UNIVERZITET CRNE GORE

Elektrotehnički fakultet, Podgorica

Materijal sa trećeg termina predavanja iz

OSNOVA MAŠINSKOG UČENJA I VJEŠTAČKE INTELIGENCIJE

Klasifikacija. Logistička regresija.

Prof. dr Vesna Popović-Bugarin

Podgorica, 2022.

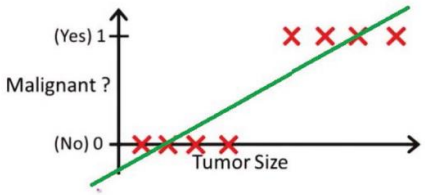
Klasifikacija

Klasifikacija je također nadgledano učenje. Razlika u odnosu na linearnu regresiju je u tipu oznaka, odnosno funkcije koju želimo estimirati. Kod klasifikacije, oznake se odnose na klase kojima pripadaju pojedini odbirci i diskretne su veličine. Dakle, naš zadatak je da na osnovu karakteristika podataka i njihovih oznaka izvršimo predviđanje diskretne veličine. Kada je u pitanju klasifikacija možemo imati proizvoljan broj klasa, ali će se klasifikacija uvoditi tako što se posmatra binarni problem – **binarna klasifikacija**, kada imamo samo dvije klase, a na kraju će se odraditi generalizacija za slučaj kada imamo više klasa - **multiklasne klasifikacije**.

Klasifikacija ima veliki broj primjena. Koristi se za prepoznavanje rukom pisanih cifara, prepoznavanje oblika i slično. Neki od primjera binarne klasifikacije su: klasifikacija tumora na maligne i benigne, klasifikacija mejlova na SPAM i regularne mejlove (nije SPAM), detekcija zloupotreba kreditni kartica klasifikacijom transakcija na regularne transakcije i transakcije u kojima je došlo do zloupotrebe kartica. Na osnovu navedenih primjera, može se zaključiti da je u binarnoj klasifikaciji zadatak da identifikujemo prisustvo određene pojave (maligni tumor, SPAM, zloupotreba). Tu klasu najčešće nazivamo **pozitivnom klasom** i uzorak koji predstavlja instancu ove klase ima oznaku 1 ($y^{(i)} = 1$). Za grafički prikaz se često koristi marker '+' kao oznaka pozitivne klase. Odsustvo pojave/klase za kojom se traga se naziva **negativnom klasom** (nije maligni tumor, nije SPAM, nije zloupotreba) i oznaka ovakvih podataka ima vrijednost 0 ($y^{(i)} = 0$), dok se za vizuelizaciju često koristi marker '-'. Dakle, u podacima kojim raspolažemo imaćemo za svaki uzorak karakteristike $x^{(i)}$ i odgovarajuće oznake $y^{(i)}$, koje mogu uzeti jednu od dvije moguće vrijednosti (`0, 1`).

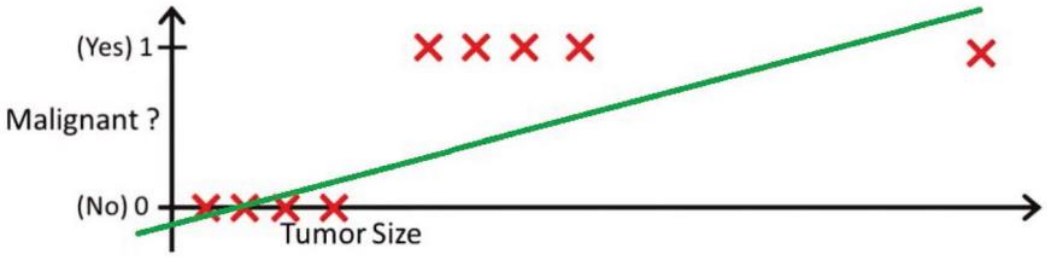
Logistička regresija

Pošavši od činjenice da se i dalje radi o problemu nadgledanog učenja i predikciji vrijednosti izlaza, a zanemarujući činjenicu da je izlaz dikretna veličina koja može imati jednu od dvije moguće vrijednosti, mogli bismo pokušati da odradimo predikciju koristeći algoritam linearne regresije. Zamislimo da imamo skup označenih podataka koji nam predstavljaju podatke o malignitetu tumora dojke u zavisnosti od njegove veličine, pri čemu je 0 oznaka za benigni tumor, a 1 oznaka za maligni tumor. Mogli bismo pokušati da estimiramo izlaznu vrijednost linearnom funkcijom, koja predstavlja zavisnost od veličine tumora. Dobili bismo nešto kao na slici Slika1.



Slika 1 Ulustracija "klasifikacije" upotrebom linearne regresije[1]

Na prvi pogled izgleda da se dobija dosta dobra klasifikacija uz prag od 0.5. Sve uzorke za koje dobijemo izlaznu vrijednost veću od 0.5 proglasimo malignim, a one za koju dobijemo vrijednost manju od 0.5 benignim. Međutim, lako se može odabrati primjer kada nam ovaj metod daje lošu klasifikaciju. **Error! Reference source not found.** predstavlja slučaj kada među podacima imamo jedan *outlier* (uzorak sa karakteristikama koje značajno odstupaku od karakteristika ostalih uzoraka), koji će nam pomjeriti liniju kojom pokušavamo predvidjeti izlaznu vrijednost, i voditi ka lošoj klasifikaciji, za dati prag odlučivanja.



Slika 2 Primjer uticaja outlier-a na "klasifikaciju" rađenu linearnom regresijom[2]

Razlog za lošu klasifikaciju nije loše odabran prag odlučivanja, već činjenica da linearna regresija nije pogodna za klasifikaciju. Pored izuzetne osjetljivosti na vrijednost odabranog praga, problem je i činjenica da pokušavamo da estimiramo funkciju za koju znamo da ima diskretne vrijednosti, $y \in \{0,1\}$, funkcijom koja može uzeti vrijednosti veće od 1 i manje od 0.

