrice jednak rangu matrice dizajna $\mathbf{X}$. Ako je taj rang jednak $n+1$, onda 17
je Gramova matrica punog ranga i pseudoinverz možemo izračunati kao što smo ovdje napisali, dakle kao $\mathbf{X}^+ = (\mathbf{X}^\mathrm{T}\mathbf{X})^{-1}\mathbf{X}^\mathrm{T}$. Međutim, ako je rang matrice dizajna manji od $n+1$ (a to će biti ako je matrica dizajna "plitka", odnosno kada $N < n+1$, tj. kada imamo manje primjera nego značajki), onda pseudoinverz ne možemo izračunati na ovaj način. Međutim, pseudoinverz tada možemo ga izračunati koristeći druge metode, tipično **rastavom na singularne vrijednosti (SVD)**. No, nećemo ovdje ići u detalje. Dovoljno je znati da pseudoinverz možemo uvijek izračunati, na ovaj ili onaj način. O računalnim aspektima izračuna pseudoinverza nastavit ćemo pričati idući put, kada ćemo se baviti prenaučenošću regresijskog modela i regularizacijom kao načinom da se ona spriječi.

5 Probabilistička interpretacija regresije

Gubitak smo definirali kao **kvadratno odstupanje**, odnosno kvadrat reziduala, a empirijsku pogrešku definirali smo kao sumu kvadrata reziduala. No, nekako smo nemušto opravdali uporabu baš kvadrata odstupanja. U strojnom učenju ne bismo se smjeli zadovoljiti takvim paušalnim objašnjenjima. Postoji li neki dublji razlog zašto bismo baš koristili kvadratni gubitak? Postoji! No da bismo to vidjeli, moramo o strojnom učenju pričati iz probabilističke perspektive. Mi ćemo tu perspektivu imati u drugoj polovici ovog predmeta, međutim već je sada korisno napraviti kratke izlete u svijet probabilističkog strojnog učenja. Vidjet ćemo da su svi ti pogledi isprepleteni, i da je korisno, a i nužno, neki algoritam razumijeti iz više perspektiva.

Pogledajmo, dakle, probabilističku interpretaciju linearne regresije. Naš model linearne regresije je:

$$h(\mathbf{x};\mathbf{w}) = \mathbf{w}^\mathrm{T}\mathbf{x}$$

Pretpostavimo da su naši označeni primjeri u stvarnosti generirani nekom funkcijom $f : \mathbb{R}^n \to \mathbb{R}$. Za tu funkciju često kažemo da je **"funkcija iza podataka"** (engl. *function underlying the data*). U stvarnosti nam, naravno, ta funkcija **nije poznata**.

Kao što već vrlo dobro znamo, u označenim podatcima neminovno postoji **šum**, koji se superponira na pravu vrijednost ishodišne funkcije. Dakle, ono što mi vidimo u oznakama jest vrijednost funkcije plus šum:

$$y^{(i)} = f(\mathbf{x}^{(i)}) + \varepsilon_i$$

I ovdje sada ulazimo u probabilističke vode. Modelirat ćemo šum kao slučajnu varijablu, distribuiranu po **normalnoj (Gaussovoj) distribuciji**, sa središtem u nuli i varijancom σ^2. To pišemo kao:

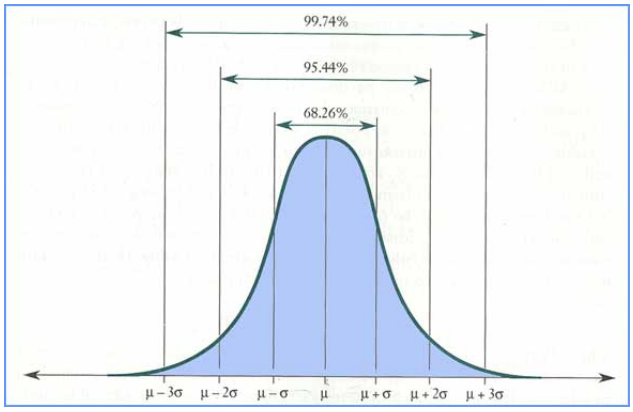
$$\varepsilon \;\sim\; \mathcal{N}(0,\sigma^2)$$

