Osnovni koncepti, metodologija i primjena objašnjive umjetne inteligencije

Duvnjak, Matej

Master's thesis / Diplomski rad

2020

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Science / Sveučilište u Zagrebu, Prirodoslovno-matematički fakultet**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:217:942014

Osnovni koncepti, metodologija i primjena objašnjive umjetne inteligencije

Duvnjak, Matej

Master's thesis / Diplomski rad

2020

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Science / Sveučilište u Zagrebu, Prirodoslovno-matematički fakultet**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:217:942014

SVEUČILIŠTE U ZAGREBU

PRIRODOSLOVNO–MATEMATIČKI FAKULTET

MATEMATIČKI ODSJEK

Matej Duvnjak

OSNOVNI KONCEPTI, METODOLOGIJA I PRIMJENA OBJAŠNJIVE UMJETNE INTELIGENCIJE

Diplomski rad

Voditelj rada:
dr. sc.Tomislav Šmuc

Zagreb, Rujan, 2020

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

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

Povjerenstvo je rad ocijenilo ocjenom _____.

Potpisi članova povjerenstva:

1. _____
2. _____
3. _____

Mojoj obitelji i tetama u menzi SD Laščina

Sadržaj

Uvod

Sve veći razvoj sustava umjetne inteligencije (AI) u domenama od visoke važnosti povezan je s povećanim zahtjevima društva za objašnjivost predviđanja takvih modela. Ova objašnjenja mogu omogućiti korisnicima da dobiju uvid u proces donošenja odluka sustava, što je ključna komponenta za održavanje povjerenja u AI sustave. Međutim, mnoge tehnike strojnog učenja, koje su odgovorne za velik napredak AI-a, nisu lako objašnjive, čak ni stručnjacima u tim područjima. Što dovodi do polja istraživanja usredotočenom na "interpretativne" ili "objašnjive" tehnike strojnog učenja. Važno je napomenuti da uz veliki broj publikacija i istraživanja u ovom polju još uvijek postoji velika praznina između društvenih potreba za objašnjenjima i interpretacijom modela te onoga što znanstvena zajednica producira. Cilj ovog rada je istražiti i proučiti različite pristupe i koncepte za postizanje objašnjive umjetne inteligencije uključujući direktno korištenje samo-objašnjivih modela te korištenje naknadno (post hoc) objašnjivih metoda za interpretaciju ponašanja cijelog modela (globalno) ili (lokalno) njegovih izlaza. Također koristimo model agnostičnih metode - ne ovisne o vrsti modela te model specifične za određenu vrstu modela s fokusom na modele dubokih neuronskih mreža za specifično strukturirane podatke kao što su slike i vremenski nizovi.

Poglavlje 1

Terminologija pojmova objašnjive umjetne inteligencije

1.1 Terminologija

Prije početka rada važno je uvesti važeću terminologiju, uspostaviti zajedničku točku razumijevanja onoga što pod pojam objašnjivosti misli u kontekstu AI, točnije objašnjivog ML-a. Zbog različitih domena primjene svaka zajednica koja koristi metode ML-a daje drugačije značenje objašnjivosti te je to jedan od razloga zašto je nekada teško specificirati što se misli konkretno pod pojmom objašnjivosti. Jedno od pitanja koje sprečava uspostavljanje zajedničkih osnova je i miješanje pojmova interpretabilnosti i objašnjivosti u literaturi. Postoje značajne razlike među tim pojmovima. Za početak, interpretabilnost se odnosi na pasivne karakteristike modela odnoseći se na razinu na kojoj određeni model ima smisla za čovjeka. Ova se značajka izražava i kao transparentnost. Suprotno, objašnjivost se može promatrati kao aktivna karakteristika modela, primjećujući bilo kakvu radnju ili postupak koji je poduzeo model s namjerom da pojašnjavanja njegovih unutarnjih funkcija.

Da bismo saželi najčešće korištenu nomenklaturu, u ovom odjeljku razjašnjavamo razlike i sličnosti među izrazima koji se često koriste u literaturi etičkog AI i XAI.

- **Razumljivost** označava karakteristiku modela da čovjek može razuminjeti njegovu funkciju - kako model funkcionira - bez potrebe za objašnjenjem njegovih internih strukturu ili algoritma pomoću kojeg model interno obrađuje podatke. Ukratko razumljivost je razumijevanje black-box funkcije algoritma.

