Bayesove neuronske mreže

Kristić, Joško

SVEUČILIŠTE U ZAGREBU

PRIRODOSLOVNO–MATEMATIČKI FAKULTET

MATEMATIČKI ODSJEK

Joško Kristić

BAYESOVE NEURONSKE MREŽE

Diplomski rad

Voditelj rada:
doc. dr. sc. Matej Mihelčić

Zagreb, Veljača, 2025.

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. _____

Mojim bližnjima

Sadržaj

Uvod

Neuronske mreže čine važan dio današnjeg strojnog učenja i umjetne inteligencije, gdje su ključne za analizu podataka, prepoznavanje uzoraka i donošenje odluka. Njihova struktura, nadahnuta načinom rada ljudskog mozga, omogućuje izgradnju modela prilagodljivih širokom rasponu zadataka – od jednostavnih klasifikacija do složenijih predviđanja i generativnih aplikacija. Iako su vrlo uspješne, tradicionalne neuronske mreže nailaze na određene izazove, a među glavnim problemima je pretreniranje, zbog kojeg modeli gube sposobnost generalizacije na nove podatke. Kako bi se ovo ublažilo, razvijeni su napredniji pristupi poput Bayesovskih neuronskih mreža. Ovi modeli koriste Bayesovsko zaključivanje pri treniranju i predikciji, omogućujući procjenu nesigurnosti modela i smanjenje pretreniranosti, čime postaju robusniji i prilagodljiviji. Ovaj rad ima za cilj proučiti teorijske osnove i praktične primjene Bayesovskih neuronskih mreža, prikazati njihove prednosti i ukazati na područja gdje mogu nadmašiti klasične pristupe. U prvom poglavlju analizirat će se opća struktura i rad neuronskih mreža te njihova primjena u nadziranom učenju. Drugo poglavlje bit će posvećeno Bayesovskom zaključivanju, osnovnom statističkom principu koji omogućuje adaptaciju modela s obzirom na nove podatke, čime se olakšava uključivanje nesigurnosti u modele. Treće poglavlje detaljno će objasniti koncept Bayesovskih neuronskih mreža, naglašavajući način na koji ove mreže spajaju Bayesov pristup sa standardnim metodama treniranja neuronskih mreža. Nadalje, u četvrtom poglavlju će se razmotriti problem pretreniranja i pokazati kako Bayesovski pristup može pomoći u rješavanju ovog problema, pružajući modelima sposobnost da bolje generaliziraju na neviđene podatke. Rad će završiti praktičnim primjerom primjene Bayesovskih neuronskih mreža, čime će se ilustrirati njihova prednost u realnim problemima gdje je kvantifikacija nesigurnosti ključna.

Poglavlje 1

Neuronske mreže

Ovo poglavlje ima za cilj pružiti razumijevanje osnovnih koncepata, struktura i funkcija neuronskih mreža, u svrhu pripreme daljnjeg razumijevanja glavne teme ovog rada - Bayesovih neuronskih mreža. U ovom poglavlju istražit ćemo ključne koncepte i teorije koje leže u osnovi neuronskih mreža. Počet ćemo s definicijom neuronskih mreža te njihovom poviješću, ističući važne prekretnice i istraživače koji su oblikovali njihov razvoj. Također, razmotriti ćemo arhitekturu neuronskih mreža, ulogu aktivacijskih funkcija te malo detaljnije postupke treniranja koji postaju bitno drugačiji prelaskom na Bayesovski pristup. Kroz ovo poglavlje, stvorit ćemo temelj za razumijevanje složenijih koncepata i primjena koje slijede uvodeći koncept Bayesove neuronske mreže.

1.1 Motivacija i povijest neuronskih mreža

Motivacija za istraživanje neuronskih mreža proizlazi iz želje da se stvore računalni sustavi koji mogu samostalno učiti i prilagođavati se okolini, analogno načinu na koji funkcioniraju biološki mozgovi. Prevodeći neuroznanost, koja je u to vrijeme bila jako slabo razvijena, u matematički okvir, pioniri poput McCullocha i Pittsa su 1943. [25] postavili temelje prve pojednostavljene neuronske mreže, McCulloch-Pitts neurona, koji je simulirao osnovne procese bioloških neurona. Sljedeći veliki iskorak napravio je Frank Rosenblatt 1958. godine razvivši perceptron, jednostavni model neuronske mreže sposoban za binarnu klasifikaciju, što je označilo početak praktične primjene neuronskih mreža u strojnom učenju [35] Kasnije, rad Geoffreya Hintona i suradnika na algoritmima učenja u neuronskim mrežama donio je novu eru u razvoju dubokog učenja [38]. Ovi pioniri su svojim radovima ucrtali put za razvoj modernih neuronskih mreža koje su danas okosnica istraživanja i razvoja u područjima kao što su računalni vid, obrada prirodnog jezika i autonomna vožnja.

Nadzirano i nenadzirano učenje

U strojnom učenju razlikujemo dvije osnovne vrste problema: nadzirano i nenadzirano učenje. U nadziranom učenju, model se trenira na označenim podacima, odnosno na skupu za učenje $(\mathbf{X}, \mathbf{y})$, gdje je $\mathbf{X}$ skup ulaznih podataka, a $\mathbf{y}$ pripadajući skup izlaznih oznaka ili vrijednosti. Cilj nadziranog učenja je naučiti funkciju $f : \mathbf{X} \to \mathbf{y}$, koja može točno predvidjeti izlaz $\mathbf{y}$ za nove, neviđene ulazne podatke $\mathbf{X}$. Ovaj pristup je posebno koristan za zadatke kao što su klasifikacija i regresija, gdje model treba predvidjeti određene oznake ili vrijednosti.

Suprotno tome, u nenadziranom učenju model dobiva samo ulazne podatke $\mathbf{X}$, bez pripadajućih oznaka $\mathbf{y}$, te samostalno otkriva strukture ili obrasce u podacima,

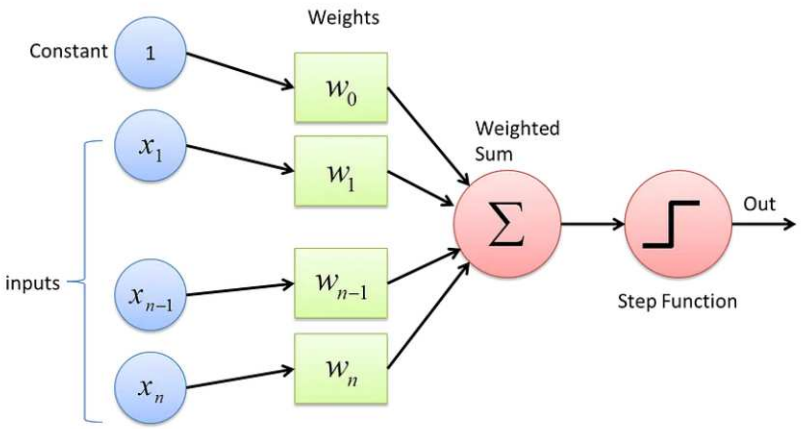
1.2 Definicija neuronske mreže

Pojam neuronske mreže obuhvaća veliku klasu modela i metoda učenja. Kroz ovo poglavlje ćemo pokazati da su neuronske mreže jedna malo kompleksnija familija nelinearnih statističkih modela. Ovdje počinjemo opisujući najčešće korištenu "jednostavnu" neuronsku mrežu, koja se ponekad naziva mreža s jednim skrivenim slojem ili jednostruki sloj neurona. Za početak, objasnimo pojam osnovne građevne jedinice neuronske mreže - neurona.

Neka je $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$ ulazni skup podataka, $\mathbf{w} = (w_0, \ldots, w_n) \in \mathbb{R}^n$ vektor te $f : \mathbb{R}^n \to \mathbb{R}$, tada je neuron definiran izborom $\mathbf{w}$ i f te je njegova "izlazna vrijednost":

$$r = f\left(w_0 + \sum_{i=1}^{n} x_i w_i\right) \tag{1.1}$$

Vrijednosti vektora $\mathbf{w}$ se zovu težine, a funkcija f aktivacijska funkcija. Slijedi shematski prikaz jednog neurona:



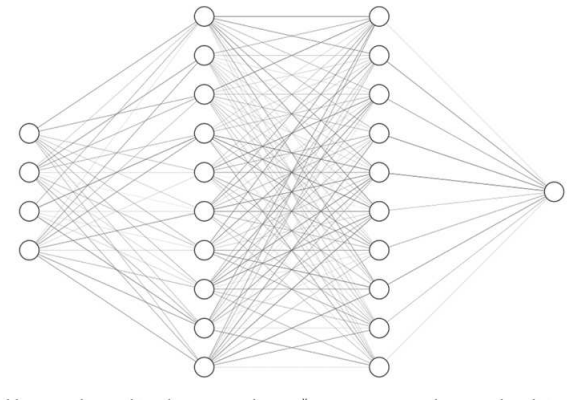
Slika 1.1: Shematski prikaz neurona sa "step" funkcijom kao aktivacijskom (preuzeto iz [39])

Aktivacijske funkcije ovdje igraju ključnu ulogu, uvodeći nelinearnost u model. Jedna od najosnovnijih aktivacijskih funkcija je sigmoidna funkcija $f(x) = \frac{1}{1+e^{-x}}$. Korištenjem sigmoidne aktivacijske funkcije na jednom neuronu, dobijamo uobičajnu logističku regresiju. Povijesno, jedna od prvih korištenih aktivacijskih funkcija je tzv. "step" funkcija. Neuron s takvom aktivacijskom funkcijom se zove perceptron [35]. Navodimo ovu aktivacijsku funkciju zbog zanimljive analogije koja motivira njen izbor. Naime, ako je ulazna vrijednost u aktivacijsku funkcija veća od neke granice, neuron se "aktivira" i poprima vrijednost 1 dok se u protivnom neuron ne aktivira i poprima vrijednost 0. Ovo je motivirano ponašanjem neurona u ljudskom mozgu gdje se kod određene doze impulsa (granica ili "threshold") određeni neuroni aktiviraju. Već na primjeru ove dvije funkcije vidimo dvije važne karakteristike aktivacijskih funkcija - normalizacija izlaznog podatka i mogućnost računanja gradijenta. Naime, ta svojstva će na važnosti dobiti kasnije kada uvedemo proces treniranja neuronskih mreža. Sada možemo definirati neuronsku mrežu sa ulaznim podacima dimenzije n, izlaznom vrijednosti dimenzije p te m "skrivenih" slojeva od kojih svaki sadrzi c_i neurona kao funkciju $f : \mathbb{R}^n \to \mathbb{R}^p$ koju računamo na sljedeći način:

1. Ulazni podaci $\mathbf{x}$ ulaze u svaki od c_0 neurona u ulaznom sloju.
2. Izlaz c_0 neurona ulaznog sloja ulazi u svaki od c_1 neurona prvog skrivenog sloja.
3. Izlaz c_i neurona u skrivenom sloju i ulazi u svaki od c_{i+1} neurona u skrivenom sloju $i + 1$
4. Izlaz c_m neurona iz zadnjeg skrivenog sloja ulazi u svaki od p neurona u izlaznom sloju.
5. Vrijednosti izlaznog sloja, vektor $\mathbf{y}$, predstavlja vrijednost funkcije u točki $\mathbf{x}$

Ulazni i izlazni slojevi mreže najčešće su određeni prirodom problema. Dimenzija ulaz-

nog sloja odgovara broju varijabli u skupu podataka, dok je dimenzija izlaznog sloja 1 u slučaju regresije ili p za klasifikacijski problem s p klasa. Osim toga, postoje i drugi tipovi zadataka, poput grupiranja podataka, segmentacije i generiranja objekata, koji zahtijevaju različite konfiguracije izlaznog sloja. Svaki od ovih zadataka može imati specifičan broj izlaznih neurona, ovisno o vrsti informacija koju mreža treba generirati ili prepoznati. Izbor hiperparametara se najčešće svodi na broj i veličinu skrivenih slojeva te izbor aktivacijskih funkcija. Aktivacijske funkcije se najčešće odabiru tako da svi neuroni u istome skrivenom sloju imaju istu aktivacijsku funkciju. Možemo reći da neuronsku mrežu definiraju dvije stvari - arhitektura i parametri. Gdje pod "arhitektura" mislimo na dimenziju ulaza i izlaza, broj skrivenih slojeva i broj neurona u svakom te izbor aktivacijskih funkcija, dok parametri predstavljaju vrijednosti težina u svakom pojedinom neuronu.



Slika 1.2: Shematski prikaz neuronske mreže gdje je $n = 4$, $p = 1$ sa 2 skrivena sloja po 10 neurona (preuzeto iz [49])

Do sada smo uveli najjednostavniji primjer neuronske mreže - tzv. "feedforward" neuronsku mrežu koja je dovoljna za daljnje razumijevanje koncepata. Naravno, postoje i mnoge druge arhitekture neuronskih mreža kao što su konvolucijske neuronske mreže, povratne neuronske mrežne, autoenkoderi, generativne neuronske mreže i dr. [11] U konačnici, svi ostali oblici se temelje na strukturi koju smo do sada objasnili. Iz mogućnosti izbora broja slojeva i neuron u pojedinom sloju vidimo da neuronske mreže mogu imati proizvoljno velik broj parametara što ih čini idealnom familijom funkcija za problem statističkog učenja s obzirom da se svaka Borel izmjeriva funkcija do na proizvoljnu razinu točnosti može aproksimirati dovoljno velikom neuronskom mrežom [15]. U nastavku prelazimo na temu "treniranja" neuronskih mreža tj. načina na koji se za dani skup podataka X pronalaze parametri neuronske mreže koja najbolje odgovara tim podacima. Napominjemo

da je sama arhitektura neuronske mreže najčešće unaprijed zadana na osnovu prijašnjih teorijskih i eksperimentalnih saznanja te se u procesu učenja traže isključivo parametri tj. skup težina **W**.

1.3 Treniranje neuronskih mreža

Treniranje neuronskih mreža ključni je proces u kojem funkcija koju smo ranije nazvali neuronska mreža postaje jedan od najkorištenijih alata u modernom statističkom učenju. Tijekom treniranja, mreža prilagođava svoje težine kako bi minimizirala razliku između predviđenih i stvarnih vrijednosti. Ovaj iterativni proces uključuje korištenje raznih algoritama optimizacije, poput gradijentnog spusta, te metoda poput propagacije unazad za izračun gradijenata [38]. Cilj treniranja je dobiti model koji se može generalizirati na nove, neviđene podatke, a pritom izbjegavajući probleme poput pretreniranja i nedovoljnog treniranja (eng. overfitting i underfitting).

Traženje "najboljih" težina

Neka je u pitanju problem regresije i $D = (x_i, y_i)_i$ je zadani skup podataka za učenje veličine N, $\mathbf{w}$ je skup težina neuronske mreže. Jedna od metoda učenja težina je metoda maksimalne vjerodostojnosti gdje vjerodostojnost definiramo kao:

$$L = P(D|\mathbf{w}) = \sum_{i=1}^{N} P(y_i \mid x_i, \mathbf{w}) \tag{1.2}$$

Za "najbolje" težine uzimamo one koje minimiziraju negativnu log-vjerodostojnost:

$$w_{\text{MLE}} = \arg\max_{w} \left[-\sum_{i=1}^{N} P(y_i|x_i, \mathbf{w}) \right] \tag{1.3}$$

Ovdje pretpostavljamo da je y dan kao $f(x; \mathbf{w})$ na koju je dodan Gaussovski šum, pa vjerojatnost iz gornjeg izraza možemo zapisati kao [26]:

$$P(y \mid x, \mathbf{w}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y - f(x; \mathbf{w}))^2}{2\sigma^2} \right) \tag{1.4}$$

Čime veličina koju minimiziramo postaje:

$$\begin{aligned} \mathcal{L}(\mathbf{w}) &= -\sum_{i=1}^{N} \log\left[\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y_i - f(x_i; \mathbf{w}))^2}{2\sigma^2} \right) \right] \\ &= -\sum_{i=1}^{N} \log(\sigma) - \frac{N}{2} \sum_{i=1}^{N} \log(2\pi) + \sum_{i=1}^{N} \frac{(y_i - \hat{y}_i)^2}{2\sigma^2} \end{aligned} \tag{1.5}$$

Gdje smo $f(x_i; \mathbf{w})$ zamijenili s $\hat{y}_i$ s obzirom da je f u ovom slučaju neuronska mreža kojoj tražimo parametre. sada vidimo da su prva dva člana konstante pa jedino što ostaje minimizirati je izraz $\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$ koji je zapravo standardna kvadratna greška. Ovime smo pokazali da pronalazak težina neuronske mreže MLE metodom ekvivalentan minimizaciji kvadratne greške na skupu za učenje. U kontekstu strojnog učenja se izraz unutar sume naziva funkcija gubitka:

$$C(y, \hat{y}) = (y_i - \hat{y}_i)^2 \tag{1.6}$$

Gore pokazana funkcija je uobičajna funkcija gubitka u problemu regresije, dok se u problemu klasifikacije najčešće koristi tzv. kategorijski gubitak entropije:

$$C(y, \hat{y}) = -\sum_{i=1}^{N} y_i \log(\hat{y}_i) \tag{1.7}$$

Ova funkcija gubitka je također funkcija cilja minimizacijskog problema u kojem dobijemo MLE procjenitelj za težine neuronske mreže, a dobije se sličnim računom kao i gore. Valja napomenuti da postoje mnoge druge u praksi korištene funkcije gubitka, koje nisu MLE procjenitelji, pogotovo u problemu klasifikacije [32]. U okviru ovog rada se držimo MLE procjenitelja tako da možemo zadržati vjerojatnosno značenje procesa treniranja kada kasnije prijeđemo na Bayesovski način traženja parametara.
Navedeni minimizacijski problem se najčešće rješava metodom gradijentnog spusta. Ideja ove iterativne optimizacijske tehnike je malim koracima u smjeru suprotnom od gradijenta funkcije gubitka mijenjati parametre mreže i pronaći globalni minimum funkcije gubitka. Jednadžba koraka gradijentnog spusta je sljedeća:

$$\mathbf{w} \leftarrow \mathbf{w} - \eta \nabla C(\mathbf{w}) \tag{1.8}$$

Gdje je η tzv. stopa učenja (eng. learning rate) - vrijednost koja kontrolira veličinu koraka koji se poduzima prilikom ažuriranja parametara. Često se uzima fiksna vrijednost između 0.001 i 0.01, a moderni optimizacijski algoritmi koriste i varijabilnu stopu učenja s ciljem brže konvergencije [37].
Ideja i jednadžba gradijentnog spusta su same po sebi jednostavne, no računanje samih gradijenata je malo složeniji problem ako se prisjetimo složenosti jedne neuronske mreže kao funkcije. Uz pretpostavku da je funkcija gubitka diferencijabilna u $\mathbf{w}$, algoritam propagacije unatrag daje efikasan i brz način za računanje gradijenata. Algoritam propagacije unatrag koristi princip ulančanog pravila za izračunavanje gradijenata funkcije gubitka u odnosu na sve težine u mreži. Sam račun nije konceptualno težak no izrazito je detaljan i nezanimljiv pa ovdje prilažemo samo glavnu jednadžbu. Detaljan raspis se može pronaći u [29]:

$$\frac{\partial C}{\partial \mathbf{w}_l} = \frac{\partial C}{\partial a_l} \cdot \frac{\partial a_l}{\partial z_l} \cdot \frac{\partial z_l}{\partial \mathbf{w}_l} \tag{1.9}$$

Gdje l označava sloj o kojem se radi, a aktivacijsku funkciju, a z linearnu kombinaciju ulaza u taj sloj. Korak u kojem se računa vrijednost neuronske mreže za ulaz x da se dobije vrijednost $y - \hat{y}$ se naziva propagacija unaprijed te je objašnjen ranije. Sada se svaka pojedina težina ažurira prema 1.8.
S obzirom da ovaj optimizacijski problem zahtijeva početne vrijednosti, vrijedi to još prokomentirati. Početne vrijednosti težina u neuronskim mrežama ključne su za uspješno treniranje modela jer pravilna inicijalizacija može značajno utjecati na brzinu konvergencije i učinkovitost mreže. Ako se težine inicijaliziraju na isti način za sve neurone, mreža ne može naučiti različite značajke, dok neodgovarajuće vrijednosti mogu uzrokovati probleme poput sporog učenja ili nestabilnosti. Tehnike poput Xavier, He, i LeCun inicijalizacije prilagođavaju početne vrijednosti na temelju broja ulaza i izlaza u sloju, čime se poboljšava stabilnost i brzina treniranja [27].

Poglavlje 2

Bayesovsko zaključivanje

Bayesovsko zaključivanje je pristup statističkoj analizi podataka temljen na Bayesovom teoremu, koji omogućava neprestano prilagođavanje modela s obzirom na nove podatke. U Bayesovskoj statistici se na parametre modela se gleda kao slučajne varijable s pripadnim distribucijama. Takav pristup omogućava fleksibilnost i prilagodbu modela, a uz same predikcije nudi i informaciju o razini neizvjesnosti. Takav pristup je osobito koristan u primjenama gdje treba razumijeti i kvantificirati razinu nesigurnosti. Za razliku od frekventističkog pristupa, Bayesovski pristup pruža mogućnost uvažavanja prethodnih znanja i ažuriranje tih uvjerenja na osnovu novih podataka.
U ovom poglavlju prvo ćemo obraditi osnovne principe Bayesovske statistike uvodeći pojmove prior i posterior distribucija te objašnjavajući njihovu ulogu. Nakon toga, navesti ćemo značaj marginalizacije i procjene nesigurnosti u parametrima. Zatim ćemo, kroz jednostavan primjer bacanja kovanice, prikazati kako Bayesovsko učenje funkcionira u praksi. Ovaj primjer omogućit će nam da jednostavno pokažemo pristup Bayesovskog učenja prije nego isti pristup uvedemo kod učenja neuronskih mreža.
Teorijski dio ovog poglavlja se temelji na izvorima [4] i [44].

