Metoda potpornih vektora s primjenama u bankarstvu

Hajdina, Marta

SVEUČILIŠTE U ZAGREBU

PRIRODOSLOVNO–MATEMATIČKI FAKULTET

MATEMATIČKI ODSJEK

Marta Hajdina

METODA POTPORNIH VEKTORA S PRIMJENAMA U BANKARSTVU

Diplomski rad

Voditelj rada:
Zlatko Drmač

Zagreb, rujan, 2020.

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

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

Povjerenstvo je rad ocijenilo ocjenom _____.

Potpisi članova povjerenstva:

1. _____
2. _____
3. _____

Sadržaj

Uvod

Jedan od glavnih napredaka tehnologije u današnjem svijetu je predviđanje i zaključivanje na temelju prijašnjeg iskustva ili zapažanjem iz okoline. Strojno učenje je područje koje se bavi metodama za rješavanje tih problema. Svaka od metoda strojnog učenja zaključuje na temelju svoje baze znanja (podataka). Ukoliko se baza znanja sastoji od podataka za koje su dostupni zaključci, tada se radi o nadziranim metodama strojnog učenja, dok metode koje u svojoj bazi znanja ne posjeduju prijašnje zaključke nazivamo nenadziranim metodama strojnog učenja. Zbog svoje uspješnosti ove se metode primjenjuju u različitim područjima kako bi omogućile donošenje zaključaka koje čovjek sam nije sposoban donijeti u efektivnom vremenskom okviru.

Porastom svijesti o važnosti podataka napredovale su i metode za skladištenje podataka, omogućavajući lakše skladištenje sve većeg broja podataka. Također, mnoge su industrije prepoznale kako se podaci osim za svrhe izvršavanja operativnog posla mogu koristiti za unapređivanje poslovanja. Iz tog razloga sve češće pohranjuju sve veće količine podataka koji nisu nužni za operativne poslove, ali su vrlo korisni za zaključivanje, analize, predviđanja i slično.

Vodeći brigu o financijama svojih klijenata banke na raspolaganje dobivaju velike količine podataka o svojim klijentima. Osim za potrebe izvršavanje svojih primarnih funkcija, pomoću tih podataka pokušavaju unaprijediti način svojeg poslovanja. Jedna od čestih primjena je predviđanje rizika kod odobravanja kredita, određivanje klijenata koji su potencijalni kupci određenog proizvoda ili usluge, segmentacija klijenata, itd. Također jedan od češćih problema s kojima se banke susreću je zadržavanje postojećih klijenata, tj. identifikacija klijenata koji bi uskoro mogli napustiti banku, kako bi se na vrijeme moglo reagirati i spriječiti njihov odlazak.

Iako je u bankarskom svijetu praksa koristiti metode koje je moguće interpretirati i uz koje je kasnije moguće lako opisati poželjne karakteristike klijenata (npr. stabla odlučivanja), u ovom radu koristimo metodu potpornih vektora. Pomoću nje ćemo rješavati problem binarne klasifikacije, odnosno odrediti da li će klijent napustiti banku.

Najprije ćemo postepeno stvoriti intuiciju i definirati teoriju potrebnu za razumijevanje metode, zatim ćemo pokazati kako metoda potpornih vektora pripada problemima kvadratičnog programiranja. U kontekstu binarne klasifikacije prikazujemo *Sequential Minimal Optimization* algoritam [5] koji koristi činjenicu da Karush–Kuhn–Tucker uvjeti skupa za učenje nisu međusobno zavisni i time uvelike ubrzava metodu. Zatim ćemo istu metodu implementirati uz pomoć LIBSVM [3] biblioteke na stvarnom skupu podataka. Također, u svojoj dualnoj formulaciji metoda potpornih vektora omogućava primjenu jezgrenih funkcija koje su se pokazale vrlo korisnima za razdvajanje nelinearnih skupova, na što će se posebno obratiti pažnja u ovome radu.

Pri izboru mjere uspješnosti modela tijekom implementacije posebno pazimo na kontekst primjene problema (hoće li za banku biti lošije ako ne identificira klijenta koji odlazi iz banke ili pogrešno klasificira klijenta koji ne odlazi iz banke), te na slabiju balansiranost klasa (mali udio klase koju je potrebno identificirati). Na kraju korištenjem *grid search* algoritma optimiziramo hiperparametre metode i iznosimo ih u radu.

Poglavlje 1

Nadzirano strojno učenje

Metode nadziranog strojnog učenja koriste se za rješavanje različitih problema klasifikacije i regresije, a u ovom radu promatrati ćemo samo problem klasifikacije. Zadatak klasifikacijske metode je pridružiti ulaznim podacima njihove oznake, tj. klase.
U ovom poglavlju najprije objašnjavamo kako se svaka metoda nadziranog strojnog učenja sastoji od dva glavna koraka koji se odvijaju na međusobno disjunktnim skupovima podataka. Zatim, definiramo skup ulaznih i izlaznih podataka u kontekstu problema klasifikacije, te kako podijeliti skup podataka na skup za učenje i skup za testiranje. Na kraju prikazujemo glavne ideje treniranja metode, te uobičajene metrike za testiranje.

1.1 Glavni koraci za nadzirano strojno učenje

Metode koje se koriste u strojnom učenju možemo podijeliti u tri skupine: nadzirane, nenadzirane i polunadzirane metode. U ovom radu ćemo proučavati nadzirane metode, tj. metode koje se treniraju na skupu podataka za učenje koji je sastavljen od ulaznih i izlaznih podataka.

Učenje (treniranje) metode zapravo podrazumijeva izgradnju modela nad podacima za treniranje(učenje). Nakon što je model izgrađen (nakon što je metoda naučena) potrebno je testirati koliko dobro klasificira na testnom skupu podataka, tj. potrebno je usporediti rezultate metode (modela) dobivene za ulazne podatke testnog skupa s pravim izlaznim podacima testnog skupa. Naravno, poželjno je da je ta razlika što manja, te je zbog toga potrebno dobro izgraditi model i zatim optimizacijom pronaći odgovarajuće hiperparametre metode .

Primjer 1.1.1. *Za potrebe ovog primjera promatramo metodu koja pomoću polinoma p klasificira točke u ravnini s obzirom na to s koje se strane grafa polinoma nalaze, tj. ako je $p(x) \geq 0$ točka x pripada pozitivnoj klasi, a ako je $p(x) < 0$ pripada negativnoj klasi. Pretpostavimo da je metoda sposobna za zadani stupanj polinoma (hiperparametar) izgraditi jedinstveni model polinoma koji klasificira točke na skupu podataka za treniranje. U ovom slučaju se skup podataka za treniranje sastoji od točaka u ravnini i njihovih pripadnih klasifikacijskih oznaka.*

Nakon što je model izgrađen procjenjuje se njegova uspješnost na skupu podataka za testiranje, tj. uspoređuje se koliko se kalsifikacija dobivana ovom metodom nad ulaznim podacima skupa za testiranje razlikuje od prave klasifikacije (izlazni podaci skupa za testiranje). S obzirom na to da je moguće mijenjati hiperparametre metode (stupanj polinoma) potrebno je pronaći optimalne hiperparametre metode i optimalne parametre modela, te zatim evaluirati njihovu uspješnost na setu podataka za testiranje.

Bitno je razlikovati hiperparametre metode od parametra modela. Kao što je već spomenuto, u ovom primjeru imamo jedan hiperparametar metode (stupanj polinoma), dok su parametri modela u ovom slučaju koeficijenti polinoma.

1.2 Podaci

Neka x vektor iz $\mathbb{R}^D$ predstavlja jedan podatak koji se sastoji od D atributa. Realan broj koji se nalazi na i-tom mjestu u tom vektoru sadrži informaciju o i-tom atributu tog vektora. Oznaka za i-tu komponentu vektora x je gornji indeks x^i.

Neka se skup podataka $\mathfrak{D}$ sastoji od N vektora (primjeraka, točaka) iz $\mathbb{R}^D$ sa pripadnim oznakama iz $\mathbf{K}$. Skup oznaka (klasa) ćemo označavati s $\mathbf{K}$. Skup klasa je konačan i poželjno je da je on niske kardinalnosti. Donji indeks koristimo za oznaku n-tog vektora i n-te oznake (vrijednosti) primjerka.

$$\mathfrak{D} = \{(x_1, y_1), \ \ldots, (x_N, y_N)\}, \quad \text{gdje je } x_n \in \mathbb{R}^D, \ y_n \in \mathbf{K} \quad za\ n = 1, ..., N.$$

Napomena 1.2.1. *Sa x_n^i označavamo i-tu komponentu n-tog primjerka u skupu za učenje.*

Napomena 1.2.2. *Primjerci x (vektori iz $\mathbb{R}^D$) su ulazni podaci metode strojnog učenja, dok su njihove pripadajuće oznake y (vrijednosti iz $\mathbb{R}$) izlazni podaci.*

Primjer 1.2.3. *U kontekstu prethodnog primjera* (1.1.1)*, skup podataka se sastoji iz točaka iz $\mathbb{R}^2$ i njihovih pripadnih binarnih oznaka. U ovom slučaju n-ti primjerak iz skupa podataka označvamo sa: $x_n = (x_n^1, x_n^2)$, a njegova pripadna oznaka $y \in \boldsymbol{K} = \{0, 1\}$.*

Kao što je već ranije spomenuto, nadziranu metodu je potrebno najprije naučiti na skupu podataka za učenje, a zatim testirati na skupu podataka za testiranje. Ta dva skupa su međusobno disjunktni, te se uobičajeno dobivaju nasumičnom podjelom dostupnog skupa podataka u prethodno zadanom omjeru. Ovisno o veličini dostupnog skupa podataka i s obzirom na to koliko su klase (oznake) u izlaznim podacima balansirane postavljamo omjer. Uobičajeno je koristiti omjer skupa za učenje i skupa za testiranje 80 : 20.

Osim ovakve standardne podjele skupa podataka, moguće je koristiti i unakrsnu validaciju.

1.3 Učenje (treniranje) metode

Skup za učenje se sastoji od ulaznih primjeraka i pripadnih izlaznih oznaka nad kojima je potrebno izgraditi optimalan model. Model je u potpunosti određen svojim parametrima, te je izgradnja modela zapravo proces pronalaska optimalnih parametara modela. Kako bi znali koji su parametri bolji, a koji lošiji, potrebno je definirati funkciju greške koja mjeri pogrešku pojedinog modela nad podacima skupa za učenje. Metoda za različite parametre modela postiže različite vrijednosti funkcije pogreške. Kako bi se pronašao optimalan model, potrebno optimizirati (pronaći minimum) funkcije pogreške nad skupom parametara modela.

Metoda $\mathbf{L}(\alpha, x_n)$ sa parametrima modela α za n-ti primjerak skupa za učenje vraća izlaznu vrijednost $\tilde{y}_n$, dok je pripadna unaprijed poznata oznaka tog vektora y_n. Najjednostavniji primjer funkcije greške za skup podataka za učenje koji se sastoji od N primjeraka je:

$$J(\alpha) = \sum_{n=1}^{N} |\tilde{y}_n - y_n| = \sum_{n=1}^{N} |\mathbf{L}(\alpha, x_n) - y_n|.$$

Funkcija greške ovisi o skupu za učenje (unaprijed zadan i fiksan tijekom cijelog procesa treniranja) i o parametrima modela. Potrebno je pronaći one parametre za koje funkcija pogreške postiže svoj minimum, tj. potrebno je optimizirati metodu:

$$\min_{\alpha} J(\alpha) = \min_{\alpha} \sum_{n=1}^{N} |\mathbf{L}(\alpha, x_n) - y_n|.$$

Primjer 1.3.1. *U kontekstu primjera klasifikacije točke u ravnini x pomoću polinom p s obzirom na to da li je $p(x) \geq 0$ ili $p(x) < 0$, parametri modela su koeficijenti polinoma. Ukoliko se radi o realnim polinomima drugog stupnja s koeficijentima a_0, a_1 i a_2 potrebno je optimizirati prethodno definiranu funkciju pogreške za $\alpha = (a_o, a_1, a_2) \in \mathbb{R}^3$.*

1.4 Testiranje metode

Kada je metode naučena, tj. kada su pronađeni parametri modela za koje funkcija pogreške postiže svoj minimum, metodu je potrebno testirati na testnom skupu podataka te ocijeniti njezinu uspješnost klasifikacije.