- **Interpretabilnost** govori o stupnju objašnjenja uzročno posljedičnih veza unutar modela. Ili, drugačije rečeno, interpretabilnost govori u kojoj mjeri možemo predvidjeti što će se dogoditi s obzirom na promjenu ulaznih ili algoritamskih parametara.

- **Objašnjivost** je stupanj u kojem unutarnji mehanizmi modela mogu biti objašnjeni u okvirima prethodnog "ljudskog" razumijevanja samog problema.

Razliku između objašnjivosti i interpretabilnosti možemo pokazati na ovaj način: recimo da radimo znanstveni eksperiment u školi. Eksperiment bi mogao biti interpretiran ako možemo vidjeti što radimo, ali stvarno je objašnjeno tek kad shvatimo kemiju iza onoga što se događa.

Možemo dati i definiciju Objašnjive umjetne inteligencije(XAI), od D.Gunning iz [4]

"XAI područje strojnog učenja koje omogućuju korisnicima da razumiju, odgovorno vjeruju i učinkovito upravljaju novim generacijama umjetno inteligentnih partnera".

1.2 Taksonomija objašnjivih metoda i postizanje objašnjivosti

U literaturi postoji jasna razlika između modela koji se mogu sami interpretirati i onih koji se mogu objasniti pomoću "vanjskih" XAI tehnika. Šire prihvaćena podjela je na Samo-objašnjive modele i post-hoc tehnike objašnjavanja.

Samo-objašnjive ili Post-hoc metode

Samo-objašnjivi modele karakterizira određeni stupanj samo-interpretabilnosti. Ova interpretacija u modelu može se postići nametanjem ograničenja na model, kao što su rijetkost, monotonost, kauzalnost ili fizička ograničenja koja proizlaze iz poznavanja domene. Intrinzična interpretabilnost se također naziva transparentnost i odgovara na pitanje kako model funkcionira.

Post hoc interpretacija odnosi se na XAI metode koje se primjenjuju nakon što je model naučen. Važno je napomenuti da postoje post-hoc metode koje se mogu primijeniti na interpretabilne modele, jer post-hoc metode se obično odvajaju od glavnog modela. Post-hoc metode koriste tekstualna objašnjenja, vizualna objašnjenja, lokalna objašnjenja, objašnjenja primjerom, objašnjenja pojednostavljivanjem i tehnike objašnjenja relevantnih značajki.

Globalna interpretabilnost ili Lokalna interpretabilnost

Alati za interpretaciju mogu se klasificirati s obzirom na njihov opseg, koji se odnosi na dio modela predviđanja koji im je cilj objasniti.

Globalna interpretabilnost se može podijeliti na holističku i modularnu razinu.

Pod *holističkom razina* je mogućnost da razumijemo ("mi ljudi") model u cijelosti. Da bismo globalno objasnili rad modela, trebamo istrenirani model, znanje o algoritmu za učenje i podatcima. Globalna interpretabilnost modela distribucije ciljanog predviđanja na temelju značajki, odgovarajući na pitanje *"Kako obučeni model izrađuje predviđanja ?"*. Iz tog razloga, to je u praksi vrlo teško postići.

Kod *Modularne razine* promatramo kako dijelovi modela donose odluke odnosno odgovara na pitanje *"Kako dijelovi modela utječu na predviđanje ciljane vrijednosti ?"*. Na primjer, za linearne modele, interpretabilni dijelovi su težine i značajke, za stabla odlučivanja dijelovi koji se mogu interpretirati su značajke i granične vrijednosti te predviđanja čvorova na listu.

Lokalno interpretabilne metode dijelimo metode za jedno predviđanje ili grupu predviđanja.

Lokalnim metodama za jedno predviđanje cilj je objasniti jedno predviđanje, općenita je ideja promatrati pojedinu instancu i pokušati razumjeti kako je model stigao do svog predviđanja. To se može postići aproksimacijom na "okolini" instance, modela crne kutije koristeći jednostavniji interpretativni model. Ovakav pristup, iako ne nudi optimalno rješenje, može biti razumno dobra aproksimacija.

Lokalne metode za grupu predviđanja Da bismo objasnili grupu predviđanja, postoje u osnovi dvije mogućnosti: primijeniti globalne metode i tretirati skupinu predviđanja od interesa kao da se radi o cijelom skupu podataka ili primjenjuju lokalne metodama za svako predviđanje pojedinačno te združiti objašnjenja nakon provođenja lokalnih metoda.