Prirodno bi bilo za binarnu klasifikaciju koristiti heurističku funkciju koja ima osobinu da je:

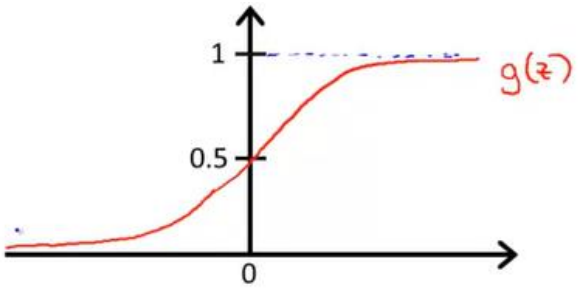
$$0 \leq h_\theta(x) \leq 1.$$

Jedna takva funkcija je **sigmoid funkcija**, poznata i kao **logistička funkcija:**

$$g(z) = \frac{1}{1+e^{-z}},$$

prikazana na **Error! Reference source not found.**.

[1] Slika preuzeta sa https://www.coursera.org/learn/machine-learning
[2] Slika preuzeta sa https://www.coursera.org/learn/machine-learning



[3]

Slika 3 Sigmoid/logistička funkcija

Sa grafika sigmoid funkcije se može primijetiti da $g(z) \to 1$ kada $z \to \infty$, odnosno da $g(z) \to 0$ kada $z \to -\infty$. Dakle, $g(z)$ uzima vrijednosti u opsegu od 0 do 1.

Sada se hipoteza može zapisati kao:

$$h_{\boldsymbol{\theta}}(\mathbf{x}) = g\left(\boldsymbol{\theta}^T \mathbf{x}\right) = \frac{1}{1+e^{-\boldsymbol{\theta}^T \mathbf{x}}},$$

gdje je:

$$\boldsymbol{\theta}^T \mathbf{x} = \theta_0 + \sum_{j=1}^{n} \theta_j x_j .$$

I dalje važi da je $x_0 = 1$, i n – broj karakteristika.

Ovako odabrana hipoteza $h_{\boldsymbol{\theta}}(\mathbf{x})$ se interpretira kao estimirana vjerovatnoća da će za date parametre $\boldsymbol{\theta}$ i karakteristike $\mathbf{x}$ izlaz y imati vrijednost 1.

Za prethodni primjer klasifikacije tumora dojke, ukoliko bismo imali uzorak:

$$\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ VelicinaTumora \end{bmatrix}$$

i za takav ulaz primjenjujući istreniranu hipotezu dobijemo da je $h_\theta(x) = 0.7$, to se tumači kao vjerovatnoća od 70 % da pacijent čiji su podaci posmatrani ima maligni tumor. Dakle, vrijednost hipoteze je uslovna vjerovatnoća da *y=1* za date karakteristike $\mathbf{x}$ i parametre $\boldsymbol{\theta}$, što zapisujemo kao:

$$h_\theta(x) = p(y=1 \mid x;\theta).$$

Obzirom da znamo da izlaz može uzeti jednu od dvije moguće vrijednosti *y=0* ili *y=1* zaključujemo da mora biti:

$$h_\theta(x) = p(y=1 \mid x;\theta) = 1 - p(y=0 \mid x;\theta)$$
$$p(y=1 \mid x;\theta) + p(y=0 \mid x;\theta) = 1$$

Za prethodnog pacijenta je vjerovatnoća da je na osnovu datih osobina tumor benigan jednaka 0.3 ili 30 %.

[3] Slika preuzeta sa https://www.coursera.org/learn/machine-learning

Granica odlučivanja (engl. Decision Boundury)

Odabirom sigmoid funkcije za predstavljanje hipoteze ograničena je funkcija na vrijednosti između 0 i 1. Međutim, i dalje je hipoteza kontinualna funkcija i može uzeti bilo koju vrijednost u opsegu 0-1. Dakle, da bi se izlaz povezao sa oznakama klase, potrebno je uvesti prag koji će na osnovu kontinualne vrijednosti heurističke funkcije izvršiti klasifikaciju i dodijeliti posmatrani uzorak pozitivnoj ili negativnoj klasi. Imajući na umu interpretaciju vrijednosti heurističke funkcije kao uslovne vjerovatnoće da analizirani uzorak sa datim karakteristikama pripada pozitivnoj klasi, i posmatrajući izgled heurističke funkcije, Slika 3, zaključuje se da važi:

$$y = \begin{cases} 1, & za\ h_\theta(x) \geq 0.5 \\ 0 & za\ h_\theta(x) < 0.5 \end{cases}$$

odnosno:

$$y = \begin{cases} 1, & za\ \boldsymbol{\theta}^T \mathbf{x} \geq 0 \\ 0 & za\ \boldsymbol{\theta}^T \mathbf{x} < 0 \end{cases}$$

Navedeni uslov nam defineše **granicu odlučivanja**. Svaki uzorak koji je sa gornje strane granice odlučivanja pridružuje se pozitivnoj klasi, dok se odbirci koji su sa druge strane granice odlučivanja klasifikuju kao pripadnici negativne klase.

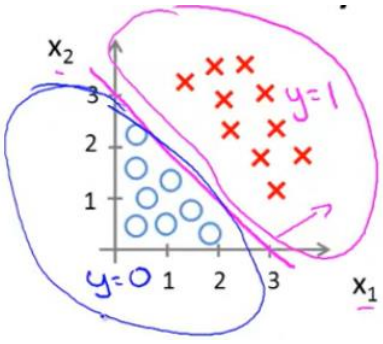
Na primjer, ukoliko imamo slučaj klasifikacije u zavisnosti od dvije karakteristike i na osnovu trening podataka estimiramo parametre heurističke funkcije, koja u ovom slučaju ima oblik $h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$ i dobijemo vrijednosti:

$$\theta = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix},$$

granica odlučivanja bi bila:

$$-3 + x_1 + x_2 \geq 0,\ x_1 + x_2 \geq 3.$$

Ovo se može ilustrovati kao na



Slika 4 Ilustracija granice odlučivanja[4]

U datom primjeru, svaki uzorak koji se na osnovu svojih karakteristika nalazi sa gornje strane granice odlučivanja, proglašava se pripadnikom pozitivne klase, dok se odbirci sa donje strane granice odlučivanja proglašavaju pripadnicima negativne klase. Ovo je

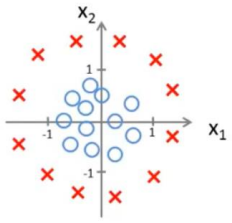
[4] Slika preuzeta sa https://www.coursera.org/learn/machine-learning

primjer koji služi kao ilustracija, dok se treba imati na umu da nakon što se odrede/nauče parametri hipoteze, granica odlučivanja postaje fiksna. Granica odlučivanja je svojstvo parametara hipoteze, a ne data seta i ne mora se crtati data set da bi se nacrtale granice odlučivanja kada se odrede parametri hipoteze.