2.1 Osnovni principi Bayesovske statistike

Bayesov teorem i uvjetna vjerojatnost

Bayesov teorem predstavlja temelj Bayesovske statistike, omogućavajući nam ažuriranje vjerojatnosti hipoteza na temelju novih podataka. U osnovi, Bayesov teorem se temelji na konceptu uvjetne vjerojatnosti, koja označava vjerojatnost događaja A pod uvjetom da se dogodio događaj B, te se zapisuje kao $P(A|B)$. Formalno, uvjetna vjerojatnost definirana je kao:

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \tag{2.1}$$

gdje je $P(A \cap B)$ vjerojatnost da se dogode oba događaja A i B, a $P(B)$ vjerojatnost događaja B, pod uvjetom da je $P(B) > 0$.

Bayesov teorem omogućava nam da izračunamo $P(A|B)$ koristeći obrnuto uvjetovanje, tj. $P(B|A)$, zajedno s *a priori* vjerojatnostima događaja A i B. Formalno, Bayesov teorem glasi:

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

gdje:

- $P(A|B)$ predstavlja *a posteriori* vjerojatnost događaja A nakon što smo promatrali događaj B,
- $P(B|A)$ je uvjetna vjerojatnost događaja B pod uvjetom da se dogodio događaj A,
- $P(A)$ je *a priori* vjerojatnost događaja A, tj. vjerojatnost prije nego što smo uzeli u obzir događaj B,
- $P(B)$ je ukupna vjerojatnost događaja B, koja se može dobiti marginalizacijom, odnosno sumiranjem ili integriranjem svih mogućih uvjeta:

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

  za diskretne slučajeve, ili

  $$P(B) = \int P(B|A)P(A)\, dA \tag{2.4}$$

  za neprekidne slučajeve.

Bayesov teorem možemo intuitivno shvatiti kao način ažuriranja početnih uvjerenja (izraženih kroz *a priori* vjerojatnost $P(A)$) u posjedovanju novih informacija (izraženih kroz $P(B|A)$). Rezultat ovog ažuriranja je *a posteriori* vjerojatnost $P(A|B)$, koja uzima u obzir i početne informacije i nove podatke.

Primjer Bayesovog teorema

Pretpostavimo jednostavan primjer dijagnostičkog testa za neku bolest. Neka je:

- A događaj da osoba ima bolest,
- B događaj da je test pozitivan.

Neka su poznate sljedeće informacije:

- $P(A)$ - *a priori* vjerojatnost da osoba ima bolest (npr., prevalencija bolesti u populaciji),
- $P(B|A)$ - vjerojatnost da je test pozitivan ako osoba ima bolest (osjetljivost testa),
- $P(B|\neg A)$ - vjerojatnost da je test pozitivan ako osoba nema bolest (lažno pozitivna stopa testa).

Na temelju Bayesovog teorema, možemo izračunati *a posteriori* vjerojatnost da osoba ima bolest s obzirom na pozitivan test, $P(A|B)$:

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B|A) \cdot P(A) + P(B|\neg A) \cdot P(\neg A)} \tag{2.5}$$

Ovaj primjer prikazuje kako Bayesov teorem koristi postojeće informacije o učestalosti bolesti i točnosti testa kako bi prilagodio vjerojatnost da osoba doista ima bolest, s obzirom na pozitivan rezultat testa. Takav način zaključivanja uobičajen je u medicinskoj dijagnostici, gdje je važno uzeti u obzir oba izvora nesigurnosti - početne informacije i rezultat testa. Bayesov teorem na taj način omogućava fleksibilno modeliranje nesigurnosti, prilagođavajući početne vjerojatnosti prema novim informacijama. Ova prilagodljivost je ključna u Bayesovskom pristupu statistici i strojnom učenju.

Pojam *prior* i *posterior* distribucije

Bayesovska statistika temelji se na ideji ažuriranja vjerojatnosti s obzirom na nove informacije. Dva ključna koncepta u tom procesu su *prior* i *posterior* distribucije, koje izražavaju naše uvjerenje o parametru prije i nakon prikupljanja podataka.

Prior distribucija

Prior distribucija, označena s $P(\theta)$, predstavlja početno uvjerenje o parametru θ prije nego što smo prikupili nove podatke. Ova distribucija odražava našu subjektivnu procjenu ili prethodno znanje o parametru i može biti informativna ili neinformativna, ovisno o tome koliko znamo o parametru. Na primjer, ako imamo prethodna istraživanja ili iskustva koja

upućuju na određene vrijednosti parametra, možemo koristiti informativnu *prior* distribuciju koja odražava ta znanja. U suprotnom, možemo odabrati neinformativnu *prior* distribuciju koja daje jednak prioritet svim mogućim vrijednostima parametra.

Posterior distribucija

Nakon što prikupimo podatke, naše početno uvjerenje o parametru se mijenja. Ažurirano uvjerenje izražavamo pomoću *posterior* distribucije, označene s $P(\theta|D)$, koja predstavlja vjerojatnost parametra θ s obzirom na nove podatke D. Bayesov teorem povezuje *prior* i *posterior* distribucije te omogućava ažuriranje vjerojatnosti. Kroz ovaj proces, *prior* i *posterior* distribucije pružaju fleksibilan okvir za izražavanje i ažuriranje vjerojatnosti uzimajući u obzir nesigurnost i nove podatke, što je osnova Bayesovske statistike.

Odabir *prior* distribucije

Odabir *prior* distribucije jedan je od ključnih koraka u Bayesovskoj statistici, jer ona predstavlja početno uvjerenje o parametru prije prikupljanja podataka. Pravilno postavljanje *priora* može značajno utjecati na rezultate analize, posebno kada je količina podataka ograničena. Ovisno o raspoloživom znanju o parametru, *prior* distribucije mogu biti različitih oblika.

Informativni i neinformativni priori

- **Informativna prior distribucija**: Koristi se kada imamo specifične informacije o parametru prije početka analize. Na primjer, u slučaju medicinskih istraživanja, raniji rezultati ili stručna znanja mogu ukazivati na određene vrijednosti ili raspon vrijednosti parametra. Informativna *prior* distribucija odražava te informacije i daje veće vjerojatnosti vrijednostima koje su konzistentne s prethodnim znanjem.

- **Neinformativna prior distribucija**: Kada nemamo prethodnih informacija o parametru, koristi se neinformativna ili slabo informativna *prior* distribucija, poput uniformne distribucije, koja jednako tretira sve moguće vrijednosti parametra. Cilj neinformativnog *priora* je omogućiti da podaci sami po sebi određuju *posterior* distribuciju. Primjeri neinformativnih *priora* uključuju uniformnu distribuciju, koja daje jednaku vjerojatnost svim vrijednostima parametra unutar određenog raspona.

Odabir oblika prior distribucije

Oblik *prior* distribucije može se birati ovisno o prirodi parametra koji procjenjujemo:

- Za parametre binarnih ili proporcijskih varijabli često se koristi *Beta* distribucija zbog svojih fleksibilnih svojstava i jednostavne interpretacije.
- Za parametre s neprekidnim vrijednostima, kao što su očekivana vrijednost ili varijanca, često se koriste *normalna* distribucija ili *inverzna gama* distribucija.

Odabir oblika i parametara *prior* distribucije trebao bi odražavati postojeće znanje i pretpostavke o parametru, ali također omogućiti fleksibilnost u ažuriranju s novim podacima. Uz pažljiv odabir, *prior* distribucija pruža okvir za formaliziranje početnih informacija i uvjerenja unutar Bayesovskog zaključivanja.

Posteriorna distribucija i ažuriranje s novim podacima

U Bayesovskoj statistici, *posterior* distribucija kombinira informacije iz *prior* distribucije i promatranih podataka, omogućavajući nam ažuriranje vjerojatnosti parametra s obzirom na nove informacije.

Bayesov teorem definira odnos između *prior*, *posterior* i funkcije vjerodostojnosti (eng. *likelihood*) na sljedeći način:

$$P(\theta|D) = \frac{P(D|\theta) \cdot P(\theta)}{P(D)} \tag{2.6}$$

gdje je:

- $P(\theta|D)$ - *posterior* distribucija, koja prikazuje ažurirano uvjerenje o parametru θ nakon promatranja podataka D,
- $P(D|\theta)$ - funkcija vjerodostojnosti, koja pokazuje vjerojatnost promatranih podataka D pod pretpostavkom da je θ točan parametar,
- $P(\theta)$ - *prior* distribucija, koja izražava naše početno uvjerenje o parametru θ prije promatranja podataka,
- $P(D)$ - marginalna vjerojatnost podataka, koja osigurava da se *posterior* distribucija normalizira tako da ukupna vjerojatnost iznosi 1.

Ažuriranje s novim podacima

Kada se pojave novi podaci, Bayesovski pristup omogućava ažuriranje uvjerenja korištenjem Bayesovog teorema. Nova *posterior* distribucija zatim može poslužiti kao *prior* za sljedeći korak ažuriranja s dodatnim podacima. Ovaj proces omogućava kontinuirano učenje u Bayesovskom pristupu.

Kroz ovaj način kombiniranja početnih uvjerenja i novih opažanja, *posterior* distribucija nudi jedan širi način procjene parametra, reflektirajući i prethodno znanje i nove dokaze. Kako se količina podataka povećava, *posterior* distribucija obično postaje sve uža, smanjujući nesigurnost oko procjene parametra θ.

Marginalizacija i izračun marginalne vjerojatnosti

U Bayesovskoj statistici, marginalizacija je proces u kojem eliminiramo određene varijable tako da integriramo njihove vjerojatnosti. Marginalizacija je ključna kada želimo izračunati marginalnu vjerojatnost podataka, $P(D)$, koja se koristi za normalizaciju *posterior* distribucije.

Marginalna vjerojatnost podataka $P(D)$ može se izraziti kao:

$$P(D) = \int P(D|\theta) \cdot P(\theta)\, d\theta \tag{2.7}$$

gdje:

- $P(D|\theta)$ predstavlja funkciju vjerodostojnosti koja pokazuje vjerojatnost podataka D pod pretpostavkom vrijednosti parametra θ,
- $P(\theta)$ je *prior* distribucija parametra θ.

Marginalizacijom integriramo nesigurnost u parametru θ kako bismo dobili ukupnu vjerojatnost podataka D, neovisno o specifičnim vrijednostima θ.

Važnost marginalizacije

Izračun marginalne vjerojatnosti podataka $P(D)$ omogućava:

- Normalizaciju *posterior* distribucije tako da ukupna vjerojatnost iznosi 1,
- Usporedbu različitih modela ili hipoteza korištenjem Bayesovog faktora, koji koristi omjer marginalnih vjerojatnosti podataka za različite hipoteze. [18]

U praksi, integrali potrebni za marginalizaciju mogu biti složeni ili nemogući za analitičko rješavanje, posebno kada radimo s višedimenzionalnim parametrima. U takvim slučajevima koriste se numeričke metode, poput Monte Carlo integracija, kako bi se aproksimativno izračunala marginalna vjerojatnosti.

Uvod u aproksimacijske metode

U Bayesovskoj statistici, izračun *posterior* distribucije često uključuje složene integrale koji mogu biti teško ili nemoguće riješiti analitički, posebno kada radimo s modelima koji sadrže veliki broj parametara. U takvim slučajevima, koristimo aproksimacijske metode koje omogućavaju procjenu *posterior* distribucije bez eksplicitnog izračunavanja svih potrebnih integrala.

Dvije najčešće korištene aproksimacijske metode u Bayesovskoj statistici su:

Monte Carlo Markovljevi Lanci (MCMC)

Monte Carlo Markovljevi Lanci (MCMC) metode koriste se za uzorkovanje iz *posterior* distribucije kada je direktno računanje složeno. MCMC metode generiraju uzorke koji slijede distribuciju od interesa (tj. *posterior* distribuciju u Bayesovskom učenju) i omogućuju aproksimaciju distribucije koristeći te uzorke.

Najpoznatiji MCMC algoritmi uključuju:

- **Metropolis-Hastings algoritam**: Generira uzorke koristeći prijedloženu distribuciju koja ovisi o prethodnom uzorku, a svaki prijedlog se prihvaća ili odbija na temelju određenih kriterija. [13]
- **Gibbs uzorkovanje**: Koristi se kada su uvjetne distribucije svih varijabli poznate, omogućavajući uzorkovanje svake varijable pojedinačno uz fiksiranje ostalih. [10]

MCMC metode su moćan alat za aproksimaciju složenih *posterior* distribucija, no zahtijevaju veliki broj iteracija kako bi se postigla konvergencija na ciljni oblik, što može biti računalno zahtjevno.

Varijacijsko zaključivanje [16]

Varijacijsko zaključivanje (eng. *Variational Inference*) je aproksimacijska metoda koja zamjenjuje stvarnu *posterior* distribuciju jednostavnijom distribucijom, koju možemo efikasnije računati. Cilj je pronaći aproksimacijsku distribuciju koja je što sličnija stvarnoj *posterior* distribuciji minimiziranjem određene mjere razlike između distribucija, kao što je Kullback-Leiblerova divergencija.

Glavne prednosti varijacijskog zaključivanja su:

- **Brzina**: U usporedbi s MCMC metodama, varijacijsko zaključivanje često može brže aproksimirati *posterior* distribuciju.
- **Efikasnost u visokoj dimenzionalnosti**: Pogodno je za modele s velikim brojem parametara, gdje MCMC metode mogu postati spore.

Iako varijacijsko zaključivanje omogućava brzo izračunavanje aproksimativne *posterior* distribucije, aproksimacija je često ograničena oblikom distribucije koju smo definirali, što može dovesti do netočne procjene u određenim slučajevima.

Aproksimacijske metode, kao što su MCMC i varijacijsko zaključivanje, omogućavaju Bayesovskoj statistici da se primjenjuje na složene modele i visoke dimenzije, gdje bi to inače bilo računalno prezahtjevno. Ove metode su ključne ako Bayesovsko učenje želimo primjeniti na složenim modelima kao što su neuronske mreže.

2.2 Primjer Bayesovskog učenja

Pogledajmo primjer procjene vjerojatnosti uspjeha kod binarnog eksperimenta, poput bacanja kovanice. Pretpostavljamo da želimo procijeniti vjerojatnost θ da kovanica padne na "glavu". Koristit ćemo Bayesov teorem kako bismo ažurirali početno uvjerenje o θ na temelju novih podataka.

Bayesov teorem

Prema Bayesovom teoremu, *posterior* distribucija $P(\theta|D)$ može se izraziti kao:

$$P(\theta|D) = \frac{P(D|\theta) \cdot P(\theta)}{P(D)} \tag{2.8}$$

gdje je:

- $P(\theta|D)$ - *posterior* distribucija, koja predstavlja ažurirano uvjerenje o parametru θ nakon promatranja podataka D,
- $P(D|\theta)$ - funkcija vjerodostojnosti (eng. *likelihood*), tj. vjerojatnost promatranih podataka D pod pretpostavkom da je θ prava vrijednost parametra,
- $P(\theta)$ - *prior* distribucija, koja predstavlja početno uvjerenje o parametru θ,
- $P(D)$ - marginalna vjerojatnost podataka, koja djeluje kao normalizacijska konstanta.

Postavljanje prior distribucije

Postavljamo *prior* distribuciju za θ kao Beta distribuciju s parametrima α i β:

$$P(\theta) = \text{Beta}(\theta; \alpha, \beta) = \frac{\theta^{\alpha-1}(1-\theta)^{\beta-1}}{B(\alpha, \beta)} \tag{2.9}$$

gdje je $B(\alpha, \beta)$ beta funkcija koja osigurava da je distribucija normalizirana. Za ovaj primjer, koristimo Beta(2, 2) distribuciju, što daje početno uvjerenje da je $\theta \approx 0.5$ s umjerenom sigurnošću.

Definiranje funkcije vjerodostojnosti

Funkcija vjerodostojnosti $P(D|\theta)$ predstavlja vjerojatnost promatranih podataka pod uvjetom da je θ vrijednost parametra. U slučaju binomnog eksperimenta (npr., n bacanja kovanice s k uspjeha), vjerodostojnost slijedi binomnu distribuciju:

$$P(D|\theta) = \theta^k(1-\theta)^{n-k} \tag{2.10}$$

gdje je:

- k broj uspjeha (glava) u n pokušaja.

Računanje posteriorne distribucije

Sada koristimo Bayesov teorem da izrazimo posteriornu distribuciju $P(\theta|D)$:

$$P(\theta|D) = \frac{P(D|\theta) \cdot P(\theta)}{P(D)} \tag{2.11}$$

Uvrštavanjem izraza za $P(D|\theta)$ i $P(\theta)$, dobivamo:

$$P(\theta|D) = \frac{\theta^k(1-\theta)^{n-k} \cdot \frac{\theta^{\alpha-1}(1-\theta)^{\beta-1}}{B(\alpha,\beta)}}{P(D)} \tag{2.12}$$

Što se može pojednostaviti kao:

$$P(\theta|D) = \frac{\theta^{k+\alpha-1}(1-\theta)^{n-k+\beta-1}}{P(D) \cdot B(\alpha,\beta)} \tag{2.13}$$

Normalizacija posteriorne distribucije

Budući da je oblik ovog izraza identičan Beta distribuciji s parametrima $\alpha + k$ i $\beta + n - k$, možemo reći da je posteriorna distribucija također Beta distribucija:

$$P(\theta|D) = \text{Beta}(\theta; \alpha + k, \beta + n - k) \tag{2.14}$$

Stoga, nakon promatranja k uspjeha u n pokušaja, ažurirani parametri za *posterior* Beta distribuciju su:

$$\alpha_{\text{posterior}} = \alpha_{\text{prior}} + k \tag{2.15}$$

$$\beta_{\text{posterior}} = \beta_{\text{prior}} + (n - k) \tag{2.16}$$

Provođenje eksperimenta

Za eksperiment, odabrali smo *prior* distribuciju Beta(2, 2), što nam daje početno uvjerenje da je $\theta \approx 0.5$. Proveli smo eksperiment bacanja kovanice $n = 10$ puta, pri čemu smo dobili $k = 7$ "glava".

Ažurirani parametri

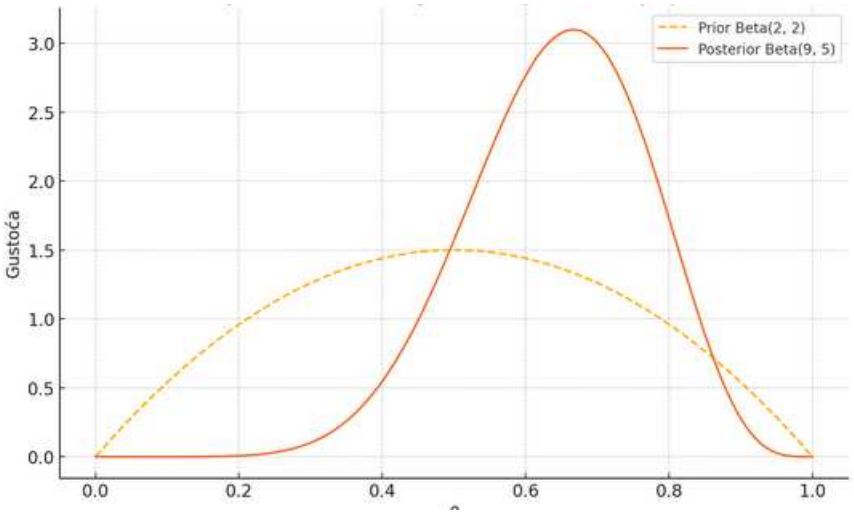
Koristeći prethodno izvedenu formulu za ažuriranje parametara, izračunavamo nove parametre *posterior* distribucije:

$$\alpha_{\text{posterior}} = 2 + 7 = 9 \tag{2.17}$$

$$\beta_{\text{posterior}} = 2 + (10 - 7) = 5 \tag{2.18}$$

Rezultat

Nova *posterior* distribucija je Beta(9, 5), koja predstavlja ažurirano uvjerenje o vjerojatnosti θ nakon promatranja podataka. Posteriorna distribucija je sada koncentriranija oko viših vrijednosti, što odražava podatke koji sugeriraju da je vjerojatnost "glave" veća od 0.5.



Slika 2.1: *Prior* i *posterior* distribucije

Ovaj primjer pokazuje osnovni princip Bayesovskog učenja: korištenje početnih informacija (*prior* distribucije) i ažuriranje uvjerenja na temelju novih podataka kako bi se dobila *posterior* distribucija. Takav pristup omogućava kontinuirano učenje i prilagodbu uvjerenja u skladu s promatranim podacima, što je osnova Bayesovske statistike i učenja.

Poglavlje 3

Bayesove neuronske mreže

3.1 Osnovne razlike

Bayesove neuronske mreže predstavljaju proširenje standardnih neuronskih mreža koristeći Bayesovski pristup za učenje i zaključivanje. Bayesove neuronske mreže su po arhitekturi, slojevima i funkcijama aktivacije identični standardnim neuronskim mrežama, dok se u tretiranju težina, metodama učenja i sposobnosti kvantifikacije nesigurnosti bitno razlikuju.

Definicija Bayesovskih neuronskih mreža [17]

Bayesovska neuronska mreža definirana je kao neuronska mreža u kojoj su težine $\mathbf{w}$ i pristranosti $\mathbf{b}$ modelirani kao slučajne varijable. Za razliku od običnih neuronskih mreža koje koriste standardne optimizacijske metode za točkovnu procjenu parametara $\mathbf{w}^*$, pritom minimizirajući funkciju gubitka, Bayesove neuronske mreže procijenjuju distribuciju parametara. Konkretno, težine $\mathbf{w}$ imaju pridruženu posteriornu distribuciju $p(\mathbf{w}|D)$, koja se računa prema Bayesovoj formuli:

$$p(\mathbf{w}|D) = \frac{p(D|\mathbf{w})p(\mathbf{w})}{p(D)}, \tag{3.1}$$

gdje:

- $p(\mathbf{w})$ predstavlja priornu distribuciju težina, koja izražava početne pretpostavke o vrijednostima parametara,
- $p(D|\mathbf{w})$ je vjerojatnost podataka uz dane parametre (vjerodostojnost),
- $p(\mathbf{w}|D)$ je posteriorna distribucija parametara,
- $p(D)$ je ukupna vjerojatnost podataka, također poznata kao *evidence*.