Model-agnostičke i Model-specifične tehnike

Jedan važan način podjele interpretabilnih tehnika strojnog učenja je na

Model-agnostičke (Model-agnostic), što znači da se mogu primijeniti na različite vrste algoritama strojnog učenja (crna kutija ili ne) . Primjenjuju se nakon obuke modela (post hoc; post-model). Ove metode se oslanjaju na analiziranje parova značajki ulaza i izlaza. Po definiciji, ove metode ne mogu imati pristup znanje o unutarnjem djelovanju modela, kao što su težine ili strukturne informacije. Još jedno svojstvo ovih metoda je to što se modeli tumače bez žrtvovanja svoje prediktivne moći.

Model-specifične(Model-specific) tehnike koje su primjenjive samo za jedan tip ili klasu algoritma. Na primjer, interpretacija težina u linearnom modelu je model-specifična interpretacija.

Poglavlje 2

Samo-objašnjive metode Strojnog učenja

Najlakši način za postizanje interpretabilnosti je uporaba **Samo-objašnjivih** modela. Linearna regresija, logistička regresija i stablo odluka najčešće su korišteni samo-objašnjivi modeli.

2.1 Linearna Regresija

Model linearne regresije predviđa vrijednost ciljane varijable kao težinski zbroj značajki ulaza. Linearnost naučenog odnosa olakšava interpretaciju. Linearni modeli mogu se koristiti za modeliranje ovisnosti regresijske ciljane varijable y o nekim značajkama x. Naučeni odnosi su linearni i mogu se napisati na sljedeći način:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_p x_p + \epsilon$$

Vrijednost β_j predstavljaju naučene težine značajki dok je ϵ slučajna greška, tj. Razlika između predviđanja i stvarnog ishoda. Pretpostavlja se da ova pogreška slijede Gaussovu distribuciju.

Za procjenu optimalnih težina mogu se koristiti različite metode. Metoda najmanjeg kvadrata obično se koristi za pronalaženje težina.U fazi treniranja pokušava se umanjiti razlika kvadrata između stvarnih rezultata i vrijednosti predviđenih linearnom regresijom:

$$\hat{\beta} = arg \min_{\beta_0,\beta_1,...\beta_p} \sum_{i=1}^{n} (y^{(i)} - (\beta_0 + \sum_{j=1}^{p} \beta_j x_j))^2$$

Više detalja o Linearnoj regresiji nalazi se u [1]. Najveća prednost modela linearne regresije je linearnost: Postupak procjene ciljane varijable čine jednostavnim, te što je najvažnije, ove linearne jednadžbe imaju lako razumljivu interpretaciju.

Interpretacija Linearne regresije

Interpretacija težina modela linearne regresije ovisi o vrsti odgovarajuće značajke:

- **Numerička značajka** - Povećanje numeričke značajke x_j za jednu jedinicu mijenja vrijednost ciljane varijable za težinu β_j.
- **Binarna značajka** - značajka koja za svaku instancu uzima jednu od dvije moguće vrijednosti. Jedna od vrijednosti smatra se referentnom kategorijom (u nekim programskim jezicima kodiranim s 0). Promjena značajke x_j iz referentne kategorije u drugu kategoriju mijenja vrijednost ciljane varijable za težinu značajke β_j.
- **Kategorijska značajka s više kategorija** - Značajka s fiksnim brojem mogućih vrijednosti. Kategorijsku značajku označavamo tzv. one-hot-encoding, što znači da svaka kategorija ima svoj binarni stupac. Za kategorijsku značajku s L kategorija potrebno je samo L-1 stupaca, jer bi L-ti stupac imao suvišne informacije (npr. Kada svi stupci 1 do L-1 imaju vrijednost 0 za jednu instancu, znamo da kategorijska značajka ovog primjerak pripada kategoriji L). Interpretacija svake kategorije tada je isto što i interpretacija svake binarne značajke.

Također imamo β_0 težinu koju interpretiramo kao vrijednost procijene modela kada bi sve numeričke vrijednosti imale vrijednost 0 te sve kategorijske varijable bile u referentnom stanju. Interpretacija β_0 težine obično nije relevantna, jer instance sa svim značajkama vrijednosti na nuli često nemaju smisla.

