Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

David Ocepek

Bayesovska optimizacija hiperparametrov v strojnem učenju

diplomsko delo

univerzitetni študijski program
prve stopnje
računalništvo in informatika

Mentor: izr. prof. dr. Matjaž Kukar

Ljubljana, 2020

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:
Bayesovska optimizacija hiperparametrov v strojnem učenju

Tematika naloge:

Osrednji cilj diplomske naloge je predstaviti in pregledati orodja za optimizacijo hiperparametrov s poudarkom na metodi Bayesovske optimizacije. Opisati razloge za njeno uporabo, jo razdeliti na osnovne sestavne dele in lete predstaviti. Na koncu pa preizskusiti metodo na nekaj praktičnih primerih ter predstaviti in interpretirati dobljene rezultate.

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
BO	bayesian optimization	BO
RS	random search	naključno iskanje
EI	expected improvement	pričakovana izboljšava
PI	probability of improvement	verjetnost izboljšave
LCB	lower confidence bounds	spodnja meja zaupanja
ES	entropy search	entropijsko iskanje
PES	predictive entropy search	napovedno entropijsko iskanje
GP	gaussian process	gausov proces
CNN	convolutional neural network	konvolucijska nevronska mreža
CA	classification accuracy	klasifikacijska točnost
AUC	area under the curve	površina pod krivuljo
TS	thompson vzorčenje	thomson sampling
ML	machine learning	strojno učenje
autoML	automated machine learning	avtomatizirano strojno učenje
SVC	support vector classifier	klasifikator podpornih vektorjev
GS	grid search	iskanje v mreži

Povzetek

Naslov: Bayesovska optimizacija hiperparametrov v strojnem učenju

Avtor: David Ocepek

Opišemo vlogo metode BO na področju ML. Spošno opišemo metodo BO, njene prednosti in slabosti. BO razčlenimo po glavnih komponentah na verjetnostni model in odločilno funkcijo. Vsako od glavnih komponent predstavimo. Pri verjetnostnih modelih predstavimo GP ter analiziramo njene latnosti. Pri odločilnih funkcijah predstavimo 3 uporabljene funkcije: EI, PI in LCB ter nekaj novejših funkcij. Predstavimo izvedene experimente na različnih datasetih in modelih (CNN, SVM, GradientBoosting, RandomForests). Pri experimentih analiziramo hitrost konvergence po iteracijah ter doseženo stopnjo optimizacije hiperparametrov. Rezultate BO primerjamo z metodama RS in GS, pogosto uporabljanima metodama za optimizacijo hiperparametrov. BO v vseh experimentih in metrikah doseže enakovredne ali boljše rezultate od RS in GS.

Ključne besede: Bayesovka optimizacija, nastavljanje hiperparametrov, avtomatizirano strojno učenje.

Abstract

Title: Bayesian optimization of hyperparameters in machine learning

Author: David Ocepek

We describe the role of the BO method in ML. We give a basic description and analyze BO its pros and cons. We split BO into its main compenents: the probability model and acquisition function. We introduce the probability model of GP and analyze its properties. We introduce 3 used acquisition functions: EI, PI, LCB and some newer functions. We perform experiments on different datasets and models (CNN, SVM, GradientBoosting, RandomForests). For each experiment we analyze the speed of convergence and optimization results. We compare the results of BO to RS and GS, two commonly used methods for hyperparameter optimization. In all experiments and metrics BO performs equal or better than RS and GS.

Keywords: Bayesian optimization, hyperparameter tunning, automated machine learning.

Poglavje 1

Uvod

ML je bilo v zadnjih desetletjih priča številnim napredkom. Te napredki so s sabo prinesli bolj natančne in hkrati bolj kompleksne modele. Te modeli imajo množico nastavljivih parametrov, imenovanih hiperparametri. Optimalne vrednosti teh hiperparametrov se razlikujejo od problema do problema, posledično je problem ročnega nastavljanja hiperparametrov časovno zahteven problem, ki zahteva veliko strokovnega znanja.