Kod linearne regresije smo zaključili da nam neće uvijek biti dovoljna linearna zavisnost od karakteristika, pa smo uveli polinomijalnu regresiju, uzimajući stepene karakteristika kao ulazne podatke. Na sličan način, kod logističke regresije, neće uvijek granica odlučivanja biti definisana linearnom funkcijom. U ovom slučaju se uzimaju, kao kod linearne regresije, stepeni karakteristika i dobija se složenija granica odlučivanja.

Jedan primjer bi bila klasifikacija podataka koji su predstavljeni sa dvije karakteristike, Slika 5. Nakon vizuelizacije podataka, zaključuje se da se ne mogu razdvojiti linearnom funkcijom, već zavise od kvadrata, ili višeg reda analiziranih karakteristika. Ukoliko bi se uzela zavisnost od kvadrata dostupnih karakteristika, heuristička funkcija bi imala oblik:

$$h_{\boldsymbol{\theta}}(\mathbf{x}) = g\left(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2\right).$$



Slika 5 Klasifikacija sa nelinearnom granicom odlučivanja[5]

Za estimirane vrijednosti parametara:

$$\boldsymbol{\theta} = \begin{bmatrix} -1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix},$$

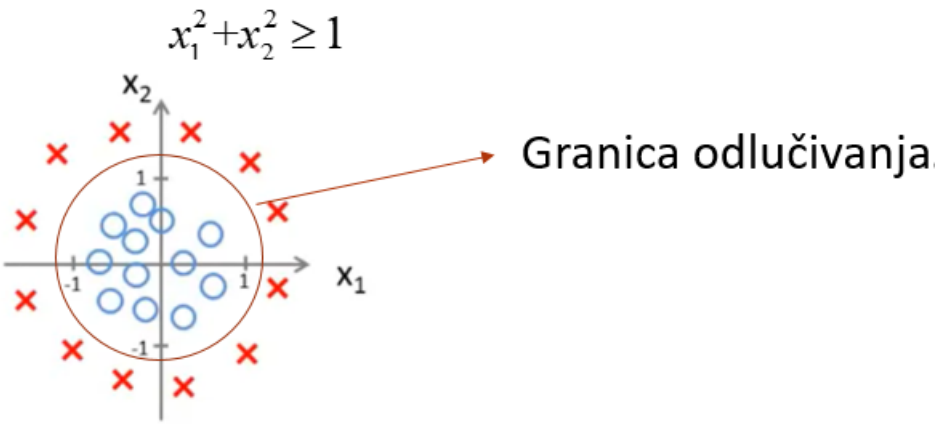
dobijamo da je uzorak pripadnik pozitivne klase ukoliko važi:

$$h_{\boldsymbol{\theta}}(\mathbf{x}) = -1 + x_1^2 + x_2^2 \geq 0\,,$$

čime se definiše granica odlučivanja kao:

$$x_1^2 + x_2^2 \geq 1.$$

[5] Slika preuzeta sa https://www.coursera.org/learn/machine-learning

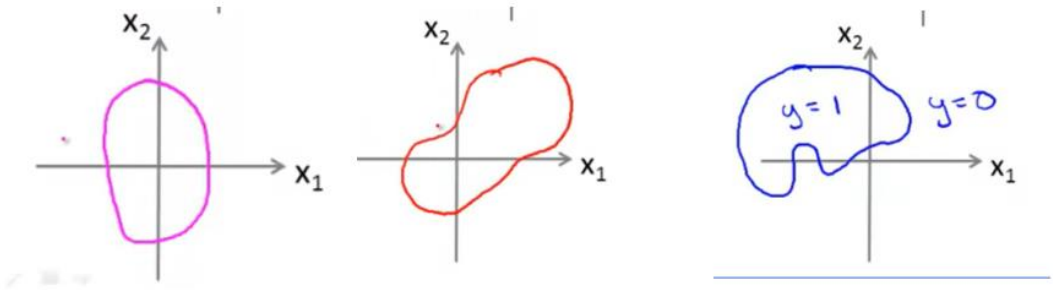


Slika 6 Ilustracija nelinearne granice odlučivanja.[6]

U opštem slučaju zavisnost heurističke funkcije od karakteristika analiziranog problema mogu biti još složenija:

$$h_{\boldsymbol{\theta}}(\mathbf{x}) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_1^2 x_2 + \theta_5 x_1^2 x_2^2 + \theta_6 x_1^3 x_2 + \cdots)$$

a samim tim i granica odlučivanja može biti proizvoljnog oblika. Neki od primjera granica odlučivanja su dati na slici 7. Svi uzorci koji se nalaze unutar zatvorene krive linije pripadaju pozitivnoj klasi, sve što je sa njene spoljašnje strane, predstavlja instance negativne klase.



Slika 7 Proizvoljne granice odlučivanja za slučaj dvije karakteristike[7]

Kroz prethodne primjere je ilustrovana granica odlučivanja koja se dobija nakon što se istrenira algoritam logističke regresije. Postavlja se pitanje na koji način se vrši njegovo treniranje, odnosno određivanje optimalnih vrijednosti za parametre θ, ukoliko imamo *m* označenih podataka u trening setu:

$$\left\{\left(\mathbf{x}^{(1)}, y^{(1)}\right), \left(\mathbf{x}^{(2)}, y^{(2)}\right), \cdots, \left(\mathbf{x}^{(m)}, y^{(m)}\right)\right\}$$

gdje je:

[6] Slika preuzeta sa https://www.coursera.org/learn/machine-learning
[7] Slika preuzeta sa https://www.coursera.org/learn/machine-learning

$$\mathbf{x} \in \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix} \qquad x_0 = 1, \quad y \in \{0,1\}$$

i heuristička funkcija:

$$h_\theta(x) = \frac{1}{1+e^{-\boldsymbol{\theta}^T \mathbf{x}}}.$$

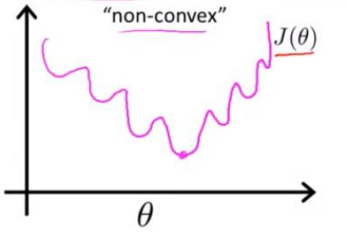
Kod linearne regresije smo vidjeli da je optimalna vrijednost bila ona vrijednost parametara za koje se dobija najmanja srednja kvadratna greška. Dakle, srednja kvadratna greška nam je bila ukupna funkcija cijene, odnosno optimizaciona funkcija. Pisali smo je kao:

$$J(\theta_0, \theta_1, \theta_2, \ldots, \theta_n) = \frac{1}{m}\sum_{i=1}^{m} \frac{1}{2}\left(h_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) - y^{(i)}\right)^2$$

Podsjetimo se dvojka je dodata u imenilac radi olakšavanja kasnijeg računa, a ne utiče na minimizaciju. Usrednjavala se kvadratna greška pojedinih uzoraka definisana kao:

$$\text{Cost}(h_{\boldsymbol{\theta}}(x^{(i)}), y^{(i)}) = \frac{1}{2}\left(h_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) - y^{(i)}\right)^2.$$

I dalje je srednja kvadratna greška logičan izbor, ali je problem činjenica da bi njeno korišćenje rezultovalo u nekonveksnoj optimizacionoj funkciji, kada je u pitanju logistička regresija. Ovo bi dalje vodilo ka nemogućnosti elegantne primjene metoda gradijentnog spusta uz garanciju konvergencije za pravilan odabir koraka učenja. Podsjetimo se, za razliku od nekonveksnih, konveksne funkcije nemaju lokalnih minimum, već samo jedan globalni minimum.



Slika 8 Ilustracija optimizacione funkcije logističke regresije, kada bi se koristila srednja kvadratana greška za optimizaciju parametara.[8]

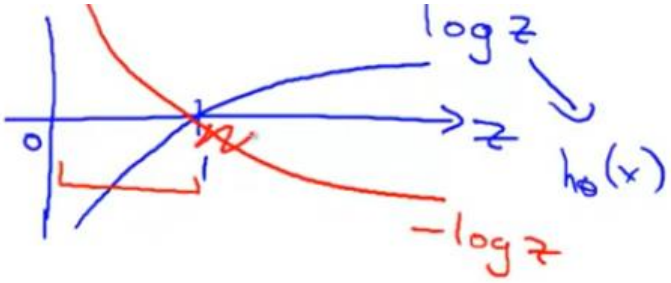
Pokazuje se da će se dobiti konveksna ukupna funkcija cijene, ako se za svaki uzorak umjesto kvadratne greške uzme cijena definisana kao:

$$\text{Cost}\left(h_\theta(x), y\right) = \begin{cases} -\log(h_\theta(x)) & \text{ako je } y = 1 \\ -\log(1 - h_\theta(x)) & \text{ako je } y = 0 \end{cases}.$$

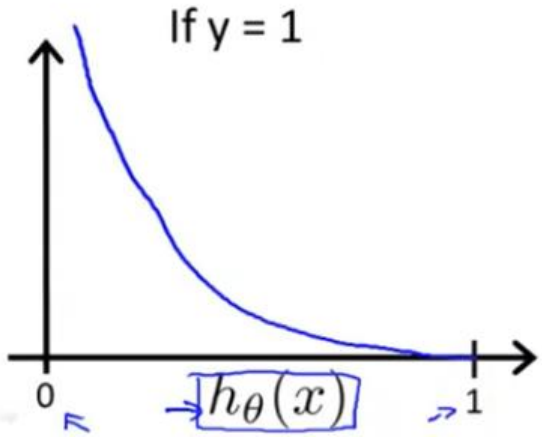
Slika 9 i Slika 10 ulustruju funkcije i $-\log(x)$ i $-\log(h_\theta(x))$. Heuristička funkcija je u oba slučaja prikazana na x-osi. Može uzeti vrijednosti između 0 i 1, pa je samo taj dio negativne logaritamske funkcije od interesa. Na slici 10, koja odgovara cijeni uyorka poyitivne klase, vrijednosti heurističke funkcije blizu 1 se odnose na estimiranje date

[8] Slika preuzeta sa https://www.coursera.org/learn/machine-learning

klase kao pozitivne i ovdje se dobija cijena bliska 0, jer je mala greška u estimaciji. Sa druge strane, ukoliko se pozitivna klasa estimira kao negativna i dobije heuristička funkcija koja ima vrijednost blisku nuli, funkcija cijene uzima veoma velike vrijednosti.

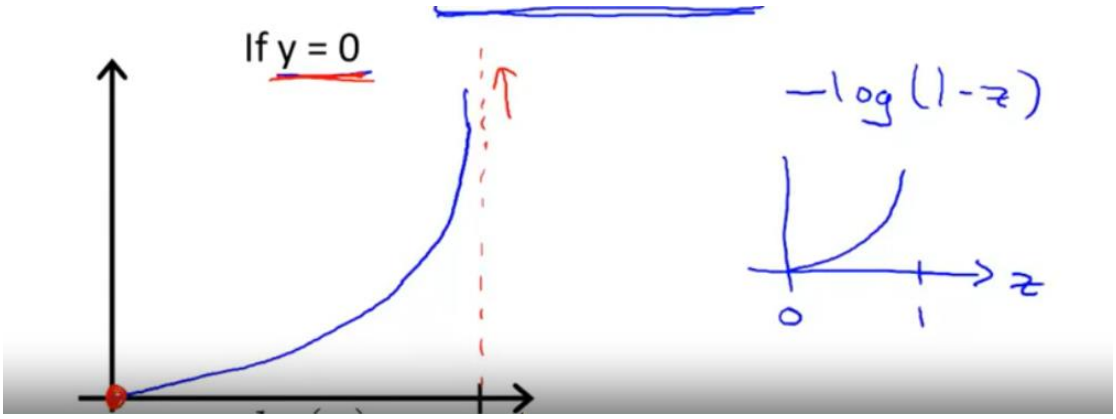


Slika 9 log(x) i log(x)[9]



Slika 10 Oblik funkcije cijene za uzorak koji odgovara pozitivnoj klasi[10]

Ista logika važi i za negativnu klasu, Slika 11. Ukoliko se uzorak koji pripada negativnoj klasi estimira ispravno dobija se heuristička funkcija bliska nuli i njoj odgovara i mala cijena. Nasuprot tome, ukoliko se odradi netačna estimacija i dobije se heuristička funkcija bliska jedinici, plaća se velika cijena.