Druga važna mjera za interpretaciju linearnih modela je R^2. R^2 nam govori koliko se ukupna varijanca našeg ciljanog ishoda objašnjava modelom. Što je veći R^2, model bolje objašnjava podatke. Formula za izračunavanje R^2 je:

$$R^2 = 1 - \frac{SSE}{SST}$$

gdje SSE i SST definiramo na sljedeći način:

$$SSE = \sum_{i=0}^{n} (y^{(i)} - \hat{y}^{(i)})$$

$$SST = \sum_{i=0}^{n} (y^{(i)} - \bar{y})$$

SSE računa kvadratnu razliku između predviđene i stvarne vrijednosti ciljane vrijednosti, dok SST računa varijancu ciljane varijable. Nije smisleno tumačiti model s vrlo niskom R^2 vrijednosti, jer takav model u osnovi ne objašnjava mnogo varijance, te bilo kakva interpretacija težina bi bila besmislena.

U stvarnim primjenama modela strojnog učenja događaju se situacije gdje imamo do nekoliko stotina značajki što ugrožava našu interpretabilnost, no postoje metode da se s time nosimo kao što su *Lasso* koji kada se koristi u linearnoj regresiji obavljao odabir značajki i regularizaciju. *Lasso* metoda dodaje izraz u optimizacijski problem

$$\min_{\beta}(\sum_{i=1}^{n}(y^{(i)} - x_i^T\beta)^2 + \lambda \|\beta\|_1)$$

Parametar λ možemo promatrati kao tuning-parametar te odabrati vrijednost λ parametra koji smanjuje cross-validacijsku grešku modela, no možemo ga koristi i kao interpretacijski parametar, naime što imamo veći lambda, više težina će biti 0 te će model biti lakše interpretabilan.

2.2 Logistička regresija

Logistička regresija modelira vjerojatnost za klasifikacijske probleme s dva moguća ishoda. To je proširenje modela linearne regresije za klasifikacijske probleme. Model logističke regresije koristi logističku funkciju za predviđanje kojoj klasi primjer pripada. Logistička funkcija je definirana kao:

$$logistic(x) = \frac{1}{1 + exp(-x)}$$

Za klasifikaciju preferiramo vjerojatnosti između 0 i 1 te definiramo vjerojatnost da $y^{(i)}$ pripada klasi 1 na sljedeći način:

$$P(y^{(i)} = 1) = \frac{1}{1 + exp(-(\beta_0 + \beta_1 x_1 + ... + \beta_p x_p))}$$

Interpretacija Logističke regresije

Interpretacija težina u logističkoj regresiji razlikuje se od interpretacije težina u linearnoj regresiji, budući da je rezultat u logističkoj regresiji vjerojatnost između 0 i 1. Težine više ne utječu na vjerojatnost linearno jer je težinska suma transformirana logističkom funkcijom u vjerojatnost. Trebamo reformulirati jednadžbu vrijednosti logističke regresije tako da samo linearni izraz bude na desnoj strani

$$log(\frac{P(y^{(i)} = 1)}{P(y^{(i)} = 0)}) = \beta_0 + \beta_1 x_1 + .. + \beta_p x_p$$

To dobivamo kada uzmemo u obzir logaritam omjera vjerojatnosti pripadnosti klas instance. Dalje izvodimo i označavamo sa $omjer$ običan omjer vjerojatnosti pripadnosti

klasa te $omjer_{x_j+1}$ omjer gdje je j-ta značajka uvećana za 1.

$$\frac{P(y=1)}{1-P(y=1)} = omjer = exp(\beta_0 + \beta_1 x_1 + ... + \beta_p x_p)$$

$$omjer_{x_j+1} = exp(\beta_0 + \beta_1 x_1 + ... + \beta_j(x_j+1) + ... + \beta_p x_p)$$

Pogledamo li odnos ova dva omjera dobivamo

$$\frac{omjer_{x_j+1}}{omjer} = exp(\beta_j(x_j+1) - \beta_j x_j) = exp(\beta_j)$$

Ovo nam govori da promjena značajke za 1, mijenja odnos omjera za faktor od $exp(\beta_j)$

Na primjer, ako imamo $omjer = 2$, to znači da je vjerojatnost za $y = 1$ dvostruko veća od $y = 0$. Ako imamo težinu $\beta_j = 0.7$, tada povećavanje dotične značajke za jednu jedinicu množi $omjer$ s $exp(0.7)$ (otprilike 2), te je sada vjerojatnost za $y = 1$ klasu četiri puta veća od $y = 0$. Također interpretacije variraju prema vrsti značajke.

