PRIMENJENI ALGORITMI

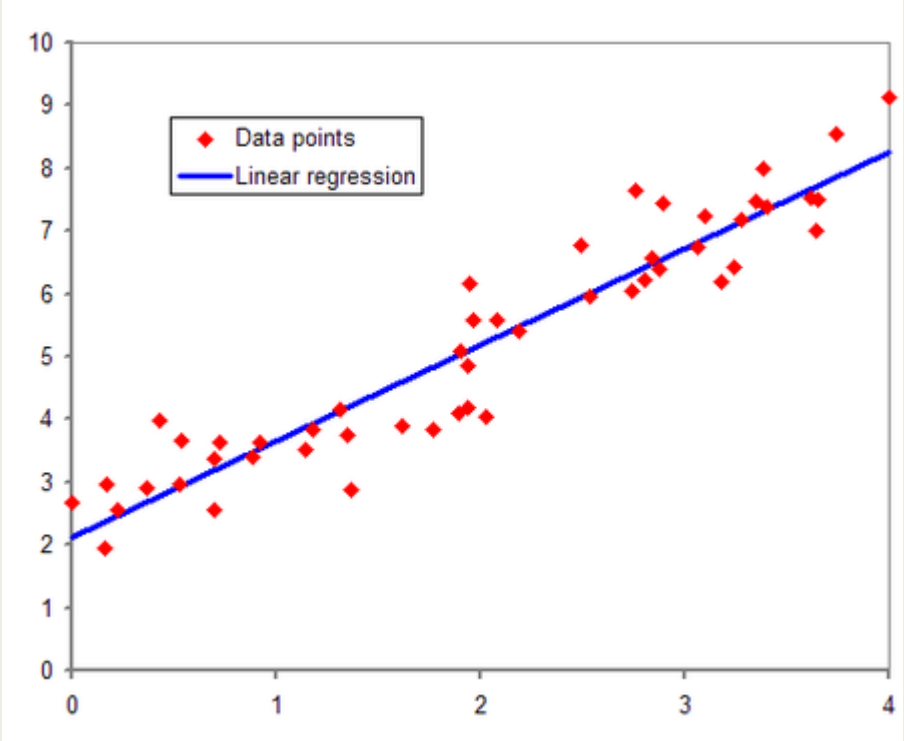
Linearna regresija

Linearna regresija

- Predstavlja jednu od osnovnih tehnika mašinskog učenja (ML, Machine learning)
- Koristi se da se pronađe linearna zavisnost između nezavisne promenljive X^n (X^n predstavlja vektor sa n vrednosti) i zavisne promenljive Y, koja zavisi od X^n
- Koristi se za **predviđanje** vrednosti Y, ukoliko je poznata vrednost X^n
- Vrednost Y najčešće predstavlja jednu vrednost iz skupa realnih brojeva
- Linearna zavisnost se izražava formulom $Y = B*X^n+E$, odnosno $y_i = b_1*x_{i1}+...+ b_n*x_{in}+e_i$, za svaku koordinatu y_i u zavisnoj promenljivoj Y
- Ako postoji samo jedna zavisna promenljiva dobija se: $\boldsymbol{y = b_1*x_1+...+ b_n*x_n+e}$
- Ako postoji samo po jedna zavisna i nezavisna promenljiva dobija se: $\boldsymbol{y = b*x+e}$

Linearna regresija

- Primer: Skup tačaka u ravni
  - *Posmatramo skup tačaka u ravni*
  - *Pokušavamo da pronađemo zavisnost koordinate y od koordinate x*
  - *Pronađena zavisnost je prikazana plavom linijom*
  - *Nagib linije predstavlja koeficijent **b** iz formule **y = b*x+e***
  - Prvo je potrebno odrediti **b** i **e** na osnovu datih tačaka
  - U predviđanju za datu vrednost **x** se uzima vrednost sa **y** prave