Naime, šum se, kao i mnogi drugi fenomeni u prirodi, može dobro opisati normalnom distribucijom, tako da je ova pretpostavka posve razumna. Nadalje, pretpostavljamo da je šum identičan za svaki pojedinačni primjer, $\varepsilon = \varepsilon_i$. 18 19

Prisjetimo se, Gaussova distribucija (iliti normalna razdioba) ima sljedeću **funkciju gustoće vjerojatnosti** (engl. *probability density function, PDF*):

$$p(Y = y|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi}\sigma}\exp\Big(-\frac{(y-\mu)^2}{2\sigma^2}\Big)\,.$$

Reda radi, prisjetimo se i kako ona izgleda:



Sada možemo napisati da će, za zadanu ulaznu vrijednost $\mathbf{x}$, izlazna vrijednost y biti distribuirana oko vrijednosti $f(\mathbf{x})$ (s odstupanjem uslijed šuma):

$$p(y|\mathbf{x}) = \mathcal{N}\big(f(\mathbf{x}), \sigma^2\big)$$

$p(y|\mathbf{x})$ je vjerojatnost da primjer $\mathbf{x}$ ima oznaku y, ako znamo da su podatci generirani funkcijom $f(\mathbf{x})$. (Zapravo, $\mathbf{x}$ je kontinuirana varijabla, pa je p gustoća vjerojatnosti oznake y, a ne vjerojatnost, ali to sad nije toliko važno, ideja stoji i dalje.)

Ovime smo modelirali vjerojatnost oznake za pojedinačni primjer. Idući korak je da modeliramo vjerojatnost svih oznaka u cijelom skupa označenih primjera $\mathcal{D}$. Prisjetimo se, skup označenih primjera $\mathcal{D}$ možemo rastaviti na matricu dizajna $\mathbf{X}$, koja sadrži samo primjere, te na vektor oznaka $\mathbf{y}$, koji sadrži oznake za sve primjere. Vjerojatnost da je skup primjera $\mathbf{X}$ označen oznakama $\mathbf{y}$ je:

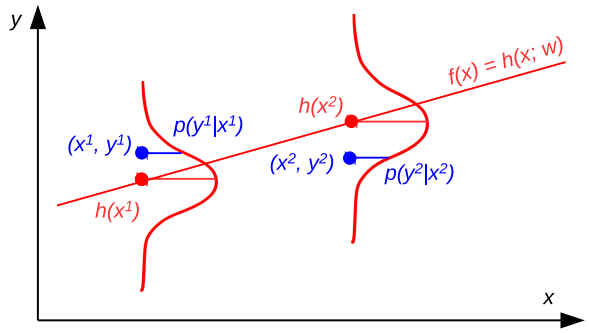
$$p(\mathbf{y}|\mathbf{X}) = \prod_{i=1}^{N} p(y^{(i)}|\mathbf{x}^{(i)}) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} \exp\Big(-\frac{(y^{(i)} - f(\mathbf{x}^{(i)}))^2}{2\sigma^2}\Big).$$

Kako smo došli do toga? Pretpostavili smo da su oznake primjera međusobno **nezavisne** i da dolaze iz **iste distribucije**. Ako su je tako, onda je vjerojatnost nekog skupa oznaka jednostavno umnožak vjerojatnosti pojedinačnih oznaka. Ta je pretpostavka ovdje razumna, i ona se često koristi u strojnom učenju, pod nazivom **nezavisno i identično distribuirane** varijable (engl. *independently and indentically distributed variables, IID*). 20