Zgodnje metode, ki so se soočale s tem problemom so bile tako imenovane "model-free" metode, ki niso uporabljale predhodno pridobljenega znanja. Primera teh metod sta GS, ki iz diskretnega prostora kombinacij množic parametrov izbere kombinacijo, pri kateri model doseže maksimum glede na metrike uspešnosti. Metoda ima polinomično časovno zahtevnost $\mathcal{O}(n^c)$, glede na število dimenzij n in moč množice c. Boljša je metoda RS, ki naključno izbira točke v omejenem prostoru ter tako omogoča večjo pokritost posameznih dimenzij. Metode RS in GS se uporabljata samo za podajanje kontexta za analizo in ovrednotenje rezultatov.

Razlog za to je velika časovna zahtevnost potrebna za učenje samotnih modelov, kar pomeni da je število točk, ki jih lahko te funkcije ovrednotijo in preizkusijo pogosto omejeno na nekaj deset ali stotero. Zato se pogosteje uporabljajo bolj efektivne metode na podlagi modelov, ki pri izbiri točk za evaluacijo uporabljajo predhodno pridobljeno znanjo iz predhodno evalui-

ranih točk. State of the art na tem področju je metoda BO, ki po vsaki evaluaciji točke posodablja model hiperravnine in na podlagi tega izbere naslednjo točko. Metoda hitreje konvergira kot zgoraj navedene metode in doseže boljše optimume kot RS in GS. V zadnjih par letih je BO podrla veliko benchmarkov, saj se je izkazalo, da razen tega, da strojno nastavljanje odstrani potrebo ročnem nastavljanju hiperparametrov ima še druge prednosti kot sta pridobivanje boljših rezultatov in boljša reproduktivnost raziskav in rezultatov.

Delo je sestavljeno iz n poglavij. V 2. poglavju predstavimo BO, zapišemo in obrazložimo osnovni algoritem BO. V 3. poglavju predstavimo sestavne dele BO optimizacije, in sicer model hiperravnine hiperparametrov in odločilno funkcijo, ki izbira naslednjo točko za evaluacijo. V 4. poglavju predstavimo predstavimo posamične eksperimente, modele in podatke, ki jih uporabljamo za posamezne eksperiment ter opišemo razlog in namen za izvedbo eksperimenta. V 5. poglavju predstavimo in analiziramo pridobljene rešitve ter ovrednotimo ali so bili eksperimenti uspešni. V zadnjem poglavju je predstavljen zaključek in sklep dela.

Poglavje 2

Bayesovska optimizacija

2.1 Osnovni pregled

V tem poglavju na grobo predstavimo Bayesovsko optimizacijo. Opišemo temeljne korake algoritma in obrazložimo kako se v algoritmu uporablja bayesovska statistika. Predstavimo prednosti in slabosti grobega algoritma in morebitne izboljšave. Na koncu pa podamo praktične primere uporabe.

Bayesovska optimizacija je iterativen algoritem za iskanje globalnega optimuma poljubne funkcije:

$$x^* = \underset{x \epsilon X}{\operatorname{argmin}} f(x) \tag{2.1}$$

Prednost Bayesovke optimizacija pred RS ali GS je, da je zelo podatkovno učinkovita, saj je bolj fokusirana na izboljšavo funkcije kot na samo preiskovanje prostora, kar je v praktičnih primerih zelo pomembno. Razlog za to je, da je optimizirana funkcija sestavljena iz učenja in evaluiranja modela, zato lahko ena evaluacija tudi enostavnejšega modela traja več GPU ur. V pogostih primerih uporabe cloud computinga pa so potrate tudi denarne in ne samo časovne.