- **Numerička značajka** - Povećanje numeričke značajke x_j za jednu jedinicu mijenja omjer između vjerojatnosti pozitivne i negativne klase t.d. $omjer_{x_j+1} = exp(\beta_j)omjer$
- **Binarna značajka** - Promjena značajke x_j iz referentne kategorije(vrijednost značajke kodirana s 0) u drugu kategoriju mijenja omjer između vjerojatnosti pozitivne i negativne klase t.d. $omjer_{x_j+1} = exp(\beta_j)omjer$
- **Kategorijska značajka s više kategorija** - Značajka s fiksnim brojem mogućih vrijednosti. Kategorijsku značajku označavamo tzv. one-hot-encoding, što znači da svaka kategorija ima svoj binarni stupac. Za kategorijsku značajku s L kategorija potrebno je samo $L - 1$ stupaca , jer bi L-ti stupac imao suvišne informacije (npr. Kada svi stupci 1 do L-1 imaju vrijednost 0 za jednu instancu, znamo da kategorička značajka ovaj primjerak preuzima kategoriju L). Interpretacija svake kategorije tada je isto što i interpretacija za binarne značajke.

2.3 Stabla odluke

Modeli linearne regresije i logističke regresije nisu podobne u situacijama kada je odnos između ulaznih i izlaznih podataka ne linearan ili gdje značajke međusobno zavise jedna o drugoj. Iz toga razloga koristimo se modelom Stabla odluka. Ideja ovog modela je podijeliti postupak predviđanja na niz izbora prema određenim značajkama, počevši od korijena stabla do listova gdje saznajemo kojem podskupu skupa podataka pripada instanca. Stabla odluke laka su za shvaćanje te mogu biti pretvoreni u *if-else pravila* pogodna za korištenje.

Postoji nekoliko različitih algoritama za konstrukciju stabla odluke, ali oni su svi gotovo varijante istog principa: algoritmi grade stablo na pohlepan način počevši od korijena, odabirom "najinformativnije" značajke u svakom koraku. Kao što smo spomenuli u svakoj fazi odabiremo značajku koja nam daje najviše informacija. Želimo kvantificirati koliko informacije nam je dano sa znanjem novih činjenica, kodiranjem ovoga bavi se Informacijska teorija.

Stabla odluke razlikujemo za *regresijske* probleme i *klasifikacijske* probleme:

Ako se radi o regresijskom problemu gdje se naši podaci sastoje od N ulaza $x_i = (x_{i1}, x_{i2}, ., x_{ip})$ te odgovora y. Particioniramo li sada ulazni prostor R^p na M dijelova $R_1, R_2, ..., R_M$. U modelu stabla odluke sljedeća formula opisuje vezu između izlazne varijable y i ulaznih značajki x.

$$\hat{y} = \hat{f}(x) = \sum_{m=1}^{M} c_m I_{\{x \in R_m\}}$$

$I_{\{x \in R_m\}}$ predstavlja funkciju identitetu skupa R_m, ako instanca upada u skup R_m, $m \in \{1, 2, ...M\}$ tada je predviđanje modela c_m.

Obično se kao procjenitelj za c_m uzima $\widehat{c_m} = ave(y_i | x_i \in R_m)$. Zadatak algoritma kojim želimo naučiti strukturu stabla je pronaći najbolju particiju ulaznog prostora R^p s obzirom na $\sum(y_i - f(x_i)^2)$. Koristimo se pohlepnim algoritmom koji rekurzivno dijeli prostor izlaznih vrijednosti na podprostore. Krajnje $R_1, R_2, ..., R_m$ dobivamo tako da razmatramo j-tu značajku i $s \in \mathbb{R}$, u svakom koraku algoritma odabiremo j, s koji smanjuju sumu kvadrata.

$$P_1(j, s) = \{X | X_j \leq s\}, P_2(j, s) = \{X | X_j > s\}$$

$$\min_{j,s}[\min_{c_1} \text{cost}(P_1) + \min_{c_2} \text{cost}(P_2)]$$

.

gdje je $cost(D) = \sum_{x_i \in D}(y_i - c)^2$
Unutarnja minimizacija je rješiva sa:

$$\widehat{c} = ave(y_i | x_i \in P_i(j, s))$$

Ako je problem koji trebamo naučiti klasifikacijski, odnosno izlazna varijabla y poprima $\{1, 2, ..., K\}$ vrijednosti. Jedina stvar koju moramo promijeniti je *cost* funkcija tako da definiramo sljedeće mjere, **Gini-index** i **Entropija**.