Linearna regresije – učitavanje podataka

- Prilikom obrade podataka, podaci se najčešće čuvaju u obliku DataFrame-a, iz paketa dataFrames
- DataFrame u sustini predstavlja slog, u kojem se nalaze i nezavisne i zavisne promenljive. Ove promenljive su date u obliku nizova
- Primer kreiranja DataFrame-a za 3 tačke: ((2, 4), (1, 6) i (3, 5))
  - *data = DataFrame(x = [2, 1, 3], y = [4, 6, 5])*
- *.csv fajlovi (comma separated values) predstavljaju tekstualne fajlove u kojima se podaci drže razdvojeni zarezom zarezom
- U Juliji, mogu da se učitaju naredbom CSV.read() ili naredbom CSV.File() iz paketa CSV i da se potom konvertuju u DataFrame nas sledeci način:
  - *data = CSV.read("file.csv", DataFrame; kwargs)*
  - *data = DataFrame(CSV.File("file.csv"))*

Linearna regresije – priprema podataka

- Da bi se mogla koristiti linearna regresija, potrebno je ulazne podatke podeliti u skup za obuku (training set) i skup za testiranje (test set)
- Najčešće skup za obuku ima između 75-90% ulaznih podataka, dok se preostalih 10-25% stavlja u skup za testiranje
- U Juliji, u paketu Lathe postoji funkcija TrainTestSplit(df, precentage) koja se može koristiti za podelu podataka na skup za obuku i testiranje.
- Primer podele u odnosu 75:25%:
  - *TrainSet, TestSet =TrainTestSplit(data, .75)*
- Kako se ova podela vrši na osnovu slučajnog ulaza, postoji mogućnost da podaci u skupu za testiranje budu više u korelaciji u odnosu na podatke u skupu za obuku. U tom slučaju kažemo da je sistem loše istreniran (overfitted)
- Ovo može biti i posledica premalog broja ulaznih podataka
- Da bi se ovo prevazišlo, često se koristi K-Fold kros validacija

Prikaz učitanih podataka

- Julija sadrži paket Plots koji omogućava kreiranje različitih grafikona. Funkcija Plot(...) može da kreira grafikon, dok funkcija Plot!(grafikon,...) može da dodaje na postojeći grafikon
- Funkcije scatter() i scatter!() mogu da kreiraju grafikone sa tačkama u ravni pogodne za linearnu regresiju sa jednom nezavisnom promenljivom
- Funkcije ima veći broj mogućih parametara koji mogu da predstavljaju ulazne podatke ili osobine grafikona, a funkcije Plot!() i scatter() kao prvi parametar mogu da imaju postojići grafikon koji mogu da menjaju. Neki parametri su:
  - *x – nezavisna promenljiva*
  - *y – zavisna promenljiva*
  - *title, xlabel, ylabel – nazivi grafikona i osa*
  - *legend – pozicija legende na grafikonu*
  - *df –postojeći data frame*
  - *...*

Korelacija

- Korelacija predstavlja zavisnost između dve promenljive. U slučaju linearne regresije, to su promenljive x i y
- Perasonov koeficjent korelacije predstavlja vrednost iz intervala [-1, 1] kojim se izražava zavisnost između promenljivih.
  - *Vrednost 1 govori nam da su vrednosti u savršenoj korelaciji, 0 da nisu u korelaciji, a –1 da su u savršenoj negativnoj korelaciji*
- Iako ne postoji precizno tumačenje vrednoszti ovog koeficijenta, može se uzeti:
  - *Za vrednosti veće od 0.9, postoji veoma jaka korelacija*
  - *Za vrednosti između 0.7 i 0.9, postoji jaka korelacija*
  - *Za vrednosti između 0.5 i 0.7, postoji umerena korelacija*
  - *Za vrednosti manje od 0.5, postoji slaba korelacija*
- U Juliji, u paketu Statistics postoji funkcija **cor()** za računanje koeficijenta korelacije