Algoritem je sestavljen je iz dveh ključnih delov: verjetnostnega modela, ki na podlagi že znanih točk modelira verjetnostno hiperravnino, ki je sestavljena iz povprečja in standardnega odklona in odločilne funkcije, ki na

podlagi verjetnostnega modela izbere naslednjo točko za evaluacijo. Apriorni verjetnostni model je v vsaki iteraciji posodobljen s podatki pridobljenimi v prejšnji iteraciji z uporabo bayesovske statistike. Na podlagi pridobljenega posteriornega verjetnostnega modela potem odločilna funkcija izbere naslednjo točko. Spodaj je podan osnovni algoritem za BO:

Algorithm 1 Bayesian optimization

1: `Inicializiraj X s naključnimi točkami in evaluiraj` $y = f(X)$`, natstavi` $D_0 = (X, y)$`.`
2: **for** `n = 1,2,...` **do**
3: `posodobi statističen model`
4: $x_{n+1} = \text{argmax}_x \, \alpha(x; D_n)$ ▷ `izberi` x_{n+1} `v optimumu odločilne funkcije` α
5: $y_{n+1} = f(x_{n+1})$ ▷ `izračunaj vrednost funkcije v izbrani točki` x_{n+1}
6: $D_{n+1} = \{D_n, (x_{n+1}, y_{n+1})\}$ ▷ `posodobi tabelo znanih vrednosti podmnožice točk` X
7: **end for**

Da bi bila Bayesovska optimizacija res uporabna pa mora veljati, da sta evaluacija verjetnostnega modela in odločilne funkcije v primerjavi s samo funkcijo f vsaj par stopenj magnitude cenejša. Na žalost pa je v praksi večina verjetnostnih modelov in odločilnih funkcij komputacijsko nerešljivih. V praksi so uporabljani sestavni deli analitično rešljivi.

Kot je bilo že zgoraj omenjeno je Bayesovska optimizacija bayesovska, zato ker uporablja bayesov teorem za posodabljanje verjetnosti v verjetnostnem modelu.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{2.2}$$

Verjetnost $P(B)$ je v pogosto računsko nerešljiva na srečo ima samo vlogo normalizacije, zato ga pri računanju ne potrebujemo upoštevati in lahko predstavimo posteriorno verjetnost kot v odvisnosti od $P(B)$.

$$P(A|B) = P(B|A)P(A) \tag{2.3}$$

To lahko naredimo, zato ker nas točna verjetnost ne zanima, zanima nas samo optimizacija te verjetnosti. V našem primeru je $A = f$ in $B = D, x$. Torej enačba izgleda takole:

$$P(f|D, x) = P(D, x|f)P(D, x) \tag{2.4}$$

Posteriorna verjetnost $P(f|D, x)$ predstavlja posodobljeno verjetnostni model funkcije f za vsako točko x glede na množico že znanih točk D, medtem ko apriorna verjetnost $P(D, x|f)$ verjetnost, je verjetnost obzervacije verjetnosti f v točki x. Apriorna verjetnost podaja verjetnostni model.

Za konec bi dodali še eno prednost bayesovske optimizacije in sicer njeno modularnost. Zaradi praktičnih razlogov je sicer izbira verjetnostnega modela in odločilnih funkcij omejena, vendar kljub temu obstajata vsaj dva v praksi uporabljana za verjetnostna modela ter veliko odločilnih funkcij, kar omogoča kombiniranje le-teh in izbiro najbolj primerne kombinacije za obravnavan problem. Za verjetnostni model se uporabljajo Gausovi procesi, ki so podobni multivariantni gausovi distribuciji in random forest. Izbira odločilnih funkcij je velika zato smo se v našem primeru omejili na pozneje opisane funkcije EI, PI in LCB.

Pogosto pa je kljub relativno hitri naravi teh komponent Bayesovska optimizacija še vedno prepočasna ali pa obstaja želja po več evaluacijah, zato obstajajo metode, ki pohitrijo algoritem za majhne izgube v natančnosti. Najenostavnejša in tudi najbolj pogosto časovna optimizacija je poganjati optimizirano funkcijo na podmnožici dataseta, treniranje algoritma na manjšem številu iteracij, na zmanjšani množici zančilk ter na eni ali nekaj prerezov pri prečni validaciji.