$$\widehat{p_c}(D) = \frac{1}{D} \sum_{x_i \in D} I(y_i \neq c)$$

Entropija je $cost(D) = \sum_{c=1}^{K} \hat{p}_c(D) log(\hat{p}_c(D))$

Gini-index je $cost(D) = \sum_{c=1}^{K} \hat{p}_c(D)(1 - \hat{p}_c(D))$

Interpretacija Stabla odluke

Interpretacija je dosta jednostavna: Polazeći od čvora korijena prema listovima prolazimo kroz unutarnje čvorove stabla te kada dođemo do listova oni govore kojem krajnjem podskupu pripada ulazna instanca te predviđena vrijednost aritmetička sredina izlaznih podataka y iz tog podskupa ako se radi o regresiji, odnosno .

Također kod modela stabla možemo izračunati i **Važnost značajke** Ukupna važnost značajke u stablu odluka može se izračunati na sljedeći način: Prođite kroz sva raskrižja u kojima je značajka korištena i izmjerimo koliko se smanjila varijanca ili Ginijev indeks u odnosu na nadređeni čvor. Zbroj svih važnosti skalira se na 100. To znači da se svaka važnost može tumačiti kao udio u ukupnoj važnosti modela.

Poglavlje 3

Model-agnostičke metode

3.1 Graf djelomične ovisnosti

Vizualizacija je jedna od najmoćnijih interpretacijskih alata. Grafički prikaz vrijednosti $\widehat{F}(x)$ kao funkcije svojih argumenata daje sveobuhvatan sažetak njegove ovisnosti na zajedničkim vrijednostima ulaznih varijabli. Nažalost, takva je vizualizacija ograničena na male dimenzije. Stoga je korisno moći vidjeti djelomičnu ovisnost aproksimacije $\widehat{F}(x)$ na odabranim malim skupinama ulaznih varijabli. Iako takav pristup vizualizaciji rijetko može pružiti sveobuhvatan prikaz aproksimacije, često može stvoriti korisne tragove.

Graf djelomične ovisnosti (Kratko PDP (Partial Dependence Plot) ili PDplot) prikazuje učinak koji jedna ili dvije značajke imaju na predviđeni ishod modela strojnog učenja. Graf djelomične ovisnosti prikazuje **Funkciju djelomičnih ovisnosti**, može pokazati je li odnos ishoda modela i određene značajke linearan, monoton ili složeniji od toga.

Definicija 3.1.1. *Skup* $U = \{x^{(i)}, y^{(i)}\}_{i=1}^{N}$*, gdje je* $x_i = (x_1^{(i)}, x_2^{(i)}, ..., x_p^{(i)})$ *vektor ulaznih značajki, dok je* $y^{(i)}$ *ciljana vrijednost.Skup* U *naziva se skup ulaznih podataka, na njemu treniramo model i dobivamo aproksimaciju* $\widehat{f}$ *tada je* ***funkcija djelomične ovisnosti*** *definirana sa:*

$$\hat{f}_{x_S}(x_S) = E_{x_C}[\hat{f}(x_S, x_C)] = \int \hat{f}(x_S, x_C) d\mathbb{P}(x_c)$$

Skup $S \subset \{1, 2, ..., p\}$*, skup* $C = \{1, 2, .., p\} \backslash S$*, p je broj ulaznih značajki. Vektor* x_S *podvektor je ulaznog vektora* $x = (x_1, x_2, .., x_p)$*, koji sadrži samo značajke iz* S*.*

U skupu S nalaze se značajke čiji odnos s ciljanom varijablom proučavamo. Uobičajeno u skupu S nalaze se 1 ili 2 značajke. Vektori značajki x_C i x_S čine potpun ulazni vektor značajki.

Pošto ne poznajemo distribucije pojedinih značajki moramo se pouzdati u procjenu Graf djelomične ovisnosti iz podataka koje imamo to činimo na sljedeći način.

$$\hat{f}_{x_S}(x_S) = \frac{1}{n}\sum_{i=1}^{n}\hat{f}(x_S, x_C^{(i)})$$

U prikazanoj formuli $x_C^{(i)}$, predstavlja stvarne vrijednosti značajki iz skupa za treniranje svih značajki koje se nalaze u C čiju povezanost s izlazom trenutno ne promatramo, te je n ukupan broj instanci u skupu za treniranje. Jedna od pretpostavki $PDP-a$ jest da značajke iz C nisu korelirane sa značajkama iz S. Ako ova pretpostavka nije zadovoljena prosjeci izračunati za funkciju djelomične ovisnosti uključivat će točke koje se rijetko pojavljuju ili su nemoguće.