Gornja vjerojatnost govori nam dakle koliko je vjerojatno da skup primjera $\mathbf{X}$ bude označen sa $\mathbf{y}$ ako su primjeri generirani funkcijom $f(\mathbf{x})$. Međutim, prisjetimo se da je funkcija $f(\mathbf{x})$ nama nepoznata: mi ne znamo koja je funkcija generirala podatke. Zapravo, naš zadatak je upravo da pronađemo tu funkciju, i tu funkciju modeliramo našom hipotezom $h(\mathbf{x}|\mathbf{w})$. Dakle, tražimo hipotezu $h(\mathbf{x};\mathbf{w})$ koja je generirala primjere, odnosno hipotezu za koju vrijedi $h(\mathbf{x};\mathbf{w}) = f(\mathbf{x})$. To sad jednostavno znači da umjesto $f(\mathbf{x})$ možemo pisati $h(\mathbf{x};\mathbf{w})$. Dakle, imamo:

$$p(\mathbf{y}|\mathbf{X}) = \prod_{i=1}^{N} p(y^{(i)}|\mathbf{x}^{(i)}) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} \exp\Big(-\frac{(y^{(i)} - h(\mathbf{x}^{(i)};\mathbf{w}))^2}{2\sigma^2}\Big).$$

Ovo je, dakle, vjerojatnost da primjeri $\mathbf{X}$ imaju oznake $\mathbf{y}$ ako su nastali funkcijom $h(\mathbf{x};\mathbf{w})$. Također, kažemo da je to "vjerojatnost skupa podataka pod modelom" (u smislu: ako je hipoteza h istinita, vjerojatnost ovako označenih primjera je ta-i-ta). Grafički, to izgleda ovako: 21



Dakle, u svakoj točki $\mathbf{x}^{(i)}$ imamo jednu normalnu distribuciju sa središtem u $f(\mathbf{x}^{(i)})$. Oznaka $y^{(i)}$ može odstupati od one koju predviđa hipoteza, $h(\mathbf{x}^{(i)}; \mathbf{w})$, i to odstupanje modelirano je normalnom distribucijom. Najvjerojatnije je da će oznaka $y^{(i)}$ biti baš u sredini normalne distribucije, tj. da je $y^{(i)} = h(\mathbf{x}^{(i)}; \mathbf{w})$, jer je tamo gustoća vjerojatnosti $p(y^{(i)}|\mathbf{x}^{(i)})$ najveća. Što je oznaka $\mathbf{x}^{(i)}$ dalje od tog središta, to je ona manje vjerojatna. Ukupna vjerojatnost (odnosno gustoća vjerojatnosti) cijelog skupa oznaka $\mathbf{y}$ je umnožak gustoća vjerojatnosti $p(y^{(i)}|\mathbf{x}^{(i)})$ za sve točke $\mathbf{x}^{(i)}$. Umnožak će biti to veći (odnosno vjerojatnost skupa oznaka će biti to veća) što više oznaka $y^{(i)}$ bude nalazilo bliže središtu njima odgovarajućih normalnih razdioba. Analogno, umnožak će biti to veći (odnosno vjerojatnost skupa oznaka će biti to veća) što funkcija $h(\mathbf{x}; \mathbf{w})$ prolazi što bliže svakoj oznaci $y^{(i)}$.

Ovo zadnje opažanje je ključno: vjerojatnost skupa označenih primjera ovisi o funkciji $h(\mathbf{x}; \mathbf{w})$. Ta vjerojatnost će biti maksimalna ako hipoteza h prođe što bliže oznakama $\mathbf{y}$. Naš je cilj onda pronaći hipotezu h, odnosno težine $\mathbf{w}$, koje maksimiziraju vjerojatnost skupa primjera. Drugim riječima, želimo pronaći hipotezu takva da ona ove naše podatke koje imamo čini najvjerojatnijima. Na taj način mi zapravo pretpostavljamo da su podatci $\mathcal{D}$ kojima raspolažemo zapravo najvjerojatniji mogući: ako smo te podatke dobili, onda ćemo pretpostaviti da su upravo ti podatci upravo najvjerojatniji mogući, jer inače je vjerojatnije da bismo dobili neke druge podatke.