Često je posteriornu distribuciju $p(\mathbf{w}|D)$ nemoguće analitički izračunati zbog njene visoke dimenzionalnost, pa se zbog toga koriste aproksimativne metode kao što su varijacijsko zaključivanje i Monte Carlo metode za uzorkovanje iz posteriora.

Inferencija u Bayesovskim neuronskim mrežama

Inferencija (proces donošenja zaključaka ili predikcija na temelju modela i ulaznih podataka) u Bayesovskim neuronskim mrežama temelji se na uzorkovanju težina iz posteriorne distribucije. Za razliku od standardnih neuronskih mreža, gdje se koristi jedna procjena težina za računanje izlaza, Bayesovske neuronske mreže svaki parametar uzorkuju iz posteriorne distribucije u svakom prolasku kroz mrežu. Proces inferencije može se opisati sljedećim koracima [17]:

1. **Uzorkovanje težina**: Iz posteriorne distribucije $p(\mathbf{w}|D)$ uzorkuje se N različitih skupova težina $\{\mathbf{w}_i\}$:
$$\mathbf{w}_i \sim p(\mathbf{w}|D), \quad i = 1, \ldots, N. \tag{3.2}$$
2. **Izračunavanje predikcija**: Za svaki skup težina $\mathbf{w}_i$, mreža generira predikciju $\hat{\mathbf{y}}_i$ za ulazne podatke $\mathbf{x}$:
$$\hat{\mathbf{y}}_i = f(\mathbf{x}; \mathbf{w}_i), \quad i = 1, \ldots, N, \tag{3.3}$$
gdje je f funkcija koja predstavlja mrežu, a prolazak kroz mrežu je identičan kao i kod običnih neuronskih mreža
3. **Agregacija predikcija**: Konačna predikcija modela izračunava se agregacijom svih pojedinačnih predikcija. U slučaju regresije, to je obično prosjek svih predikcija:
$$\hat{\mathbf{y}} = \frac{1}{N} \sum_{i=1}^{N} \hat{\mathbf{y}}_i. \tag{3.4}$$
Za klasifikaciju, konačna predikcija može biti definirana kao najvjerojatnija klasa:
$$\hat{y} = \arg\max_k \left(\frac{1}{N} \sum_{i=1}^{N} \hat{y}_i^{(k)} \right), \tag{3.5}$$
gdje $\hat{y}_i^{(k)}$ predstavlja vjerojatnost klase k za i-ti uzorak težina.

Ovaj pristup je zapravo kombinacija predikcija generiranih s različitim skupovima težina, čime se obuhvaća nesigurnost modela i postiže robusnija procjena. Veći broj uzoraka povećava preciznost predikcije, ali zahtijeva veće računalne resurse.

Odabir prior distribucija [40]

Odabir prior distribucije ključan je korak u dizajnu Bayesovskih neuronskih mreža jer direktno utječe na sposobnost modela da generalizira i uči iz podataka. Prior distribucija $p(\mathbf{w})$

odražava naše početne pretpostavke o težinama i pristranostima mreže prije nego što se uzmu u obzir podaci D. Ove pretpostavke utječu na regularizaciju, mogućnost izražavanja nesigurnosti i računsku efikasnost modela.

Važnost prior distribucija

Za postavljanje bilo kojeg modela Bayesove neuronske mreže, prvi korak je definiranje prior distribucija. Uloge prior distribucija su sljedeće:

- **Regularizacija**: Prior ograničava složenost modela i pomaže s problemom pretreniranja.
- **Ubacivanje domenskog znanja**: Specifična znanja o problemu se mogu ugraditi kroz priore. Npr., težine možemo "nagnuti" na pretežno pozitivne ako postavimo eksponencijalni prior.
- **Kvantifikacija nesigurnosti**: Odabir priora izravno utječe na sposobnost modela da izrazi nesigurnost uzrokovanu nedostatkom podataka.
- **Utjecaj na posterior**: Kombinacija priora i vjerodostojnosti definira posteriornu distribuciju, koja je najzahtjevniji dio računa u procesu učenja. Kvalitetan prior pomaže u stabilizaciji i ubrzanju procesa učenja.

Uobičajene prior distribucije

Za neuronske mreže često se koriste sljedeće prior distribucije:

- **Gaussova distribucija**:
$$p(\mathbf{w}) = \mathcal{N}(\mu, \sigma^2), \tag{3.6}$$
gdje su μ srednja vrijednost i σ^2 varijanca. Ova distribucija je često prvi izbor zbog svoje jednostavnosti i svojstva da kažnjava kvadratno velike težine.
- **Laplaceova distribucija**:
$$p(\mathbf{w}) = \frac{1}{2b} \exp\left(-\frac{|\mathbf{w} - \mu|}{b}\right), \tag{3.7}$$
koja favorizira težine s malim apsolutnim vrijednostima.
- **Uniformna distribucija**:
$$p(\mathbf{w}) = \begin{cases} \frac{1}{b-a}, & a \leq \mathbf{w} \leq b \\ 0, & \text{inače} \end{cases} \tag{3.8}$$
Koristi se kao neutralan prior, kada nemamo nikakvih specifičnih saznanja o težinama.

- **Složeni priori**: Po potrebi se mogu koristiti i složenije distribucije kako bi se modelirale i složenije pretpostavke o težinama.

Utjecaj odabira priora na performanse modela

Odabir prior distribucije može značajno utjecati na performanse Bayesove neuronske mreže:

- Preuski prior (npr. s malom varijancom) može ograničiti fleksibilnost modela, dok preširoki prior može dovesti do pretjerano složenog modela.
- Ako je prior kontradiktoran s podacima, npr. favorizira težine koje u kontekstu problema nemaju smisla, može krivo usmjeriti posterior, što direktno utječe na točnost predikcija.

Razlike klasičnih i Bayesovih neuronskih mreža

Iako su po arhitekturi i operacijama (slojevi, funkcije aktivacije, način propagacije signala) obične i Bayesove neuronske mreže gotovo identične, razlikuju se u sljedećim ključnim aspektima:

- **Parametri kao distribucije**: U običnoj neuronskoj mreži težine $\mathbf{w}$ i pristranosti $\mathbf{b}$ su deterministčke vrijednosti koje se optimiziraju kako bi se minimizirala funkcija gubitka:
$$\mathbf{w}^* = \arg\min_{\mathbf{w}} L(\mathbf{w}, D), \tag{3.9}$$
gdje je L funkcija gubitka, a D podaci. Nasuprot tome, u Bayesovim neuronskin mrežama težine i pristranosti su slučajne varijable s pridruženom posteriornom distribucijom $p(\mathbf{w}|D)$.
- **Uzorkovanje težina**: Umjesto jednog skupa optimiziranih težina, Bayesove neuronske mreže više puta uzorkuju težine $\{\mathbf{w}_i\}$ iz distribucije:
$$\mathbf{w}_i \sim p(\mathbf{w}|D), \quad i = 1, \ldots, N, \tag{3.10}$$
gdje N predstavlja broj uzoraka. Svaki uzorak težina predstavlja zaseban model koji pridonosi ukupnoj predikciji.
- **Kvantifikacija nesigurnosti**: Dok standardne neuronske mreže daju jednu determinističku predikciju za ulazne podatke, Bayesove neuronske mreže omogućuju kvantifikaciju nesigurnosti u predikcijama. Ovo se postiže analizom varijacija u rezultatima uzorkovanih modela.

- **Regularizacija**: Priorna distribucija težina u Bayesovim neuronskim mrežama prirodno djeluje kao metoda regularizacije, dok standardne mreže često zahtijevaju dodatne tehnike regularizacije navedene ranije.
- **Pristup učenju**: Obične neuronske mreže koriste optimizaciju (npr. gradijentni spust) za minimizaciju funkcije gubitka. U Bayesovim neuronskim mrežama, umjesto klasične optimizacije, procijenjuje se posteriorna distribucija koristeći aproksimacijske metode.

Praktične implikacije

Iako je zahtijevno implementirati i trenirati Bayesovske neuronske mreže, sposobnost kvantifikacije nesigurnosti ih čini idealnim u okolinama gdje se greške u predikciji skupo plaćaju. Neki od takvih problema su autonomna vožnja, medicinska dijagnostika i procjena rizika [17].

3.2 Metode učenja

Glavni izazov kod učenja Bayesovskih neuronskih mreža je procjena posteriorne distribucije $p(\mathbf{w}|D)$, koja povezuje parametre $\mathbf{w}$ s podacima D. Posteriorna distribucija može se izraziti pomoću Bayesove formule:

$$p(\mathbf{w}|D) = \frac{p(D|\mathbf{w})p(\mathbf{w})}{p(D)}, \tag{3.11}$$

gdje su:

- $p(D|\mathbf{w})$ (*likelihood*) vjerojatnost podataka uz dane parametre,
- $p(\mathbf{w})$ priorna distribucija težina,
- $p(D)$ ukupna vjerojatnost podataka, izračunata kao:

$$p(D) = \int p(D|\mathbf{w})p(\mathbf{w})\, d\mathbf{w}. \tag{3.12}$$

Glavni razlog zašto je teško izračunati posterior je što je integral u nazivniku $p(D)$ visokodimenzionalan i često neizvediv za složene modele poput neuronskih mreža. U praksi, ovaj integral nije moguće izračunati analitički zbog velikog broja parametara (neuronske mreže nerijetko imaju milijune parametara) i nelinearnih aktivacijskih funkcija u slojevima mreže.

Zbog ovih izazova, učenje Bayesovih neuronskih mreža oslanja se na aproksimacijske metode za procjenu posteriorne distribucije. Te metode mogu se podijeliti u nekoliko glavnih kategorija:

Metode za aproksimaciju posteriora [3]

- **Markov Chain Monte Carlo (MCMC)**: - Algoritmi kao što su Metropolis-Hastings, Hamiltonian Monte Carlo (HMC) i No-U-Turn Sampler (NUTS) omogućuju uzorkovanje iz posteriora. Iako su vrlo precizni, kod velikih mreža do izražaja dolazi njihova računalna zahtijevnost.

- **Varijacijsko zaključivanje**: - Umjesto uzorkovanja, varijacijsko zaključivanje aproksimira posterior koristeći jednostavnije distribucije (npr. Gausovske distribucije). Optimizacija se postiže minimizacijom Kullback-Leiblerove divergencije između aproksimacijske distribucije i stvarnog posteriora.

- **Monte Carlo Dropout**: - Koristi uobičajnu dropout metodu regularizacije kako bi se simuliralo uzorkovanje iz više modela. Ova tehnika aproksimira Bayesovsko učenje i daje mogućnost procjene nesigurnosti modela. [9]

- **Stohastičke gradijentne metode**: - Kombiniraju gradijentni spust i uzorkovanje. Neke od metoda su Stohastičke gradijentne Langevin dinamike (SGLD) [45] i Stohastičke Hamiltonove Monte Carlo metode [5].

- **Bayes-by-Backprop**: - Specifična metoda varijacijskog zaključivanja prilagođena za neuronske mreže, koja koristi stohastičko uzorkovanje tijekom propagacije unatrag za učenje posteriora. [3]

U nastavku ćemo detaljnije obraditi dvije najvažnije metode: *Markov Chain Monte Carlo* i *Varijacijsko zaključivanje*.

Markov Chain Monte Carlo (MCMC) [28]

Monte Carlo Markovljevi Lanci (MCMC) predstavlja jednu od najtočnijih metoda za aproksimaciju posteriorne distribucije $p(\mathbf{w}|D)$ u Bayesovskim neuronskim mrežama. MCMC metode omogućuju uzorkovanje iz posteriorne distribucije konstruiranjem Markovljevog lanca koji asimptotski konvergira prema željenoj distribuciji.

Osnovni princip

Cilj MCMC-a je generirati uzorke $\{\mathbf{w}_i\}_{i=1}^{N}$ iz posteriorne distribucije:

$$p(\mathbf{w}|D) = \frac{p(D|\mathbf{w})p(\mathbf{w})}{p(D)}. \tag{3.13}$$

Problem je što normalizacijska konstanta $p(D) = \int p(D|\mathbf{w})p(\mathbf{w})\,d\mathbf{w}$ često nije izračunljiva. MCMC zaobilazi direktan izračun $p(D)$ tako što generira uzorke proporcionalno nenormaliziranom posterioru:

$$p^*(\mathbf{w}) \propto p(D|\mathbf{w})p(\mathbf{w}). \tag{3.14}$$

Markovljev lanac

MCMC metode se temelje na konstrukciji Markovljevog lanca $\mathbf{w}_1, \mathbf{w}_2, \ldots$ koji zadovoljava sljedeće:

- **Markovljevo svojstvo**: Svaka nova točka $\mathbf{w}_{i+1}$ ovisi samo o trenutnoj točki $\mathbf{w}_i$, a ne o prethodnima.
- **Stacionarna distribucija**: Lanac je konstruiran tako da njegova stacionarna distribucija odgovara posteriornoj distribuciji $p(\mathbf{w}|D)$.

Nakon inicijalnog razdoblja (*burn-in*), uzorci iz lanca mogu se koristiti za aproksimaciju posteriora. Procjena očekivanja bilo koje funkcije $f(\mathbf{w})$ pod posteriorom tada je:

$$\mathbb{E}[f(\mathbf{w})] \approx \frac{1}{N}\sum_{i=1}^{N} f(\mathbf{w}_i), \tag{3.15}$$

gdje su $\mathbf{w}_i \sim p(\mathbf{w}|D)$ uzorci generirani MCMC-om.

Metropolis-Hastings algoritam

Jedan od osnovnih algoritama za MCMC je Metropolis-Hastings (MH) algoritam. On koristi predloženu distribuciju $q(\mathbf{w}'|\mathbf{w})$ za generiranje prijedloga $\mathbf{w}'$ za novu točku u lancu. Nova točka se prihvaća s vjerojatnošću:

$$\alpha = \min\left(1, \frac{p^*(\mathbf{w}')q(\mathbf{w}|\mathbf{w}')}{p^*(\mathbf{w})q(\mathbf{w}'|\mathbf{w})}\right). \tag{3.16}$$

Ako je prijedlog odbijen, trenutna točka $\mathbf{w}$ se ponavlja u lancu:

$$\mathbf{w}_{i+1} = \begin{cases} \mathbf{w}', & \text{ako } u \le \alpha, \\ \mathbf{w}, & \text{inače,} \end{cases} \tag{3.17}$$

gdje je $u \sim \text{Uniform}(0, 1)$ slučajni broj.

MH algoritam je fleksibilan, ali može biti spor za visoke dimenzije zbog osjetljivosti predložene distribucije $q(\mathbf{w}'|\mathbf{w})$.

Hamiltonian Monte Carlo (HMC)

Hamiltonian Monte Carlo (HMC) proširuje MCMC uvodeći pomoćne varijable i koristi gradijentne informacije posteriora za bolje uzorkovanje. Posterior $p(\mathbf{w}|D)$ se kombinira s pomoćnom varijablom $\mathbf{r}$, koja simulira trenutak u sustavu:

$$p(\mathbf{w}, \mathbf{r}|D) \propto \exp\left(-\mathcal{H}(\mathbf{w}, \mathbf{r})\right), \tag{3.18}$$

gdje je $\mathcal{H}$ Hamiltonijan definiran kao:

$$\mathcal{H}(\mathbf{w}, \mathbf{r}) = -\ln p(D|\mathbf{w}) - \ln p(\mathbf{w}) + \frac{1}{2}\mathbf{r}^\top \mathbf{r}. \tag{3.19}$$

HMC koristi Hamiltonovu dinamiku za simulaciju gibanja kroz prostor parametara koristeći gradijent log-vjerodostojnosti:

$$\frac{\partial \mathbf{w}}{\partial t} = \frac{\partial \mathcal{H}}{\partial \mathbf{r}}, \quad \frac{\partial \mathbf{r}}{\partial t} = -\frac{\partial \mathcal{H}}{\partial \mathbf{w}}. \tag{3.20}$$

Numerički integratori, poput Leapfrog metode, koriste se za izračun gibanja:

$$\mathbf{r}_{t+\epsilon/2} = \mathbf{r}_t - \frac{\epsilon}{2}\nabla_{\mathbf{w}}\mathcal{H}(\mathbf{w}_t) \tag{3.21}$$

$$\mathbf{w}_{t+\epsilon} = \mathbf{w}_t + \epsilon \mathbf{r}_{t+\epsilon/2} \tag{3.22}$$

$$\mathbf{r}_{t+\epsilon} = \mathbf{r}_{t+\epsilon/2} - \frac{\epsilon}{2}\nabla_{\mathbf{w}}\mathcal{H}(\mathbf{w}_{t+\epsilon}), \tag{3.23}$$

gdje je ϵ duljina koraka.

HMC efikasno obliazi prostor parametara, ali računanje gradijenata dovodi do velike računalne složenosti.

Prednosti i ograničenja MCMC-a

Prednosti:

- Visoka preciznost aproksimacije posteriora.
- Teorija osigurava asimptotsku točnost uzorkovanja.

Ograničenja:

- Računski intenzivno, posebno za velike neuronske mreže.
- Velika potreba za prilagodbom hiperparametara (npr. koraci u HMC-u).
- Spora konvergencija u visokodimenzionalnim prostorima.

MCMC metode su najpopularnija metoda za Bayesovsku inferenciju, ali njihova računalna zahtijevnost često stvara problem u primjeni kod kompleksnih modela kao što su neuronske mreže.

Varijacijsko zaključivanje [22]

Varijacijsko zaključivanje (eng. *Variational Inference*, VI) predstavlja aproksimacijsku metodu za procjenu posteriorne distribucije $p(\mathbf{w}|D)$, koristeći optimizaciju umjesto uzorkovanja. Cilj varijacijskog zaključivanja je aproksimirati posterior jednostavnijom distribucijom $q(\mathbf{w}|\boldsymbol{\phi})$, gdje $\boldsymbol{\phi}$ predstavlja skup parametara aproksimacijske distribucije. Ova metoda ima puno manji problem s dimenzionalnosti kao što je bio slučaj kod MCMC metoda, što je izrazito bitno kod Bayesovih neuronskih mreža.

Osnovni princip

Varijacijsko zaključivanje optimizira razliku između aproksimacijske distribucije $q(\mathbf{w}|\boldsymbol{\phi})$ i prave posteriorne distribucije $p(\mathbf{w}|D)$. Razlika se mjeri pomoću Kullback-Leiblerove divergencije (KL divergencije):

$$\mathrm{KL}(q(\mathbf{w}|\boldsymbol{\phi}) \,||\, p(\mathbf{w}|D)) = \int q(\mathbf{w}|\boldsymbol{\phi}) \ln \frac{q(\mathbf{w}|\boldsymbol{\phi})}{p(\mathbf{w}|D)} \, d\mathbf{w}. \tag{3.24}$$

S obzirom na to da je posterior $p(\mathbf{w}|D)$ proporcionalan $p(D|\mathbf{w})p(\mathbf{w})$, KL divergencija se može preoblikovati u oblik koji ne ovisi o $p(D)$:

$$\mathrm{KL}(q(\mathbf{w}|\boldsymbol{\phi}) \,||\, p(\mathbf{w}|D)) = \mathbb{E}_{q(\mathbf{w}|\boldsymbol{\phi})}[\ln q(\mathbf{w}|\boldsymbol{\phi}) - \ln p(D|\mathbf{w}) - \ln p(\mathbf{w})] + \text{konst.} \tag{3.25}$$

Umjesto minimizacije KL divergencije, varijacijsko zaključivanje maksimizira tzv. *varijacijski donji rub* (eng. *Evidence Lower Bound*, ELBO):

$$\mathcal{L}(\boldsymbol{\phi}) = \mathbb{E}_{q(\mathbf{w}|\boldsymbol{\phi})}[\ln p(D|\mathbf{w})] - \mathrm{KL}(q(\mathbf{w}|\boldsymbol{\phi}) \,||\, p(\mathbf{w})). \tag{3.26}$$

Komponente varijacijskog donjeg limita su:

- $\mathbb{E}_{q(\mathbf{w}|\boldsymbol{\phi})}[\ln p(D|\mathbf{w})]$: - Predstavlja očekivanu log-vjerodostojnost podataka pod aproksimacijskom distribucijom $q(\mathbf{w}|\boldsymbol{\phi})$. - Potiče $q(\mathbf{w}|\boldsymbol{\phi})$ da se uskladi s podacima.
- $-\mathrm{KL}(q(\mathbf{w}|\boldsymbol{\phi}) \,||\, p(\mathbf{w}))$: - Kažnjava razliku između aproksimacijske distribucije i prirorne distribucije težina $p(\mathbf{w})$. - Djeluje kao oblik regularizacije.

Maksimizacijom $\mathcal{L}(\boldsymbol{\phi})$, $q(\mathbf{w}|\boldsymbol{\phi})$ postaje blisko posterioru $p(\mathbf{w}|D)$.

Aproksimacijska distribucija

U praksi, aproksimacijska distribucija $q(\mathbf{w}|\boldsymbol{\phi})$ obično se bira iz jednostavnijih oblika kako bi optimizacija bila izvediva. Najčešće korištena je faktorizirana Gausovska distribucija:

$$q(\mathbf{w}|\boldsymbol{\phi}) = \prod_{i=1}^{N} \mathcal{N}(w_i|\mu_i, \sigma_i^2), \tag{3.27}$$

gdje su $\boldsymbol{\phi} = \{\mu_i, \sigma_i^2\}_{i=1}^N$ parametri aproksimacijske distribucije.

Prednost ovog oblika je da omogućuje lako uzorkovanje težina $\mathbf{w}$ i učinkovito računanje gradijenata potrebnih za optimizaciju.

Stohastičko varijacijsko zaključivanje

Za velike neuronske mreže, izravno računanje očekivanja u varijacijskom donjem limitu je nepraktično zbog velikog broja težina. Umjesto toga koristi se Monte Carlo procjena:

$$\mathbb{E}_{q(\mathbf{w}|\boldsymbol{\phi})}[\ln p(D|\mathbf{w})] \approx \frac{1}{S} \sum_{s=1}^{S} \ln p(D|\mathbf{w}^{(s)}), \quad \mathbf{w}^{(s)} \sim q(\mathbf{w}|\boldsymbol{\phi}), \tag{3.28}$$

gdje je S broj uzoraka iz aproksimacijske distribucije.