Za dane ulazne podatke testnog skupa metoda vraća oznake koje zatim uspoređujemo sa istinitim izlaznim podacima skupa za testiranje. Ta usporedba dobivenih i istinitih podataka se vrši pomoću različitih mjera uspješnosti. Uobičajene mjere uspješnosti ćemo definirati za skup binarne klasifikacije, a zatim se one mogu lako generalizirati za ostale probleme.

U stvarnosti je često zahtjevno doći do izlaznih podataka skupa za učenje. Na primjer, za klasifikaciju slike s obzirom na to nalazi li se na njoj određeni objekt, skup podataka za učenje (i za testiranje) se sastoji od slika s pripadajućim oznakama (klasama). Ljudsko oko svaku sliku mora prikladno obilježiti, što nije komplicirano. Imajući na umu da kvalitetni skupovi podataka sadržavaju nekoliko desetaka tisuća slika, taj posao postaje itekako zahtjevan.

S obzirom na to da definiramo mjere uspješnosti binarne klasifikacije, primjerak iz skupa za testiranje klasificiramo u pozitivnu ili negativnu klasu te u odnosu na njegovu istinitu oznaku možemo reći da pripada skupu:

- *true positives* (TP) ukoliko mu je klasa dobivena metodom pozitivna i istinita klasa pozitivna
- *false positives* (FP) ukoliko mu je klasa dobivena metodom pozitivna i istinita klasa negativna
- *true negatives* (TN) ukoliko mu je klasa dobivena metodom negativna i istinita klasa negativna
- *false negatives* (FN) ukoliko mu je klasa dobivena metodom negativna i istinita klasa pozitivna

Sada pomoću broja primjeraka iz skupa za testiranje definiramo mjere koje se najčešće koriste:

- osjetljivost (*recall / sensitivity*) = $\frac{TP}{TP+FN}$ (omjer točno pozitivno klasificiranih primjeraka i svih primjeraka kojima je prava klasa pozitivna)
- specifičnost (*specificity*) = $\frac{TN}{TN+FP}$ (omjer točno negativno klasificiranih primjeraka i svih primjeraka kojima je prava klasa negativna)

- precisznost (*precision*) = $\frac{TP}{TP+FP}$ (omjer točno pozitivno klasificiranih primjeraka i svih primjeraka koji su pozitivno klasificirani)
- točnost (*accuracy*) = $\frac{TP+TN}{TP+TN+FP+FN}$ (omjer točno klasificiranih primjeraka i svih primjeraka)
- *F1 score* = $\frac{2TP}{2TP+FP+FN}$ (harmonijska sredina mjera recall i precision)

Napomena 1.4.1. *Ukoliko su istinita pozitivna i negativna klasa otprilike podjednako balansirane sve mjere dobro informiraju o uspješnosti metode. No, ukoliko imamo loše balansirane istinite klase tada moramo dobro razmisliti koje mjere dobro opisuju uspješnost metode. Na primjer ako je udio pozitivne klase 2% i za problem je bitno koliko je primjeraka točno pozitivno klasificirano, točnost i specifičnost nam neće biti dobri pokazatelji uspješnosti metode, jer će ju procijeniti kao dobrom čak i ako metoda loše klasificira pozitivnu klasu. U tom slučaju je puno korisnije koristi osjetljivost i preciznost kao validacijske mjere.*

Čest oblik validacije uspješnosti metode za binarnu klasifikaciju je ROC (*Receiver Operating Characteristic*). ROC krivulju dobivamo grafičkim prikazom ovisnosti mjere recall s obzirom na udio FP (1-*specificity*) za različite hiperparametre metode. Za slučajni klasifikator ROC krivulja je zapravo pravac $x = y$ (y-os za *recall* i x os za udio FP).
Uz ROC krivulju usko je povezana mjera AUC (*Area Under Curve*) koja mjeri vjerojatnost da metoda bolje klasificira od slučajnog klasifikatora. Njezina vrijednost odgovora površini ispod ROC krivulje. Primijetimo da ukoliko se radi o slučajnom klasifikatoru AUC iznosi 0.5.

Osim standardne validacije na testnom skupu podataka, puno realnijom i manje pristranom se pokazala unakrsna validacija. Ona prvotni skup podataka particionira u proizvoljnih k podskupova, te zatim u k koraka nalazi prosječnu uspješnost metode. U svakom koraku drugi podskup smatra testnim, a uniju ostalih koristi za treniranje metode.

Algoritam unakrsne validacije:

1. podijeli skup podataka u k disjunktnih podskupova
2. za $i = 1, ..., k$
    - i-ti podskup spremi kao skup podataka za testiranje
    - na preostalih k-1 podskupova nauči model
    - testiraj naučeni model na testnom skupu podataka
    - spremi rezultate testiranja i odbaci naučeni model

3. izračunaj aritmetički prosjek spremljenih rezultata

Unakrsna metoda validacije se koristi za optimizaciju hiperparametara metode. Prilikom završnog testiranja metode cilj nam je pokazati koliko dobro metoda generalizira. Iz tog razloga za završno testiranje metode uvijek koristimo podatke koji nisu ni na koji način sudjelovali u optimizaciji parametara modela i hiperparametara metode. Stoga, skup podataka najprije dijelimo na skup za učenje i testiranje, zatim na skupu podataka za učenje radimo optimizaciju hiperparmatera pomoću unakrsne validacije (koja ponovo dijeli skup na učenje na svojih k podskupova), te na kraju metodu za optimalne parametre testiramo na testnom skupu podataka.

Ukoliko je prvotni skup podataka loše balansiran, prilikom podjele skupa podataka na skupove za učenje i testiranje potrebno je obratiti pažnju na balansiranost klasa u skupovima za učenje i testiranje. Također je bitno naglasiti da testni skup podataka nikada ne balansiramo. Cilj testnog skupa podataka je da pokaže što realniju sliku o prediktivnim/klasifikacijskim sposobnostima metode.

Poglavlje 2

Teorija optimizacije

U ovom poglavlju navodimo teoreme optimizacije koji su usko vezani uz metodu potpornih vektora. Najprije navodimo nužne uvjete za postizanje lokalnog ekstrema (Fermatov teorem), zatim definiramo pojam Lagrangiana koji je krucijalan za pronalaženje rješenja matematičkog programiranja sa uvjetima tipa jednakosti. Na kraju navodimo Kuhn-Tuckerov teorem koji će se pokazati ključnim za pronalazak rješenja metode potpornih vektora.

2.1 Fermatov teorem

Definicija 2.1.1. *Točka x_0 je točka lokalnog minimuma (maksimuma) funkcije f definirane na $\mathbb{R}^D$ ako postoji $\epsilon > 0$ tako da za svaki x takav da $\|x - x_0\| < \epsilon$ vrijedi $f(x_0) \leq f(x)$, odnosno $f(x_0) \geq f(x)$.*
Točke lokalnog minimuma i maksimuma nazivamo točke lokalnih ekstrema.

Definicija 2.1.2. *Funkcija f(x) definirana na $\mathbb{R}$ je diferencijabilna u točki x_0 ako postoji $\alpha = (\alpha_1, ..., \alpha_D)$ tako da:*

$$f(x_0 + \lambda) = f(x_0) + \alpha\lambda + r(\lambda),$$

gdje je $r(\lambda) = o(|\lambda|)$, tj. za proizvoljan $\epsilon > 0$ postoji $\delta > 0$ tako da $\forall \lambda \in \mathbb{R}$ uvjet $|\lambda| < \delta$ povlači:

$$|r(\lambda)| < \epsilon|\lambda|.$$

Vrijednost α nazivamo diferencijal funkcije f u točki x_0 i označavamo sa $f'(x_0)$.
Odnosno:

$$f'(x_0) = \lim_{\lambda \to 0} \frac{f(x_0 + \lambda) - f(x_0)}{\lambda} = \alpha.$$

Teorem 2.1.3. *Neka je $f(x)$ funkcija jedne varijable, diferencijabilna u točki x_0. Ako je x_0 točka lokalnog ekstrema, tada $f'(x_0) = 0$.*

Definicija 2.1.4. *Funkcija f(x) definirana na $\mathbb{R}^D$ je diferencijabilna u točki x_0 ako postoji α tako da:*

$$f(x_0 + \lambda) = f(x_0) + \sum_{i=1}^{D} \alpha_i h_i + r(h)$$

gdje je $r(h) = o(\|h\|)$, tj, za proizvoljan $\epsilon > 0$ postoji $\delta > 0$ tako da $\|h\| = \sqrt{h_1 + ... + h_D} < \delta$ povlači:

$$\|r(h)\| \leq \epsilon \|h\|.$$

Vektor $\alpha = (\alpha_1, ..., \alpha_D)$ nazivamo diferencijal funkcije f u točki x_0 i označavamo sa $\nabla f(x_0)$. Vrijednost α_i :

$$\alpha_i = \lim_{\lambda \to 0} \frac{f(x_0 + \lambda e_i) - f(x_0)}{\lambda}, \quad gdje\ je\ e_i = (0, ..., 0, 1, 0, ..., 0)$$

nazivamo parcijalna derivacija i označavamo sa $f'_{x_i}(x_0)$ ili sa $\frac{\partial f(x_0)}{\partial x_i}$.
Vrijedi: $\nabla f(x_0) = (f'_{x_1}(x_0), ..., f'_{x_D}(x_0))$.

Korolar 2.1.5. *Neka je f funkcija definirana na $\mathbb{R}^D$ diferencijabilna u točki x_0. Ako je x_0 točka lokalnog ekstrema funkcije f tada $\nabla f(x_0) = 0$.*

Napomena 2.1.6. *$\nabla f(x_0) = 0$ za funkciju definiranu na $\mathbb{R}^D$ povlači:*

$$f_{x_1}(x_0) = ... = f_{x_D}(x_0) = 0$$

2.2 Lagrangeov teorem

Sljedeći korak je riješiti optimizacijski problem minimizacije ili maksimizacije funkcije f_0 ($f_0(x) \to \min$ ili $f(x) \to \max$) tako da vrijede jednakosti: $f_1(x) = ... = f_m(x) = 0$. Naravno, za funkcije $f_1, ..., f_m$ pretpostavljamo da su dovoljno glatke.
Ovaj problem nazivamo zadaća matematičkog programiranja i zapisujemo ga kao:

$$\min_x f_0(x)$$
$$\text{uz uvjete : } f_i(x) = 0, \quad i = 1, ..., m.$$

Funkciju $\mathbf{L}(x, \lambda, \lambda_0)$ gdje $\lambda = (\lambda_1, ..., \lambda_m)$ definiranu s:

$$\mathbf{L}(x, \lambda, \lambda_0) = \sum_{k=0}^{m} \lambda_k f_k(x)$$

nazivamo *Lagrangeova funkcija* ili *Lagrangian*, a $\lambda_0, ..., \lambda_m$ nazivamo *Lagrangeovi multiplikatori*.