Definicija 3.1.2. *Graf djelomične ovisnosti definira se kao* $G(\widehat{f_{x_S}}) = \{(x, y)|\widehat{f_{x_s}}(x) = y, x \in \mathbb{R}^{|S|}\})$

Primjeri.

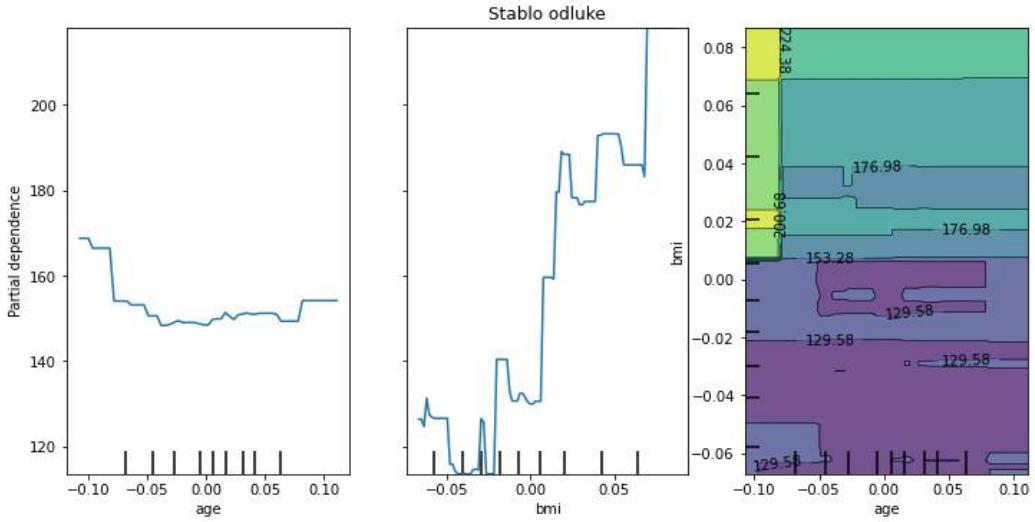
Uzmemo li skup podataka koji sadrži deset osnovnih značajki: dob, spol, indeks tjelesne mase(bmi), prosječni krvni tlak i šest mjerenja krvnog seruma za svakog od $n = 442$ bolesnika s dijabetesom, te je izlazna varijabla y kvantitativna mjera napredovanja bolesti unutar godinu dana nakon početka mjerenja. Ulazni podaci su prije korištenja centralizirani oko 0.

Istrenirali smo 2 modela regresijsko Stablo odluke te neuronsku mrežu, te prikazali njihove grafove djelomične ovisnosti, za značajke dobi, indeks tjelesne mase (bmi) te dobi i indeks tjelesne mase zajedno.

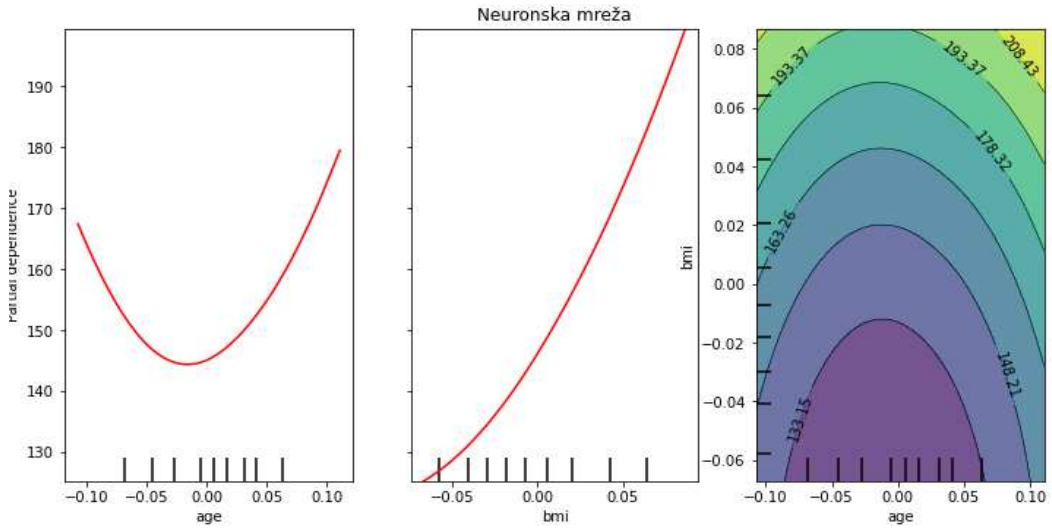
Prednosti ove metode su intuitivnost, jasno je da funkcija $\hat{f}$ vrača prosječnu vrijednost po svim x_C uz zadanu x_S, te tako možemo proučavati odnos značajki iz skupa S i izlaza modela. Jednostavna je za implementaciju. Mane su joj što možemo vizualizirati samo mali broj značajki, realistično samo dvije. Važnija mana je pretpostavka ne koreliranosti između značajki iz skupa S i skupa C, ako su varijable korelirane u izračun $\hat{f}$ ulaze instance koje ne postoje ili su iznimno rijetke što dovodi do mogućih krivih zaključaka. Također može doći do krivih zaključaka ako polovica vrijednosti značajke pozitivno korelirana s izlaznom varijablom, dok je druga polovica negativno korelirana, tada će vrijednost PDP varirati oko 0, te se time može protumačiti da promatrana značajka nema utjecaja na ciljanu varijablu.