Slika 11 Funkcija cijene za uzorak koji pripada negativnoj klasi[11]

[9] Slika preuzeta sa https://www.coursera.org/learn/machine-learning
[10] Slika preuzeta sa https://www.coursera.org/learn/machine-learning
[11] Slika preuzeta sa https://www.coursera.org/learn/machine-learning

Imajući na umu da je u pitanju binarna klasifikacija i da *y* može imati vrijednost 0 ili 1, može se pisati funkcija cijene za binarnu logističku regresiju kao:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})$$
$$= \frac{1}{m}\sum_{i=1}^{m}\left[-y^{(i)}\log h_\theta(x^{(i)}) - (1-y^{(i)})\log(1-h_\theta(x^{(i)}))\right]$$

za pozitivnu klasu i y=1 uzima se prvi član, dok je za negativnu klasu y=0 i u sumu ulazi drugi član, dok je prvi jednak nuli.

Optimizacion funkcija za logističku regresiju se može statistički izvesti koristeći Maximum likelihood estimation princip. Smatra se da vodi ka statistički efikasnoj procjeni parametara. Detalji će se izučavati iz *Teorije slučajnih procesa*.

Parametre ranije definisane heurističke funkcije logističke regresije određujemo slijedeći ranije uvedeni metod gradijentnog spusta. Nakon što odredimo parametre, za novi uzorak se isti koriste i računa se vrijednost hipoteze, koja se tumači kao vjerovatnoća da je uzorak pripadnik pozitivne klase y = 1, za date parametre i karakteristike.

Metod gradijentog spusta ima za zadatak da odredi parametre θ na osnovu karakteristika dostupnih odbiraka i njima pridruženih oznaka klasa, tako što će optimizovati funkciju cijene definisanu kao:

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})$$
$$= \frac{1}{m}\sum_{i=1}^{m}\left[-y^{(i)}\log h_\theta(x^{(i)}) - (1-y^{(i)})\log(1-h_\theta(x^{(i)}))\right]$$

I to tako što će na početku svim parametrima dodijeliti vrijednosti nula, ili slučajno odabrane vrijednosti i do konvergencije ponavljati ažuriranje podataka po sljedećoj proceduri:

$$\theta_j := \theta_j - \alpha\frac{\partial}{\partial\theta_j}J(\theta), \quad \text{za } j = 0,1,2,\ldots n$$

Razlika u odnosu na linearnu regresiju je u obliku hipoteze. Sada se koristi sigmoid funkcija, čiji je gradijent:

$$\frac{\partial}{\partial\theta}h_\theta(x) = \frac{\partial}{\partial\theta}\frac{1}{1+e^{-\theta^T x}} = -\frac{1}{\left(1+e^{-\theta^T x}\right)^2}e^{-\theta^T x}(-x)$$
$$= \frac{1}{1+e^{-\theta^T x}}\left(1-\frac{1}{1+e^{-\theta^T x}}\right)x = h_\theta(x)(1-h_\theta(x))x$$

Sada se može odrediti gradijent optimizacione funkcije kao:

$$\frac{\partial J(\theta)}{\partial \theta_j} = \sum_{i=1}^{m}\left[\frac{-y^{(i)}}{h_\theta(x^{(i)})} + \frac{(1-y^{(i)})}{(1-h_\theta(x^{(i)})}\right]\frac{\partial h_\theta(x^{(i)})}{\partial \theta_i} =$$

$$= \sum_{i=1}^{m}\left[\frac{-y^{(i)}}{h_\theta(x^{(i)})} + \frac{(1-y^{(i)})}{(1-h_\theta(x^{(i)})}\right]h_\theta(x^{(i)})(1-h_\theta(x^{(i)})x_j^{(i)}$$

$$= \sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)x_j^{(i)}$$

Metod grupnog gradijentnog spusta se sada svodi na istovremeno ažuriranje svih parametara po sljedećoj formuli:

$$\theta_j := \theta_j - \alpha\sum_{i=1}^{m}\left(h_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) - y^{(i)}\right)x_j^{(i)} \quad \text{za} \quad j = 0,1,2,\ldots n.$$

Iako gore navedene formule izgledaju identično kao za linearnu regresiju, sada je drugačiji oblik hipoteze:

$$h_\theta(x) = \frac{1}{1+e^{-\boldsymbol{\theta}^T\mathbf{x}}}.$$

Prethodni algoritam se treba realizovati tako što se ažuriranje vrši koristeći formule u vektorskom obliku, pa će biti:

$$\boldsymbol{\theta} := \boldsymbol{\theta} - \alpha\sum_{i=1}^{m}\left(h_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) - y^{(i)}\right)\mathbf{x}^{(i)}$$

gdje je:

$$\mathbf{x}^{(i)} = \begin{bmatrix} x_0^{(i)} \\ x_1^{(i)} \\ \vdots \\ x_n^{(i)} \end{bmatrix} \qquad \boldsymbol{\theta} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{bmatrix}.$$

Može se i suma po *m* eliminisati koristeći matrični zapis kao:

$$\boldsymbol{\theta} := \boldsymbol{\theta} - \alpha\mathbf{X}^T(h_{\boldsymbol{\theta}}(\mathbf{X}\boldsymbol{\theta}) - \mathbf{y})$$

gdje je:

$$\mathbf{X} = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & \cdots & x_n^{(1)} \\ x_1^{(2)} & x_2^{(2)} & \cdots & x_n^{(2)} \\ \vdots & \vdots & \vdots & \vdots \\ x_1^{(m)} & x_2^{(m)} & \cdots & x_n^{(m)} \end{bmatrix}_{mxn} = \begin{bmatrix} -\left(\mathbf{x}^{(1)}\right)^T - \\ -\left(\mathbf{x}^{(2)}\right)^T - \\ -\vdots- \\ -\left(\mathbf{x}^{(m)}\right)^T - \end{bmatrix}_{mxn}$$

i

$$\mathbf{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix}_{mx1} .$$

Što se tiče same realizacije algoritma, konvergencija se prati na isti način kao kod linearne regresije. Posmatra se ponašanje funkcije cijene u zavisnosti od broja iteracija. Ukoliko sa povećavanjem iteracija dobijamo da je funkcija cijene opadajuća funkcija dobro smo odabrali korak učenja. Ukoliko dobijemo rastuću funkciju cijene u zavisnosti od iteracija, loše smo odabrali korak učenja.