Teorem 2.2.1. *Neka su funkcije $f_k,\ k = 0, 1, ..., m$ neprekidne i diferencijabilne u okolini točke x_0. Ako je x_0 točka lokalnog ekstrema, tada postoje Lagrangeovi multiplikatori $\lambda^* = (\lambda_1^*, ..., \lambda_m^*)$ i λ_0 koji nisu svi istovremeno jednaki 0 tako da vrijedi:*

$$\nabla \mathbf{L}(x_0, \lambda, \lambda_0) = 0.$$

To jest:

$$\mathbf{L}'_{x_i}(x_0, \lambda, \lambda_0) = 0, \quad i = 1, ..., D.$$

Kako bi vrijedilo da $\lambda_0 \neq 0$ dovoljno je da su vektori $\nabla f_1(x_0), ..., \nabla f_m(x_0)$ linearno nezavisni.

Napomena 2.2.2. *Kako bi se pronašla stacionarna točka, potrebno je riješiti $m + D$ jednadžbi s $m + D + 1$ nepoznanica:*

$$\frac{\partial}{\partial x_i}(\sum_{k=0}^{m} \lambda_k f_k(x)) = 0, \quad i = 1, ..., D \tag{2.1}$$

$$f_1(x) = ... = f_m(x) = 0 \tag{2.2}$$

Lagrangeovi multiplikatori su međusobno zavisni, tj. za $\lambda_0 \neq 0$ moguće je pronaći konstantu tako da množenjem Lagrangeovih multiplikator istom dobivamo $\lambda_0 = 1$. Na taj način broj jednadžbi i broj nepoznanica postaje jednak.
Jednadžbe (2.1) *i* (2.2) *moguće je zapisati u obliku:*

$$\nabla \mathbf{L}_x(x_0, \lambda, 1) = 0$$

$$\nabla \mathbf{L}_\lambda(x_0, \lambda, 1) = 0$$

2.3 Karush-Kuhn-Tuckerov teorem

Lagrange je uveo metodu *Lagrangeovih* multiplikatora za rješavanje optimizacijskog problema sa uvjetima koji su određeni pomoću jednakosti. Kuhn i Tucker predlažu rješenje za problem konveksne optimizacije, tj. za minimizaciju funkcije s (konveksnim) uvjetima koji su određeni pomoću nejednakosti.

Definicija 2.3.1. *Podskup A vektorskog prostora je konveksan ako za svake dvije točke x i y iz A, skup A sadrži interval povezan tim točkama:*

$$[x, y] = \{z \ : \ z = \alpha x + (1 - \alpha)y, \quad 0 \leq \alpha \leq 1\} \subseteq A.$$

Definicija 2.3.2. *Za funkciju f definiranu na skupu X kažemo da je konveksna ako za svake dvije točke x, y iz X vrijedi Jensenova nejednakost:*

$$f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha)f(y), \quad 0 \leq \alpha \leq 1.$$

Neka je X vektorski prostor, i neka je A njegov konveksni potprostor, a f_k $k = 0, 1, ..., m$ konveksne funkcije.

Konveksni optimizacijski problem podrazumijeva:
Minimizaciju funkcionala:

$$f_0(x) \to \min f_0(x) \tag{2.3}$$

s obzirom na uvjete:

$$x \in A \tag{2.4}$$

$$f_k(x) \leq 0, \quad k = 1, ..., m \tag{2.5}$$

Teorem 2.3.3. *Ako x^* minimizira funkciju* (2.3) *s obzirom na uvjete* (2.4) *i* (2.5)*, tada postoje Lagrangeovi multiplikatori λ_0^* i $\lambda^* = (\lambda_1^*, ..., \lambda_m^*)$ takvi da nisu svi istovremeno jednaki 0 i takvi da su sljedeće tri tvrdnje istinite:*
(a) Princip minimuma:

$$\min_{x \in A} L(x, \lambda_0^*, \lambda^*) = L(x^*, \lambda_0^*, \lambda^*).$$

(b) Nenegativnost multiplikatora:

$$\lambda_k^* \geq 0, \quad k = 0, 1, ..., m.$$

(c) Kuhn-Tuckerovi uvjeti:

$$\lambda_k^* f_k(x^*) = 0, \quad k = 0, 1, ..., m.$$

Ako je $\lambda_0 \neq 0$, tada su uvjeti (a), (b) i (c) dovoljni da x^* bude rješenje optimizacijskog problema.
Za $\lambda_0 \neq 0$ je dovoljno da postoji $\tilde{x}$ takav da Slaterov uvjet vrijedi:

$$f_i(\tilde{x}) < 0, \quad i = 1, ..., m.$$

Korolar 2.3.4. *Ako je Slaterov uvjet zadovoljen, tada se može izabrati $\lambda_0 = 1$ i Lagrangian napisati u obliku:*

$$L(x, 1, \lambda) = f_0(x) + \sum_{k=1}^{m} \lambda_k f_k(x).$$

Sada je Lagrangian definiran s $m + D$ varijabli, te su uvjeti Kuhn-Tuckerovog teorema ekvivalentni postojanju sedlaste točke (x^*, λ^*) Lagrangiana, to jest:

$$\min_{x \in A} L(x, 1, \lambda^*) = L(x, 1, \lambda^*) = \max_{\lambda > 0} L(x^*, 1, \lambda).$$

Napomena 2.3.5. *Dokazi teorema iz ovog poglavlja mogu se naći u [7].*

Poglavlje 3

Metoda potpornih vektora

U ovom poglavlju predstavljamo metodu potpornih vektora. Najprije definiramo pojam hiperravnine, a zatim definiramo metodu potpornih vektora za binarnu klasifikaciju pomoću hiperravnine. Kako bismo osigurali jedinstvenost hiperravnine razdvajanja (a i imali bolju metodu klasifikacije) želimo da su primjerci i pozitivne i negativne klase maksimalno udaljeni od hiperravnine. Ovakva klasifikacija tvrdom marginom moguća je jedino u slučaju kada je skup podataka koji je potrebno klasificirati linearno razdvojiv. Za linearno nerazdvojive skupove uvodimo pojam meke margine i jezgreni trik pomoću kojeg skup podataka koji nije linearno razdvojiv preslikavamo u prostor u kojem dobivamo linearno razdvojiv skup. Također predstavljamo i dualni problem traženja optimalne hiperravnine razdvajanja, koji olakšava rješavanje kompliciranog pripadnog primarnog problema.

3.1 Hiperravnina

Definicija 3.1.1. *Neka je V vektorski prostor, $x_0 \in V$ vektor, , te $U \subseteq V$ potrostor. Tada potprostor:*

$$L = x_0 + U = \{x_0 + u \ : \ u \in U\} = \{v \in V \ : \ \exists u \in U \text{ tako da } v = x_0 + u\} \subseteq V$$

zovemo afini potprostor ili linearna mnogostrukost u V.
Smjer (potprostor) afinog potrostora je određen s U, a točku x_0 nazivamo potporna točka.

Napomena 3.1.2. *Primijetimo da niti jedan afini potprostor ne sadrži nul-vektor.*

Napomena 3.1.3. *Neka je $L = x_0 + U$ k-dimenzionalni afini potprostor vektorskog prostora V. Ako je $U = [\{b_1, ..., b_k\}]$, tada za svaki $x \in L$ postoje jedinstveni koeficijenti $\lambda_1, ..., \lambda_k$ iz $\mathbb{R}$ tako da:*

$$x = x_0 + \lambda_1 b_1 + ... + \lambda_k b_k.$$

Napomena 3.1.4. *Jednodimenzionalne afine potprostore nazivamo pravci, dvodimenzionalne afine potprostre ravninama.*

Definicija 3.1.5. *Neka je V n-dimenzionalni vektorski prostor. Afini potprostor od V dimenzije* $n-1$ *nazivamo hiperravnina.*

3.2 Binarna klasifikacija hiperravninom - Intuicija

Neka je $x \in \mathbb{R}^D$ primjerak iz skupa podataka. Promatramo funkciju f parametriziranu sa $\omega \in \mathbb{R}^D$ i $b \in \mathbb{R}$, uz oznaku $\langle \cdot, \cdot \rangle$ za skalarni produkt:

$$f : \mathbb{R}^D \longrightarrow \mathbb{R} \quad f(x) = \langle \omega, x \rangle + b$$

Hiperravninu u $\mathbb{R}^D$ za afino preslikavanje f možemo definirati pomoću:

$$\mathbf{H} = \{x \in \mathbb{R}^D \ : \ f(x) = 0\}$$

gdje je vektor ω normala hiperravnine $\mathbf{H}$.

Napomena 3.2.1. *Geometrijski gledano hiperravnina dijeli prostor u kojem se nalazi na dva dijela. U* $\mathbb{R}^2$ *hiperravnina je zapravo pravac koji dijeli prostor* $\mathbb{R}^2$ *na dvije poluravnine. Stoga, svaku točku iz* $\mathbb{R}^2$ *možemo klasificirati s obzirom na to u kojoj se poluravnini nalazi.*

Zadatak binarne klasifikacije je vektoru $x \in \mathbb{R}^D$ pridružiti oznaku klase $y \in \{-1, +1\}$. Iz perspektive hiperravnina, taj problem možemo poistovjetiti sa određivanjem s koje "strane" hiperrravnine se točka x nalazi u prostoru $\mathbb{R}^D$. Ukoliko se nalazi sa "gornje strane" pridružujemo joj klasu $+1$, a ukoliko se nalazi sa donje strane hiperravnine pridružujemo joj oznaku -1. Sada smo problem binarne klasifikacije u prostoru $\mathbb{R}^D$ sveli na:

$$\text{Za } x \ \in \mathbb{R}^D \text{odrediti oznaku } y = \begin{cases} +1, & \text{ako } \langle \omega, x \rangle + b \geq 0 \\ -1, & \text{ako } \langle \omega, x \rangle + b < 0 \end{cases}$$

Što možemo kraće zapisati u obliku:

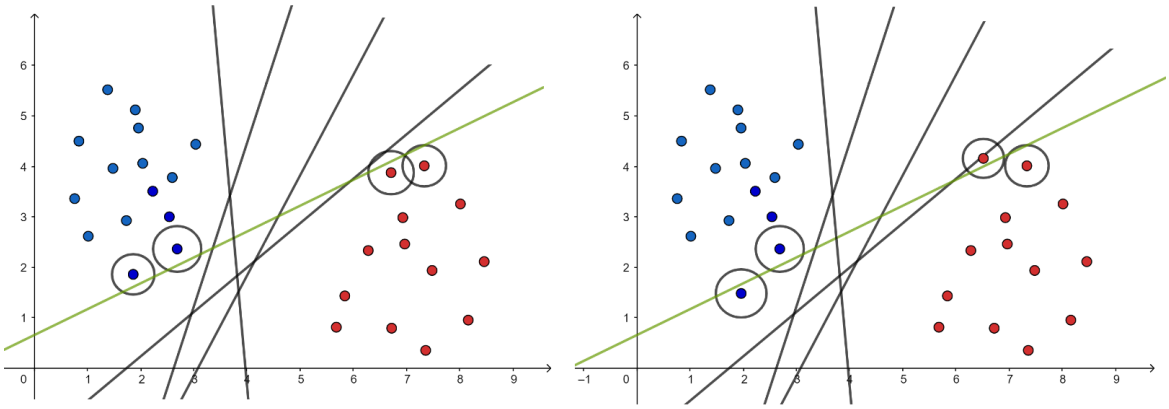
$$\text{za } x \ \in \mathbb{R}^D \text{odrediti oznaku } y \text{ tako da vrijedi } y \cdot (\langle \omega, x \rangle + b) \geq 0$$

Margina

Za linearno razdvojiv skup podataka $\{(x_1, y_1), ..., (x_N, y_N)\}$ imamo beskonačno mnogo kandidata za hiperravninu koje određuju binarnu klasifikaciju. Primjer u prostoru $\mathbb{R}^2$ na slici (3.1) lako vizualizira kako hiperravnina za problem binarne klasifikacije nije jedinstvena.