Parametri $\boldsymbol{\phi}$ optimiziraju se korištenjem gradijentnih metoda. Gradijenti ELBO-a izračunavaju se pomoću reparametrizacijskog trika:

$$\mathbf{w} = \mu + \sigma \odot \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I}), \tag{3.29}$$

što omogućuje računanje gradijenata u odnosu na μ i σ.

Prednosti i ograničenja varijacijskog zaključivanja

Prednosti:

- Brza i skalabilna metoda pogodna za velike mreže.
- Jednostavna implementacija u većini modernih programskih jezika.
- Jako precizna aproksimacija posteriora.

Ograničenja:

- Aproksimacija ovisi o obliku odabrane distribucije $q(\mathbf{w}|\boldsymbol{\phi})$.
- Zbog pretpostavki na q, nesigurnost se često zna podcijeniti.
- Teorija ne garantira da će aproksimacija uvijek biti bliska stvarnom posterioru.

Zbog svoje računalne efikasnosti i skalabilnosti, varijacijsko zaključivanje je često korišteno kod učenja Bayesovih neuronskih mreža gdje se broj parametara može mjeriti u milijunima.

3.3 Kvantificiranje neizvjesnosti

Jedna od ključnih prednosti Bayesovskih neuronskih mreža u usporedbi s klasičnim neuronskim mrežama je njihova sposobnost kvantificiranja nesigurnosti u predikcijama. Kvantifikacija neizvjesnosti je od presudne važnosti u aplikacijama gdje su pouzdane procjene ključne, poput medicinske dijagnostike, autonomnih vozila i procjena rizika u financijama.

Vrste neizvjesnosti [9]

Bayesovske neuronske mreže omogućuju kvantifikaciju dviju glavnih vrsta nesigurnosti [19]:

- **Aleatorička (stohastička) nesigurnost**: Ova nesigurnost proizlazi iz prirodne slučajnosti. Primjerice, u postavljanju modela u strojnom učenju često zavisnu varijablu modeliramo kao $f(x) + \epsilon$, gdje je f funkcija koju pokušavamo naučiti, a ϵ tzv. šum. Ova vrsta nesigurnosti se ne može smanjiti povećavanjem skupa za treniranje s obzirom da je ona svojstvena problemu.
- **Epistemološka (modelska) nesigurnost**: Ova nesigurnost proizlazi iz ograničenja modela ili nedostatka podataka. Primjerice, u problemu linearne regresije kao rezultat dobijemo točkovne procjene parametara. Tada se epistemološka nesigurnost očituje kroz razinu statističke značajnosti tih parametara. Ova vrsta nesigurnosti se smanjuje povećavanjem skupa za treniranje.

Aleatorička nesigurnost [2]

Aleatorička nesigurnost modelira se kroz distribuciju izlaznih vrijednosti. U slučaju regresije, često se pretpostavlja da izlazna vrijednost sadrži šum koji je normalno distribuiran. Tada možemo modelirati $p(\mathbf{y}|\mathbf{x}, \mathbf{w})$ kao Gaussovu distribuciju:

$$p(\mathbf{y}|\mathbf{x}, \mathbf{w}) = \mathcal{N}\left(\mathbf{y} \,\middle|\, f(\mathbf{x}; \mathbf{w}), \sigma_a^2 \mathbf{I}\right), \tag{3.30}$$

gdje je:

- $f(\mathbf{x}; \mathbf{w})$ funkcija modela koja daje srednju vrijednost predikcije za dani ulaz $\mathbf{x}$ i parametre $\mathbf{w}$,
- σ_a^2 varijanca aleatoričke nesigurnosti,
- $\mathbf{I}$ jedinična matrica odgovarajuće dimenzije.

Varijanca σ_a^2 može biti konstantna (homoskedastički šum) ili može ovisiti o ulazu $\mathbf{x}$ (heteroskedastički šum).

Epistemološka nesigurnost [28]

Epistemološka nesigurnost proizlazi iz nesigurnosti u parametrima modela $\mathbf{w}$, koje Bayesovske neuronske mreže modeliraju kroz posteriornu distribuciju $p(\mathbf{w}|D)$. Ova nesigurnost pokazuje neznanje o točnim vrijednostima parametara zbog ograničenih ili šumovitih podataka.

Za opisivanje epistemološke nesigurnosti u predikciju, potrebno je uzeti u obzir varijabilnost predikcije čiji je uzrok slučajnost u parametrima (s obzirom da su parametri u Bayesovoj neuronskoj mreži slučajne varijable). Dakle, ukupna prediktivna distribucija izlaza $\mathbf{y}$ za dani ulaz x i podatke $\mathbf{D}$ na kojima je treniran model se dobije integriranjem po svim mogućim vrijednostima parametara $\mathbf{w}$:

$$p(\mathbf{y}|\mathbf{x}, D) = \int p(\mathbf{y}|\mathbf{x}, \mathbf{w})\, p(\mathbf{w}|D)\, d\mathbf{w}. \tag{3.31}$$

Ovdje je:

- $p(\mathbf{w}|D)$ posteriorna distribucija parametara nakon što uz dane podatke D,
- $p(\mathbf{y}|\mathbf{x}, \mathbf{w})$ je već definirana vjerojatnost izlaza s obzirom na ulaz i parametre.

Integriranje preko $\mathbf{w}$ zapravo uzima u obzir sve moguće vrijednosti parametara s obzirom da njihovu posteriornu distribuciju.

Mjerenje ukupne nesigurnosti

Ukupna nesigurnost u predikciji može se kvantificirati izračunavanjem ukupne varijance prediktivne distribucije $p(\mathbf{y}|\mathbf{x}, D)$ [3]. Ukupna varijanca može se rastaviti na dva dijela: aleatoričku varijancu i epistemološku varijancu. Ako pretpostavimo da je $\mathbf{y}$ skalarna, ukupna varijance sa računa kao:

$$\mathrm{Var}(\mathbf{y}|\mathbf{x}, D) = \underbrace{\mathbb{E}_{p(\mathbf{w}|D)}\left[\mathrm{Var}(\mathbf{y}|\mathbf{x}, \mathbf{w})\right]}_{\text{Aleatorička nesigurnost}} + \underbrace{\mathrm{Var}_{p(\mathbf{w}|D)}\left[\mathbb{E}(\mathbf{y}|\mathbf{x}, \mathbf{w})\right]}_{\text{Epistemološka nesigurnost}}. \tag{3.32}$$

Prvi član je očekivana varijanca izlaza s obzirom na aleatoričku nesigurnost, a drugi varijanca očekivanih izlaza s obzirom na nesigurnost u parametrima. Dalje se računa:

1. **Aleatorička nesigurnost**:

$$\mathbb{E}_{p(\mathbf{w}|D)}\left[\mathrm{Var}(\mathbf{y}|\mathbf{x}, \mathbf{w})\right] = \int \mathrm{Var}(\mathbf{y}|\mathbf{x}, \mathbf{w})\, p(\mathbf{w}|D)\, d\mathbf{w}. \tag{3.33}$$

Ovdje je:

- $\text{Var}(\mathbf{y}|\mathbf{x}, \mathbf{w})$ varijanca aleatoričke nesigurnosti za dane parametre $\mathbf{w}$, obično je to σ_a^2 ako je distribucija šuma neovisna o $\mathbf{x}$ [31].

Ako je σ_a^2 konstantna, tada je aleatorička nesigurnost jednostavno jednaka toj konstanti.

2. **Epistemološka nesigurnost**:

$$\text{Var}_{p(\mathbf{w}|D)}\left[\mathbb{E}(\mathbf{y}|\mathbf{x}, \mathbf{w})\right] = \int \left(\mathbb{E}(\mathbf{y}|\mathbf{x}, \mathbf{w}) - \mathbb{E}_{p(\mathbf{w}|D)}[\mathbb{E}(\mathbf{y}|\mathbf{x}, \mathbf{w})]\right)^2 p(\mathbf{w}|D)\, d\mathbf{w}. \tag{3.34}$$

Ovdje je:

- $\mathbb{E}(\mathbf{y}|\mathbf{x}, \mathbf{w})$ očekivana vrijednost izlaza za dane parametre $\mathbf{w}$, što je obično $f(\mathbf{x}; \mathbf{w})$,
- $\mathbb{E}_{p(\mathbf{w}|D)}[\mathbb{E}(\mathbf{y}|\mathbf{x}, \mathbf{w})]$ ukupna očekivana vrijednost preko posteriora parametara, tj.:

$$\mathbb{E}_{p(\mathbf{w}|D)}[f(\mathbf{x}; \mathbf{w})] = \int f(\mathbf{x}; \mathbf{w})\, p(\mathbf{w}|D)\, d\mathbf{w}. \tag{3.35}$$

Dakle, epistemološka nesigurnost kvantificira koliko se očekivane vrijednosti predikcija razlikuju zbog nesigurnosti u parametrima.

Završni korak je računanje dobivenih integrala koje se radi ranije opisanim metoda za integriranje po prostoru parametara.

Poglavlje 4

Problem pretreniranja

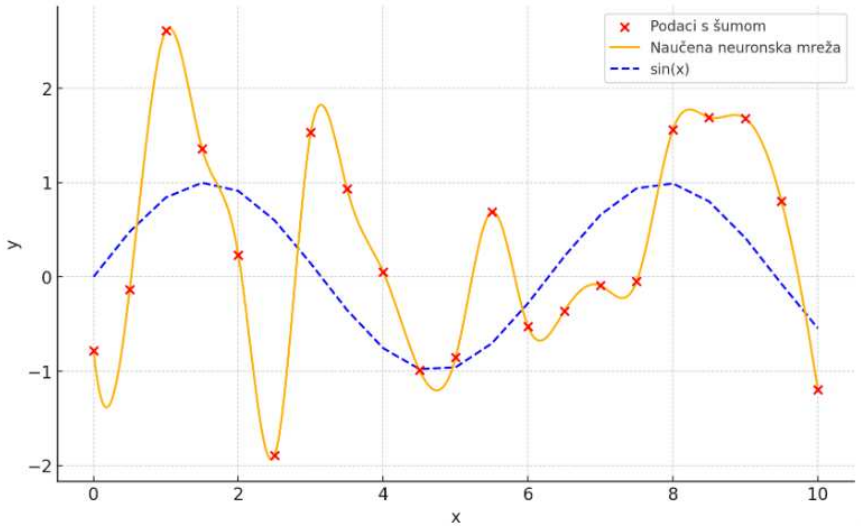
Pretreniranje (eng. overfitting) je čest problem u modelima strojnog učenja, uključujući neuronske mreže, ali se može javiti u gotovo svim oblicima modeliranja podataka. Do pretreniranja dolazi kada model previše precizno "zapamti" uzorke u skupu za učenje, umjesto da nauči općenite odnose koji bi se mogli primijeniti na nove, neviđene podatke. Kada je model pretreniran, performanse na trening podacima su izuzetno visoke, no kad se model primijeni na novi skup podataka (npr. testni skup ili podaci iz stvarnog svijeta), njegova točnost se bitno razlikuje. Razlog za to je što model postaje vrlo dobar u prepoznavanju specifičnih uzoraka i iznimaka unutar trening podataka, no ne uspijeva uhvatiti šire trendove koji bi mogli pomoći u generalizaciji. Pretreniranje se najčešće događa kada model ima previše parametara u odnosu na količinu i složenost dostupnih podataka. Takav model ne samo da uči korisne obrasce iz podataka, već i šum koji nije karakteristika podataka.

4.1 Primjer pretreniranja

Ilustrirajmo problem pretreniranja pomoću jednostavnog primjera učenja gdje pokušavamo naučiti funkciju $f : \mathbb{R} \to \mathbb{R}$ na podacima generiranim na sljedeći način:

$$
\begin{aligned}
x &= (0, 0.5, 1, \ldots, 9.5, 10) \\
y &= \sin(x) + \epsilon \\
\epsilon &\sim \mathcal{N}(0, 1)
\end{aligned}
$$

Učenjem neuronske mreže s dovoljno velikim brojem parametara na gornjim podacima, dobijamo model koji "savršeno" opisuje podatke u skupu za učenje te izgleda ovako:



Slika 4.1: Fittana neuronska mreža, originalni podaci te sinusoida

Vidimo da naša neuronska mreža prolazi kroz svaku točku skupa za učenje. Iako to na prvu može zvučati kao pozitivan ishod (gubitak na skupu za učenje je 0), sjetimo se da bi najbolji ishod ovog problema učenja bio kada bi neuronska mreža poprimila oblik sinusoide, s obzirom da su podaci generirani sinusoidom. Naime, ideja statističkog učenja nije da model nauči šum, koji je slučajan, već uzorke i obrasce koji generiraju podatke. Vidimo da je u gornjem slučaju neuronska mreža poprimila oblik neke vrste polinomijalnog splinea. Ovo ne treba čuditi s obzirom da je pokazano da ”dovoljno velika” neuronska mreža može ”dovoljno dobro” aproksimirati proizvoljnu funkciju [38].

4.2 Metode borbe protiv pretreniranja kod neuronskih mreža

Borba protiv pretreniranja u strojnome učenju i dubokim neuronskim mrežama jedan je od ključnih izazova u samoj konstrukciji neuronskih mreža. U slučaju običnih determinističkih neuronskih mreža, neke od najefikasnijih metoda su [48]:

- **Regularizacija**
  - *L1 regularizacija (Lasso)*: Penalizira apsolutnu vrijednost težina.
  - *L2 regularizacija (Ridge)*: Penalizira kvadratnu vrijednost težina.
- **Kros validacija (Cross-validation)**

  - Metoda podjele podataka u k podskupova (npr. *k-fold cross-validation*), pri čemu se model više puta trenira i testira na različitim podskupovima.

- **Dropout**
  - Tijekom treniranja neuronskih mreža, slučajno se isključuje određeni postotak neurona.
- **Rana zaustavljanja (Early Stopping)**
  - Treniranje modela se prekida kada se performanse na validacijskom skupu počnu pogoršavati.
- **Smanjenje složenosti modela**
  - Upotreba jednostavnijih modela, s manje slojeva ili neurona.

Regularizacija

Regularizacija je tehnika kojom pokušavamo smanjiti pretreniranje, tako što smanjujemo složenost modela. Regularizacija dodaje "kazne" u funkciju gubitka kako bi težine modela bile manje, i time potiče jednostavniji model. Navodimo glavne oblike regularizacije.

L2 Regularizacija (Ridge)

Kod L2 regularizacije, dodajemo kaznu koja je proporcionalna kvadratu vrijednosti težina modela. Funkcija gubitka $L(\mathbf{w})$ sada izgleda ovako:

$$C(\mathbf{w}) = C_{\text{original}}(\mathbf{w}) + \lambda \sum_{i=1}^{n} w_i^2 \tag{4.1}$$

Ovdje je $C_{\text{original}}(\mathbf{w})$ originalna funkcija gubitka, $\mathbf{w}$ su težine modela, a λ je hiperparametar koji kontrolira snagu regularizacije. L2 regularizacija smanjuje težine modela, ali ih ne postavlja na nulu.

L1 Regularizacija (Lasso)

Kod L1 regularizacije, kazna je proporcionalna apsolutnoj vrijednosti težina. Funkcija gubitka sada postaje:

$$C(\mathbf{w}) = C_{\text{original}}(\mathbf{w}) + \lambda \sum_{i=1}^{n} |w_i| \tag{4.2}$$

Za razliku od L2 regularizacije, ovdje mnoge težine postaju točno nula, što znači da se neke varijable (točnije neuroni čiji je input neka kombinacija ulaznih varijabli) eliminiraju.

ElasticNet Regularizacija

ElasticNet kombinira prednosti L1 i L2 regularizacije. Funkcija gubitka izgleda ovako:

$$C(\mathbf{w}) = C_{\text{original}}(\mathbf{w}) + \lambda_1 \sum_{i=1}^{n} |w_i| + \lambda_2 \sum_{i=1}^{n} w_i^2 \tag{4.3}$$

Ovdje λ_1 kontrolira snagu L1 regularizacije, a λ_2 snagu L2 regularizacije. ElasticNet je koristan kada postoji mnogo koreliranih značajki, jer kombinira prednosti obje metode.

U konačnici, regularizacija prisiljava težine da budu manje, što model čini jednostavnijim, dok L1 regularizacija u potpunosti eliminira neke težine. Također, regularizacija smanjuje varijabilnost modela, pa je manja šansa da će se model previše prilagoditi skupu za treniranje.

Dropout

Dropout je popularna tehnika u dubokim neuronskim mrežama koja pomaže smanjiti pretreniranje. Ideja je da, tijekom treniranja, nasumično "isključimo" (drop-out) određeni postotak neurona. Ovo sprječava mrežu da previše ovisi o specifičnim neuronima i prisiljava je da nauči generalnije obrasce.

Tijekom treniranja, za svaki korak (batch), isključuje se određeni postotak neurona, npr. 20% ili 50%. Dakle, samo podskup neurona doprinosi treniranju u svakom koraku. Kad je treniranje završeno, svi neuroni su ponovno aktivni prilikom testiranja, ali se njihove težine skaliraju kako bi se kompenzirao dropout. Ako je p postotak neurona koji ostaju aktivni tijekom jednog batcha, svaki neuron h_i se isključuje s vjerojatnošću $1 - p$. Funkcija koja modelira ovo je:

$$h_i' = \begin{cases} 0 & \text{sa vjerojatnošću } 1 - p \\ h_i/p & \text{sa vjerojatnošću } p \end{cases} \tag{4.4}$$

Ovdje je h_i' nova aktivacija neurona nakon dropout-a, a h_i je originalna aktivacija. Da bi se očuvala očekivana vrijednost aktivacija tijekom testiranja, težine se skaliraju s $1/p$. Dropout je odličan jer smanjuje preveliku povezanost između neurona čime model uči generalnije obrasce i smanjuje rizik pretreniranja [41].

Dropout sprječava neuronsku mrežu da previše nauči specifične uzorke iz podataka čime povećava robusnost modela, čineći ga fleksibilnijim i sposobnijim za bolje generaliziranje na nove podatke. Jedna od ključnih prednosti Dropout-a je njegova jednostavna primjena,

jer se samo tijekom treniranja nasumično ”isključuju” neuroni, bez potrebe za promjenom arhitekture mreže

Rano zaustavljanje

Rano zaustavljanje (eng. Early stopping) je tehnika koja sprječava pretreniranje tako što zaustavlja treniranje modela kada performanse na validacijskom skupu prestanu napredovati. Time se izbjegava da model previše nauči specifične uzorke iz skupa za učenje. Ova metoda povećava sposobnost modela da generalizira, jer se treniranje zaustavlja prije nego što model počne previše prilagođavati podatke. Early stopping je jednostavan za implementaciju jer samo prati performanse na validacijskom skupu i zaustavlja treniranje čim primijeti pogoršanje, bez potrebe za dodatnim prilagođavanjem arhitekture modela.

4.3 Metode borbe protiv pretreniranja kod Bayesovih neuronskih mreža

Za razliku od standardnih neuronskih mreža, Bayesovske neuronske mreže zbog svog probabilističkog pristupa učenju prirodno uvode metode za smanjenje pretreniranosti bez potrebe za puno dodatnih intervencija kao kod klasičnih neuronskih mreža. Glavni mehanizmi koji to omogućuju su:

Odabir priora [40]

Jedan od temeljnih načina na koji Bayesove neuronske mreže izbjegavaju pretreniranost je kroz uvođenje priornih distribucija za parametre modela. Priori $p(\mathbf{w})$ odražavaju početne pretpostavke o težinama $\mathbf{w}$, prije nego što se uzmu u obzir podaci.

Regularizirajući učinak priora [43]

Priorne distribucije u Bayesovim neuronskim mrežama djeluju kao oblik regularizacije ograničavajući raspon mogućih vrijednosti za parametre. Dvije najkorištenije prior distribucije, Gaussova i Laplaceova, rade to na sljedeći način:

Gaussovski prior definira distribuciju težina s očekivanjem 0 i varijancom σ^2:

$$p(\mathbf{w}) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{w_i^2}{2\sigma^2}\right), \tag{4.5}$$

gdje je N broj težina, a w_i pojedinačna težina.

Maksimizacija posteriorne distribucije $p(\mathbf{w}|D)$ pod ovim priorom ekvivalentna je minimizaciji negativne log-vjerodostojnosti uz kvadratnu penalizaciju težina:

$$-\ln p(\mathbf{w}) \propto \sum_{i=1}^{N} w_i^2, \tag{4.6}$$

što je identično L2 regularizaciji ($\lambda\|\mathbf{w}\|_2^2$) u klasičnim neuronskim mrežama, gdje je $\lambda = \frac{1}{2\sigma^2}$.

Gaussovski prior favorizira kvadratno manje vrijednosti težina, čime se preferiraju jednostavniji modeli. Ovo smanjuje rizik od prevelikog prilagođavanja na šum u podacima.

Laplaceov prior definira distribuciju težina kao:

$$p(\mathbf{w}) = \prod_{i=1}^{N} \frac{1}{2b} \exp\left(-\frac{|w_i - \mu|}{b}\right), \tag{4.7}$$

gdje su μ očekivanje i b faktor skaliranja.

Maksimizacija posteriora uz ovaj prior ekvivalentna je minimizaciji apsolutnih vrijednosti težina:

$$-\ln p(\mathbf{w}) \propto \sum_{i=1}^{N} |w_i - \mu|, \tag{4.8}$$

što odgovara L1 regularizaciji ($\lambda\|\mathbf{w}\|_1$), gdje je $\lambda = \frac{1}{b}$.

Laplaceova priora potiče "rijetkost" težina, čime dosta težina u modelu završi na 0. Ovo smanjuje složenost modela i sprječava pretreniranost u situacijama s ograničenom količinom podataka.