Ukoliko imamo više karakteristika koje uzimaju vrijednosti iz međusobno neproporcionalnih opsega, i ovdje će skaliranje ubrzati konvergenciju. Skaliranje se vrši na isti način kao kod linearne regresije.

Kada je u pitanju računska složenost algoritma, najzahtjevije je što je u svakoj iteraciji potrebno izračunati funkciju cijene i njen gradijent – parcijalne izvode za $j = 0, 1, 2, \ldots N$

Gradijentni spust se sa nivoa implementacije može posmatrati kao funkcija koja ima ulazni argument gradijent i na osnovu njega vrši ažuriranje parametara. Doduše, za gradijentni spust je dovoljan gradijent funkcije cijene, ali se i funkcija cijene najčešće računa zbog praćenja konvergencije, odabira koraka učenja i slično.

Postoji veliki broj sofisticiranijih algoritama za realizaciju logističke regresije i optimizaciju parametara. Neki od njih su:

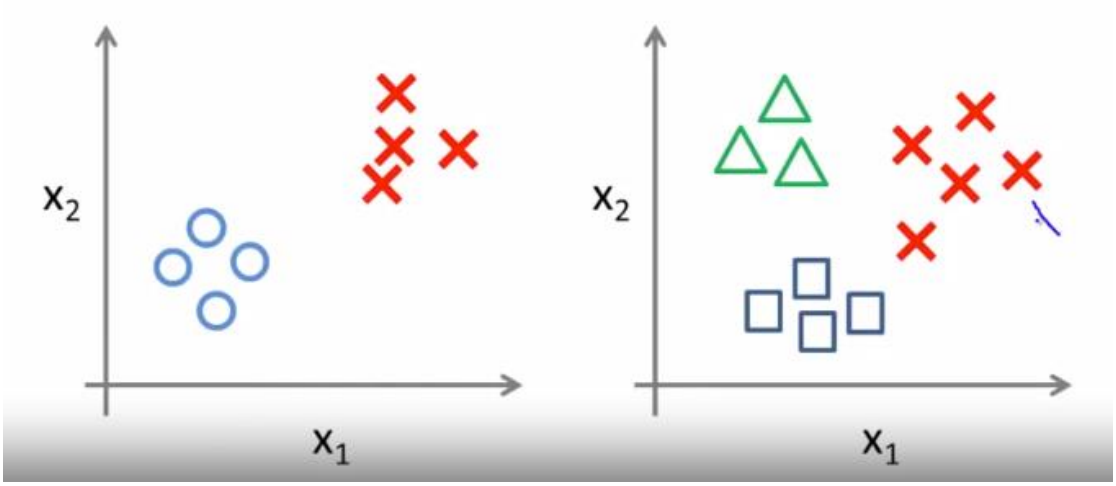
- *Conjugate gradient*
- *BFGS*
- *L - BFGS*

Prednosti navedenih algoritama su činjenice da nema potrebe za ručnim određivanjem koraka učenja α i najčešće su brži od metoda gradijentnog spusta. Sa druge strane ovi algortimi su kompleksni su za implementaciju, ali postoji veliki broj biblioteka koje nude implementiranu verziju ovih metoda. Treba provjeriti koja biblioteka nudi dobru realizaciju.

Višeklasna klasifikacija

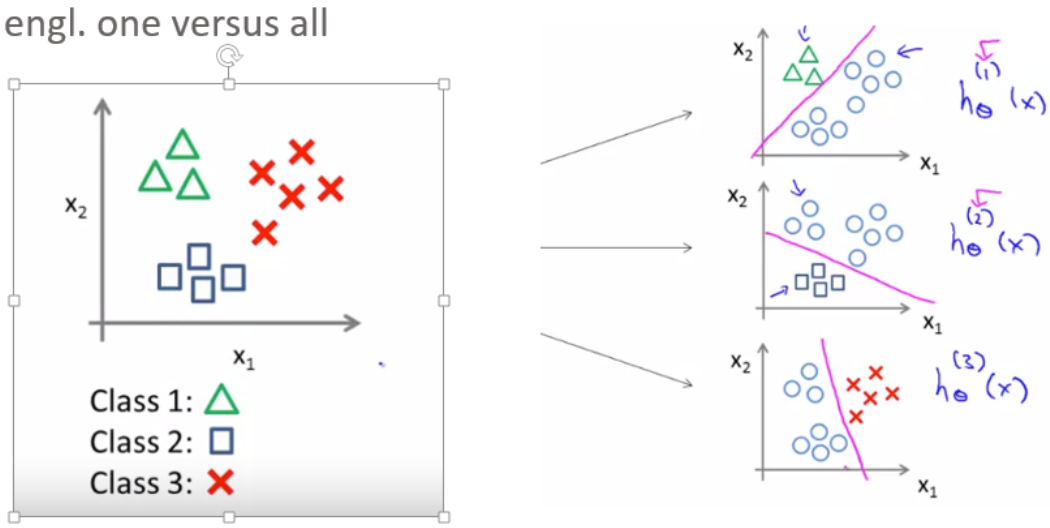
Do sada je posmatrana binarna klasifikacija, međutim, često imamo potrebu izvršiti klasifikaciju u veći broja klasa, tzv. **višeklasnu ili multiklasnu klasifikaciju**. Veliki je broj primjena ovakve klasifikacije, a može se i posmatrani primjer klasifikacije pošte proširiti tako da želimo dolaznu poštu klasifikovati u: poslovnu, privatnu-prijatelji, privatnu – porodica, hobi,....

Kada imamo veći broj klasa imamo i više vrijednosti za oznake uzoraka, pri čemu svaka označava pripadnost jednoj klasi. Na primjer, za četiri klase oznake bi bile: y = 1, y =2, y = 3, y = 4.