Kako bi imali jedinstvenu hiperravninu, uzimamo onu hiperravninu koja maksimizira marginu između pozitivnih i negativnih primjerka skupa podataka, gdje pod pojmom margina podrazumijevamo udaljenost hiperravnine od najbližeg primjerka iz skupa podataka koji je potrebno klasificirati.

U realnosti, zbog pogrešaka prilikom mjerenja podaci su rijetko kada u potpunosti točni. Stoga, za primjerke kod kojih je prisutan šum (pogreška u mjerenju) u blizini hiperravnine razdvajanja čija je margina uska lako može doći do pogrešnog klasificiranja (slika 3.1. desno)

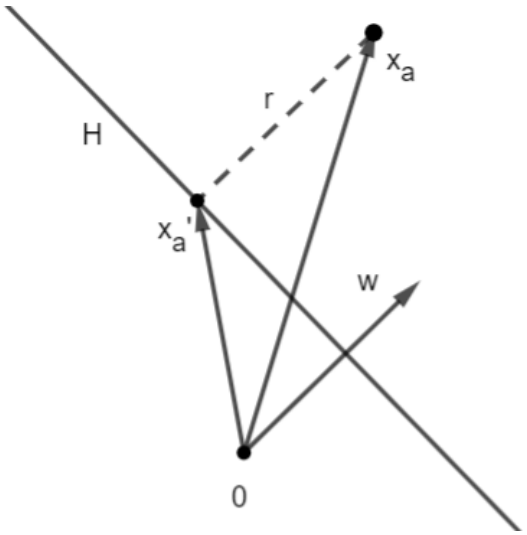


Slika 3.1: U $\mathbb{R}^2$ linearno razdvojiv skup podataka je prikazan crvenim i plavim točkama koje ujedno prikazuju i njihovu klasu. Primijetimo da postoji beskonačno mnogo hiperravnina (pravaca) koje razdvajaju taj skup podataka. Pretpostavljamo da je kod podataka prisutan šum, tj. da se njihova prava vrjednost može nalaziti u njihovoj bliskoj okolini (prikazano kružnicama). Zelena hiperravnina razdvajajnja u ovome slučaju krivo klasificira podatke na slici desno, jer se zbog svoje uske margine ne može nositi sa šumom.

Promatramo hiperravninu $\mathbf{H} = \langle \omega, x \rangle + b$ i primjerak x_a iz skupa podataka. Bez smanjenja općenitosti, neka x_a ima pozitivnu oznaku, tj. $\langle \omega, x_a \rangle + b > 0$. Želimo izračunati udaljenost $r > 0$ primjerka x_a od hiperravnine $\mathbf{H}$. Kako je ω ortogonalan na hiperravninu $\mathbf{H}$, određivanje udaljenosti točke od hiperravnine $\mathbf{H}$ je zapravo određivanja koeficijenta kojim je potrebno skalirati normalu ω.
Vektor x_a možemo zapisati kao zbroj njegove projekcije na hiperravninu x'_a i normale ω skalirane udaljenošću vektora x_a od $\mathbf{H}$:

$$x_a = x'_a + r\frac{\omega}{||\omega||}.$$

Na ovaj način smo implicitno definirali udaljenost točke x_a od hiperravnine **H**. Cijela ideja je očita za prostor $\mathbb{R}^2$, te je prikazana na slici (3.2).



Slika 3.2: Svaki vektor x_a iz $\mathbb{R}^2$ možemo prikazati kao zbroj njegove projekcije na hiperravninu x'_a i vekotra ω skaliranog udaljenošću x_a od **H**.

Neka je x_a primjerak koji se nalazi najbliže hiperravnini **H**, tada je r margina, te svi primjerci moraju biti udaljeni od ravnine za barem r u pozitivnom ili negativnom smjeru, ovisno o tome koje oznake imaju:

$$y \cdot (\langle \omega, x \rangle + b) \geq r, \quad \text{za primjerak } x \text{ i njemu pripadnu oznaku } y.$$

Kako je za normalu jedino bitno da su primjerci udaljeni u pozitivnom ili negativnom smjeru za r, bez smanjenja općenitosti možemo pretpostaviti da je $\|\omega\| = 1$.

Sada problem pronalaska hiperravnine razdvajanja možemo napisati kao problem maksimizacije, tj. tražimo hiperravninu sa maksimalnom marginom na skupu podataka za treniranje $\{(x_1, y_1), ..., (x_N, y_N)\}$:

$$\max_{\omega, b} \; r \tag{3.1}$$

$$\text{uz uvjete} : y_i \cdot (\langle \omega, x_i \rangle + b) \geq r, \quad i = 1, ..., N \tag{3.2}$$

$$\|\omega\| = 1 \tag{3.3}$$

$$r > 0 \tag{3.4}$$

3.3 Tvrda margina metode potpornih vektora

U prijašnjem odjeljku dolazimo do (3.1) nakon zaključka da nam je bitniji smjer normale ω od njene duljine, te dodajemo pretpostavku $\|\omega\| = 1$.
U ovom odjeljku izvodimo problem maksimizacije margine uz drugačije pretpostavke. Umjesto pretpostavke da je normala normalizirana, ovdje biramo normalu takve veličine da je $\langle \omega, x_a \rangle + b = 1$ za najbliži primjerak x_a hiperravnini **H**.

Neka je x'_a ortogonalna projekcija primjerka x_a na hiperravninu **H**. Kako x'_a leži na hiperravnini, po definiciji hiperravnine vrijedi:

$$\langle \omega, x'_a \rangle + b = 0.$$

Pomoću $x_a = x'_a + r\frac{\omega}{\|\omega\|}$ dobivamo:

$$\langle \omega, x_a - r\frac{\omega}{\|\omega\|} \rangle + b = 0$$

$$\langle \omega, x_a \rangle + b - r\frac{\langle \omega, \omega \rangle}{\|\omega\|} = 0$$

$$1 - r\frac{\|\omega\|^2}{\|\omega\|} = 0$$

Konačno dobivamo da za udaljenost najbližeg vektora od hiperravnine (margine r) vrijedi:

$$r = \frac{1}{\|\omega\|} \tag{3.5}$$

Sada, problem maksimizacije margine za hiperravninu **H** na skupu podataka za učenje $\{(x_1, y_1), ..., (x_N, y_N)\} \subset (\mathbb{R}^D \times \mathbb{R})$ svodimo na:

$$\max_{\omega \in \mathbb{R}^D, b \in \mathbb{R}} \frac{1}{\|\omega\|} \tag{3.6}$$

$$\text{uz uvjete} : y_n \cdot (\langle \omega, x_n \rangle + b) \geq 1, \quad n = 1, ..., N \tag{3.7}$$

Problem maksimizacije recipročne vrijednosti norme normale jednak je problemu minimizacije kvadrata norme normale:

$$\min_{\omega \in \mathbb{R}^D, b \in \mathbb{R}} \frac{1}{2}\|\omega\|^2 \tag{3.8}$$

$$\text{uz uvjete} : \quad y_n \cdot (\langle \omega, x_n \rangle + b) \geq 1, \quad n = 1, ..., N \tag{3.9}$$

Gornji minimizacijski problem nazivamo tvrda margina metode potpornih vektora, zato što takav model ne dopušta narušavanje uvjeta margine.

3.4 Meka margina metode potpornih vektora

Za razliku od tvrde margine koja ne dopušta narušavanje uvjeta margine, sad uvodimo model u kojem su dopuštene klasifikacijske pogreške. Zbog mogućnosti narušavanja uvjeta margine, taj problem se naziva meka margina metode potpornih vektora.

Svakom paru (x_n, y_n) iz skupa podataka za učenje pripisujemo slabu (eng. slack) varijablu ξ_n . Njezina svrha je dopustiti određenim primjercima skupa za učenje da prekrše uvjet margine, tj. da se nalaze s krive strane hiperravnine ili unutar margine. Oduzimanjem vrijednosti varijable ξ_n od margine, te ograničavajući ξ_n na samo nenegativne vrijednosti dobivamo formulaciju:

$$\min \frac{1}{2}\|\omega\|^2 + C\sum_{n=1}^{N}\xi_n \tag{3.10}$$

$$\text{uz uvjete } y_n \cdot (\langle \omega, x_n \rangle + b) \geq 1 - \xi_n, \quad n = 1, ..., N \tag{3.11}$$

$$\xi_n \geq 0, \quad n = 1, ..., N \tag{3.12}$$

Vrijednost parametra $C > 0$ označava težinu svake slack varijable. Ukoliko je slack varijabla različita od 0, tj. ako se ne poštuje uvjet tvrde margine, vrijednost funkcije koja se minimizira je uvećana za pogrešku margine pomnoženu sa parametrom C ($\xi \cdot C$). Stoga velika vrijednost parametra C stavlja veliku težinu na svako narušavanje uvjeta tvrde margine te je margina u tom slučaju uska, dok je za male vrijednosti parametra C margina široka.
Slika (3.3) prikazuje kako veličina parametra C utječe na širinu margine.

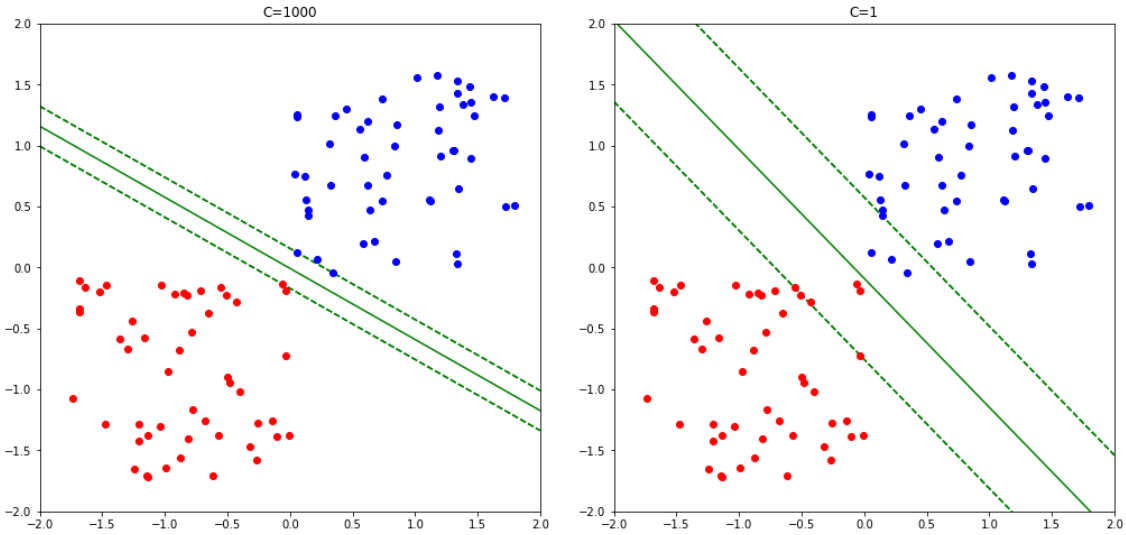
3.5 Funkcija pogreške

Sada ćemo promatrati ranije definirane formulacije metode potpornih vektora iz perspektive funkcije pogreške. U poglavlju 1 je spomenuto kako se parametri modela (u ovom slučaju parametri hiperravnine ω i b) biraju tako da funkcija pogreške na skupu za učenje poprima minimalnu vrijednost. Prisjetimo se da je hirperravnina definirana sa:

$$f(x) = \langle \omega, x \rangle + b.$$

U slučaju kada x leži na hiperravnini $f(x) = 0$, a ukoliko x leži na margini $f(x) = 1$ za x iz pozitivne klase i $f(x) = -1$ za x iz negativne klase.