Utjecaj priora na posterior

Posteriorna distribucija $p(\mathbf{w}|D)$ u Bayesovskim neuronskim mrežama kombinira informacije iz podataka (vjerodostojnost) i naših početnih pretpostavki (prior distribucija):

$$p(\mathbf{w}|D) = \frac{p(D|\mathbf{w})p(\mathbf{w})}{p(D)}. \tag{4.9}$$

Kada je količina podataka mala ili su podaci šumni, vjerodostojnost $p(D|\mathbf{w})$ može postati neinformativna. U tim slučajevima, prior $p(\mathbf{w})$ dominira posteriornom distribucijom, stabilizirajući procjenu težina.

Recimo da je prior Gausovski ($p(\mathbf{w}) = \mathcal{N}(0, \sigma^2)$). On tada penalizira ekstremne težine čak i kad vjerodostojnost favorizira prilagodbu šumu u podacima [43]. Posterior postaje manje osjetljiv na neinformativne ili šumne podatke jer prior „vuče" težine prema nuli. U konačnici, s obzirom na broj podataka ponašanje u učenju izgleda ovako:

- **Velika količina podataka**: Vjerodostojnost dominira nad priorom, a posterior se usklađuje s podacima:

$$p(\mathbf{w}|D) \approx p(D|\mathbf{w}). \tag{4.10}$$

- **Mala količina podataka**: Prior dominira nad vjerodostojnosti, ograničavajući ekstremne vrijednosti težina:

$$p(\mathbf{w}|D) \approx p(\mathbf{w}). \tag{4.11}$$

Kombinacija priora i vjerodostojnosti osigurava da model ne prilagođava previše šumne podatke te da težine ostanu unutar granica koje postavlja prior. Ovo pomaže u postizanju boljeg balansa između prilagodbe podacima i generalizacije.

Marginalizacija težina

Za razliku od klasičnih neuronskih mreža, gdje se koristi jedna točkasta procjena težina $\mathbf{w}^*$ (najčešće dobivena optimizacijom), Bayesove neuronske mreže koriste marginalizaciju težina kako bi dobili predikcije [14]:

$$p(\mathbf{y}|\mathbf{x}, D) = \int p(\mathbf{y}|\mathbf{x}, \mathbf{w}) p(\mathbf{w}|D)\, d\mathbf{w}. \tag{4.12}$$

Ovaj postupak integrira preko svih mogućih vrijednosti težina, skaliranih njihovom vjerojatnošću prema posterioru. Marginalizacija sprječava pretreniranost na sljedeće načine.

- **Smanjenje osjetljivosti na pojedinačne uzorke**: Umjesto da se koristi jedan skup težina, marginalizacija uzima prosječnu predikciju preko svih težina. Ekstremni (često šumni) podaci nemaju velik utjecaj na model s obzirom da posterior preferira modele koji bolje opisuju cijeli skup podataka.

- **Učinkovitije iskorištavanje podataka**: Slično primjeru opisanom ranije, čak i uz jako malu količinu podataka prior indirektno (preko utjecaja na posterior) utječe na predikcije i ograničava ekstremne vrijednosti čime poboljšava generalizaciju.

Kvantificiranje nesigurnosti

Kod običnih neuronskih mreža pretreniranje je najčešće prevelika sigurnost mreže u područjima gdje je mreža (zbog manjka podataka ili prevelikog šuma) slabo naučila generalizirati. S druge strane, Bayesove neuronske mreže imaju prirodno rješenje ovog problema iz svoje mogućnosti kvantificiranja nesigurnosti [19]:

- **Epistemološka nesigurnost**: Reflektira nesigurnost u parametrima modela $\mathbf{w}$, čiji je razlog manjak podataka. U regijama gdje podaci nisu dostupni, epistemološka nesigurnost se povećava, što rezultira širim intervalima pouzdanosti i manje ekstremnim predikcijama.
- **Aleatorička nesigurnost**: Modelira šum ugrađen u podatke čime se izbjegava prevelika prilagodba pojedinačnim uzorcima.

Kombinacija ove dvije vrste nesigurnosti omogućava Bayesovim neuronskim mrežama da u područjima visokog rizika od pretreniranja izbjegnu preveliku sigurnost u svoje procjene.

Dropout kao aproksimacija Bayesovskog učenja

Iako prave Bayesovske neuronske mreže zahtijevaju kompleksne i računski ”skupe” postupke za procjenu posteriora, u praksi se često koriste aproksimacijske metode poput Monte Carlo Dropout-a (MC Dropout). Dropout, tehnika obično korištena kao regularizacija u neuronskim mrežama, može se interpretirati kao aproksimacija varijacijskog zaključivanja za Bayesove neuronske mreže [9].

- **Stohastičko ponašanje**: Tijekom inferencije, višestruko nasumično uzorkovanje težina zapravo implicitno aproksimira posterior $p(\mathbf{w}|D)$.
- **Kvantifikacija nesigurnosti**: MC Dropout omogućuje izračunavanje varijance predikcija čime se dobija informacija o nesigurnosti procjene.

Ovaj pristup smanjuje pretreniranost jer uvodi stohastičnost u proces učenja i inferencije, slično pravim Bayesovim neuronskim mrežama.
Pokazano je da su koncepti poput odabira priora, marginalizacije i kvantificiranja nesigurnosti dovoljne da pokriju sve metode regularizacije i dodaju ono malo više što fali običnim neuronskim mrežama - nositi se s ograničenim i šumnim podacima. Uzrok toga je upravo probabilistički (Bayesov) pristup arhitekturi i treniranju Bayesovih neuronskih mreža koji omogućuje bolju generalizaciju od klasičnih neuronskih mreža.

Poglavlje 5

Primjena Bayesovih neuronskih mreža

5.1 Eksperiment na *Wine quality* skupu podataka

Uvod u skup podataka

Prvi skup podataka korišten u ovom radu dolazi iz javno dostupnog *Wine Quality* skupa podataka. Ovaj skup podataka sadrži fizikalno-kemijske značajke vina te njihovu subjektivnu ocjenu kvalitete označenu kao $y \in \{0, 1, \ldots, 10\}$. Modeliranje ovog skupa podataka posebno je zanimljivo zbog prisutnosti nesigurnosti u ocjenama, koje su subjektivne i mogu varirati između ocjenjivača. U skupu podataka nalazi se ukupno $n = 1599$ uzoraka crnog vina i $d = 11$ značajki. Predviđanje ocjene kvalitete je iznimno izazovno zbog neravnoteže u podacima. Na primjer, većina ocjena se nalazi u rasponu od 5 do 7, dok su ekstremne ocjene rijetke. Ovo čini skup idealnim za primjenu Bayesovih neuronskih mreža, koje pružaju dodatne informacije o nesigurnosti u predikcijama.

Podjela skupa podataka

Za učinkovito treniranje i evaluaciju modela, podaci su podijeljeni na 3 dijela:

- **Trening skup** (D_{train}): Koristi se za optimizaciju parametara modela. Sadrži 80% ukupnih podataka, odnosno $n_{train} = 1279$ uzoraka.
- **Validacijski skup** (D_{val}): Koristi se za procjenu performansi modela tijekom treniranja i odabir hiperparametara. Sadrži 10% ukupnih podataka, odnosno $n_{val} = 160$ uzoraka.
- **Testni skup** (D_{test}): Koristi se za procjenu konačnih performansi modela na neviđenim podacima. Sadrži preostalih 10% podataka, odnosno $n_{test} = 160$ uzoraka.

Za dani skup podataka D veličine n se bira nasumični podskup veličine $0.8n$ koji predstavlja skup za treniranje, dok se preostalih $0.2n$ podataka nasumično razdvaja na skupove za validaciju i testiranje, oba veličine $0.1n$. Nasumično dijeljenje skupa podataka provodi se kako bi svaki primjerak iz skupa podataka imao jednaku šansu biti dodijeljen u bilo koji od podskupova, čime se minimizira potencijalna pristranost u treniranju. Podjela skupa podataka osigurava da se performanse modela mjere na neviđenim podacima izvan skupa za treniranje. Validacijski skup omogućuje prilagodbu hiperparametara modela i praćenje napretka treniranja, dok testni skup pruža mjeru njegove sposobnosti generalizacije. [11]

Podjela skupa podataka je iznimno važna u pristupima poput Bayesovskih neuronskih mreža gdje se osim točkovne procjene ciljne varijable $\hat{\mathbf{y}}$ dobijemo i procjenjenu distribuciju nesigurnosti $p(\hat{\mathbf{y}}|x)$. Validacijski i testni skup ne služe više samo za evaluaciju procjene već ih je moguće iskoristiti i za evaluaciju nesigurnosti.

Istraživanje skupa podataka

Skup podataka sadrži sljedeće varijable:

Varijabla	**Tip**	**Značenje**
fixed acidity	neprekidna	Koncentracija nehlapljivih kiselina
volatile acidity	neprekidna	Koncentracija octene kiseline
citric acid	neprekidna	Koncentracija limunske kiseline
residual sugar	neprekidna	Količina šećera preostala nakon fermentacije
chlorides	neprekidna	Koncentracija soli
free sulfur dioxide	neprekidna	Slobodni oblik SO_2 u vinu
total sulfur dioxide	neprekidna	Ukupna koncentracija SO_2
density	neprekidna	Gustoća vina
pH	neprekidna	Razina kiselosti vina
sulphates	neprekidna	Koncentracija sulfata
alcohol	neprekidna	Sadržaj alkohola u vinu
quality	cjelobrojna	Ocjena kvalitete vina (ciljana varijabla)

Tablica 5.1: Pregled varijabli u skupu podataka

Na prvi pogled očekujemo da se informacije dobivene iz nekih od ovih varijabli poklapaju s obzirom da nekoliko varijabli predstavlja neku razinu kiselosti, a nekoliko neki oblik koncentracije sulfata.

Osnovne deskriptivne statistike prikazane su u sljedećoj tablici:

Varijabla	**Min**	**Max**	**Prosjek**	**Std**
fixed acidity	4.60	15.90	8.32	1.74
volatile acidity	0.12	1.58	0.53	0.18
citric acid	0.00	1.00	0.27	0.19
residual sugar	0.90	15.50	2.54	1.41
chlorides	0.012	0.611	0.087	0.047
free sulfur dioxide	1.00	72.00	15.87	10.46
total sulfur dioxide	6.00	289.00	46.47	32.90
density	0.99007	1.00369	0.99675	0.00189
pH	2.74	4.01	3.31	0.15
sulphates	0.33	2.00	0.66	0.17
alcohol	8.40	14.90	10.42	1.07
quality	3.00	8.00	5.64	0.81

Tablica 5.2: Osnovne deskriptivne statistike

Normalizacija varijabli [24]

Sve ulazne varijable bit će normalizirane kako bi se osigurala stabilnost i učinkovitost optimizacije tijekom treniranja modela. Normalizacija transformira vrijednosti varijable tako da imaju srednju vrijednost 0 i standardnu devijaciju 1, prema formuli:

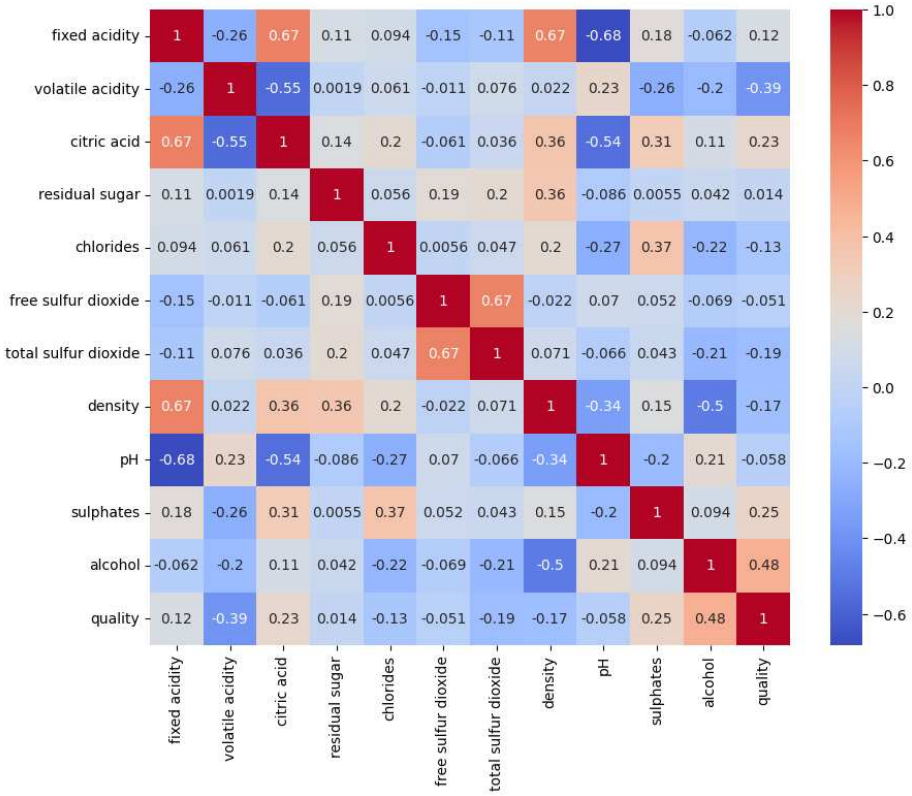
$$x_{\text{norm}} = \frac{x - \mu}{\sigma},$$

gdje x označava originalnu vrijednost, μ srednju vrijednost varijable, a σ standardnu devijaciju varijable.

Normalizacija ima ključnu ulogu u postizanju boljih rezultata treniranja modela pomažući algoritmima gradijentnog spusta da brže konvergiraju s obzirom da ulazne vrijednosti imaju usporedive skale. Također, usporedive skale smanjuju rizik od numeričkih nestabilnosti te omogućuju ravnopravnije učenje među varijablama.

Vizualizacije

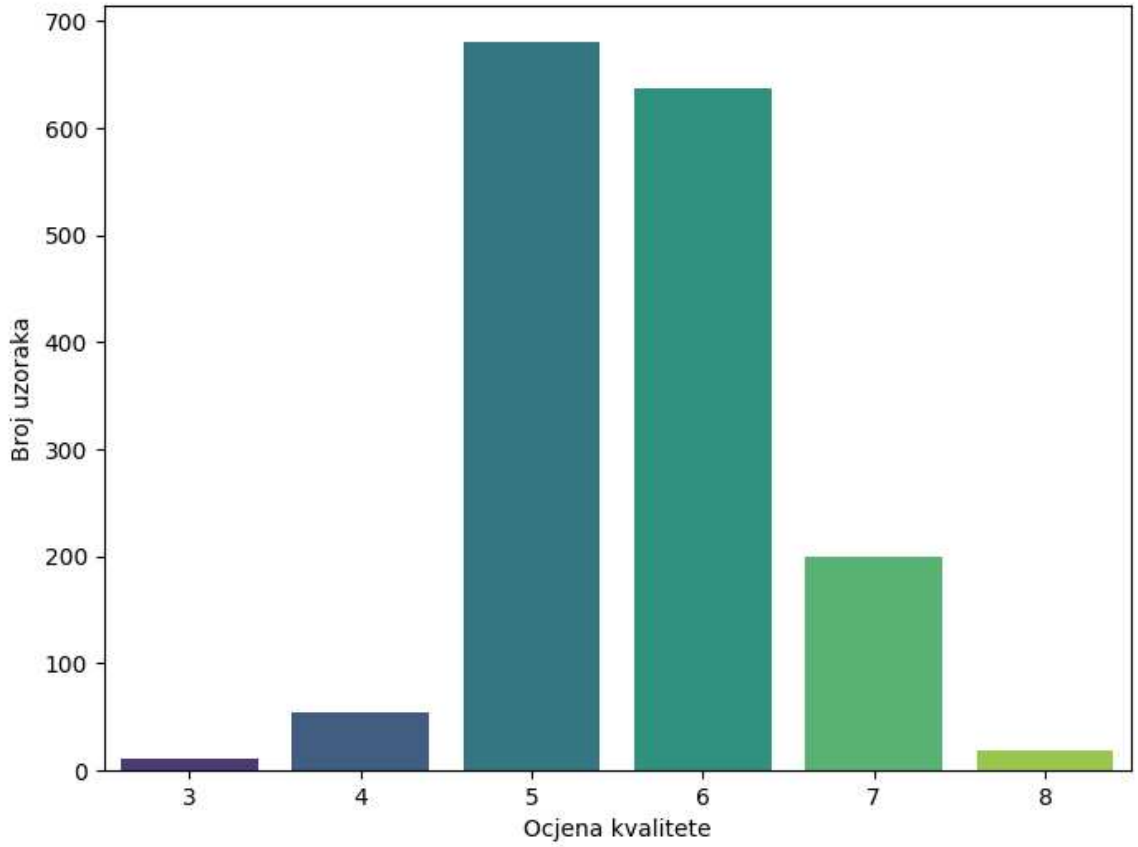
Navodimo i dvije korisne vizualizacije kojima možemo dobiti dublji uvid u skup podataka. Korelacijska matrica daje grubi uvid u povezanost između varijabli, što može biti jako korisno u modelima koji ne trpe koreliranost među nezavisnim varijablama:



Slika 5.1: Korelacijska matrica varijabli

Na slici vidimo nekoliko jakih korelacija od kojih neke, poput korelacije pH vrijednosti i nekih kiselina, možemo i laički objasniti - pH vrijednost je po definiciji mjera kiselosti neke otopine ili smjese. Jakost robusnih modela poput neuronskih mreža dolazi do izražaja upravo u ovakvim situacijama gdje ne moramo birati podskup dostupnih varijabli da izbjegnemo koreliranost nezavisnih varijabli čime gubimo informacije iz skupa podataka.

Na sljedećoj slici je prikazana distribucija ocjena kvalitete (zavisna varijabla):



Slika 5.2: Distribucija ocjena kvalitete vina

Iz vizualnog pregleda je vidljiva neravnoteža ocjena te njihova koncentriranost oko vrijednosti 5, 6 i 7. Vidimo da ekstremne ocjene poput 1, 2, 9 i 10 uopće ne postoje u skupu podataka, a ocjena 3 i 8 je jako malo. Ovako mala frekvencija pojedinih vrijednosti ciljane varijable uvodi rizik pretreniranosti.

Metode

U svrhu modeliranja skupa podataka koristit će se dvije vrste modela: klasična neuronska mreža i Bayesova neuronska mreža. Prvo, kroz proces treniranja cilj je pokazati da su efekti regularizacijskih tehnika u običnim neuronskim mrežama implicitno uključeni kroz koncept Bayesovih neuronskih mreža. Također, kroz proces evaluacije modela na tes-

tnom skupu cilj je, osim poželjno boljih rezultata (u smislu bilo koje metrike), pokazati mogućnost kvantifikacije nesigurnosti na nekoliko primjera iz skupa za testiranje.

Za početak ćemo naučiti model linearne regresije na danom skupu za treniranje kako bi dobili neki minimalni model s kojim možemo usporediti ostale kompleksnije modele u smislu točnosti. Srednje kvadratna greška linearnog modela na skupu za testiranje iznosi 0.422 gdje su svi parametri osim gustoće vina statistički značajni.

Standardna neuronska mreža

Standardna neuronska mreža implementirana je kao sekvencijalni model s dva skrivena sloja, svaki s 64 neurona i funkcijom aktivacije *ReLU*. Izlazni sloj sadrži jedan neuron koji daje predviđenu vrijednost ciljne varijable. S obzirom da zavisna varijabla poprima jednu od 11 vrijednosti (0 do 10), problem bi se mogao shvatiti kao problem klasifikacije. S druge strane, između tih 11 klasa postoji jasno definiran uređaj, pa je prirodno problem pretvoriti u problem regresije. Treniranje se izvršava standardnom metodom propagacije unatrag gdje je algoritam optimiziranja *Adam* [21], s početnim koeficijentom učenja $\eta = 0.001$ i funkcijom gubitka definiranoj kao srednja kvadratna pogreška.

Bayesova neuronska mreža

Bayesova neuronska mreža koristi isti arhitekturni raspored kao standardna mreža uz jednu važnu razliku - zadnji sloj mreže ima dimenziju izlaza 2, gdje jedan izlaz predstavlja predviđenu srednju vrijednost, a drugi varijancu ciljane varijable. Ovaj pristup koristimo nastavno na (3.30) u pokušaju da modeliramo aleatoričku nesigurnost. U većini problema koji rade na podacima iz stvarnog svijeta je dobro pokušati modelirati aleatoričku nesigurnost s obzirom da šum koji je ugrađen u podatke je najčešće heteroskedastičan [33]. Ipak, u samoj arhitekturi se ne modelira točno varijanca ciljane varijable već log-varijanca. Ovaj pristup donosi brojne prednosti kao što su osiguravanje pozitivnosti varijance, lakši pristup širokim rasponima varijance te stabilizacija računanja reziduala i gradijenata [23]. Mreža također uključuje Bayesovske pristupe za modeliranje težina kao slučajnih varijabli kako je opisano ranije, a ostali detalji vezani uz Bayesovski pristup su sljedeći:

- **Priorna distribucija težina**: Za modeliranje priora koristila se normalna distribucija $N(0, 1)$, s obzirom da nema posebnih pretpostavnih i saznjanja o njihovim vrijednostima. Osim prirodnog uvođenja regularizacije opisanog ranije, ovaj prior ne nameće prevelika ograničenja kod učenja parametara. Isto tako, ovaj posterior može biti prilagođen zadatku smanjivanjem ili povećavanjem varijance. [28]

- **Aproksimacijska distribucija posteriora**: Posteriorna distribucija aproksimirana je tzv. *faktoriziranim Gaussovim posteriorom* koji pretpostavlja da su sve težine

nezavisne i da svaka slijedu vlastitu Gaussovu distribuciju. Ovo znači da se posterior aproksimira produktom nezavisnih Gaussovih distribucija što uvelike smanjuje složenost modela i omogućuje učinkovitiju optimizaciju čak i kod velikih mreža te skupova podataka. [22]

Treniranje provodimo varijacijskim zaključivanjem, gdje se optimizira varijacijski donji rub uz male preinake u odnosu na (3.26):

$$\mathcal{L} = -\mathbb{E}_{q(\mathbf{w}|\boldsymbol{\phi})}[\ln p(D|\mathbf{w})] + \beta \cdot \mathrm{KL}(q(\mathbf{w}|\boldsymbol{\phi}) \,||\, p(\mathbf{w})) + \lambda. \tag{5.1}$$

gdje je faktor β uveden kao faktor skaliranja KL divergencije koji se u pravilu uzima kao $\frac{1}{|D_{train}|}$ [17], a λ dio koji regularizira log-varijancu i time dodatno osigurava stabilnost učenja i raspon veličina log-varijance [36]. Od već poznatih izraza tu je $q(\mathbf{w}|\boldsymbol{\phi})$ kao aproksimacijska distribucija posteriora, $p(\mathbf{w})$ kao priorna distribucija, te KL divergencija koja mjeri odstupanje između posteriora i priora. Optimizacija se provodi algoritmom *Adam*, s početnim koeficijentom učenja $\eta = 0.001$. Inferencija se provodi pomoću Monte Carlo uzorkovanja iz posteriorne distribucije težina. Tijekom predikcije, višestruka uzorkovanja omogućuju izračunavanje srednje vrijednosti predikcija i procjenu epistemološke nesigurnosti, što je uz već modeliranu aleatoričku nesigurnost ključna razlika Bayesovih neuronskih mreža u odnosu na klasične.