2.2 Knjižnice

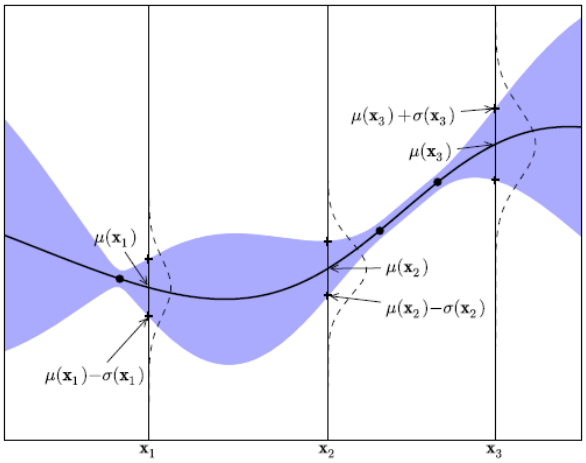
Na območju bayesovske optimizacije obstaja mnogo knjižnic, v katerih je implementirana bayesovska optimizacija. Primeri takih knjižnic so BayesianOptimization, GPyOpt, Spearmint, GPflowOpt, scikit-optimize, sklearn GPR in GPClassifier, hyperotp, optuna, HpBandSter, Botorch in verjetno še druge.

Poglavje 3

Verjetnostni modeli

V tem poglavju razložimo razliko med parametričnimi in neparametričnimi modeli, potem natančneje raziščemo najpogosteje uporabljan verjetnostni model GP, predstavimo razloge za njegovo uporabo, opišemo in razložimo njegovo teoretično podlago in podamo primere za lažje razumevanje.

Parametrični modeli $p(f|X, \theta)$ so modeli, ki imajo fiksno število parametrov. Pogosto se uporabljajo pri strojnem učenju za razlago in modeliranje podatkov, pogosto pa uporabimo regresijo, da dobimo tak θ, ki optimizira $p(f|X, \theta)$. Glede na to da so dataseti čedalje večji in čedalje bolj kompleksni so potrebnimi modeli, ki imajo čedalje več parametrov. Posledično so se začeli uporabljati neparametrični modeli. Ime neparametrični je zavajajoče in ne pomeni, da modeli nimajo parametrov, ampak da jih imajo poljubno mnogo. Eden s takih modelov so Gausovi procesi. Grobo si lahko predstavljamo GP kot multivariantno funkcijo $f(x) = [\mu(x), \delta(x)]$, ki vsaki točki v prostoru hiperparametrov določi povprečje $\mu(x)$ in standardni odklon $\delta(x)$.



Slika 3.1: Enostaven 1D Gausov proces s tremi evaluiranimi točkami. Črna črta predstavlja predpostavljano povprečje optimizirane funkcije glede na podatke in osenčeno območje okoli predstavlje povprečje plus ali minus standardna deviacija. Prekrivajoče Gausove krivulje predstavljajo GP povprečje in standardno deviacijo v točkah $x_{1:3}$

Za bolj natančno razumevanje pa je potrebno razumevanje multivariantne gausove distribucije, saj je prehod iz te na gausove procese zelo preprost.

Najprej predstavimo enostavno univariantno gausovo distribucijo $X \sim N(\mu, \sigma)$, ki se glasi

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(x-\mu)^2} \tag{3.1}$$

Enačba za multivariantno distribucijo k-dimenzionalno $X \sim N(\mu, \Sigma)$ je slednja

$$p(x) = \frac{1}{\sqrt{2\pi\delta^k|\Sigma|}} e^{-\frac{1}{2}(x-\mu)^T\Sigma(x-\mu)} \tag{3.2}$$

kjer je μ k-dimenzionalen vektor povprečnih vrednosti glede na vsako dimenzijo

$$\mu = E[X] = (E[X_1], E[X_2], E[X_3], ..., E[X_4]) \tag{3.3}$$

in Σ kovariančna matrika, kjer je $\Sigma(n, m)$ enako kovarianci med n-to in m-to gausovo ditribucijo.

$$\Sigma_{i,j} = E[(X_i - \mu_i)(X_j - \mu_j)] = Cov[X_i, X_j] \tag{3.4}$$

Podobnost je pogosto v primeru Bayesovske optimizacije podana s posebno funcijo jedra K.