Želimo, dakle, naći hipotezu koja **maksimizira vjerojatnost oznaka**. U nastavku će nam biti lakše prebaciti se u logaritamsku domenu (a vidjet ćemo uskoro i zašto). Također, umjesto hipoteze h možemo pisati težine $\mathbf{w}$, jer zapravo njih tražimo. Vjerojatnost oznaka $\mathbf{y}$ za težine $\mathbf{w}$ je:

$$\ln p(\mathbf{y}|\mathbf{w}) = \ln \prod_{i=1}^{N} p(y^{(i)}|x^{(i)}) = \ln \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{ - \frac{\left(y^{(i)} - h(\mathbf{x}^{(i)}; \mathbf{w})\right)^2}{2\sigma^2} \right\}$$

Sada možemo malo srediti izraz (logaritmom ući u umnožak), pa dobivamo:

$$\underbrace{-N \ln(\sqrt{2\pi}\sigma)}_{=\text{konst.}} - \frac{1}{2\sigma^2} \sum_{i=1}^{N} \left(y^{(i)} - h(\mathbf{x}^{(i)}; \mathbf{w})\right)^2$$

Kako nas zanima maksimizacija po $\mathbf{w}$, to ovaj prvi član te σ^2 možemo zanemariti jer su konstantni. Ostaje nam:

$$-\frac{1}{2} \sum_{i=1}^{N} \left(y^{(i)} - h(\mathbf{x}^{(i)}; \mathbf{w})\right)^2$$

Prisjetimo se, ovo je vjerojatnost koju želimo **maksimizirati**. To znači da izraz nakon znaka minus želimo **minimizirati**. A taj izraz je upravo **pogreška kvadratnog odstupanja**! Drugim riječima, uz pretpostavku normalno distribuiranog šuma, hipoteza koja će maksimizirati vjerojatnost da su podatci označeni tako kako jesu jest upravo ona koja minimizira kvadratno odstupanje predviđenih i stvarnih oznaka. I to je, konačno, probabilističko opravdanje za uporabu kvadratne funkcije gubitka!

Postupak koji smo odradili ovdje može se sažeti u sljedećem: izrazili smo vjerojatnost oznaka tako da smo svaku oznaku modelirali kao normalno distribuiranu varijablu sa srednjom vrijednošću jednakom predviđanju hipoteze, i zatim smo izrazili vjerojatnost skupa oznaka, pretpostavivši IID. Zatim tražimo parametre hipoteze koji maksimiziraju tu vjerojatnost. To nas je dovelo do izraza koji želimo maksimizirati, za koji se ispostavio da je jednak negativnoj empirijskoj pogrešci, koju želimo minimizirati. Ovaj princip nije bezveze, nego je jedan univerzalan princip kako u strojnom učenju možemo definirati funkcije pogreške. O tome ćemo još govoriti u nekim narednim predavanjima.

Sažetak

- Regresija služi predviđanju **numeričkih vrijednosti** (zavisne varijable) na temelju ulaznih primjera (nezavisne varijable)
- Kod **linearne regresije**, izlazna vrijednost je linearna kombinacija ulaznih vrijednosti
- Funkcija gubitka je **kvadratni gubitak** (kvadratna razlika predviđene i stvarne oznake)
- Optimizacija se provodi **postupkom najmanjih kvadrata**, koji izračunavamo pomoću **pseudoinverza** matrice dizajna
- Uz pretpostavku normalno distribuiranog šuma, rješenje najmanjih kvadrata istovjetno je maksimizaciji vjerojatnosti oznaka, što daje **probabilističko opravdanje** za kvadratni gubitak

Bilješke

[1] Ovo predavanje je lepršavija varijanta poglavlja o regresiji iz (nedovršene) skripte: http://www.fer.unizg.hr/_download/repository/SU-Regresija.pdf.

[2] Regresija je jedan od osnovnih i nevjerojatno moćnih alata statističkog zaključivanja, koji se pogotovo često koristi u analizi podataka u društvenim znanostima i medicini. Želite li uroniti u regresiju iz aspekta primijenjene statistike, preporučam vam da krenete od (Kutner et al., 2005), a zatim da pročitate (Harrell Jr, 2015). Računajte na to kao višemjesečni projekt. Mi u strojnom učenju nećemo inati ništa za reći o statističkom pristupu regresiji.