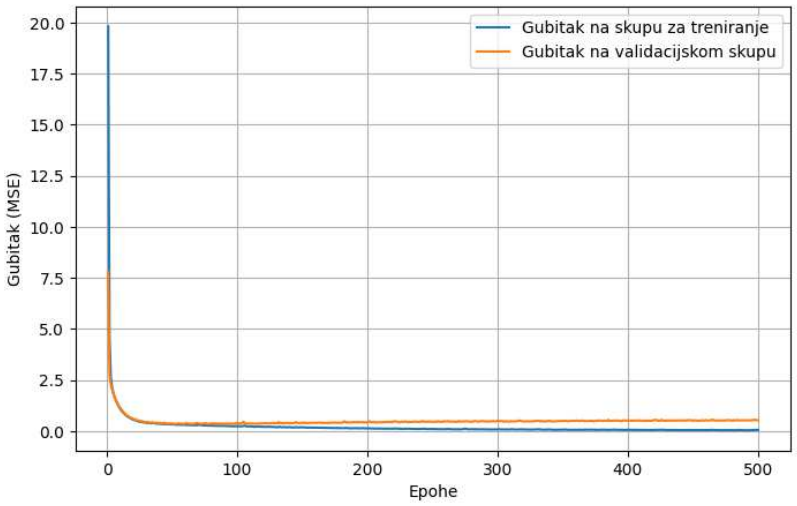
Treniranje modela

Proces treniranja organiziran je kroz epohe koje imaju ključnu ulogu u optimizaciji modela. Epoha predstavlja jedan potpuni prolazak kroz cijeli skup za treniranje. S jedne strane, nedovoljan broj epoha može uzrokovati nedovoljno učenje modela, a s druge strane, preveliki broj epoha može uzrokovati pretreniranost, pri čemu model postaje previše prilagođen podacima za treniranje i gubi sposobnost generalizacije [6].

Tijekom treniranja, mjerimo funkcije gubitka na skupu za treniranje i skupu za validaciju kako bi se pratila konvergencija modela i izbjegla pretreniranost. Praćenje pogreške na skupu za validaciju ključno je za identificiranje optimalnog broja epoha. Kada pogreška na skupu za validaciju prestane opadati ili počne rasti, to ukazuje na početak pretreniranosti. Takva situacija može zahtijevati ranu obustavu treniranja (eng. *early stopping*) kako bi se osigurale bolje performanse modela na neviđenim podacima. Ovakav odabir epoha je primjenjiv na obje vrste modela - standardne te Bayesove neuronske mreže.

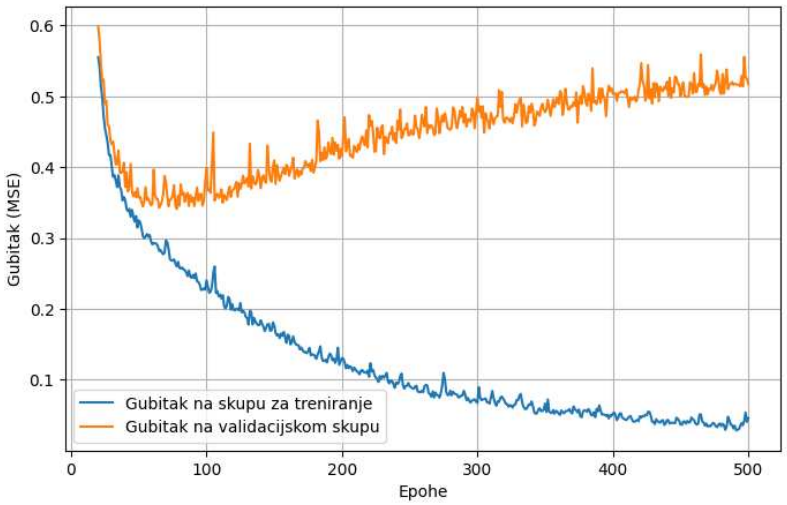
Rezultati klasične neuronske mreže

Da bi bolje ilustrirali koncept treniranja i epoha, treniramo običnu neuronsku mrežu kroz 500 epoha u svrhu identifikacije optimalnog broja epoha. Gubitak (MSE) na skupu za treniranje i validacijskom skupu kroz 500 epoha je prikazan na grafu:



Slika 5.3: Gubitak na trening i validacijskom skupu obične neuronske mreže kroz 500 epoha

Nagli pad gubitka na početku pokazuje tipično ponašanje u treniranju neuronskih mreža - mreža jako brzo nauči glavne obrasce u podacima. Vizualno ćemo jasnije moći identificirati optimalan broj epoha iz grafa izuzmemo prvih nekoliko epoha s velikim gubitkom:

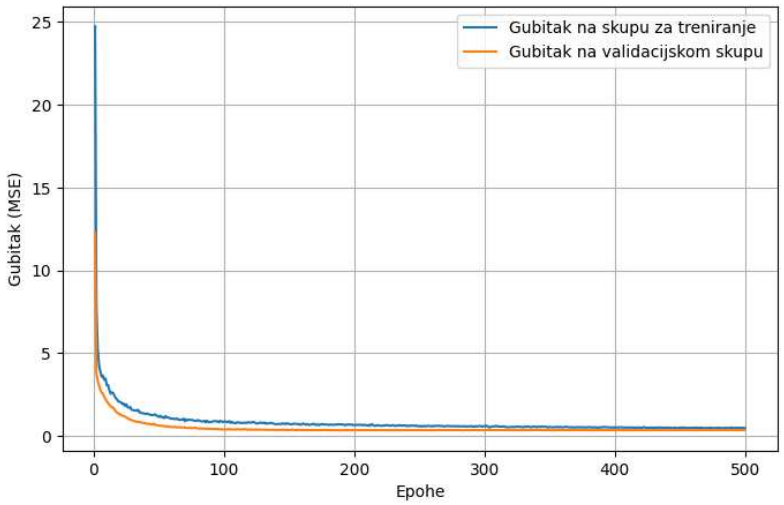


Slika 5.4: Gubitak na trening i validacijskom skupu obične neuronske mreže kroz epohe 20-500

Vizualnom inspekcijom vidimo da se gubitak na validacijskom skupu počinje povečavati nakon 50 epoha, te uzimamo 50 epoha kao optimalan broj epoha za treniranje modela. Nakon evaluacije oba modela na skupu za testiranje dobijamo da model treniran kroz 500 epoha ima srednje kvadratnu grešku 0.5317, dok model treniran kroz 50 epoha ima srednje kvadratnu grešku 0.3804. Ovime je pokazano da se model nakon 50 epoha počinje previše prilagođavati skupu za treniranje (greška na skupu za treniranje nastavlja padati), dok, s druge strane, gubi sposobnost generalizacije (greška na validacijskom skupu raste). Ovime smo zapravo proveli regularizacijsku metodu ranog zaustavljanja [34].

Rezultati regularizirane neuronske mreže

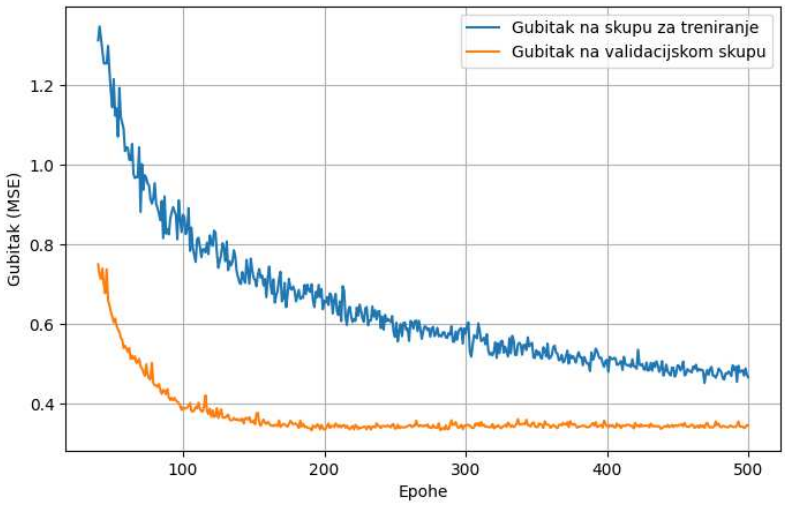
Sada istu neuronsku mrežu regulariziramo kombinacijom L2 regularizacije i Dropout tehnike. L2 regularizacija penaliziralo velike vrijednosti težina čime smanjuje složenost modela, dok Dropout tehnika nasumično isključuje 30% neurona čime smanjuje zavisnost između njih i dodatno smanjuje rizik od pretreniranja. Gubitak na trening i validacijskom skupu je na sljedećem grafu:



Slika 5.5: Gubitak na trening i validacijskom skupu regularizirane neuronske mreže kroz 500 epoha

U ovom slučaju vidimo da je gubitak na skupu za treniranje kroz epohe veći od gubitka na validacijskom skupu, čime se vidi sposobnost regularizirane mreže da odmah počne učiti generalne obrasce i ignorirati šum, a ne se prilagođavati isključivo skupu za treniranje.

Kada izbacimo prve epohe s velikim gubitkom imamo sljedeće:



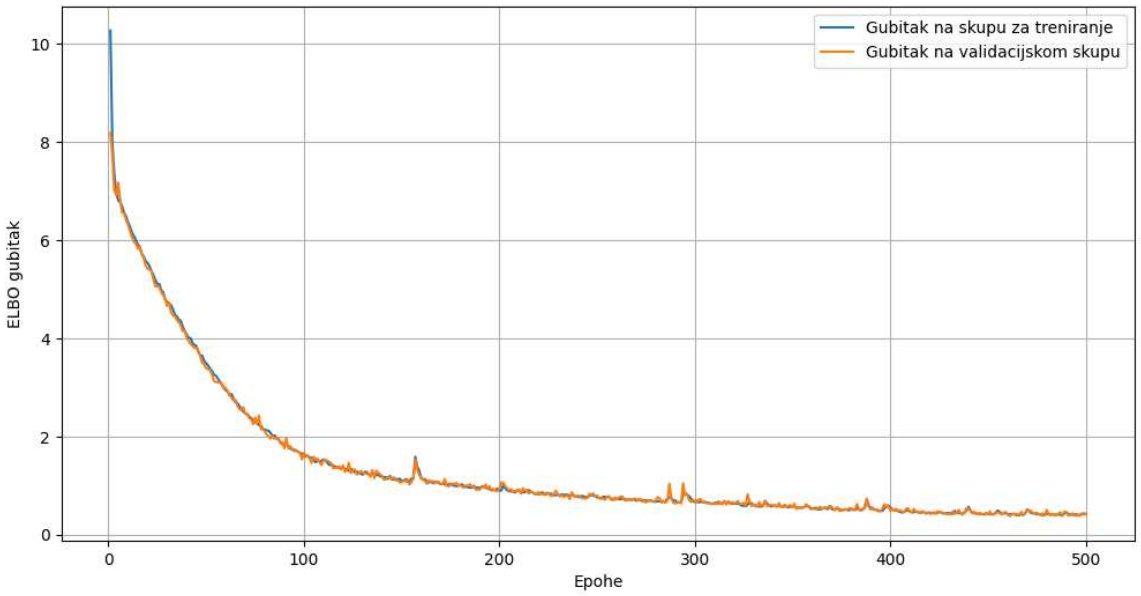
Slika 5.6: Gubitak na trening i validacijskom skupu regularizirane neuronske mreže kroz epohe 40-500

Za optimalan broj epoha uzimamo 200, ali uz uočavanje glavne razlike - greška na validacijskom skupu u ovom slučaju ne raste nakon 200te epohe već ostaje ista. Ovo ponašanje je pokazatelj snage regularizacije u borbi protiv pretreniranja. Nakon što je mreža naučila sve što može naučiti iz podataka više se ne prilagođava dodatno skupu za treniranje čime bi pogoršala sposobnost generalizacije. Nakon evaluacije oba modela na skupu za testiranje dobijamo da model treniran kroz 500 epoha ima srednje kvadratnu grešku 0.3674, dok model treniran kroz 200 epoha ima srednje kvadratnu grešku 0.3629.

Iz ovih rezultata je vidljiva prednost regularizacije u treniranju neuronskih mreža. Za početak, dobro regularizirana mreža nam omogućava da samo jednom treniramo model kroz proizvoljno velik broj epoha bez straha da će predugo treniranje utjecati na sposobnost generalizacije modela. Također, bolji rezultati na skupu za testiranje su pokazatelj bolje mogućnosti generalizacije regularizirane neuronske mreže. S druge strane, regularizirana neuronska mreža je trebala puno više epoha da dođe do optimalnih preformansi što se svakako očituje u vremenu i resursima potrebnim za treniranje.

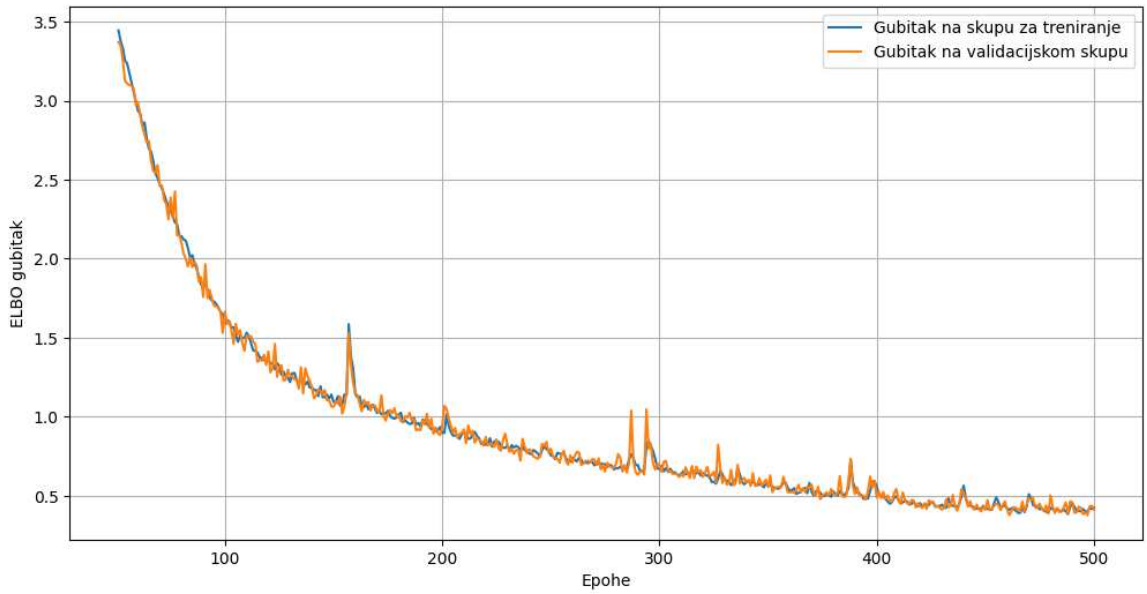
Rezultati Bayesove neuronske mreže

Za razlike od obične neuronske mreže, kod Bayesovske neuronske mreže u procesu treniranja pratimo varijacijski donji limit na validacijskom i skupu za treniranje s obzirom da je to vrijednost koja se minimizira u procesu treniranja. Očekujemo postići slične rezultate u smislu regularizacije - treniranje i na prevelikom broju epoha neće pogoršati generalizacijsku sposobnsot modela. Također, očekujemo da ćemo u smislu točnosti, koju ćemo kasnije evaluirati kroz srednje kvadratnu grešku na skupu za testiranje, dobiti slične ili bolje rezultate. Ono što je jedinstveno kod Bayesove neuronske mreže u odnosu na obične neuronske mreže je sposobnost kvantifikacije nesigurnosti. Usporedit ćemo točkovnu predikciju obične neuronske mreže kod par primjera na skupu za testiranje s procjenom Bayesove neuronske mreže gdje procjena Bayesove neuronske mreže pokazuje visoku razinu nesigurnosti. Gubitak na skupu za treniranje i validacijskom skupu je prikazan na sljedećem grafu:



Slika 5.7: Gubitak na trening i validacijskom skupu Bayesove neuronske mreže kroz 500 epoha

Za razliku od klasičnih neuronskih mreža, vidi se puno veća varijacija gubitka kroz epohe što ne čudi s obzirom na stohastičku prirodu modela. Također, vidi se i sporija brzina konvergencije u odnosu na klasične neuronske mreže. Kada izbacimo prve epohe s velikim gubitkom imamo:



Slika 5.8: Gubitak na trening i validacijskom skupu Bayesove neuronske mreže kroz epohe 50-500

Na ovom prikazu je još vidljivija varijacija gubitka kroz epohe, a pred kraj vidimo regularizacijski učinak Bayesove neuronske mreže - gubitak na validacijskom skupu ne divergira od gubitka na skupu za treniranje te se počinje stabilizirati.

Za predikciju na skupu za testiranje potrebno je uzorkovati iz posteriorne distribucije težina naučene tijekom treniranja s obzirom da mreža ne generira točkovnu predikciju kao obična neuronska mreža već za svaki ulaz generira distribuciju izlaza. Za svaku ulaznu vrijednost mreža se ”pokreće” više puta (težine se uzorkuju iz posteriorne distribucije) čime se generiraju višestruke predikcije. Nakon dovoljnog broja koraka (npr. 100), uprosječena vrijednost svih predikcija srednje vrijednosti se koristi kao konačna predikcija dok se standardna devijacija tih 100 srednjih vrijednosti uzima kao mjera epistemološke nesigurnosti za svaki primjer iz skupa za testiranje. S druge strane, drugi izlaz predikcije je log-varijanca iz koje pravilnom agregacijom dobijemo mjeru aleatoričke nesigurnosti. Formalnije, rezultati koje dobijemo za $N = 100$ primjeraka mreže su:

$$\hat{y} = \frac{1}{100}\sum_{i=1}^{100}\mu_i \tag{5.2}$$

$$\sigma_{\mathrm{e}} = \sqrt{\frac{1}{100}\sum_{i=1}^{100}(\mu_i - \hat{y})^2} \tag{5.3}$$

$$\sigma_{\mathrm{a}} = \sqrt{\frac{1}{100}\sum_{i=1}^{100}\exp(\log\sigma_i^2)} \tag{5.4}$$

$$\sigma = \sqrt{\sigma_{\mathrm{e}}^2 + \sigma_{\mathrm{a}}^2} \tag{5.5}$$

gdje su μ_i i σ_i procijenjena srednja vrijednost i standardna devijacija i-tog primjerka mreže, $\hat{y}$ konačna predikcija srednje vrijednosti, σ_{e} epistemološka nesigurnosti, σ_{a} aleatorička nesigurnost te σ ukupna nesigurnost.

Srednje kvadratna greška modela treniranog kroz 500 epoha je 0.4025, čime vidimo neznatno pogoršanje u točnosti u odnosu na prijašnja 2 modela klasičnih neuronskih mreža. U sljedećoj tablici vidimo srednje kvadratne greške sva 4 modela:

Model	**MSE**
Linearna regresija	0.422
Klasična neuronska mreža	0.3804
Regularizirana neuronska mreža	0.3674
Bayesova neuronska mrežea	0.4025

Tablica 5.3: Vrijednosti MSE za različite modele

Vidimo da uvodeći mnogo kompleksnije modele od linearne regresije nismo znatno napredovali u smislu točnosti. Pokušajmo pokazati prednosti Bayesove neuronske mreže kroz sposobnost kvantifikacije nesigurnosti.

Primjeri nesigurnosti

Uzmimo nekoliko primjeraka iz skupa za testiranje gdje Bayesova neuronska mreža daje visoku razinu nesigurnosti, skupa s procijenjenim vrijednostima svih modela, stvarnom vrijednosti te vrijednostima nezavisnih varijabli. Vrijednosti nezavisnih varijabli ćemo ostaviti u normaliziranom obliku tako da se na prvi pogled može očitati koliko su ekstremne bez referiranja na deskriptivne statistike. U tablici ispod navodimo primjerke s najvećom ukupnom nesigurnosti, najvećom epistemološkom nesigurnosti te najvećom aleatoričkom nesigurnosti:

Primjer	**1**	**2**	**3**
Ukupna nesigurnost	1.761	1.219	1.398
Epistemološka nesigurnost (šum)	0.622	0.473	0.429
Aleatorička nesigurnost (parametri)	1.648	1.123	1.330
Stvarna ocjena kvalitete	6	6	8
Procjena ocjene BNN	5.71	5.91	6.07
Procjena ocjene regularizirane ANN	5.81	5.76	6.26
Procjena ocjene klasične ANN	5.67	5.99	6.18
Konc. nehlapljivih kiselina	-1.406	-1.696	0.044
Konc. octene kiseline	-1.342	1.169	-0.896
Konc. limunske kiseline	-0.115	-1.395	1.318
Količina šećera	7.556	-0.944	-0.526
Konc. soli	-0.435	-0.962	6.581
Slobodni oblik SO_2 u vinu	5.444	0.012	-0.667
Ukupna konc. SO_2	3.442	-0.020	-0.627
Gustoća	0.235	-2.261	0.607
Razina kiselosti	0.119	4.536	-1.635
Konc. sulfata	-0.688	-0.401	2.292
Sadržaj alkohola	-0.112	1.978	-1.253

Tablica 5.4: Primjerci s najvećom nesigurnosti

Odmah se uočava da primjeri s visokom razinom nesigurnosti bilo kakve vrste u nekim nezavisnim varijablama imaju ekstremne vrijednosti. Ovakvo ponašanje ne čudi s obzirom na to da primjerci s ekstremnim vrijednostima u prostoru nezavisnih varijabli imaju manje manje bližih podataka, pa su samim time i to područja s manjkom podataka za treniranje. Primjerci u takvim područjima po definiciji imaju visoku razinu epistemološke nesigurnosti, no ne nužno i aleatoričke. Kod definiranja modela se često šum modelira samo na zavisnoj varijabli (ocjena kvalitete), no u stvarnim primjerima je pokazano da i nezavisne varijable često imaju šum koji onda utječe na aleatoričku nesigurnost. S obzirom na to da

je dani skup podataka rezultat raznih fizikalno-kemijskih mjerenja, očekivano je da i nezavisne varijable sa sobom nose određeni šum, pogotovo one koje imaju mnogo ekstremnih vrijednosti. Dakle, nije začuđujuće da primjeri s visokom razinom epistemološke nesigurnosti nose sa sobom i još višu razinu aleatoričke nesigurnosti [42]. Također, aleatorička nesigurnost ovisi o veličini skupa za treniranje u odnosu na broj parametara modela što bi moglo sugerirati da je skup za treniranje premalen za ovakvu arhitekturu mreže. Uistinu, 1279 primjeraka za treniranje je u teoriji premalo za treniranje neuronske mreže s ovoliko parametara [1].