Medtem ko je multivariantna gausova distribucija porazdelitev nad vektorji je gausov proces porazdelitev nad funkcijami.

$$f(x) \sim GP(m(x), k(x, x^{'})) \tag{3.5}$$

kjer je katerokoli končna podmnožica $X = x_1, x_2..., x_n$ v domeni x, porazdeljena multivariantno gausovo distribucijo:

$$f(X) \sim N(m(X), k(X, X)) \tag{3.6}$$

Edina razlika med Gausovim procesom in distribucijo je, da gausov proces ni omejen končno število distribucij.

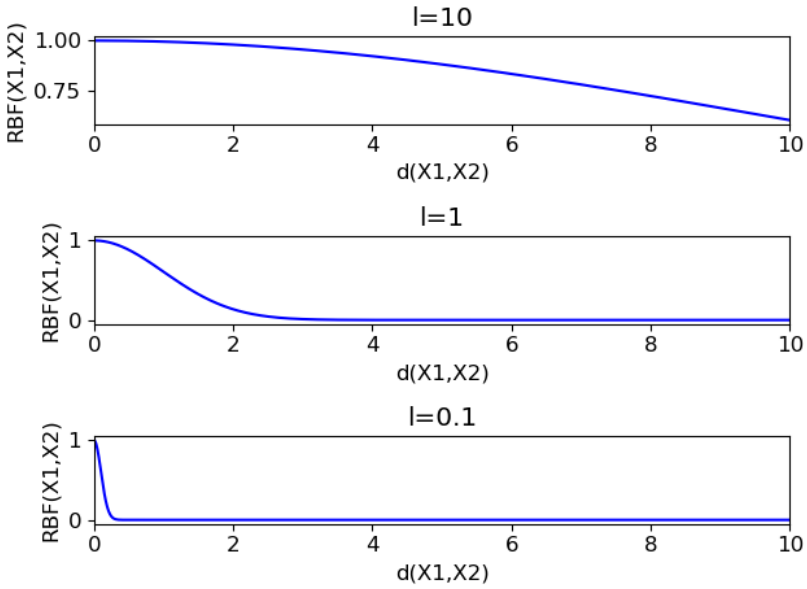
3.1 Jedra

Za gausove procese potrebujemo funkcijo $m(x)$ in kovariančno funkcijo podobnosti točk $k(x_i, x_j)$. Funkcijo $m(x)$ se pogosto nastavi na 0, saj so podatki normalizirani in kljub normalizaciji GP neizgubi na moči izražanja. Za funkcijo $k(x_i, x_j)$ pa mora veljati, da točkam, ki pri evaluaciji so si blizu vrne visoko vrednost medtem ko bolj odaljenim točkam vrne različne vrednosti. Pogosto se uporablja funkcija RBF ali radial basis function, ki ima naslednjo obliko:

$$k(x_i, x_j) = e^{(-\frac{||x_i - x_j||^2}{2\delta^2})} \tag{3.7}$$

δ je prost parameter in omogoča spreminjanje skale vhodov v funkcijo k. V primeru, da je δ vektor je možno za vsak hiperparameter nastaviti

skalo posebej. To je potrebno in uporabno, saj imajo hiperparametri različna definicijska območja, ki jih je potrebno normalizirati za pravilno Gausovih procesov. Dodatna prednost tega jedra je, da je stacionarno, kar pomeni, da je ni odvisno od samih vrednosti x_i in x_j in je odvisno le od velikosti razdalje med njima.



Slika 3.2: Slika prikazuje RBF jedro s parametrom $l = 10$, $l = 1$ in $l = 0.1$. Iz slike je razvidno, kako s večanjem parametra l se večja predpostavljana razdalja na katerih so si točke še podobne podobnost točk, ki so si med sabo bolj oddaljene. Funkcija je omejena samo na pozitivne vrednosti, saj je parameter d vedno večji od 0.