Slika 12 Ilustracija klasifikaciji u: dvije klase (lijevo), tri klase (desno)[12]

Za višeklasnu klasifikaciju koristi **pristup jedan naspram svih**, engl. **one versus all**. Ovaj problem posmatramo kao tri odvojena problema binarne klasifikacije tako što kreiramo tri nova trening seta – za svaku klasu po jedan. Trening setovi se kreiraju tako što klasa za koju se vrši klasifikacija predstavlja pozitivne primjere y = 1, a ostale klase negativne y = 0, Slika 13.



Slika 13 Ilustracija principa jedan naspram svih[13]

Nakon formiranja željenog broja setova za treniranje za svaki od njih se pokrene binarna klasifikacija i optimizacija funkcije cijene. Za novi uzorak se računaju sve naučene heurističke funkcije i uzorak se pridružuje onoj klasi za koju se dobije najveća vrijednost heurističke funkcije – vjerovatnoće da uzorak sa datim karakteristikama pripada klasi.

Dakle, ukoliko imamo *K* klasa. Treniranje vršimo tako da dobijemo *K* heurstičkih funkcija:

[12] Slika preuzeta sa https://www.coursera.org/learn/machine-learning
[13] Slika preuzeta sa https://www.coursera.org/learn/machine-learning

$$h_\theta^{(i)}(x) = p(y = i \mid x, \theta) \qquad i = 1, 2, 3, \ldots, K$$

Nakon treniranja, za novi ulazni podatak ***x*** kao predikcija se uzima klasa koja ima maksimalnu vrijednost heurističke funkcije:

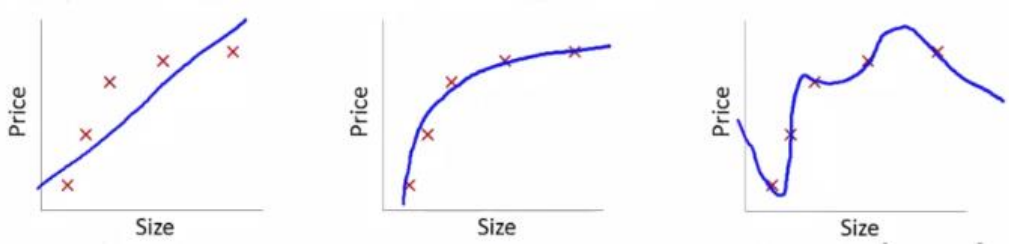
$$\max_i h_\theta^{(i)}(x).$$

Dakle, bira se klasa kojoj novi podatak pripada sa najvećom vjerovatnoćom.

Pretjerana prilagođenost modela podacima

Pokazali smo da se i kod linearne i kod logističke regresije mogu vršiti složenije procjene tako što će se uzeti u obzir veći stepen polinoma prilikom definisanja heurističke funkcije. Takođe, posmatrali smoi proizvoljan broj karakteristika kojima se opisuje posmatrani problem. Može djelovati da se sa povećanjem stepena polinoma kojim se modelira neka funkcija, granica odlučivanja, odnosno sa povećanjem karakteristika dobija bolja klasifikacija i da važi pravilo što više, to bolje. Međutim, nije uvijek tako.

Na slici 14 je data ilustracija slučaja kada je kod linearne regresije uzet linearni model zavisnosti, koji je nedovoljan i dovodi do premale (engl. underfit) prilagođenosti modela podacima i velikog bijasa (lijevo), optimalne prilagođenosti (sredina) i prevelike prilagođenosti (engl. overfit) podacima, koja dovodi do velike varijanse (desno). Naime, iako bi se za red polinoma korišćen za dobijanje desne slike dobila najmanja funkcija cijene, računata kao srednja kvadratna greška, prilagođenost modela podacima na kojima je treniran bi bila prevelika i rezultovala bi u velikoj grešci prilikom primjene na nove podatke kojih nije bilo u trening setu.



Slika 14 Ilustracija uticaja prilagođenosti modela podacima[14]

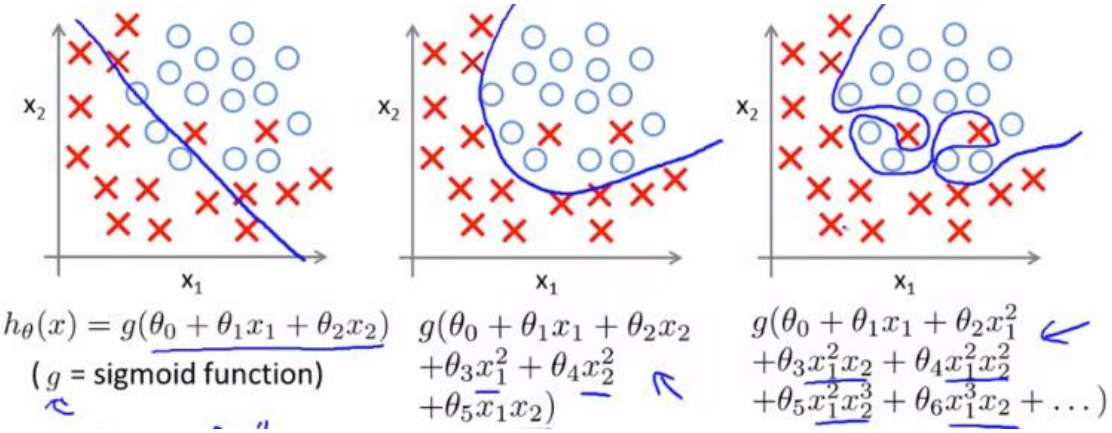
Pretjerana prilagođenost podacima – overfitting se javlja kada koristimo veliki broj karakteristika (ili veći stepen polinoma), tako da se hipoteza u toku učenja jako dobro prilagodi podacima iz trening seta i da malu funkciju cijene:

$$\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \min_{\boldsymbol{\theta}} \frac{1}{2m} \sum_{i=1}^{m} \left(h_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) - y^{(i)} \right)^2 \approx 0,$$

ali slabo vrši generalizaciju i daje veliku grešku kada se primijeni nad novim podacima.