[3] U pogledu naziva regresijskih koeficijenata u statistici postoji terminološka nekonzistentnost: ponekad se ti koeficijenti nazivaju betama (beta-koeficijentima) i označavaju sa β, dok se nekad pod tim nazivom podrazumijevaju isključivo **standardizirani koeficijenti**: koficijenti dobiveni nakon transformacije značajki tako da imaju srednju vrijednost nula i standardu devijaciju jednaku 1, tj. transformacije na z-vrijednosti. Standardizirani koeficijenti omogućavaju nam da uspoređujemo relativnu važnost neke značajke u odnosu na sve druge značajke.

[4] Vidjet ćemo da je ovo česta tema – često u strojnom učenju malo manipuliramo matematičke izraze kako bi se u optimizaciji nešto pokratilo ili pojednostavilo. Naravno, to je legitimno, dok god ne utječe na krajnje rješenje. Npr., množenje funkcije koju minimiziramo konstantnim faktorom nema utjecaja na minimizator funkcije. Primjena **monotone transformacije** (npr., logaritamske funkcije) na funkciju koju minimiziramo također nema utjecaja na minimizator funkcije, a može pojednostaviti algebarski oblik funkcije koju minimiziramo i povećati numeričku stabilnost rješenja.

[5] U nastavku je puni izvod za parametre (w_0, w_1) jednostavne regresije. Pronađimo najprije minimum

s obzirom na parametar w_0:

$$\frac{\partial}{\partial w_0}\Big[\frac{1}{2}\sum_i^N \big(y^{(i)} - (w_1 x^{(i)} + w_0)\big)^2\Big] = \tag{1}$$
$$\sum_i^N (-y^{(i)} + w_1 x^{(i)} + w_0) = -\sum_i^N y^{(i)} + w_1 \sum_i^N x^{(i)} + N w_0 = 0\,.$$

Rješavanjem za w_0 i uvrštenjem $\bar{x} = \sum_i^N x^{(i)}/N$ i $\bar{y} = \sum_i^N y^{(i)}/N$ dobivamo

$$w_0 = \frac{1}{N}\big(\sum_i^N y^{(i)} - w_1 \sum_i^N x^{(i)}\big) = \frac{1}{N}\sum_i^N y^{(i)} - w_1 \frac{1}{N}\sum_i^N x^{(i)} = \bar{y} - w_1 \bar{x}\,. \tag{2}$$

Za w_1 dobivamo

$$\frac{\partial}{\partial w_1}\Big[\frac{1}{2}\sum_i^N \big(y^{(i)} - (w_1 x^{(i)} + w_0)\big)^2\Big] = \tag{3}$$
$$\sum_i^N -x^{(i)}(y^{(i)} - w_1 x^{(i)} - w_0) = -\sum_i^N x^{(i)} y^{(i)} + w_1 \sum_i^N (x^{(i)})^2 + w_0 \sum_i^N x^{(i)} = 0\,.$$

Uvrštenjem (2) u (3) te zamjenom $\sum_i^N x^{(i)} = N\bar{x}$ dobivamo

$$-\sum_i^N x^{(i)} y^{(i)} + w_1 \sum_i^N (x^{(i)})^2 + (\bar{y} - w_1 \bar{x}) N \bar{x} = w_1\big(\sum_i^N (x^{(i)})^2 - N\bar{x}^2\big) + N\bar{x}\bar{y} - \sum_i^N x^{(i)} y^{(i)} = 0\,.$$

iz čega slijedi

$$w_1 = \frac{\sum_i^N x^{(i)} y^{(i)} - N\bar{x}\bar{y}}{\sum_i^N (x^{(i)})^2 - N\bar{x}^2}\,.$$