Posplošitev RBF jedra je družina pogosto uporabljanih stacionarnih jeder Matern, ki imajo slednjo obliko:

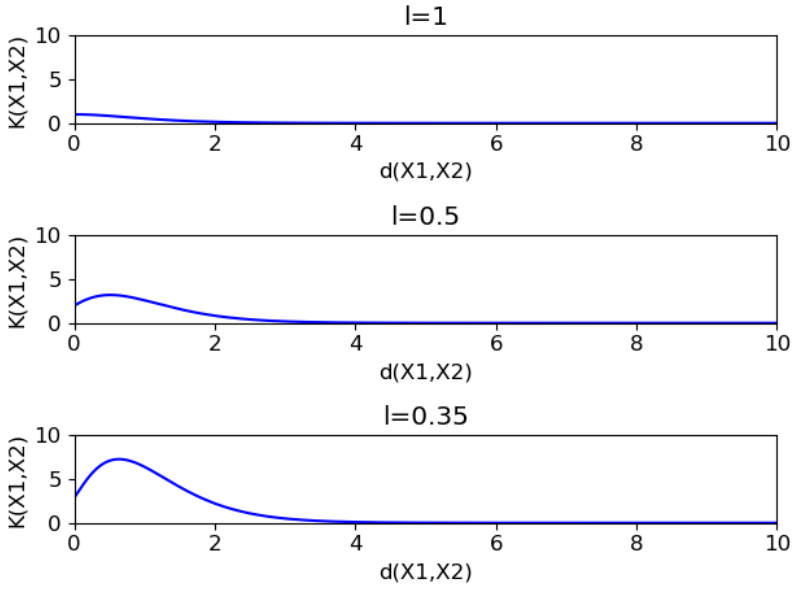
$$C_v(d) = \sigma^2 \frac{2^{1-\nu}}{\Gamma(\nu)} (\sqrt{2\nu}\frac{d}{\rho})^{\nu} K_\nu(\sqrt{2\nu}\frac{d}{\rho}) \tag{3.8}$$

kjer je Γ gamma funkcija, K_v je modificirana Bessel funkcija. ρ in ν sta prosta parametra funkcije.

Velja da lahko Gausov proces, ki uporablja Metern jedro C_v odvajamo $\lceil v \rceil - 1$ krat. V primeru, da je $x = 1$ je Matern jedro ekvivalentno exponentnemu jedru, medtem ko v $\lim_{v\to\infty}$ je jedro ekvivalentno jedru RBF. V praksi se uporabljajo jedra oblike $v = p + 1/2$, kjer je p pozitivno celo število. Za to podmnožico Matern jeder je značilno, da imajo poenostavljeno analitično obliko. Kot primer podamo najpogosteje uporabljano jedro Matern5/2.

$$K_{5/2} = \sigma^2(1 + \frac{\sqrt{5}d}{l} + \frac{5d^2}{3l^2})e^{-\frac{\sqrt{5}d}{l}} \tag{3.9}$$

Matern5/2 je uporabljana v knnjižnici, ki je bila uporabljena za to diplomsko. Spodaj je prikazan graf funkcije Matern5/2:



Slika 3.3: Slika prikazuje RBF jedro s parametrom $l = 1$, $l = 0.5$ in $l = 0.35$. Iz slike je razvidno, da se s večanjem parametra l večja predpostavljana razdalja počasneje kot pri RBF ter da predpostavljana točka s največjo podobnostjo ni točka na razdalji nič, vendar malce oddaljena. Funkcija je omejena samo na pozitivne vrednosti, saj je parameter d vedno večji od 0.

3.2 Napovedovanje vrednosti GP

Zdaj ko imamo jedro predstavimo enačbo verjetnostni funkcije $p(f|X)$:

$$p(f, X) = N(f|\mu, K) \tag{3.10}$$

V zgornji enački je $f = [f(x_1), ..., f(x_N)]^T$, $\mu = [m(x_1), ..., m(x_N)]^T$ in $K_{ij} = \kappa(x_i, x_j)$. Pogosto uporabimo $m(x) = 0$, saj je GP dovolj fleksibilen model, da lahko modelira povprečje poljubno dobro. Posledično je Gausov proces distribucija, katere obliko določa K.