Za potrebe binarne klasifikacije potrebno je izabrati funkciju pogreške koja je sposobna dobro mjeriti razliku između klase dobivene metodom ($f(x_i)$) i istinite klase (y_i) svakog primjerka skupa za učenje. Jedan od kandidata je $zero - one\ loss$ funkcija moja mjeri broj



Slika 3.3: Na lijevoj slici prikazana je hiperravnina razdvajanja (puni pravac) i margine (isprekidani pravci) za $C = 1000$, a desna hiperravninu razdvajanja i margine za $C = 1$. Velika vrijednost parametra C daje veću težinu svakom prekršenom uvjetu tvrde margine i na taj način stvara uže margine.

krivo klasificiranih primjeraka na skupu za učenje, no takva funkcija stvara kombinatorne probleme koji su složeni za rješavanje.

Jedna od čestih funkcija pogreške za ovakav problem je *hinge loss* funkcija ($l(\cdot)$) definirana sa:

$$l(t) = \max\{0, 1 - t\}, \quad \text{gdje je } t = y \cdot f(x) = y \cdot (\langle \omega, x \rangle + b) \tag{3.13}$$

Primijetimo da ukoliko se $f(x)$ nalazi sa točne strane hiperravnine, te je udaljenost od hiperravnine veća ili jednaka 1, tada dobivamo da je $t \geq 1$ te funkcija gubitka poprima vrijednost 0. Za $f(x)$ sa dobre strane hiperravnine, ali za x unutar margine hiperravnine funkcija gubitka poprima pozitivnu vrijednost manju od 1. Ukoliko se primjerak nalazi sa krive strane hiperravnine, funkcija gubitka poprima vrijednosti veće od 1.

Za dani skup za učenje $\{(x_1, y_1), ..., (x_N, y_N)\}$ želimo minimizirati funkciju gubitka. Pomoću (3.13) funkcije gubitka dobivamo optimizacijski problem:

$$\min_{\omega,b} \frac{1}{2}\|\omega\|^2 + C \sum_{n=1}^{N} \max\{0, 1 - y_n(\langle \omega, x \rangle + b)\} \tag{3.14}$$

Optimizacijski problem (3.14) jednak je definiciji problema slabe margine (3.10).

3.6 Dualni problem

Napomena 3.6.1. *Optimizacijski problem* (3.10) *nazivamo primarni problem meke margine metode potpornih vektora, zato varijable ω, b i ξ nazivamo primarnim varijablama.*

Za definiciju dualnog problema uvodimo Lagrangian:

$$L(\omega, b, \xi, \alpha, \gamma) = \frac{1}{2}\|\omega\|^2 + C\sum_{n=1}^{N}\xi_n - \sum_{n=1}^{N}\alpha_n \cdot (y_n \cdot (\langle \omega, x_n\rangle + b) - 1 + \xi_n) - \sum_{n=1}^{N}\gamma_n\xi_n. \tag{3.15}$$

Lagrangeove multiplikatore $\alpha_n \geq 0$ koristimo kao odgovarajuću zamjenu za uvjet koji osigurava da su primjerci dobro klasificirani:

$$y_n \cdot (\langle \omega, x_n\rangle + b) \geq 1 - \xi_n \quad n = 1, ..., N.$$

Dok Lagrangeovi multiplikatori $\gamma_n \geq 0$ osiguravaju ne-negativnost slack varijabli.

Po Fermatovom teoremu točke minimuma zadovoljavaju da je derivacija jednaka 0. Deriviranjem Lagrangiana po primarnim varijablama ω, b i ξ dobivamo:

$$\frac{\partial L}{\partial \omega} = \omega^T - \sum_{n=1}^{N}\alpha_n y_n x_n^T \tag{3.16}$$

$$\frac{\partial L}{\partial b} = -\sum_{n=1}^{N}\alpha_n y_n \tag{3.17}$$

$$\frac{\partial L}{\partial \xi_n} = C - \alpha_n - \gamma_n \tag{3.18}$$

Izjednačavanjem gornje jednadžbe (3.16) sa 0 dobivamo da se ekstrem Lagrangiana postiže za:

$$\omega = \sum_{n=1}^{N}\alpha_n y_n x_n$$

Izjednačavanjem jednadžbe (3.17) sa 0 zaključujemo da je vektor ω afina kombinacija primjeraka iz skupa za učenje.

Napomena 3.6.2. *Primjerak x_n za kojeg je pripadajući parametar $\alpha_n = 0$ ne pridonosi ω, dok primjerci x_n za koje $\alpha_n > 0$ zovemo potporni vektori, jer je hiperravnina dobivena afinom linearnom kombinacijom upravo takvih primjeraka iz skupa za učenje.*

Uvrštavanjem rješenja ω u Lagangian (3.15) dobivamo dualni problem:

$$\begin{aligned} D(\xi,\alpha,\gamma) =& \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \langle x_i, x_j\rangle - \sum_{i=1}^{N} y_i \alpha_i \langle \sum_{j=1}^{N} y_j \alpha_j x_j, x_i\rangle \\ &+ C\sum_{i=1}^{N}\xi_i - b\sum_{i=1}^{N} y_i\alpha_i + \sum_{i=1}^{N}\alpha_i - \sum_{i=1}^{N}\alpha_i\xi_i - \sum_{i=1}^{N}\gamma_i\xi_i \end{aligned}$$

Gornju jednakost možemo zapisati u obliku:

$$D(\xi,\alpha,\gamma) = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \langle x_i, x_j\rangle + \sum_{i=1}^{N}\alpha_i + \sum_{i=1}^{N}(C-\alpha_i-\gamma_i)\xi_i$$

Napomena 3.6.3. *Primjetimo da se dualni problem dobiva maksimizacijom primarnog, ukoliko je primarni problem minimizacijski. Kako je maksimizacija neke funkcije ekvivalentna minimizaciji iste negativne funkcije, dualni problem možemo ponovo zapisati kao minimizacijsku zadaću.*

Sada dobivamo zapis dualnog problema meke margine:

$$\min_{\alpha} \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \langle x_i, x_j\rangle - \sum_{i=1}^{N}\alpha_i \tag{3.19}$$

$$\text{uz uvjete : } \sum_{i=1}^{N} y_i\alpha_i = 0 \tag{3.20}$$

$$0 \le \alpha_i \le C \quad i = 1, ..., N \tag{3.21}$$

Primijetimo da moraju za točke u kojima se postiže ekstrem moraju zadovoljavati Karush-Kuhn-Tuckerove uvjete:

$$\begin{aligned} \alpha_n(y_n(\langle\omega, x_n\rangle + b) - 1 + \xi_n = 0 \\ \gamma_n\xi_n = 0 \\ y_n(\langle\omega, x_n\rangle + b) - 1 + \xi_n \ge 0 \end{aligned}$$

Kada pronađemo optimalan dualni parametar α^* iz njega možemo izvesti primarni parametar $\omega^* = \sum_{n=1}^{N}\alpha_n^* y_n x_n$. Parametar b nalazimo pomoću primjerka x_n koji leži točno na margini, tada $b* = y_n - \langle\omega^*, x_n\rangle$. Ukoliko se niti jedan primjerak ne nalazi na margini jedna od mogućnosti je računati $|y_n - \langle\omega^*, x_n|$ za sve potporne vektore i kao b^* uzimeti medijan.

3.7 Jezgreni trik

Primijetimo da se u dualnom problemu metode potpornih vektora (3.19) - (3.21) skalarni produkt odnosi samo na primjerke skupa za učenje x_i. Ta činjenica nam omogućava da metodu primijenimo na primjerke skupa za učenje reprezentirane pomoću drugih značajki $\phi(x_i)$, tj. da primjerke skupa za učenje najprije preslikamo u prostor u kojem ih je lakše razdvojiti, a zatim primijenimo metodu.

Izbor nove reprezentacije vektora x_i i parametra metode potpornih vektora su dva zasebna problema, što nam omogućuje da se ta dva problema rješavaju nezavisno. Jedno od glavnih ograničenja metode potpornih vektora je to što je ona općenito linearan optimizacijski problem, tj. može separirati samo linearno razdvojive skupove. No, nezavisnost funkcije $\phi(\cdot)$ nam omogućuje da ona bude i nelinearna, tj. da skupove koji nisu linearno separabilni transformira u linearno separabilne skupove, čime smo omogućili da metodu potpornih vektora primjenjujemo i na skupove koji nisu linearno separabilni.

Umjesto da eksplicitno definiramo preslikavanje $\phi(\cdot)$ i računamo vrijednost skalarnog produkta za primjerke x_i i x_j, definiramo funkciju sličnosti $k(\cdot,\cdot)$ za primjerke x_i i x_j. Funkcije sličnosti koje implicitno definiraju nelinearno mapiranje $\phi(\cdot)$ nazivamo jezgrene funkcije.

Definicija 3.7.1. *Funkciju* $k : \mathbf{X} \times \mathbf{X} \to \mathbb{R}$ *za koju postoji Hilbertov prostor* $\mathbf{H}$ *i preslikavanje* $\phi : \mathbf{X} \to \mathbf{H}$ *tako da vrijedi:*

$$k(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle_{\mathbf{H}}$$

nazivamo jezgrena funkcija.

Dualni problem metode potpornih vektora uz pomoć jezgrenog trika glasi:

$$\min_{\alpha} \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j k(x_i, x_j) - \sum_{i=1}^{N} \alpha_i$$

$$\text{uz uvjete : } \sum_{i=1}^{N} y_i \alpha_i = 0$$

$$0 \leq \alpha_i \leq C, \quad i = 1, ..., N$$

Matricu $\mathbf{K} \in \mathbb{R}^{N \times N}$ dobivenu primjenom jezgrene funkcije na ulazne podatke skupa za učenje $\{(x_1, y_1), ..., (x_N, y_N)\}$ nazivamo *Gramova matrica*. Prednost jezgrenog trika je sadržana u tome da se Grammova matrica može izračunati prije primjenjivanja same metode, čime se dodatno smanjuje vrijeme izvršavanja.

Jezgrene funkcije moraju biti simetrične i pozitivno semidefinitne:

$$z^T \mathbf{K} z \geq 0, \quad \forall z \in \mathbb{R}^N.$$

Neki od uobičajenih izbora za jezgrenu funkciju su:

- polinomijalna stupnja d: $k(x_i, x_j) = (\gamma \langle x_i, x_j \rangle + r)^d$
- radijalna bazna funkcija: $k(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|^2)$
- sigmoid: $k(x_i, x_j) = \tanh(\gamma \langle x_i, x_j \rangle + r)$

Jezgrene funkcije imaju svoje parametre (hiperparamterti modela) koje je moguće podesiti kako bi se skup podataka za učenje preslikao u što više razdvojiv skup. Kako bi stvorili intuiciju o tome kako se ponašaju funkcije koje definiraju jezgre promatramo linearno nerazdvojiv skup. Dvodimenzionalni podaci su najprije prikazani u ravnini (lijeva slika (3.4)) te su obojeni u žuto ukoliko su bliže ishodištu, a ljubičasto ukoliko su više udaljeni od njega. Zatim iste podatke prikazujemo u trodimenzionalnom prostoru gdje se na z-osi nalaze vrijednosti funkcije koja definira jezgru. Na slici (3.5) je prikazan utjecaj polinomijalne jezgre stupnja 2 i utjecaj radijalne jezgre, a na lijevoj slici (3.4) utjecaj sigmoidalne jezgre na podatke.