6 Jednostavan opis što je to rješenje u zatvorenoj formi: http://riskencyclopedia.com/articles/closed_form_solution/

7 A što ako bismo željeli raditi multivarijatnu regresiju, gdje bismo za jedan primjer predviđali više brojeva (npr., cijenu nekretnine ali i energetsku potrošnju te iste nekretnine)? U principu, to uvijek možemo napraviti pomoću univarijatne regresije, tako da za svaku izlaznu varijablu trenirano zaseban model univarijatne regresije. Međutim, sa statističkog stajališta dekompozicija multivarijatne regresije u skup univarijatnih regresijskih modela nije uvijek zadovoljavajuća, jer između izlaznih varijabli postoje zavisnosti, pa će i pogreške univarijatnih regresijskih modela biti donekle korelirane, a to, ako želimo raditi propisno statističko zaključivanje, svakako treba uzeti u obzir. Više ovdje: https://stats.stackexchange.com/q/254254/93766

8 U literaturi iz strojnog učenja (npr., u Bishopovog knjizi, a također i u našoj skripti) ponekad se uvodi posebna notacija za tako proširene vektora $\mathbf{x}$ i $\mathbf{w}$:

$$\tilde{\mathbf{x}} = (1, \mathbf{x})$$
$$\tilde{\mathbf{w}} = (w_0, \mathbf{w})$$

Onda model možemo definirati kao: $h(\mathbf{x}; \tilde{\mathbf{w}}) = \tilde{\mathbf{w}}^{\mathrm{T}} \tilde{\mathbf{x}}$. Jednostavnosti radi, mi to u ovim materijalima nećemo raditi. Vjerujem da će iz konteksta biti jasno radi li se o proširenom vektoru ili ne.

9 Zapravo, postupak **obični najmanji kvadrati** (engl. *ordinary least squares, OLS*) samo je jedna vrsta postupka **najmanjih kvadrata** (engl. *least squares, LS*). Potonji uključuju linearne i nelinearne postupke, ovisno o tome ovise li reziduali o prediktorima linearno ili nelinearno. Postupak običnih kvadrata pripada skupini **linarnih najmanjih kvadrata** (engl. *linear least squares, LLS*), kojoj pripadaju i postupci **težinski najmanji kvadrati** (engl. *weighted least squares, WLS*) i poopćeni najmanji kvadrati (engl. *generalized least squares, GLS*).

10 Prisjetimo se, **rang** (engl. *rank*) matrice $\mathbf{A}$, označen kao $\mathrm{rank}(\mathbf{A})$, je broj linearno nezavisnih stupaca matrice $\mathbf{A}$, i taj je broj jednak broju linearno nezavisnih redaka matrice $\mathbf{A}$. Kvadratna matrica $\mathbf{A}$

dimenzija $n \times n$ je **punog ranga**, $\text{rank}(\mathbf{A}) = n$, ako i samo ako nema linearno zavisnih redaka niti linearno zavisnih stupaca. Matrica ima puni rang ako i samo ako nije **singularna** i ako i samo ako ima **inverz**. Npr., ova matrica ima rang 3, dakle puni rang:

$$\begin{pmatrix} 1 & 5 & 1 \\ 2 & 0 & 4 \\ 3 & 2 & 6 \end{pmatrix}$$

dok ova matrica nema puni rang, već rang 2, jer je treći stupac linearna kombinacija prvog:

$$\begin{pmatrix} 1 & 5 & 2 \\ 2 & 0 & 4 \\ 3 & 2 & 6 \end{pmatrix}$$

Pravokutna matrica $\mathbf{B}$ dimenzija $m \times n$ je punog ranga ako ima maksimalni mogući rang, tj. ako $\text{rank}(\mathbf{B}) = \min(m, n)$. Npr., ova matrica ima puni rang, jer ima rang 2, što je maksimalni mogući rang za matricu dimenzija 3×2:

$$\begin{pmatrix} 1 & 5 & 1 \\ 2 & 0 & 4 \end{pmatrix}$$

dok ova matrica nema puni rang, jer ima rang jednak 1:

$$\begin{pmatrix} 1 & 5 & 2 \\ 2 & 10 & 4 \end{pmatrix}$$

[11] Kronecker-Capellijev teorem: https://en.wikipedia.org/wiki/Rouch%C3%A9%E2%80%93Capelli_theorem

[12] Ovo je dobro mjesto da napomenemo da kad kažemo "skup" označenih primjera, zapravo i ne mislimo doista na skup u smislu kolekcije elemenata koji se ne ponavljaju. Zapravo, u strojnom učenju skup elemenata je N-torka. Redoslijed uglavnom nije bitan, ali ponavljanja su moguća, i općenito je moguće da se jedan te isti primjer u skupu pojavljuje više puta te moguće s različitim oznakama.