3.2.1 Napovedovanje vrednosti v datasetu brez šuma

Če uporabljamo dataset brez šuma in imamo vrednosti f in X kot vhoda, lahko pretvorimo zgoraj podan apriorno verjetnost $p(f|X)$ v posteriorno verjetnost $p(f_*|D, x)$, ki se lahko potem uporabi za napovedovanje verjetnosti za vsako vrednost v točki x. Po definiciji GP ima skupna porazdelitev že evaluiranih vrednosti f in še neevaluiranih vrednosti f_* prav tako normalno obliko, ki jo lahko razdelimo v

$$\begin{pmatrix} f \\ f_* \end{pmatrix} \sim N\left(0, \begin{pmatrix} K & K_* \\ K_*^T & K_{**} \end{pmatrix}\right) \tag{3.11}$$

kjer je $K_* = \kappa(X, X_*)$ in $K_{**} = \kappa(X_*, X_*)$. Če je število že evaluiranih točk enako n in število točk, katerih vrednosti hočemo napovedati n_*, potem imajo matrike K, K_*, K_{**} dimenzije $n \times n$, $n \times n_*$in $n_* \times n_*$. Z uporabo standardnih pravil za izračun pogojne verjetnostni normalne distribucije dobimo formulo za verjetnost napovedi v točki x:

$$\begin{aligned} p(f_*|D, x) &= N(f|\mu_*, \Sigma_*) \\ \mu_* &= K_*^T K^{-1} f \\ \Sigma_* &= K_{**} - K_*^T K^{-1} K_* \end{aligned} \tag{3.12}$$

Iz tukaj izhaja kubična časovna kompleksnost za napoved točke, saj je čas potreben za izračun K^{-1} $O(n^3)$.

3.2.2 Napovedovanje vrednosti v datasetu s šumom

V praksi pa imamo pogosto opravka s šumom v naših opazovanjih. Ne samo v naših podatkih, ampak tudi v naših napovedih. Ker naj bila Bayesovska optimizacija uporabljena na poljubnih problemih je potrebno te pojave zajeti v napovedni funkciji. Šum je podan s funkcijo $f^{'} = f + \epsilon$, kjer je $\epsilon \sim N(0, \sigma^2 I)$ in je neodvisno dodan vsaki naši evaluaciji, potem naša funcija napovedi izgleda takole:

$$\begin{aligned} p(f_*|X, f_\epsilon, x) &= N(f|\mu_*, \Sigma_*) \\ \mu_* &= K_*^T K_\epsilon^{-1} f_\epsilon \\ \Sigma_* &= K_{**} - K_*^T K_\epsilon^{-1} K_* \end{aligned} \tag{3.13}$$

kjer je $K_\epsilon = K + \sigma^2 I$.

S tem smo zajeli še šum v podatkih, zdaj moramo zajeti še šum v napovedih. V praksi je šum spodnja meja za varianco vsake točke, saj varianca ne more biti manjša kot šum. Ker je varianca definirana kot diagonala kovariančne matrike Σ dodamo šum diagonali in dobimo končno enačbo, ki upošteva tako šum v podatkih kot v napovedih.

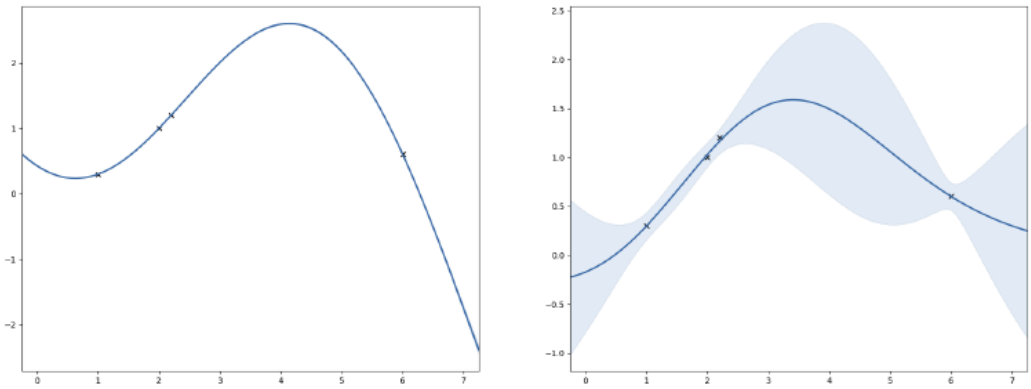
$$p(f_*|X, f_\epsilon, x) = N(f|\mu_*, \Sigma_* + \sigma^2 I) \tag{3.14}$$