Kada bismo laički pokušali objasniti značenje nesigurnosti u kontekstu ovog problema, mogli bismo reći da aleatorička nesigurnost za pojedini primjerak predstavlja kombinaciju varijabilnosti u ocjenama kvalitete (model smatra da su ocjene za primjerak s takvim vrijednostima varijabli veoma različite) te varijabilnosti u nezavisnim varijablama (model ne daje veliku težinu vrijednostima varijabli koje dobije, misleći da su jako šumovite u tim regijama). S druge strane, kroz epistemološku nesigurnost za pojedine primjerke model pokušava reći da kroz proces treniranja nije susreo velik broj primjeraka s takvim vrijednostima nezavisnih varijabli te stoga nije siguran u svoju procjenu (nesigurnost u parametre).

Zanimljiv je primjerak br. 3 na kojem se vidi da vino s velikom ocjenom kvalitete (koja je vrlo rijetka) nosi veliku razinu nesigurnosti, što je i za očekivati s obzirom na to da je takvih podataka u skupu za treniranje bilo jako malo. Taj primjerak također nosi sa sobom i veliku grešku (općenito su predikcije jako koncentrirane između vrijednosti 5 i 6), što bi moglo sugerirati da se model jako usmjerio na srednju vrijednost cijelog skupa za treniranje. Ovo ponašanje je primjer nedovoljne naučenosti modela kojoj se pristupa povećavanjem složenosti modela ili povećavanjem skupa za treniranje [11].

Kroz same procjene se vidi manjkavost ovog skupa podataka za evaluaciju modela strojnog učenja. Naime, vidjeli smo da velika većina ocjena kvalitete iznosi 5 ili 6 (čemu se model možda i previše prilagodio), čime smo na samom početku ograničeni što se tiče bilo kakve ozbiljnije evaluacije modela po pitanju točnosti. Šteta je što vrijednosti zavisne varijable nisu neprekidne i malo više raspršene, gdje bismo onda mogli pokazati razinu nesigurnosti za primjerke s ekstremnim ocjenama.

U konačnici, čini se da kompleksni modeli poput neuronskih mreža nisu idealni za dani skup podataka zbog njegove veličine i nekvalitete. Unatoč tome, uspjeli smo pokazati barem neke od prednosti Bayesovih neuronskih mreža. Kroz treniranje modela je pokazana sposobnost "samoregularizacije" i izbjegavanja overfittinga, a kroz analizu izlaza neuronske mreže i sposobnost kvantificiranja kako aleatoričke tako i epistemološke nesigurnosti. U nastavku pokušavamo naučiti klasičnu i Bayesovu neuronsku mrežu na malo većem skupu podataka s ciljem boljeg i jasnijeg prikaza prednosti Bayesove neuronske mreže u odnosu na klasičnu.

5.2 Eksperiment na *MNIST* skupu podataka

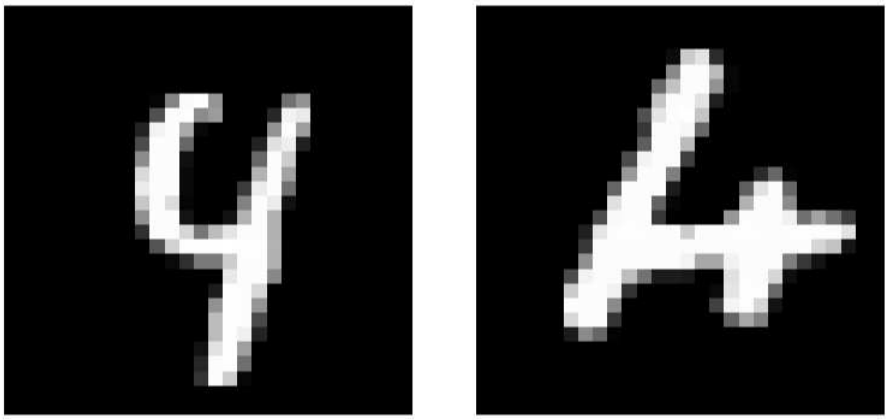
O skupu podataka i klasifikaciji

MNIST skup podataka (eng. *Modified National Institute of Standards and Technology dataset*) jedan je od poznatih i često korištenih skupova podataka u području računalnog vida i strojnog učenja. Ovaj skup sadrži slike rukom pisanih znamenki (brojeva od 0 do 9) u sivim tonovima, gdje je svaka slika dimenzija 28x28 piksela, odnosno sadrži ukupno 784 piksela. Svaki od 784 piksela je reprezentiran vrijednošću od 0 do 256 koja predstavlja intenzitet crne boje na tom pikselu. Kod korištenja neuronskih mreža u području računalnog vida često se slike šalju u model kao dvodimenzionalne matrice u svrhu primjene tzv. *konvolucijskih neuronskih mreža* [20]. Međutim pokazano je da se za ovaj skup podataka mogu postići odlični rezultati pretvaranjem ulaza u vektor dimenzije $28 \cdot 28 = 784$ i zanemarivanjem prirodne dvodimenzionalnosti ulaza. Uz svaki primjerak skupa dolazi i vrijednost od 0 do 9 koja označava o kojoj se znamenki radi.

Cilj je na ovom skupu podataka naučiti model koji može za dani neviđeni primjerak (sliku) odrediti o kojoj se znamenci (klasi) radi. Ovakav problem se u okviru strojnog učenja zove problem klasifikacije s p klasa, gdje je u ovom slučaju $p = 10$. Točnost modela u problemu klasifikacije se mjeri kroz postotak primjera iz neviđenog skupa za testiranje koje model ispravno klasificira. Iako najbolje rezultate na ovom problemu postižu konvolucijske neuronske mreže (preko 99% točnosti [46]), zbog konzistentnosti s eksperimentom od prije držimo se arhitekture kao i u prijašnjem primjeru.

Ovaj skup je posebno interesantan za primjenu Bayesovih neuronskih mreža s obzirom da otvara razne mogućnosti po pitanju analize ponašanja modela. Iako je ulaz u model vektor dimenzije 784, značenje tog vektora se može jasno prikazati - pomoću slike koju predstavlja. Postavlja se pitanje što ako naučenom modelu damo neku jako neuredno napisanu znamenku ili pak nešto što uopće nije znamenka? U ovom slučaju bi bilo iznimno korisno kada bi mogli ocijeniti sigurnost modela u svoju predikciju što je upravo ono što očekujemo od Bayesovih neuronskih mreža. Očekujemo da kroz aleatoričku i epistemološku nesigurnost za dani testni primjerak možemo ocijeniti koliko je model siguran u svoju predikciju. Konkretno, visoka aleatorička nesigurnost može sugerirati da je slika nejasna, da joj nedostaje piksela ili da ulazni primjerak sliči na više znamenaka (npr. broj 1 napisan slično kao broj 7). S druge strane, epistemološka nesigurnost bi nam trebala reći da se model možda suočava s primjercima van svoje domene na kojoj je treniran. Npr. znamenka napisana stilom koji do sada nije viđen ili slika nečega što uopće ne predstavlja znamenku.

Originalni MNIST skup podataka sadrži 60000 primjeraka u skupu za treniranje i 10000 u skupu za testiranje, gdje skup za testiranje nasumično dijelimo na dva skupa od 5000 - jedan za validaciju i jedan za testiranje.



Slika 5.9: Primjer jednog ”urednog” i jednog ”neurednog” primjerka iz MNIST skupa podataka

Arhitektura modela

Cilj nam je na skupu za treniranje naučiti dva modela - regulariziranu neuronsku mrežu i Bayesovu neuronsku mrežu. Za regulariziranu neuronsku mrežu koristimo istu arhitekturu kao prije s 2 skrivena sloja po 64 s jednom bitnom razlikom u izlaznom sloju - izlazni sloj je veličine 10 (po jedan za svaku klasu). S obzirom da vrijednosti u izlaznom sloju mogu biti proizvoljne, često je korisno vrijednosti izlaza pretvoriti u vjerojatnosti korištenjem tzv. *softmax* funkcije:

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{p} e^{z_j}} \tag{5.6}$$

gdje je z_i vrijednost izlaznog sloja za $i - tu$ klasu. Nakon ove operacije, izlazne vrijednosti predstavljaju vjerojatnost pripadnosti pojedinoj klasi [8]. Ovaj izlaz se onda može interpretirati na više načina. Najčešće se kao konačna predikcija uzima klasa s najvećom vjerojatnosti (što se moglo napraviti i prije primjene softmax funkcije s obzirom da je transformacija monotona). Vjerojatnosni izlaz je jako koristan jer predstavlja najprimitivniju vrstu kvantifikacije nesigurnosti. Npr, može se postaviti proizvoljna granica (npr. 0.5) te za konačnu predikciju uzeti klasa s najvišom vjerojatnosti ako prelazi 0.5 ili se izlaz može definirati kao ”nepoznato” ako ni jedna vjerojatnost ne prelazi 0.5. Ovdje je već vidljiva i mogućnost klasičnih neuronskih mreža da neformalno definiraju nesigurnost u procjenu. U postupku treniranja klasične neuronske mreže se sve odvija kao i prije osim što je funkcija gubitka sada kategorijski gubitak entropije definiran u 1.7.

Što se tiče Bayesovih neuronskih mreža, koristimo istu arhitekturu kao i prije osim što izlazni sloj i računanje konačne predikcije postaje malo kompliciranije. U Bayesovoj

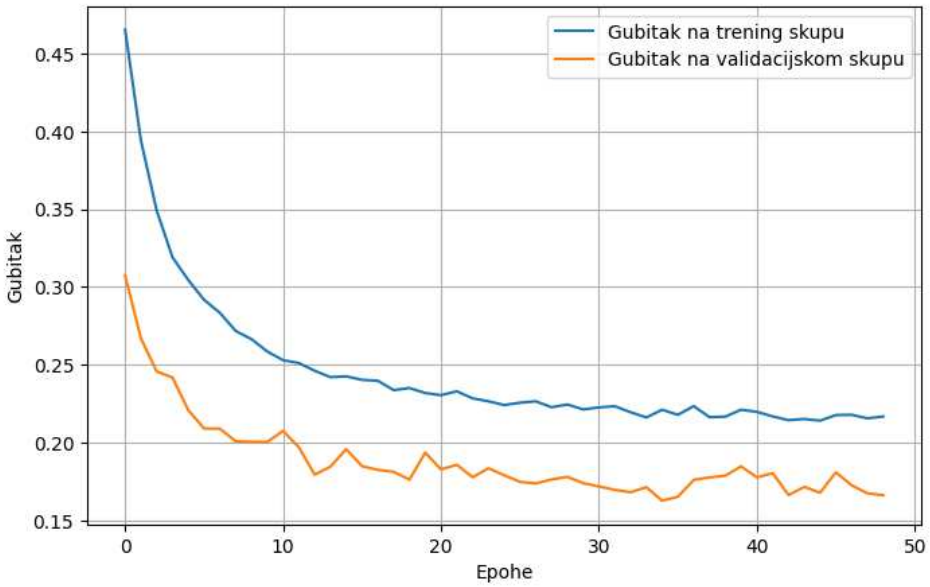
neuronskoj mreži će izlazni sloj biti dimenzije 20 s obzirom da svaka klasa za izlaz ima srednju vrijednost i log-varijancu. Prije primjene softmax funkcije nužno je na neki način skalirati izlazne vrijednosti log-varijanci kako bi se nakon softmax funkcije vjerojatnosti i procijenjene mjere nesigurnosti tih vjerojatnosti nalazili na istoj skali [7]. Recimo:

$$\text{softmax}_{\text{std}}(\log \sigma^2)_i = \frac{\sigma_i^2}{\left(\sum_{j=1}^{p} e^{z_j}\right)^2} \tag{5.7}$$

čime smo došli na skalu standardne devijaciju kao izlaz softmax funkcije. Dobivene vrijednosti nakon primjene softmax funkcije na srednje vrijednosti i modificirane softmax funkcije na log-varijance su sada vjerojatnosti pripadnosti klasama i pripadne standardne devijacije tih vjerojatnosti. U ovom radu ćemo te vjerojatnosti interpretirati tako da uzmemo klasu s najvećom vjerojatnosti kao konačnu predikciju. Sami proces treniranja se provodi kao i prije - varijacijskim učenjem minimizirajući varijacijski donji limit.

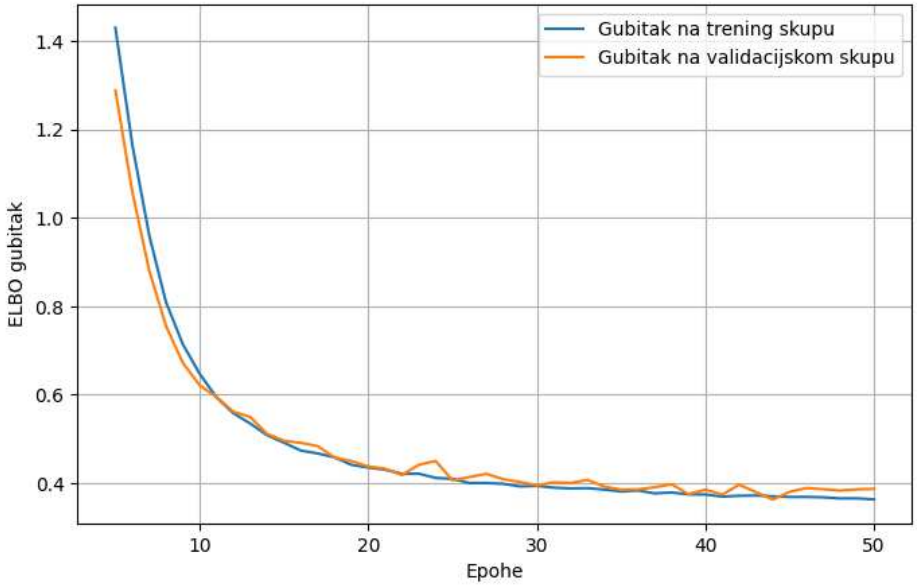
Treniranje modela

Na sljedećem grafu vidimo gubitak na trening i validacijskom skupu regularizirane neuronske mreže:



Slika 5.10: Gubitak regularizirane neuronske mreže kroz epohe 2-50

Ponašanje je slično kao i u prijašnjem primjeru s tim da izgleda da se gubitak stabilizirao u puno manjem broju epoha. Slično ponašanje uočavamo i za Bayesovu neuronsku mrežu:



Slika 5.11: Gubitak Bayesove neuronske mreže kroz epohe 4-50

Opet se vidi da Bayesova neuronska mreža konvergira sporije, a ta konvergencija je postignuta u puno manjem broju epoha nego u prijašnjem primjeru. Kod obje mreže je vidljiv regularizacijski učinak s obzirom da validacijski gubitak prati gubitak na skupu za treniranje.

U ovom primjeru smo u procesu treniranja Bayesove neuronske mreže postupili malo opreznije. Uz malo truda i sreće uspjeli smo optimizirati neke postavke i hiperparametre:

- **Smanjenje varijanci priora i prvog posteriora** - omogućuje modelu da krene s konzervativnijim pretpostavkama o parametrima, što osigurava stabilniju optimizaciju, realističnije nesigurnosti i bolju generalizaciju na testnim podacima [28].
- **Uvođenje planiranog učenja** - tehnika učenja gdje se stopa učenja mjenja prema unaprijed definiranom rasporedu, najčešće eksponencijalno ili linearno. Ova tehnika poboljšava konvergenciju modela smanjujući rizik od ”zapinjanja” u lokalnom minimumu i smanjivanjem oscilacije parametara [11].
- **Ograničavanje i regularizacija log-varijance** - prisilno ograničavanje log-varijance osigurava numeričku stabilnost u računanju gradijenata i gubitaka. S druge strane,

regularizacija log-varijance spriječava "eksploziju" nesigurnosti i mogućnost da model "zloupotrebljava" varijance kako bi smanjio gubitak, a pritom žrtvujući generalizacijsku sposobnost [12].

Optimizacija postavki i hiperparametara je provedena metodom pokušaja i pogreške gdje se za svaku konfiguraciju pratila točnost na testnom skupu. Sama ideja smjera kretanja hiperparametara je preuzeta s [30]. S obzirom da je ovo jako raširen i poznat skup podataka, poznato je da točnost od 95% i više sugerira dobru naučenost modela pa se optimizacija provodila sve dok nije pronađena konfiguracija koja je rezultirala zadovoljavajućom točnosti na skupu za validaciju.

Iste su intervencije pokušane i na prijašnjem *Wine quality* skupu podataka međutim nisu donijele neki značajan pomak. Prijeđimo sada na rezultate modela.

Rezultati

U tablici su navedene točnosti u postotcima za oba modela te njihove točnosti po svakoj od 10 klasa na skupu za testiranje:

Klasa	Bayesova NN (%)	Klasična NN (%)
0	98.5	97.1
1	98.0	96.9
2	97.5	95.1
3	96.9	95.3
4	94.2	95.6
5	94.4	94.6
6	96.9	94.5
7	97.98	94.9
8	95.2	91.7
9	94.5	92.9
Ukupno	96.50	94.94

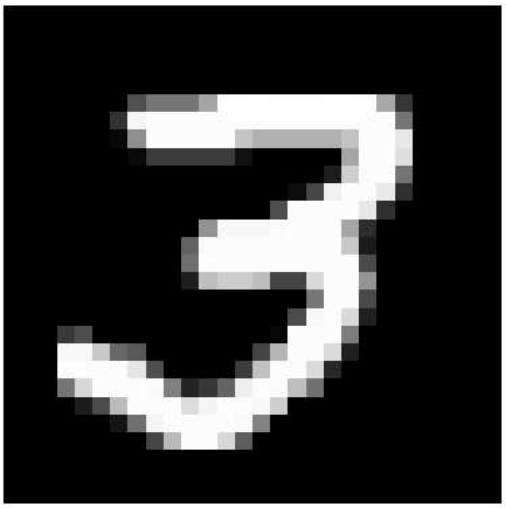
Tablica 5.5: Točnosti modela u postocima na skupu za testiranje

Vidimo da Bayesova neuronska mreža postiže bolji ukupni rezultat te bolji rezultat u skoro svakoj od 10 klasa na skupu za testiranje. Također, vidi se da klasična neuronska mreža postiže najgori rezultat na znamenci 8 dok Bayesova ima najgori rezultat na znamenci 4. S druge strane, obje mreže postižu najbolji rezultat za znamenku 1 što ne čudi s obzirom da znamenka 1 ima daleko najjednostavniji oblik. Prijeđimo sada na analizu i interpretaciju nesigurnosti.

Procjena nesigurnosti

Vjerojatno najzanimljivi dio ovog eksperimenta je analiza nesigurnosti Bayesove neuronske mreže na nekim zanimljivim primjerima. Još jednom se prisjetimo da aleatorička nesigurnost u ovom kontekstu daje mjeru kvalitete i šuma u slici te nesigurnosti u procjeni između klasa. S druge strane, epistemološka nesigurnost daje mjeru o tome koliko je model upoznat s ovakvim primjerom. Epistemološka nesigurnost u ovom kontekstu je idealan alat za prepoznavanje primjeraka koji uopće ne pripadaju ovom skupu podataka (eng. *out-of-distribution detection* [47]).