Linearna regresija u Juliji

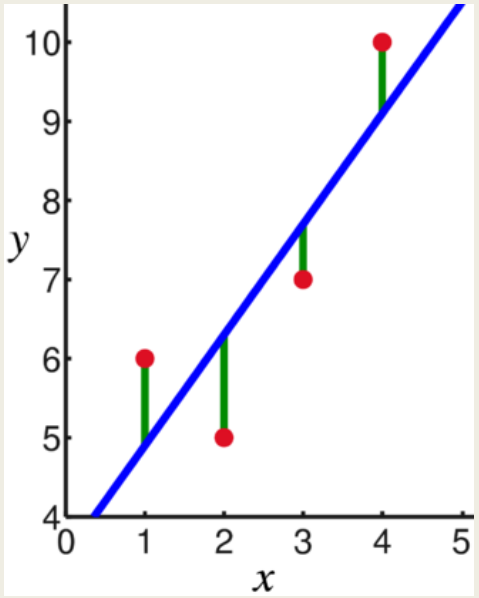
- Za kreiranju i obuku regresora, koristi se funkcija lm() iz paketa GLM. Funkciji se kao parametar prosleđuju ulazi podaci iz seta za obuku, a kao rezultat dobija se regresor. Prilikom obuke, vrši se određivanje parametara B i E iz formule Y=B*X+E
- Regresor predstavlja objekat pomoću kojeg se vrši linearna regresija i koji u sebi sadrži parametre B i E
- Za predvidjanje vrednosti, koristi se finkcija predict() koja kao parametre prima regresor i vrednosti za predvidjanje (npr. X vrednosti iz skupa za testiranje). Kao izlaz funkcija vraća predviđene vrednosti za ulazne podatke, tj. predviđene Y vrednosti

Ocena kvaliteta linerane regresije

- Koeficijent determinacije, r^2, je deskriptivna mera jačine regresione veze, koja meri koliko se dobro regresiona linija prilagođava podacima, tj. koliko ona odstupa od podataka
- r^2 se uvek kreće u intervalu [0,1], pri čemu veće vrednosti pokazuju bolji stepen veze između podataka
- Iako ne postoji precizno tumačenje vrednoszti ovog koeficijenta, može se uzeti:
  - *Za vrednosti veće od 0.9, model je jako dobar za predvidjanje*
  - *Za vrednosti između 0.7 i 0.9, model je veoma dobar za predvidjanje*
  - *Za vrednosti između 0.5 i 0.7, model je dobar za predviđanje*
  - *Za vrednosti manje od 0.5, model nije dobar za predvidjanje*
- U linearnoj regresiji, ova vrednost se koristi slično koeficijentiu korelacije

Ocena kvaliteta linerane regresije

- Standardna greška u predviđanju računa se kao $e_i=y_i-yp_i$, pri čemu je y_i izmerena vrednost, a yp_i vrednost dobijena regresijom
- Koristeći ovu vrednost, često se koriste sledeće mere za grešku:
  - *Prosek absolutnih vrednosti greske: mean(abs(e_i))*
    - Ne obraća pažnju na to da li greške jako variraju
  - *Prosečna relativna greška: mean(abs(e_i/y_i))*
    - Ne obraća pažnju na to da li greške jako variraju
  - *Prosek kvadrata greške (MSE): mean(e_i^2)*
    - Osetljiva ako postoji neka velika greška
  - *Koren proseka kvadrata greške*
    - Osetljiva ako postoji neka velika greška
- Za sve ove vrednosti često ne postoje bitne granice, već samo treba da budu što manje



Ocena kvaliteta linerane regresije

- Primer: neka su vrednosti 10, 12 i 14 predvidjene linearnom regresijom kao vrednosti 9, 13 i 18. Tada imamo da su greske redom 1, -1 i -4. Dobijamo:
  - *Prosek absolutnih vrednosti greske: mean(abs(e_i))*
    - (|1|+|-1|+|-4|)/3 = 2
  - *Prosečna relativna greška: mean(abs(e_i/y_i))*
    - (1/10+1/13+4/14) = 0,248352
  - *Prosek kvadrata greške (MSE): mean(e_i^2)*
    - $1^2+(-1)^2+(-4)^2=1+1+16 = 18$
  - *Koren proseka kvadrata greške (RMSE)*
    - $\text{sqrt}(1^2+(-1)^2+(-4)^2) = \text{sqrt}(1+1+16) = \text{sqrt}(18) = 4{,}242641$
- U slučaju kada se greške ujednačene prva i poslednja analiza daju slične rezultate, dok u situaciji kada greške nisu ujednačene, RMSE daje mnogo veće rezultate i zato se smatra boljom (informativnijom) analizom
- Samo druga analiza zavisi od vrednosti y i predstavlja relativno objektivnu meru