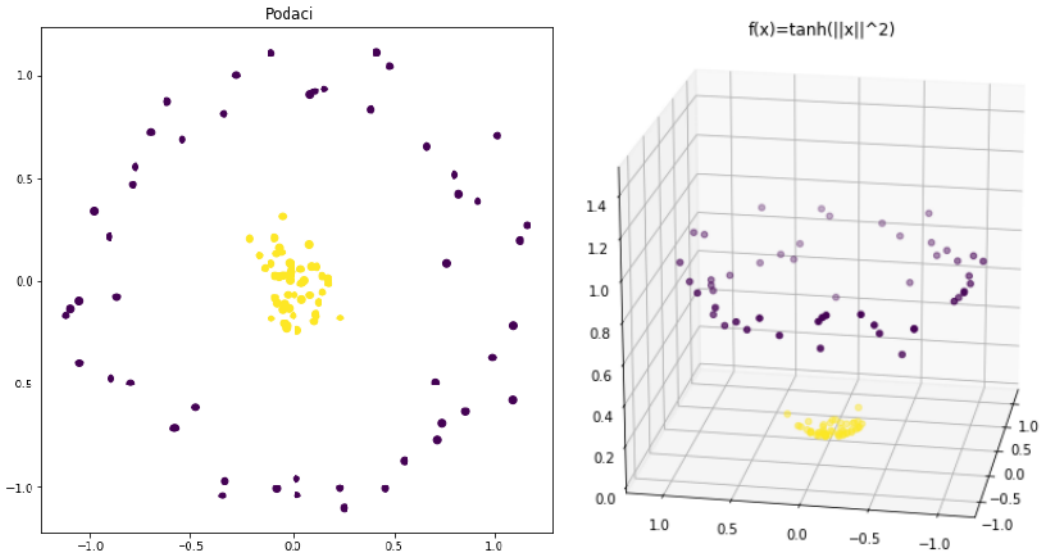
Zatim prikazujemo granice klasifikatora za dva linearno nerazdvojiva skupa podataka za linearnu jezgru i radijalnu jezgru (slika (3.6)) . Naravno, metoda potpornih vektora sa linearnom jezgrom nije sposobna uspješno razdvojiti linearno nerazdvojiv skup.

3.8 Numeričko rješenje

Kod primarnog problema (3.10) optimizacijske varijable su dimenzije D (broj značajki/atributa u svakom vektoru iz skupa za učenje), dok kod dualnog problema (3.19) - (3.21) optimizacijske varijable su veličine N (broj primjeraka u skupu za učenje). Kako bi zapisali problem (3.10) u matričnom zapisu, preoblikujemo jednadžbe te koristeći matrični zapis za skalarni produkt dobivamo:

$$\begin{aligned} &\min \frac{1}{2}\|\omega\|^2 + C \sum_{n=1}^{N} \xi_n \\ \text{uz uvjete :} \quad &- y_n x_n^T \omega - y_n b - \xi_n \leq -1, \quad n = 1, ..., N \\ &- \xi_n \leq 0, \quad n = 1, ..., N \end{aligned}$$

Gornji zapis se lako svodi na matrični zapis:

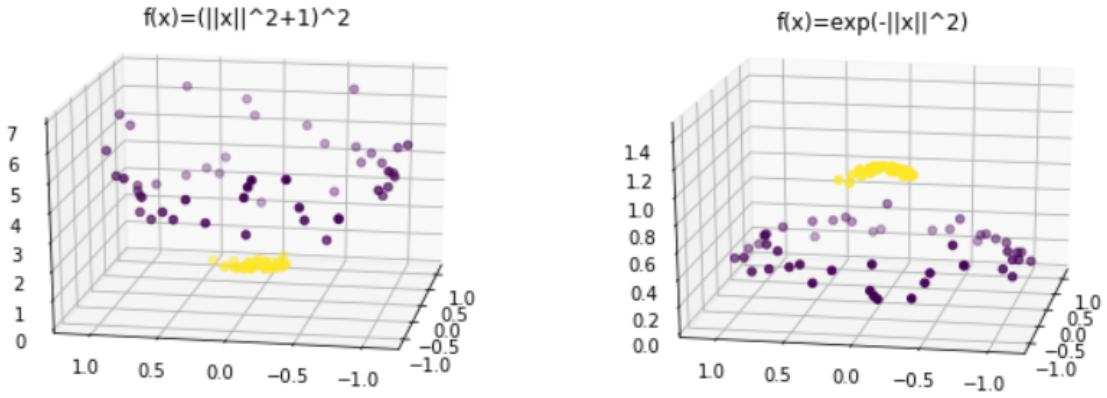


Slika 3.4: Prikaz linearno nerazdvojivog skupa na koji primijenjujemo funkciju za sigmoidalnu jezgru $f(x) = \tanh((\gamma\langle x, x\rangle + r)$ za $\gamma = 1$ i $r = 0$. Podaci iz $\mathbb{R}^2$ (lijeva slika) su pomoću sigmoidalne jezgre preslikani u prostor $\mathbb{R}^3$ (desna slika), gdje ih je moguće razdvojiti hiperravninom.

$$\min \frac{1}{2}\begin{bmatrix}\omega\\ b\\ \xi\end{bmatrix}^T \begin{bmatrix} I_D & 0_{D,N+1} \\ 0_{N+1,D} & 0_{N+1,N+1}\end{bmatrix}\begin{bmatrix}\omega\\ b\\ \xi\end{bmatrix} + \begin{bmatrix}0_{D+1,1} & C\cdot \mathbf{1}_{N,1}\end{bmatrix}^T \begin{bmatrix}\omega\\ b\\ \xi\end{bmatrix}$$

$$\text{uz uvjete : } \begin{bmatrix}-\mathbf{YX} & -y & -I_N \\ 0_{N,D+1} & & -I_N\end{bmatrix}\begin{bmatrix}\omega\\ b\\ \xi^T\end{bmatrix} \leq \begin{bmatrix}-\mathbf{1}_{N,1}\\ 0_{N,1}\end{bmatrix}$$

Gornji problem optimizacije se odnosi na parametre $\begin{bmatrix}\omega^T & b & \xi\end{bmatrix}^T \in \mathbb{R}^{D+1+N}$, a I_m označava jediničnu matricu veličine $m \times m$, $0_{m,n}$ označava matricu nula veličine $m \times n$, $1_{m,n}$ označava matricu jedinica veličine $m \times n$, oznake skupa za učenje se nalaze u dijagonalnoj matrici $\mathbf{Y} = diag(y)$ za $y = \begin{bmatrix}y_1, ..., y_N\end{bmatrix}^T$ veličine $n \times n$, a ulazni podaci skupa za učenje se nalaze u matrici $\mathbf{X} \in \mathbb{R}^{N\times D}$, čiji i-ti redak predstavlja i-ti pirmjerak skupa za učenje.



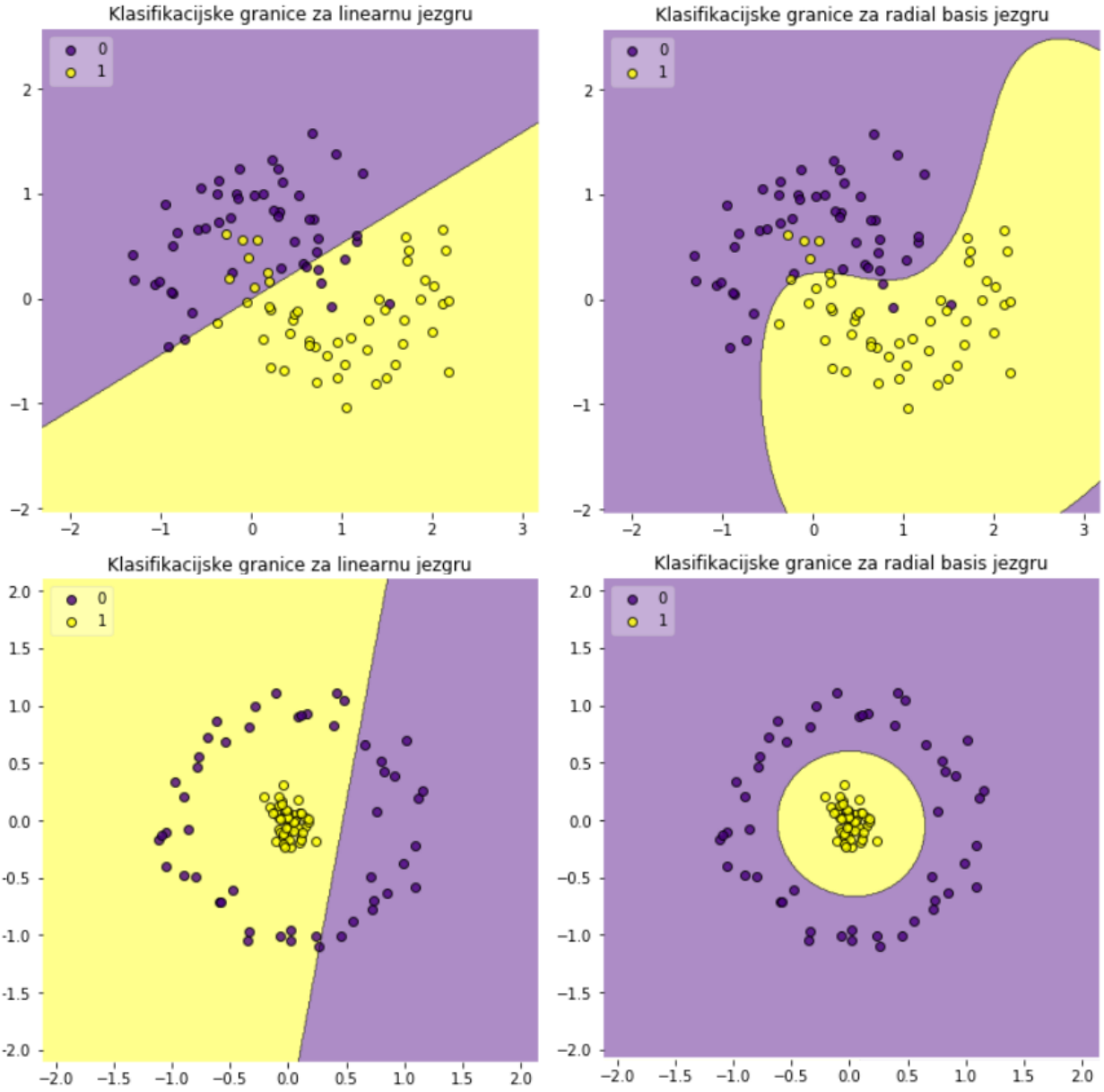
Slika 3.5: Prikaz funkcije za polinomijalnu jezgru $f(x) = (\langle x, x\rangle + 1)^2$ (lijeva slika) i funkcije za radijalnu jezgru $f(x) = \exp(-\|x\|^2)$ (desna slika). U oba slučaja vidimo da su podaci iz $\mathbb{R}^2$ (slika (3.4) lijevo) preslikani u prostor $\mathbb{R}^3$, gdje ih je moguće razdvojiti hiperravninom.

Dualni problem metode potpornih vektora je zapravo problem konveksnog kvadratičnog programiranja (optimizacija uz ograničenja). Neka matrica $\mathbf{K} \in \mathbb{R}^{N\times N}$ predstavlja jezgrenu funkciju primijenjenu na ulazne podatke skupa za učenje $\mathbf{X}$, tj. $\mathbf{K}_{i,j} = k(x_i, x_j)$. Dualni problem (3.19) - (3.21) možemo zapisati u obliku:

$$\min \frac{1}{2}\alpha^T \mathbf{YKY}\alpha - \mathbf{1}^{\mathbf{T}}_{\mathbf{N,1}}\alpha \tag{3.22}$$

$$\text{uz uvjete : } \begin{bmatrix} y^T \\ -y^T \\ -I_N \\ I_N \end{bmatrix} \alpha \leq \begin{bmatrix} 0_{N+2,1} \\ C \cdot \mathbf{1}_{N,1} \end{bmatrix} \tag{3.23}$$

Ovako definirani problem možemo rješavati direktim korištenjem već ranije spomenutih KKT uvjeta koji definiraju sistem jednadžbi ili pomoću neke od iterativnih aproksimacijskih metoda ([4]). Metoda potpornih vektora u ovom radu je implementirana pomoću SMO algoritma ([6]).