[13] Matrični račun (odnosno linearna algebra općenitije) jedan je od osnovnih matematičkih alata u strojnom učenju, premda ćemo se mi u ovom predmetu vrlo ograničeno koristiti matričnim računom (i linearnom algebrom). Dobar pregled matričnog računa u možete naći u https://www.math.uwaterloo.ca/~hwolkowi/matrixcookbook.pdf. Standardna referenca za algoritme za operacije nad matricama je (Van Loan and Golub, 1983). Standardna (i didaktički izvrsna) referenca za linearnu algebru je (Strang et al., 1993).

[14] Pseudoinverz je prvi definirao Eliakim Moore 1921. godine, ponovo ga je otkrio Arne Bjerhammar 1951., a Roger Penrose ga je "ponovo ponovo otkrio" (?) 1955. godine. Roger Penrose je britanski matematičar, fizičar i filozof, poznat po raznim lijepim "Penroseov X" stvarima, na primjer "Penroseovom trokutu", nemogućem trokutu koji je inspiriran sličnim grafikama nizozemskog slikara Eschera, "Penroseovom uzorku", "Penroseovom procesu", itd. U AI krugovima poznat je po svojim radovima na temu filozofije uma, točnije povezanosti svijesti i fizike. Zainteresiranima preporučujem pročitati njegovu knjigu *Carev novi um* (Penrose, 1989), u kojoj argumentira protiv mogućnosti inteligencije današnjih računala kao algoritamski deterministčkih sustava i time u osnovi protiv mogućnosti jake umjetne inteligencije.

[15] L_2**-norma** (druga norma, euklidska norma) vektora $\mathbf{x}$ je $\|\mathbf{x}\|_2 = \sqrt{\mathbf{x}^{\mathrm{T}}\mathbf{x}}$. Funkciju pogreške $E(\mathbf{w}|\mathcal{D})$ definirali smo kao sumu kvadrata reziduala (razlike između predikcije modela i ciljne oznake), a to je upravo kvadrat L_2-norme vektora $(\mathbf{X}\mathbf{w} - \mathbf{y})$:

$$E(\mathbf{w}|\mathcal{D}) = \frac{1}{2}(\mathbf{X}\mathbf{w} - \mathbf{y})^{\mathrm{T}}(\mathbf{X}\mathbf{w} - \mathbf{y}) = \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2$$

Kvadriranje norme je monotona transformacija, pa je rješenje koje minimizira $\|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2$ isto ono koje minimizira $\|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2$. Zato možemo reći da je minimizacija $E(\mathbf{w}|\mathcal{D})$ jednaka minimizaciji L_2-norme vektora $(\mathbf{X}\mathbf{w} - \mathbf{y})$, tj. druge norme vektora reziduala.

[16] **Gramova matrica** skupa vektora je matrica unutarnjih produkata tih vektora (odnosno skalarnih produkata, kada govorimo o euklidskom prostoru). Ako je $\mathbf{A}$ matrica dimenzija $m \times n$ sastavljena

od m vektor-stupaca dimenzija n, $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_m]$, onda je Gramova matrica definirana kao $\mathbf{G} = \mathbf{A}^{\mathrm{T}}\mathbf{A}$. To je matrica čiji je element $\mathbf{G}_{i,j}$ jednak skalarnom produktu vektora $\mathbf{a}_i$ s vektorom $\mathbf{a}_j$, tj. $\mathbf{a}_i^{\mathrm{T}}\mathbf{a}_j$. Gramova matrica je simetrična, te je pozitivno semidefinitna ($\mathbf{x}^{\mathrm{T}}\mathbf{G}\mathbf{x} \geqslant 0$). Još važnije, svaka pozitivno semidefinitna matrica je Gramova matrica za neki skup vektora, što znači da svaka pozitivno semidefinitna matrica odgovara skalarnom produktu vektora u nekom vektorskom prostoru. Ovo će nam biti jako korisna spoznaja kada ćemo pričati o stroju potpornih vektora i jezgrenim strojevima. Općenito, koncept Gramove matrice često se pojavljuje u strojnom učenju.