3.2.3 Optimizacija hiperparametrov od GP

Pri opisu jeder smo omenili, da imajo jedra svoje lastne hiperparametre, prav tako imamo še parameter šuma σ. Te hiperparametre združimo v vektor hiperaparametrov θ in dobimo posodobljeno vrednost funkcije $f(x) \sim N(0, K(\theta, x, x^{'}))$. Da dobimo optimalne hiperparametre θ maximiziramo po logaritmu mejne verjetnosti:

$$logp(f(x)|\theta, x) = -\frac{1}{2} f(x)^T K(\theta, x, x^{'})^{-}1 f(x^{'}) - \frac{1}{2} logdet K(\theta, x, x^{'}) - \frac{n}{2} log 2\pi \tag{3.15}$$

Za izračun te enačbe je potrebno izračunati inverz in determinanto kovariančne matrike Σ, oba od teh izračunov imata kubično časovno zahtevnost v odvisnosti od števila točk že evaluiranih. Zaradi te slabosti je bilo izdelanih več metod za aproksimacijo teh vrednosti, ki se pogosto uporabljajo. Za optimizacijo te enačbe lahko uporabimo množico numeričnih optimizacijskih iterativnih metod kot sta na primer L-BFGS, Rprop, CG ali SGD. Glede na to da je lahko funkcija hiperparametrov ne-konveksna in ima več lokalnih optimumov, ki so mnogo slabši od glabalnega optimuma se pogosto uporablja tehnika naključnega ponavljanja za povečanje verjetnosti, da bo algoritem konvergiral v globalni optimum.



Slika 3.4: Slika prikazuje Gausov proces s neoptimiziranimi parametri na levi in z optimiziranimi parametri na desni.

Poglavje 4

Odločilne funkcije

V tem poglavju predstavimo uporabljene odločilne funkcije EI, PI in LCB in novejše funkcije kot so ES in KG ter jih umestimo na skalo med exploracijo in exploitacijo. Za to poglavje predpostavimo, da želimo funkcije minimizirati, torej da je optimum funkcije v minimumu, v praksi to ni nujno zato pogosto preoblikujemo funkcijo v tako obliko, kjer to drži.

4.1 Verjetnost izboljšave

Verjetnost izboljšave ali PI je bila prva odločilna funkcije ustvarjena za Bayesovko optimizacijo. Predpostavimo, da je f_{min} do zdaj najmanjša dosežena vrednost:

$$f_{min} = minf \tag{4.1}$$

Verjetnost izboljšave evaluira funkcijo f v točki x, kjer je verjetnost največja. To je enakovredno naslednji funkciji u:

$$u(x) = \{ \tag{4.2}$$

...

Poglavje 5

Experimenti

5.1 Gradient Boosting

Gradient Boosting je ML algoritem za regresijo in klasifikacijo problemov, ki ustvari napovedni model v obliki ansambla slabih napovednih modelov, tipično odločilnih dreves.

https://towardsdatascience.com/probability-concepts-explained-marginalisation-2296846344fc https://machinelearningmastery.com/what-is-bayesian-optimization/ https://distill.pub/2020/bayesian-optimization/ http://krasserm.github.io/2018/03/19/gauss processes/ https://en.wikipedia.org/wiki/Multivariate normal distribution https://peterroela process-tutorial/ https://en.wikipedia.org/wiki/Mathttps://towardsdatascience.com/radial-basis-function-rbf-kernel-the-go-to-kernel-acf0d22c798a https://stats.stackexchange.com/ques is-the-rationale-of-the-mat