Slika 3.6: Na slici su prikazane granice razdvajanja za dva linearno nerazdvojiva skupa podataka. Slika gore desno i dolje desno prikazuje kako je radijalna jezgra u oba slučaja napravila prikladnije granice razdvajanja, dok su granice razdvajanja za linearnu jezgru (slika gore lijevo i dole lijevo) za oba linearno nerazdvojiva skupa podataka neupotrebljive.

Poglavlje 4

Implementacija

Metoda opisana u ranijim poglavljima je primijenjena na problem predviđanja odlaska klijenata iz banke pomoću LIBSVM biblioteke ([3]) koja koristi SMO algoritam. U ovom poglavlju prvo objašnjavamo glavne ideje SMO algoritma, te zatim izlažemo potpunu implementaciju za odabrani problem u bankarstvu. Najprije skup podataka pripremamo kako bi bio prikladan za sklearn.svm.SVC modul (baziran na LIBSVM) u Python scikit-learn biblioteci [2], zatim tražimo optimalne hiperparametre metode i prikazujemo dobivene rezultate.

4.1 SMO algoritam

Algoritam SMO (Sequential minimal optimization) koji je osmislio John Platt [5] služi za iterativno rješavanje problema kvardatičnog programiranja.

Promatramo binarnu klasifikaciju skupa za učenje $\{(x_1, y_1), ..., (x_N, y_N)\}$ sa oznakama $y_i \in \{-1, 1\}$ u obliku dualnog problema:

$$\max_{\alpha} \sum_{i=1}^{N} \alpha_n + \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j K(x_i, x_j) \alpha_i \alpha_j \tag{4.1}$$

$$\text{uz uvjete} : 0 \leq \alpha_i \leq C, \quad i = 1, ..., N \tag{4.2}$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0 \tag{4.3}$$

Jezgrena funkcija je kao i ranije dana sa $K(\cdot, \cdot)$, a C je hiperparametar hiperravnine razdvajanja.

Podsjetimo se da KKT uvjeti tvrde da su u točki postiže maksimum problema (4.1) ako i samo ako za nju vrijede KKT uvjeti:

$$\begin{aligned} \alpha_i = 0 &\Rightarrow y_i f(x_i) \geq 1 \quad i = 1, ..., N \\ 0 \leq \alpha_i \leq C &\Rightarrow y_i f(x_i) = 1 \quad i = 1, ..., N \\ \alpha_i = C &\Rightarrow y_i f(x_i) \leq 1 \quad i = 1, ..., N \end{aligned}$$

Također možemo primijetiti da je KKT uvjete moguće provjeravati za svaki primjerak skupa za učenje zasebno.

SMO razdvaja definirani optimizacijski problem na najmanje moguće potprobleme koji su zatim riješeni analitički. Zbog uvjeta jednakosti najmanji mogući podproblem uključuje dva Lagangeova multiplikatora. U svakoj iteraciji SMO bira dva Lagrangeova multiplikatora te ih optimizira (rješava problem definiran sa sa (4.1)). Problem kvadratične optimizacije koji je definiran za samo dva Lagrangeova multiplikatora se može riješitit analitički, te je na taj način izbjegnuto numeričko rješavanje problema kvadratičnog programiranja, što pridonosi i brzini algoritma.

SMO se sastoji od tri glavne komponente:

- analitička metoda za rješavanje problema (4.1) za $N = 2$ (optimizacija)
- heuristika za odabira multiplikatora koji će biti optimizirani
- izračun parametra b

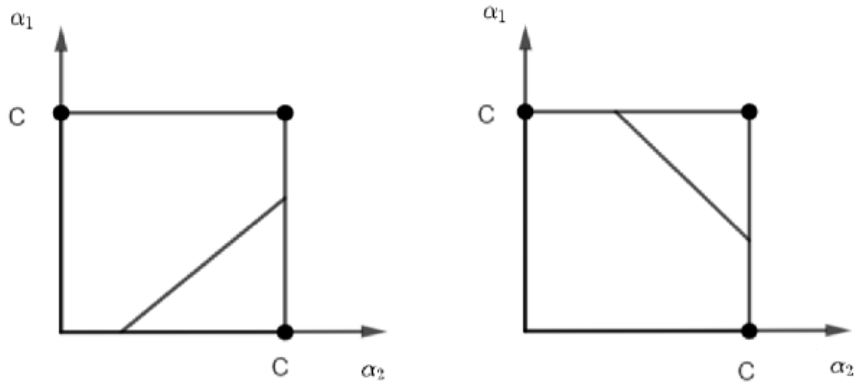
Optimizacija dva Lagrangeova multiplikatora

Zbog jednostavnosti sve vrijednosti vezane uz prvi multiplikator označavamo indeksom 1, a sve veličine vezane uz drugi multiplikator indeksom 2. Ograničenje da $0 \leq \alpha_1, \alpha_2 \leq C$ nam omogućava da lako vizualiziramo prostor nad kojim se optimizira (jednak je kvadratu stranice C), a uvjet $\alpha_1 y_1 + \alpha_2 y_2 = 0$ nam govori da Lagrangeovi multiplikatori leže na dijagonalnoj liniji unutar tog kvadrata. Cijela situacija je vizualizirana na slici (4.1). Bez smanjena općenitosti pretpostavljamo da algoritam prvo računa α_2. Vrijednosti u lijevom i desnom kraju dijagonalne linije u terminima α_2 u slučaju kada je $y_1 \neq y_2$ su dani sa :

$$L = \max(0, \alpha_2^{stari} - \alpha_1^{stari}), \qquad D = \min(C, C + \alpha_2^{stari} - \alpha_1^{stari})$$

U slučaju kada je $y_1 = y_2$ imamo:

$$L = \max(0, \alpha_2^{stari} + \alpha_1^{stari} - C), \qquad D = \min(C, C + \alpha_2^{stari} + \alpha_1^{stari})$$



Slika 4.1: Na slici je prikazan prostor u kojem se nalaze Lagrangeovi multiplikatori (kvadrat stranice C) i dijagonala na kojoj leže multiplikatori. Lijevi prikaz prikazuje slučaj kada je $y_1 \neq y_2$, tada je dijagonalna linija dana sa $\gamma = \alpha_1 + \alpha_2$. Na desnom prikazu je slučaj kada je $y_1 = y_2$ te je dijagonalna linija dana sa $\gamma = \alpha_1 + \alpha_2$.

Druga derivacija funkcije koja se optimizira na dijagonalnoj liniji je dano sa:

$$\nu = 2k(x_1, x_2) - k(x_1, x_1) - k(x_2, x_2)$$

U normalnim okolnostima (za uobičajene izbore jezgrenih funkcija) ν postiže negativne vrijednosti i SMO računa maksimum u smjeru:

$$\alpha_2^{novi} = \alpha_2^{stari} - \frac{y_2(E_1 - E_2)}{\nu}$$

gdje sa E_i označavamo grešku primjerka za učenje, tj. $E_i = f^{stari}(x_i) - y_i$ za $i = 1, 2$. Podsjetimo kako sa $f(x)$ označavamo ishod metode za vektor x.
Potrebno je osigurati da novi Lagrangeovi multiplikator bude dobro definiran, stoga uvodimo $\alpha_2^{novi,korekcija}$:

$$\alpha_2^{novi,korekcija} = \begin{cases} D, & \alpha_2^{novi} \geq D \\ \alpha_2^{novi} & L \leq \alpha_2^{novi} \geq D \\ L, & \alpha_2^{novi} \leq L \end{cases}$$

Sada dobivamo novi α_1^{novi} kao:

$$\alpha_1^{novi} = \alpha_1^{stari} - y_1 y_2(\alpha_2^{stari} - \alpha_2^{novi,korekcija})$$

Heuristika za odabir multiplikatora koji će biti optimizirani

SMO optimizira dva Lagrangeova multiplikatora u svakom koraku iteracije. Stoga funkcija koja se optimizira raste u svakom koraku iteracije, a u isto vrijeme Lagrangeovi multiplikatori ostaju unutar dopuštenog područja. Kako bi se ubrzala brzina konvergencije metode koristimo heuristički odabir za indekse Lagrangeovih multiplikatora koje je potrebno optimizirati. U algoritmu koristimo dvije heuristike, jednu za odabir ineksa prvog multiplikatora koji se optimizira, a drugu za odabir indeksa drugog multiplikatora kojeg je potrebno optimizirati.

Prva heuristika iterira po primjercima skupa za treniranje i bira primjerak koji narušava KKT uvjete. Nakon što je pronađen prvi multiplikator, drugi primjerak za optimizaciju tražimo pomoću druge heuristike. Cilj druge heuristike je izabrati drugi multiplikator na način da se postigne što veći korak prilikom optimizacije multiplikatora. Veličina koraka u optimizaciji je aproksimirana sa $|E_1 - E_2|$, tj. ukoliko je E_1 pozitivan heuristika bira primjerak s minimalnim E_2, a ukoliko je E_1 negativan bira primjerak sa maksimalnim E_2. Ukoliko su prvi i drugi primjerak iz skupa za učenje jednaki stagniramo u optimizaciji. Kako bi se to izbjeglo druga heuristika pazi da je drugi primjerak izabran na način da se postiže pozitivan napredak u optimizaciji.

Izračun parametra b

Rješavanjem problema (4.1)ne dobivamo vrijednost parametra b te on mora biti izračunat posebno.

Nakon svakog koraka, b se ponovo računa i pri tome se pazi da su zadovoljeni KKT uvjeti za oba multiplikatora. Kada novi α_1 nije unutar granica b je dan sa:

$$b_1 = E_1 + y_1(\alpha_1^{novi} - \alpha_1^{stari})k(x_1, x_1) + y_2(\alpha_2^{novi,korekcija} - \alpha_2^{stari})k(x_1, x_2) + b^{stari}$$

Ukoliko je novi α_2 nije unutar granica parametar b je dan sa:

$$b_2 = E_2 + y_1(\alpha_1^{novi} - \alpha_1^{stari})k(x_1, x_2) + y_2(\alpha_2^{novi,korekcija} - \alpha_2^{stari})k(x_2, x_2) + b^{stari}$$

Ukoliko su vrijednosti b_1 i b_2 valjane tada su one i jednake. No, ako su oba nova Lagrangeova multiplikatora na granicama i ako je L jednako D, vrijednosti u intervalu između b_1 i b_2 su sve konzistentne sa KKT uvjetima. U tom slučaju SMO izabere aritmetičku sredinu b_1 i b_2 kao vrijednost parametra b.

4.2 Opis implementiranog problema i podaci

Metodu potpornih vektora u ovom radu primjenjujemo za predviđanje odlaska klijenta iz banke. Skup podataka je preuzet sa [1] i sadrži podatke za 10 000 klijenata, od kojih je 2 037 napustilo banku.

U prvobitnom skupu podataka svaki je klijent opisan pomoću 14 atributa, neki od njih su nepotrebni (jedinstveni identifikator klijenta) te su iz tog razloga ti atributi odmah izbačeni iz skupa podataka. Također atribute koji nisu numerički (npr. spol klijenta) najprije je bilo potrebno kodirati u numeričke, a za atribute koji nisu kontinuirani (npr. zemlja u kojoj se klijent nalazi) je bilo potrebno dodati nove atribute u kojima je numeričkim vrijednostima 0 i 1 kodirana informacija koju od mogućih vrijednosti nekontinuirane varijable klijent ima (npr. klijent ima vrijednost 1 za varijablu koja se odnosi na zemlju u kojoj stanuje, a za sve ostale varijable koje opisuju geografski položaj ima vrijednost 0).