17 Vrijedi $\mathrm{rank}(\mathbf{A}^{\mathrm{T}}\mathbf{A}) = \mathrm{rank}(\mathbf{A})$. Dokaz: https://math.stackexchange.com/q/349738/273854

18 No, pogledajte zanimljivu diskusiju o sveprisutnosti i opravdanosti normalne distribucije: https://stats.stackexchange.com/q/204471/93766. Jedan od žešćih kritičara pretjeranog oslanjanja na normalnu distribuciju u pokušajima da se predvidi nepredvidivi ili rijetki događaji je Nassim Taleb; preporučam pročitati njegovog *Crnog labuda* (Taleb, 2007).

19 Ova se pretpostavka u statistici naziva **homoskedastičnost** (engl. *homoscedasticity*) reziduala. To je jedna od osnovnih pretpostavki modela linearne regresije (pored pretpostavke o normalnosti reziduala, linearnosti zavisne varijable u odnosu na nezavisne i nepostojanju multikolinearnosti, uz temeljne pretpostavke o uključenosti svih varijabli u model i nezavisnosti opažanja). Ako ova pretpostavka nije ispoštovana, ne postoje garancije da je statistička analiza (intervali pouzdanosti, statistički testovi) valjana.

20 Pretpostavka o **nezavisno i identično distribuiranim (IID)** primjerima i/ili njihovim oznakama je centralna pretpostavka mnogih algoritama strojnog učenja, i zapravo svih algoritama koje ćemo raditi na ovom predmetu. Slučajne variable X i Y su IID ako su međusobno nezavisne i imaju istu distribuciju. Formalno, slučajne varijable X i Y su identično distribuirane akko $P_X(X) = P_Y(Y)$ i ako $P(X, Y) = P(X)P(Y)$.

21 Ovo je zapravo **funkcija izglednosti** (engl. *likelihood function*) hipoteze h na skupu primjera za učenje $\mathcal{D}$. Izglednost je jedan od središnjih koncepata strojnog učenja i statistike, koji ćemo formalno uvesti malo kasnije. Ljude obično buni razlika između vjerojatnosti i izglednosti. Razlika se svodi na to što je fiksirano a što varira. Izglednost hipoteze h uz fiksirane podatke $\mathcal{D}$ je vjerojatnost podataka $\mathcal{D}$ uz fiksiranu hipotezu h. Dakle, da izglednost od h jest vjerojatnost, samo što to nije vjerojatnost od h nego vjerojatnost od $\mathcal{D}$. Međutim, kod izglednosti hipoteza h je ona koja varira, dok je $\mathcal{D}$ fiksirana varijabla.

Literatura

F. E. Harrell Jr. *Regression modeling strategies: with applications to linear models, logistic and ordinal regression, and survival analysis*. Springer, 2015.

M. H. Kutner, C. J. Nachtsheim, J. Neter, W. Li, et al. *Applied linear statistical models*, volume 5. McGraw-Hill Irwin New York, 2005.

R. Penrose. *The Emperor's New Mind: Concerning Computers, Minds, and the Laws of Physics*. Oxford University Press, 1989.

G. Strang, G. Strang, G. Strang, and G. Strang. *Introduction to linear algebra*, volume 3. Wellesley-Cambridge Press Wellesley, MA, 1993.

N. N. Taleb. *The black swan: The impact of the highly improbable*, volume 2. Random house, 2007.

C. F. Van Loan and G. H. Golub. *Matrix computations*. Johns Hopkins University Press Baltimore, 1983.