Pogledajmo kako izgledaju izlazi i predikcije oba modela te procjena nesigurnosti za jedan primjerak koji je Bayesova neuronska mreža točno klasificirala s niskom nesigurnosti. p_i predstavlja procijenjenu vjerojatnost i-te klase, $\hat{y}$ konačnu procijenjenu klasu te σ_a, σ_e i σ odgovarajuće nesigurnosti. Vrijednosti u svim tablicama će biti zaokružene na dvije decimale:



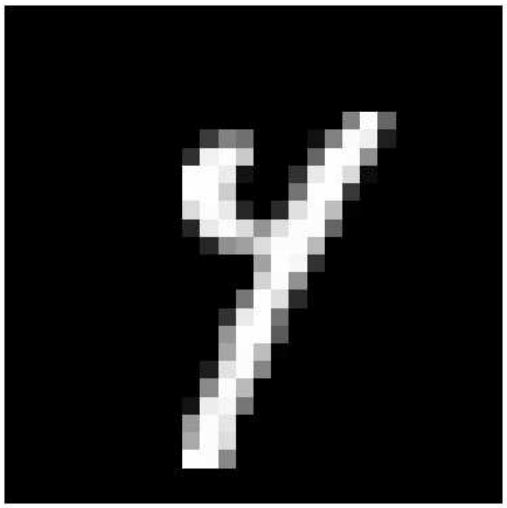
Slika 5.12: Lagan primjerak za Bayesovu neuronsku mrežu - stvarna klasa 3

Dobiveni rezultati ne čude. Znamenka se čini uredna i jednostavna za razumijeti. Za očekivati je da veliki broj znamenaka 3 ima jasno uočljive uzorke kao na ovoj slici. Jednostavnost ovog primjera se očituje u visokim izlazima vjerojatnosti oba modela i niskoj razini nesigurnosti.

	p_0	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	$\hat{y}$
BNN	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	3
NN	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	3
	σ_a	σ_e	σ								
	$< 10^{-3}$	$< 10^{-3}$	$< 10^{-3}$								

Tablica 5.6: Tablica izlaza modela za gornji primjerak

Pogledajmo sada primjerak koji je Bayesova neuronska mreža krivo klasificirala:



Slika 5.13: Težak primjerak za Bayesovu neuronsku mrežu - stvarna klasa 4

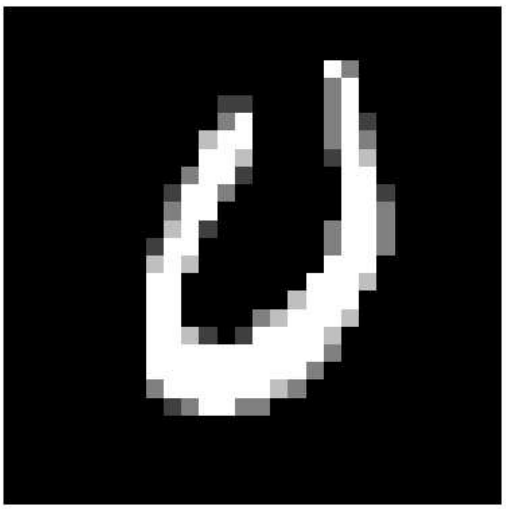
	p_0	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	$\hat{y}$
BNN	0.00	0.05	0.00	0.00	0.28	0.01	0.00	0.40	0.09	0.16	7
NN	0.00	0.00	0.00	0.00	0.86	0.00	0.00	0.05	0.01	0.08	4
	σ_a	σ_e	σ								
	0.25	0.24	0.34								

Tablica 5.7: Tablica izlaza modela za gornji primjerak

Ni ljudskom oku ova znamenka nije u potpunosti jasna. Iskustveno znamo da je ova znamenka najvjerojatnije 4, no nebi čudilo i da je 1, 7 ili 9. Vidimo da je Bayesova neuronska mreža ovo krivo klasificirala, ali ipak nije otišla u skroz pogrešnom smjeru s obzirom

da je klasa 4 na drugom mjestu. Vidljive su i značajne razine obje vrste nesigurnosti što i ne čudi s obzirom na ne tipičan stil pisanja ove znamenke.

Uzmimo sada jedan primjerak s velikom aleatoričkom nesigurnosti:



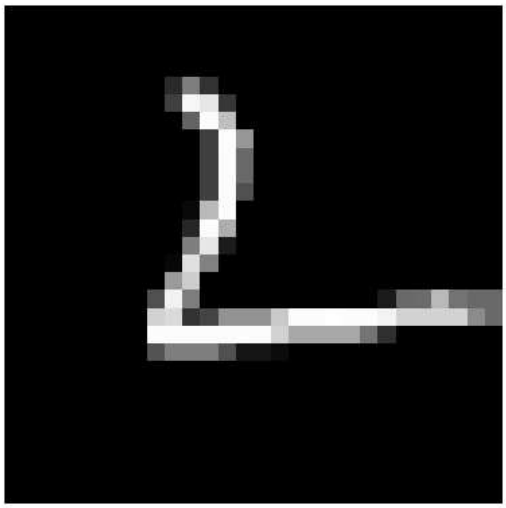
Slika 5.14: Primjerak s velikom aleatoričkom nesigurnosti - stvarna klasa je 0

	p_0	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	$\hat{y}$
BNN	0.66	0.00	0.07	0.00	0.04	0.00	0.22	0.00	0.00	0.01	0
NN	0.86	0.00	0.01	0.00	0.02	0.00	0.09	0.00	0.00	0.01	0
	σ_a	σ_e	σ								
	0.27	0.20	0.34								

Tablica 5.8: Tablica izlaza modela za gornji primjerak

Ovo je tipičan primjer aleatoričke nesigurnosti. Na ovom primjeru se jasno vidi da fale neki pikseli. Vrlo je vjerojatno da ovako ne izgleda tipična znamenka 0 u skupu za treniranje, pa zato ne čudi visoka razina nesigurnosti. S druge strane, to povlači i visoku razinu epistemološke nesigurnost s obzirom da model sigurno nije vidio puno ovakvih primjeraka u skupu za treniranje. Kako god, to nije spriječilo oba modela da naprave točnu procjenu s obzirom da i u ovako "krnjem" obliku ova znamenka najviše sliči znamenci 0 nego bilo kojoj drugoj.

Uzmimo i jedan primjerak s velikom epistemološkom nesigurnosti:



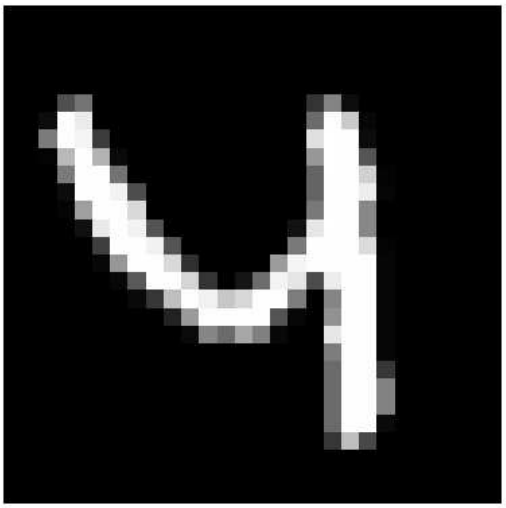
Slika 5.15: Primjerak s velikom epistemološkom nesigurnosti - stvarna klasa 2

	p_0	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	$\hat{y}$
BNN	0.07	0.00	0.69	0.02	0.00	0.00	0.00	0.20	0.01	0.00	2
NN	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	2
	σ_a	σ_e	σ								
	0.14	0.40	0.43								

Tablica 5.9: Tablica izlaza modela za gornji primjerak

Na slici se nalazi jedna jako čudna i netipična verzija znamenke 2. Moglo bi se reći da ovo uopće nije znamenka iz skupa za treniranje nego neko slovo, što potvrđuje visoka razina epistemološke nesigurnosti Bayesove neuronske mreže. Zanimljivo je kako je Bayesova neuronska mreža dala i značajnu vjerojatnost klasi 7 iako ova slika ne malo ne sliči na znamenku broj 7. Vjerojatno je do takvog ponašanja došlo zbog velikog preklapanja piksela s primjercima iz klase 7.

Pogledajmo i jedan primjerak s velikom ukupnom nesigurnosti:



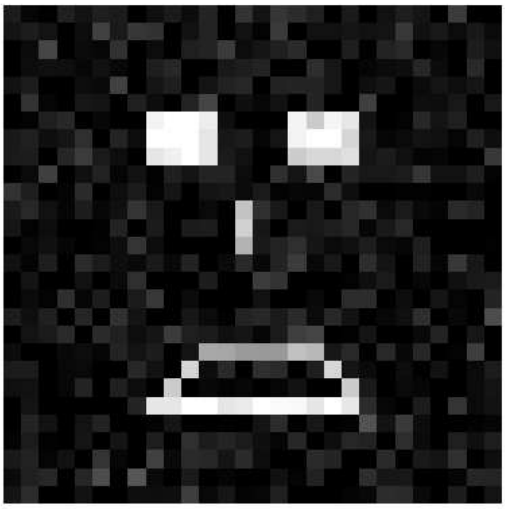
Slika 5.16: Primjerak s velikom ukupnom nesigurnosti - stvarna klasa 4

	p_0	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	$\hat{y}$
BNN	0.00	0.08	0.15	0.03	0.43	0.00	0.00	0.29	0.01	0.02	4
NN	0.00	0.00	0.60	0.01	0.38	0.00	0.00	0.01	0.00	0.00	2
	σ_a	σ_e	σ								
	0.19	0.40	0.44								

Tablica 5.10: Tablica izlaza modela za gornji primjerak

Ovaj primjerak je zanimljiv s obzirom da Bayesova neuronska mreža daje značajne vjerojatnosti 3 različite klase. Ipak, zbog razine ukupne nesigurnosti, procjena Bayesove neuronske mreže ovdje nema neku veliku težinu. S druge strane vidimo i da obična neuronska mreža nije u potpunosti sigurna u svoju odluku. Na većini primjera neuronska mreža daje jako samouvjerene predikcije čak i ako je u krivu dok ovdje izbacuje vjerojatnost od samo 0.6. Zanimljivo je također kako obična neuronska mreža ne prirodaje skoro nikakvu vjerojatnost klasi 7 dok Bayesova daje čak 0.29.

Na kraju, pogledajmo što modeli kažu na jednu sliku koja nema nikakve veze s ovim skupom podataka:



Slika 5.17: Primjerak van MNIST skupa podataka

	p_0	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	$\hat{y}$
BNN	0.01	0.05	0.12	0.05	0.01	0.62	0.06	0.07	0.00	0.01	5
NN	0.01	0.08	0.09	0.05	0.03	0.55	0.02	0.17	0.01	0.01	5
	σ_a	σ_e	σ								
	0.40	0.43	0.59								

Tablica 5.11: Tablica izlaza modela za prilagođenu sliku

Iako prikazana slika ne sliči ni jednoj znamenki iz skupa za treniranje obje mreže su poprilično samouvjereno napravili procjenu za nešto što uopće ne dolazi iz distribucije podataka. Vidimo također da su distribucije na izlazu poprilično korelirane u ovom slučaju što opet pokazuje da je sličnost između informacija koje su mreže usvojile jako visoka. U ovom slučaju niti malo ne čudi izrazito visoka razina epistemološke i ukupne nesigurnosti Bayesove neuronske mreže s obzirom da ovaj primjerak uistinu nije iz distribucije originalnih podataka.

U konačnici, vidimo da su Bayesova i klasična neuronska mreža usvojile slične obrasce s obzirom da se preklapaju u jako puno primjeraka. Također, kroz raspršenost vjerojatnosti u izlaznom sloju vidimo da Bayesova neuronska mreža pristupa predikciji puno konzervativnije u odnosu na klasičnu koja često daje vjerojatnosti jako blizu 1.

Za kraj, vidimo da je MNIST skup podataka zbog svoje veličine i prirode izrazito dobar za demonstrirati prednosti Bayesove neuronske mreže. Na danome smo skupu podataka uspijeli pokazati kako Bayesova neuronska mreža može vrlo lijepo prikazati veličinu i izvor nesigurnosti kada se model uspješno nauči pritom ne žrtvujući svoju prediktivnu sposobnost.

Kodovi korišteni u provedbi eksperimenata se nalaze na sljedećoj GitHub poveznici: `https://github.com/josko19kristic50/Bayesove_neuronske_mre-e`.

Bibliografija

[1] Ahmad Alwosheel, Sander van Cranenburgh, and Caspar G. Chorus. Is your dataset big enough? sample size requirements when using artificial neural networks for discrete choice analysis. *Journal of Choice Modelling*, 28:167–182, 2018. doi:10.1016/j.jocm.2018.07.002.

[2] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, New York, 2006.

[3] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks, 2015. URL: https://arxiv.org/abs/1505.05424, arXiv:1505.05424.

[4] William M. Bolstad and James M. Curran. *Introduction to Bayesian Statistics, 3rd Edition*. John Wiley Sons, Hoboken, NJ, 2016.

[5] Tianqi Chen, Emily Fox, and Carlos Guestrin. Stochastic gradient hamiltonian monte carlo. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, pages 1683–1691, 2014.

[6] Romain Egele, Felix Mohr, Tom Viering, and Prasanna Balaprakash. The unreasonable effectiveness of early discarding after one epoch in neural network hyperparameter optimization. *arXiv preprint arXiv:2404.04111*, 2024. URL: https://arxiv.org/abs/2404.04111.

[7] Stanislav Fort, Huiyi Hu, and Balaji Lakshminarayanan. Deep ensembles: A loss landscape perspective. *arXiv preprint arXiv:2008.01289*, 2020.

[8] Michael Franke and Judith Degen. The softmax function: Properties, motivation, and interpretation. 2021. URL: https://alpslab.stanford.edu/papers/FrankeDegen_submitted.pdf.

[9] Yarin Gal. *Uncertainty in Deep Learning*. PhD thesis, University of Cambridge, 2016. URL: https://www.repository.cam.ac.uk/handle/1810/261084.

[10] Stuart Geman and Donald Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6):721–741, 1984. doi:10.1109/TPAMI.1984.4767596.

[11] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[12] Danijar Hafner, Dustin Tran, Timothy Lillicrap, Alex Irpan, and James Davidson. Reliable uncertainty estimates in deep neural networks using noise contrastive priors. *arXiv preprint arXiv:1807.09289*, 2018.

[13] W. Keith Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970. doi:10.1093/biomet/57.1.97.

[14] J. W. Higgins et al. Bayesian neural networks: An introduction and survey. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, pages 85–115. Springer, 2020.

[15] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989. URL: https://www.sciencedirect.com/science/article/pii/0893608089900208, doi:10.1016/0893-6080(89)90020-8.

[16] Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37(2):183–233, 1999. doi:10.1023/A:1007665907178.

[17] Laurent Valentin Jospin, Hamid Laga, Farid Boussaid, Wray Buntine, and Mohammed Bennamoun. Hands-on bayesian neural networks—a tutorial for deep learning users. *IEEE Computational Intelligence Magazine*, 17(2):29–48, May 2022. URL: http://dx.doi.org/10.1109/MCI.2022.3155327, doi:10.1109/MCI.2022.3155327.

[18] Robert E. Kass and Adrian E. Raftery. Bayes factors. *Journal of the American Statistical Association*, 90(430):773–795, 1995. doi:10.1080/01621459.1995.10476572.

[19] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5574–5584, 2017. URL: https://arxiv.org/abs/1703.04977.

[20] Salman Khan, Hossein Rahmani, Syed Afaq Ali Shah, and Mohammed Bennamoun. *A Guide to Convolutional Neural Networks for Computer Vision*. Springer, 2018. URL: https://link.springer.com/book/10.1007/978-3-031-01821-3.

[21] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. URL: https://arxiv.org/abs/1412.6980.

[22] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[23] Mladen Kolar and James Sharpnack. Variance function estimation in high-dimensions. *arXiv preprint arXiv:1205.4770*, 2012.

[24] Yann LeCun, Léon Bottou, Genevieve B. Orr, and Klaus-Robert Müller. Efficient backprop. In *Neural Networks: Tricks of the Trade*, pages 9–48. Springer, 2012. doi:10.1007/978-3-642-35289-8_3.

[25] Warren S. McCulloch and Walter H. Pitts. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5(4):115–133, 1943. doi:10.1007/BF02478259.

[26] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.

[27] Meenal V. Narkhede, Prashant P. Bartakke, and Mukul S. Sutaone. A review on weight initialization strategies for neural networks. *Artificial Intelligence Review*, 55:291–322, 2021. URL: https://api.semanticscholar.org/CorpusID:237793845.

[28] Radford M. Neal. *Bayesian Learning for Neural Networks*, volume 118 of *Lecture Notes in Statistics*. Springer, 1996. URL: https://link.springer.com/book/10.1007/978-1-4612-0745-0, doi:10.1007/978-1-4612-0745-0.

[29] Michael A. Nelsen. *Neural Networks and Deep Learning*. Determination Press, 2015.

[30] Neptune.ai. Bayesian neural networks—implementing, training, inference with the jax framework. 2023. Accessed: 2025-01-08. URL: https://neptune.ai/blog/bayesian-neural-networks-with-jax.

[31] David A. Nix and Andreas S. Weigend. Estimating the mean and variance of the target probability distribution. In *Proceedings of the 1994 IEEE International Conference on Neural Networks (ICNN)*, volume 1, pages 55–60. IEEE, 1994. doi:10.1109/ICNN.1994.374138.

[32] Mathew Mithra Noel, Arindam Banerjee, Yug Oswal, Geraldine Bessie Amali D, and Venkataraman Muthiah-Nakarajan. Alternate loss functions for classification and

robust regression can improve the accuracy of artificial neural networks, 2024. URL: https://arxiv.org/abs/2303.09935, arXiv:2303.09935.

[33] Farhad Pourkamali-Anaraki, Jamal F. Husseini, and Scott E. Stapleton. Probabilistic neural networks (pnns) for modeling aleatoric uncertainty in scientific machine learning. *arXiv preprint arXiv:2402.13945*, 2024.

[34] Lutz Prechelt. Early stopping—but when? In *Neural Networks: Tricks of the Trade*, pages 53–67. Springer, 2012.

[35] Frank Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958. doi: 10.1037/h0042519.

[36] Jonas Rothfuss, Fabio Ferreira, and Andreas Krause. Effective bayesian heteroscedastic regression with deep neural networks. In *Advances in Neural Information Processing Systems*, 2023.

[37] Sebastian Ruder. An overview of gradient descent optimization algorithms, 2017. URL: https://arxiv.org/abs/1609.04747, arXiv:1609.04747.

[38] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986. doi: 10.1038/323533a0.

[39] Sagar Sharma. What the hell is perceptron? https://towardsdatascience.com/what-the-hell-is-perceptron-626217814f53, 2017. Pristupljeno: 2024-10-11.

[40] Matteo Silvestro and L. Andermann. The role of prior distributions in bayesian neural networks: A study of classification and out-of-distribution detection, 2020. URL: https://arxiv.org/abs/2005.04987, arXiv:2005.04987.

[41] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[42] Matias Valdenegro-Toro and Daniel Saromo. A deeper look into aleatoric and epistemic uncertainty disentanglement. *arXiv preprint arXiv:2204.09308*, 2022.

[43] Maria Vladimirova, Jakob Verbeek, and Rémi Gribonval. Understanding priors in bayesian neural networks at the unit level, 2018. URL: https://arxiv.org/abs/1810.05193, arXiv:1810.05193.

[44] Larry Wasserman. *All of Statistics: A Concise Course in Statistical Inference*, chapter 11, pages 175–190. Springer, New York, 2004. doi:10.1007/978-0-387-21736-9.

[45] Max Welling and Yee Whye Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, pages 681–688, 2011.

[46] Papers with Code. Image classification on mnist. https://paperswithcode.com/sota/image-classification-on-mnist, 2025. Accessed: 2025-01-15.

[47] Jingkang Yang, Kaiyang Zhou, Yixuan Li, and Ziwei Liu. Generalized out-of-distribution detection: A survey. *arXiv preprint arXiv:2110.11334*, 2021. URL: https://arxiv.org/abs/2110.11334.

[48] Xue Ying. An overview of overfitting and its solutions. *Journal of Physics: Conference Series*, 1168(2):022022, 2019. doi:10.1088/1742-6596/1168/2/022022.

[49] K. Šura. Primjena neuronskih mreža u modeliranju svojstava atomskih jezgara (diplomski rad). https://urn.nsk.hr/urn:nbn:hr:217:519531, 2023. Pristupljeno: 2024-11-01.

Sažetak

U ovom diplomskom radu obrađena je tema Bayesovih neuronskih mreža, koje predstavljaju spoj standardnih neuronskih mreža i Bayesovske statistike. U svrhu razumijevanja Bayesovih neuronskih mreža obrađuju se teme klasičnih neuronskih mreža i Bayesovskog zaključivanja, s naglaskom na Bayesov teorem te priorne i posteriorne distribucije. Detaljno se analizira struktura i razlike Bayesovih neuronskih mreža u odnosu na klasične, s fokusom na metode treniranja poput varijacijskog zaključivanja i Monte Carlo metoda. Mogućnost kvantifikacije neizvjesnosti Bayesovih neuronskih mreža prezentira se kroz uvođenje vrsta neizvjesnosti i njihove teorijske podloge. Za kraj teorijskog dijela, detaljno se analizira problem pretreniranosti u kontekstu strojnog učenja te metode kojima klasične i Bayesove neuronske mreže njemu pristupaju.

U eksperimentalnom dijelu, na stvarnim skupovima podataka implementiraju se modeli klasičnih i Bayesovih neuronskih mreža. Kroz rezultate eksperimentalnog dijela pokazuju se mogućnosti Bayesovih neuronskih mreža i prednosti u odnosu na klasične.

Summary

This thesis explores the topic of Bayesian neural networks, which represent a combination of standard neural networks and Bayesian statistics. To facilitate an understanding of Bayesian neural networks, the concepts of classical neural networks and Bayesian inference are discussed, with an emphasis on Bayes' theorem and prior and posterior distributions. The structure and differences of Bayesian neural networks compared to classical ones are analyzed in detail, focusing on training methods such as variational inference and Monte Carlo methods. The ability of Bayesian neural networks to quantify uncertainty is presented through the introduction of types of uncertainty and their theoretical foundations. The theoretical section concludes with a detailed examination of the problem of overfitting in the context of machine learning and the methods used by classical and Bayesian neural networks to address it.

In the experimental section, models of classical and Bayesian neural networks are implemented on real datasets. The results of the experimental section demonstrate the capabilities of Bayesian neural networks and their advantages over classical neural networks.

Životopis

Rođen sam 24.04.1999. u Splitu gdje sam nakon završetka osnovne škole školovanje nastavio u Prirodoslovnoj-matematičkoj gimnaziji u Splitu. Uz redovno školovanje sam se rekreativno bavio košarkom i ragbijem te redovno sudjelovao na županijskim i državnim smotrama iz Matematike, Fizike i Logike. Godine 2018. sam upisao preddiplomski studij Matematika na Prirodoslovno-matematičkom fakultetu u Zagrebu koji sam i redovno završio 2021. te upisao diplomski sveučilišni studij Matematička statistika.

U svojoj profesionalnoj karijeri sam radio kao kreator matematičkog sadržaja i poslovni analitičar u *Photomath d.o.o.* od 2019. do 2023. kada sam se zaposlio kao podatkovnik znanstvenik u tvrtki *Maistra d.d.* gdje sam još uvijek zaposlen.