Nakon što je na opisan način definirat set atributa (10 atributa) koji će se koristiti za daljnju izgradnju metode, prema preporukama autora metode LIBSVM svaka varijabla (atribut) je skalirana ((varijabla - aritmetička sredina) / standardna devijacija) tako da su joj vrijednosti sadržane u segmentu $[-1, 1]$.

4.3 Optimizacija hiperparametara metode

Nakon što skup podataka pripremljen kao što je opisano ranije, skup dijelimo u omjeru 0.33:0.67 na skup za testiranje i skup za učenje.

U ovom primjeru je vrlo važno identificirati klijente koji bi mogli napustiti banku (pozitivna klasa), te kako je omjer pozitivne i negativne klase u danom skupu podataka loše balansiran (20.37% klijenata je napustilo banku), za mjeru uspješnosti modela koristimo osjetljivost (omjer točno klasificiranih klijenata koji su napustili banku i svih klijenata koji su napustili banku).

Optimizaciju hiperparametara metode radimo pomoću *grid search* metode (dio biblioteke *scikit-learn*) za različite jezgrene funkcije. Kako skup nije linearno razdvojiv, linearnu jezgru isključujemo iz *grid search* metode. Najprije je za svaku jezgru napravljeno nekoliko probnih pokušaja optimizacije kako bi se odredilo u kojem je smjeru potrebno tražiti parametre, a zatim se metoda ponovo optimizira na profinjenim mrežama.

Prilikom optimizacije svake jezgre bilo je potrebno zadati parametar γ koji skalira skalarni produkt. Za njegov izbor smo najprije isprobali različite vrijednosti, no najboljim izborom su se pokazali recipročna vrijednost dimenzije primjeraka iz skupa $1/D$ (za primjerke $x \in \mathbb{R}^D$) i vrijednost $1/(D \cdot var(X))$, gdje $var(X)$ označava varijancu podataka X.

Također, važno je napomenuti kako *grid search* metoda dopušta paralelno računanje čime se uvelike smanjuje vrijeme potrebno za optimizaciju hiperparametara.

Metoda s radijalnom jezgrom je optimizirana za parametar penalizacije i parametar jezgre. Nakon što smo više puta trenirali metodu na proizvoljnoj mreži parametara, zaključujemo kako se najbolji rezultati postižu za $C \in [700, 900]$ i za $\gamma \in [0.15, 0.05] \cup \{1/D, 1/(D \cdot var(X))\}$. Stoga konstruiramo profinjenu mrežu parametara za te dobivene segmente, te definiramo mrežu parametara za C: $\{700, 750, 775, 800, 810, 825, 850, 900\}$, i mrežu parametara za γ: $\{0.15, 0.12, 0.1, 0.9, 1/(D \cdot var(X)), 1/D, 0.08, 0.7, 0.6, 0.05\}$. Pomoću *grid search* algoritma dobivamo optimalne parametre: $C = 900$ i $\gamma = 1/D$ čija osjetljivost iznosi 0.4964. Na testnom skupu podataka postižemo osjetljivost 0.48.

Za sigmoidalnu jezgru koristimo isti princip te završnu optimizaciju radimo na mreži parametara C: $\{50, 100, 150, 200, 250, 300\}$, γ: $\{1/(D \cdot var(X)), 1/D, 0.5, 0.4, 0.3, 0.2\}$ i r: $\{0, 1, 5, 10, 50\}$. Najbolju osjetljivost (0.2877) na skupu za učenje dobivamo za parametre $C = 150$, $\gamma = 1/D$ i $r = 1$. Na skupu za testiranje metoda definirana tim hiperparametrima postiže osjetljivost od 0.27.

Finalna optimizacija parametara metode za polinomijalnu jezgru se izvodi na mreži za C: $\{1, 10, 50, 100\}$, γ: $\{1, 0.5, 0.3, 0.1\}$, r: $\{0, 1, 1.5\}$ i d: $\{2, 3, 4\}$. Najbolju osjetljivost na skupu za učenje postižu parametri $C = 1$, $\gamma = 1/D$, $r = 1$ i $d = 4$, te ona iznosi 0.5036. S tim parametrima osjetljivost na skupu za testiranje iznosi 0.53.

U tablici prilažemo vrijednosti ostalih relevantnih mjera koje metoda postiže za optimalne parametre svake jezgre:

Rezultati na testnom skupu podataka				
Jezgra	Osjetljivost	Preciznost	F1-score	AUC
Radijalna	0.48	0.48	0.48	0.68
Sigmoid	0.27	0.30	0.28	0.55
Polinomijalna	0.53	0.62	0.57	0.73

Iz priložene tablice vidimo kako polinomijalna jezgra postiže najbolje rezultate.

Kao što smo ranije napomenuli najvažnije metrike su osjetljivost i preciznost. $F1 - score$ računa harmonijsku sredinu preciznosti i osjetljivosti, te je iz tog razloga poželjno da ima vrijednosti što bliže 1. Ukoliko preciznost i osjetljivost poprimaju iste vrijednosti $F1 - score$ je jednak njima, u protivnom $F1 - score$ poprima vrijednost koja se nalazi između njih. Osim navedenih metrika prikazujemo i AUC vrijednost. Slučajni klasifikator postiže AUC vrijednost 0.5, stoga nam je bitno da se vrijednosti te mjere nalaze unutar segmenta [0.5, 1].

Osjetljivost prestavlja omjer točno klasificiranih klijenata koji odlaze iz banke i svih klijenata koji uistinu odlaze iz banke. Kako želimo što veći broj točno klasificiranih klijenata koji odlaze iz banke poželjno je da vrijednost koju postiže osjetljivost bude što bliža 1. Preciznost u kontekstu našeg primjera prikazuje omjer točno klasificiranih klijenata koji odlaze iz banke i svih klijenata koji su klasificirani kao da odlaze iz banke. Za preciznost je također poželjno da ima vrijednost što bliže 1, kako banka ne bi nepotrebno reagirala prema klijentima koji u stvarnosti neće napustiti banku.

Bibliografija

[1] *Bank churn prediciton dataset from Kaggle*, rujan 2020, `https://www.kaggle.com/adammaus/predicting-churn-for-bank-customers`.

[2] *scikit-learn 0.23.2*, rujan 2020, `https://scikit-learn.org/stable/`.

[3] Chih Chung Chang i Chih Jen Lin, *LIBSVM: A library for support vector machines*, ACM Transactions on Intelligent Systems and Technology **2** (2011), 27:1–27:27, Software dostupan na `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[4] Stephen J. Wright Jorge Nocedal, *Numerical Optimization*, Springer, 2006.

[5] John C. Platt, *Fast Training of Support Vector Classifiers, in Advances in Kernel Methods – Support Vector Learning, B. Scholkopf, C. Burges, A. Smola, eds.,s*, MIT Press, 1998.

[6] Chih Jen Lin Rong-En Fan, Pai-Hsuen Chen, *Working Set Selection Using Second Order Information for Training Support Vector Machines*, Journal of Machine Learning Research 6 1889–1918 (2005).

[7] Vladimir N. Vapnik, *Statistical Learning Theory*, John Wiley and Sons, 1989.

Sažetak

Strojno učenje obuhvaća metode koje se koriste za rješavanje različitih problema klasteriranja, klasifikacije, regresije. Osnovi koraci i ideje strojnog učenja za nadzirane metode nalaze se u prvom poglavlju. U drugom poglavlju prikazujemo tri najvažnija teorema optimizacije (Fermatov teorem, Lagrangeov teorem, Karush–Kuhn–Tuckerov teorem) koji imaju značajnu ulogu u pronalasku rješenja metode potpornih vektora.

U trećem poglavlju polako razvijamo ideju na kojoj je bazirana metoda potpornih vektora, i objašnjavamo dvije velike prednosti dualne formulacije problema. Prva prednost je to što je metoda u dualnoj formulaciji problem kvadratičnog programiranja čije rješenje postoji i poznato je (KKT teorem). Druga prednost je mogućnost korištenja jezgrenih funkcija, koje omogućavaju primjenu metode na skup podataka reprezentiran u prostoru koji je jednostavnije razdvojiti hiperravninom.

Na kraju rada prikazujemo implementaciju metode baziranu na Sequentail Minimal Optimization algoritmu. Za razliku od drugih poznatih algoritama za rješavanje problema metode potpornih vektora, SMO odlazi u krajnost i rješava optimizacijske potprobleme za samo dva Lagrangeova multiplikatora. Upravo zbog toga se SMO pokazao kao jedan od najefikasnijih algoritama pri rješavanju problema metode potpornih vektora.

U završnom poglavlju iznosimo rezultate metode primijenjene na binarni klasifikacijski problem u bankarstvu. Cilj ovog primjera je izgraditi metodu koja se može koristiti za identifikaciju klijenata koji odlaze iz banke. Nakon što je prvotni skup podataka pripremljen za metodu i osjetljivost izabrana kao mjera uspješnosti metode, krećemo trenirati metodu. Najprije dijelimo skup podataka na skup za testiranje i skup za treniranje, te pomoću unakrsne validacije optimiziramo hiperparametre metode na skupu podataka za treniranje, a zatim testiramo moć generalizacije metode za dobivene optimalne parametre na skupu podataka za testiranje. Najbolje rezultate metoda postiže kada se koristi polinomijalna jezgra.

Summary

Machine learning involves all the methods used to find a solution to the problems of clustering, classification, regression. In Chapter 1 the basic steps and ideas of machine learning for supervised methods are presented. The goal of the Chapter 2 is to build a theory foundation for the Support Vector Machine (SVM) on the three optimizations theorems (Fermat theorm, Lagrange theorem, Karush–Kuhn–Tucker theorem) that play a major role in finding solution to the SVM.

In the Chapter 3, the main concepts on which SVM is based are presented and we remark how to exploit the two great adventages obtained from the dual formulation. The first advantage is that the method in dual formulation is a quadratic programming problem whose solution exists and is known (KKT theorem). The second advantage is the possibility of using kernels, which allow the application of the method to a dataset represented by the features on which is possible to build a separating hyperplane.

In the final chapter of this thesis we apply the method which is based on the Sequential Minimal Optimization algorithm. Unlike the other known algorithms for solving problems of SVM, SMO goes to extremes and solves optimization subproblems for only two Lagrange multipliers. This is why SMO has proven to be one of the most efficient algorithms in solving the problem of support vector methods.

At the end of the thesis we present the results of the described method applied to the binary classification problem in banking. The goal of this example is to build a method that can be used to identify clients leaving the bank. Once the initial data set has been prepared for the method and recall is selected as a validation measure of the method, we start the training process. Using cross-validation we optimize the hyperparameters of the method on the train data set, and finally we test the generalization power of the method for the optimal parameters on the test data set. The best results are achieved for the method based on the polynomial kernel.

Životopis

Rođena sam 9. veljače 1996. godine u Zgrebu. Nakon završene osnovne škole upisala sam XV. gimnaziju u Zagrebu, smjer informatika. Svoje daljenje obrazovanje nastavila sam u Zagrebu gdje sam 2014. godine upisala prediplomski studij Matematika na Prirodoslovno-matematičkom fakultetu Sveučilišta u Zagrebu. Završetkom prediplomskog studija postala sam sveučilišni prvostupnik matematike, te nastavila obrazovanje na diplomskom studiju Primijenjene matematike. U prvom semestru druge godine diplomskog studija sudjelovala sam na Erasmus studentskoj razmjerni te pohađala predavanja na sveučlištu u Ghentu. Na zadnjoj godini diplomskog studija počinjem raditi u banci kao dio tima za upravljanje odnosima s klijentima. Diplomski studij završavam u rujnu 2020. godine ovim radom.