Do pretjerane prilagođenosti podacima može doći i kod klasifikacije,

[14] Slika preuzeta sa https://www.coursera.org/learn/machine-learning



Slika 15 Ilustracija uticaja prilagođenosti podacima kod klasifikacije: premala (lijevo), optimalna (sredina), prevelika (desno).[15]

Pretjerana prilagođenost podacima se može spriječiti na više načina. Prvi koji se nameće jeste da se smanjiti broj karakteristika, odnosno red polinoma, kada se utrdi da je došlo do pretjerane prilagođenosti. Ovo se može učiniti na dva načina, i to ručnim uklanjanjem karakteristike za koje mislimo da dovode do pretjerane prilagođenosti (ali odbacujemo dio informacija) i korišćenjem algoritama za automatsku selekciju modela – učićete naredne godine. Princip koji se često koristi u mašinskom učenju je automatizovan i naziva se **regularizacija**. Ideja regularizacije je da se zadrže sve karakteristike, ali smanji vrijednosti parametara koji određuju koliko koja karakteristika doprinosi predikciji. Dobro funkcioniše kada imamo veliki broj karakteristika i svaka pomalo doprinosi predikciji izlazne vrijednosti y.

Posmatrajmo jedan primjer u kojem smo zaključili da smo previše prilagodili hipotezu podacima i željeli bismo da smanjimo uticaj članova trećeg i četvrtog reda na nju:

$$h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3$$

Ali ne želimo da u potpunosti odbacimo ove članove, možemo modifikovati funkcije cijene na sljedeći način:

$$\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \min_{\boldsymbol{\theta}} \left[\frac{1}{2} \sum_{i=1}^{m} \left(h_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) - y^{(i)} \right)^2 + 1000 \cdot \theta_3 + 1000 \cdot \theta_4 \right]$$

Kako bi modifikovana funkcija cijene postala bliska nuli, moramo smanjiti vrijednost dodatih članova, a to možemo učiniti tako što ćemo odabrati jako male vrijednosti za θ_3 i θ_4, tako da se dobije funkcija koja je bliska kvadratnoj, ali i dalje na njen oblik utiču članovi trećeg i četvrtog reda.

Ovaj se pristup može generalizovati. Polazi od ideje da se dobiju manje vrijednosti za parametre $\theta_i \, i = 1,2,\dots,n$, jer se tako dobijaju jednostavnije hipoteze i manja je mogućnost da se dobije pretjerano prilagođavanje podacima – overfitting. Manje vrijednosti parametara se dobijaju modifikacijom funkcije cijene, tako da se svaki parametar smanji, jer ne znamo koja je karakteristika uzrokovala pretjeranu prilagođenost. θ_0 **se po konvenciji ne regularišе.**

[15] Slika preuzeta sa https://www.coursera.org/learn/machine-learning

Sada se optimizacija parametara za linearnu regresiju, uz regularizaciju vrši minimizacijom modifikovane optimizacione funkcije:

$$J(\boldsymbol{\theta}) = \frac{1}{2}\left[\sum_{i=1}^{m}\left(h_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}) - y^{(i)}\right)^2 + \lambda\sum_{j=1}^{n}\theta_j^2\right],$$

gdje je λ - **faktor regularizacije**.

Ovako definisanom optimizacionom funkcijom se postiže regulisanje uticaja dva člana na funkciju cijene čime se postižu dva cilja. Minimizacija prvog člana obezbjeđuje malu grešku i dobro prilagođavanje podacima iz trening seta, a drugog zadržavanje malih vrijednosti za parametre i dobru generalizaciju – izbjegavanje pretjerane prilagođenosti podacima iz trening seta.

Faktor realizacije igra ključnu ulogu u procesu regularizacije i prilikom njegovog odabira mora se voditi računa o tome da se za velike vrijednosti faktora regularizacije mora odabrati jako mala vrijednost za svako θ, kako bi se funkcija cijene održala malom. To vodi do zaključka da θ ne smije biti previše malo, jer će se dobiti slaba prilagođenost podacima – underfitting. Svi će parametri će biti bliski nuli. S druge strane premala vrijednost faktora regularizacije će voditi do slabe regularizacije i velikog uticaja karakteristika/polinoma, što može voditi ka prevelikoj prilagođenosti modela podacima. Odabir faktora regularizacije je kompromis.

Kada se uzme u obzir regularizacija metod gradijentog spusta se mora modifikovati, tako da se posmatra novodefinisana funkcija cijene, pa će se do konvergencije ponavljati ažuriranje podataka:

$$\theta_0 := \theta_0 - \alpha\ \frac{1}{m}\ \sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_0^{(i)}$$

$$\theta_j := \theta_j - \alpha\left[\left(\frac{1}{m}\ \sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)}\right) + \frac{\lambda}{m}\,\theta_j\right] \qquad j \in \{1, 2...n\}$$

I prilikom rješavanja linearne regresije korišćenjem normalne jednačine mora se izvršiti regularizacija, pa su modifikovane formule:

$$\boldsymbol{\theta} = \left(\mathbf{X}^T\mathbf{X} + \lambda I\right)^{-1}\mathbf{X}^T\mathbf{y}$$

gdje je:

$$I = \begin{bmatrix} 0 & & & \\ & 1 & & \\ & & \ddots & \\ & & & 1 \end{bmatrix}_{n+1\times n+1}.$$

I kod logističke regresije će regularizacija promijeniti način ažuriranja podataka, pa se moifikuje model gradijentnog spusta tako da se odvoji ažuriranje θ_0 od ažuriranja ostalih parametara, jer se θ_0 ne ragularizuje. Sada se ponavlja do konvergencije ažuriranje podataka:

$$\theta_0 := \theta_0 - \alpha \sum_{i=1}^{m} \left(h_{\mathbf{\theta}}(\mathbf{x}^{(i)}) - y^{(i)} \right)$$

$$\theta_j := \theta_j - \alpha \left[\sum_{i=1}^{m} \left(h_{\mathbf{\theta}}(\mathbf{x}^{(i)}) - y^{(i)} \right) x_j^{(i)} + \lambda \theta^{(i)} \right] \quad j = 1, 2, \ldots, n$$

Dok je funkcija cijene:

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} \left[y^{(i)} \log\left(h_\theta\left(x^{(i)} \right) \right) + \left(1 - y^{(i)} \right) \log\left(1 - h_\theta\left(x^{(i)} \right) \right) \right] + \frac{\lambda}{2} \sum_{j=1}^{n} \theta_j^{(2)}$$

LITERATURA

[1] https://see.stanford.edu/Course/CS229, posljednji put pristupano, 20. 10. 2021. godine.
[2] *https://www.coursera.org/learn/machine-learning*, posljednji put pristupano, 20. 10. 2021. godine.