Ocena kvaliteta linerane regresije

- Za linearnu regresiju kažemo da previše dobro predviđa (overfitt) ukoliko su predviđanja sa njom mnogo bolja od očekivanog
- Pod očekivanim predviđanjima podrazumeva se često predviđanje nad skupom za obuku. Ako je npr. RMSE skupa za obuku veća od RMSE skupa za testiranje, tada kažemo da je model overfitted i da nije pogodan za korišćenje
- Ovo može biti posledica nekoliko uzroka:
  - *Uzorak je premali, pa model ne može adekvatno da se podesi.*
    - Primer: ako imamo 10 vrednosti, sa 2 velike greške, tada se greške ne mogu ravnomerno raspodeliti u skup za obuku i testiranje
  - *Nije adekvatna podela, skup za test je previše mali*
    - Primer: ako imamo 100 vrednosti, 95 u skupu za obuku i 5 u skupu za testiranje, mala je verovatnoća da je skup za testiranje adekvatno izabran
  - *Nije adekvatna podela, skup za test je pogrešno izabran*
    - Primer: ako imamo 100 vrednosti i 20 vrednosti u skupu za testiranje, a nijedna od 10 velikih grešaka se ne nalazi u njemu

Primer 1: LR sa jednom nezavisnom promenljivom

- Primer 1: Neka je dat skup tačaka u ravni. Napisati skript kojim se određuje zavisnost vrednosti y koordinate u zavisnosti od x koordinate tačke. Pri tome:
  - *Odrediti korelaciju između x i y koordinata*
  - *Predvideti vrednosti y koordinata za test skup. Skupove podeliti u odnosu 80:20*
  - *Proceniti grešku za skup za obuku i test skup.*
  - *Proveriti da li je model dobar za predvidjanja ovog tipa*
  - *Proveriti da li je model overfitted*
  - *Nacrtati ove skupove na grafikonu*

One hot encoding

- One hod encoding predstavlja mogućnost u preprocesiranju da se jedna kolona koja ima N razlicitih vrednosti zameni sa N kolona, u kojima će biti vrednosti Tačno i Netačno, u zavisnosti od vrednosti početne kolone
- Primer: ako imamo crvene i crne tačke u ravni, tada DataFrame ima 3 kolone: x, y i boja. Kada primenimo ovo enkodiranje, mozemo dobiti DataFrame sa 4 kolone: x, y, crvena i crna, pri čemu npr. crvene tačke imaju vrednost Tačno u koloni Crvena i vrednost netačno u koloni crna
- Ovo može da poboljša model za predviđanja
- U Juliji, postoji funkcija Lathe.preprocess.OneShotEncoder() koja vraća enkoder, a njegova funkcija predict(), može da razdvoji jednu kolonu na više kolona

Primer 2: LR sa tri nezavisne promenljive

- Primer 2: Neka je data baza automobila, pri čemu je za svaki automobil dat njegov proizvođač, model, zapremina motora (kubikaža) i godište proizvodnje.
  - *Napraviti model linearne regresije za ovaj problem, a predviđanja cene izvršiti na osnovu modela, kubikaže i godišta automobila*
  - *Polje model enkodirati u više kolona u zavisnosti od vrednosti u toj koloni*
  - *Izračunati greške predviđanja, kao i prosečnu absolutnu grešku i prosečnu relativnu grešku*

K-fold kros validacija

- Da bi se izbegao slučaj da nije dobra podela na skupove za obuku i testiranje, uvodi se K-fold kros validacija
- Ona podrazumeva podelu skupa podataka na K delova i testiranje svakog dela u odnosu na ostatak. Nakon toga može da se traži prosečna greška za K testiranja
- Na ovaj način dobija se pouzdanija mera o kvalitetu testiranja, tj. o njegovoj grešci
- Primer: Ako imamo skup A sa 100 podataka i K = 5, tada se skup A deli na 5 skupova A1, A2, A3, A4, A5 sa po 20 elemenata i vrši se 5 testiranja, pri čemu se jedan skup uzima za test skup, a unija ostala 4 za skup za obuku

Primer 3: K-fold kros validacija

- Primer 1: Neka je dat skup tačaka u ravni. Napisati skript kojim se određuje zavisnost vrednosti y koordinate u zavisnosti od x koordinate tačke. Pri tome:
  - *Testiranja izvršiti K-fold kros validacijom za k = 5*
  - *Pronaći prosečnu absolutnu grešku za svako od 5 testiranja*
  - *Pronaći prosečnu absolutnu grešku za tih 5 testiranja*