Obrazložimo mane na primjerima:

- **Pretpostavka ne koreliranosti** Uzmemo li primjer u kojem je Y izlazna varijabla mjerenje brzine hodanja osoba, značajke skupa X visina i težina, promatramo li parcijalnu ovisnost recimo visine, pretpostavljamo da težina nije korelirana s njom, što



Slika 3.1: Graf djelomične ovisnosti za Stabla odluke



Slika 3.2: Graf djelomične ovisnosti za neuronske mreže

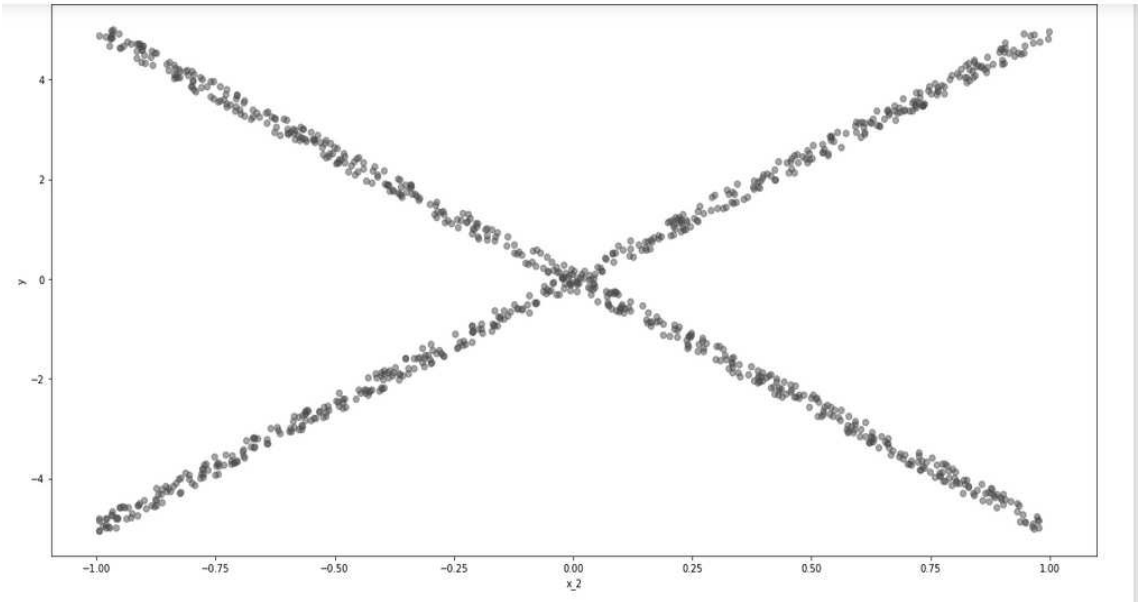
je očito ne točno. Za izračun $\hat{f}$ za zadanu visinu, recimo 200cm, tražimo prosjek preko svih težina u skupu za treniranje što može uključivati težine manje od 50kg, a iz života znamo da je izrazito ne izgledno da osoba ima 200cm i manje od 50kg. U kasnijim poglavljima predstavljamo rješenje za ovu manu.

- **Pozitivna \ negativna koreliranost značajke** Generiramo uzorak prema sljedećim uvjetima s

$$Y = 0.2X_1 - 5X_2 + 10X_2 I(X_3 > 0) + \epsilon$$

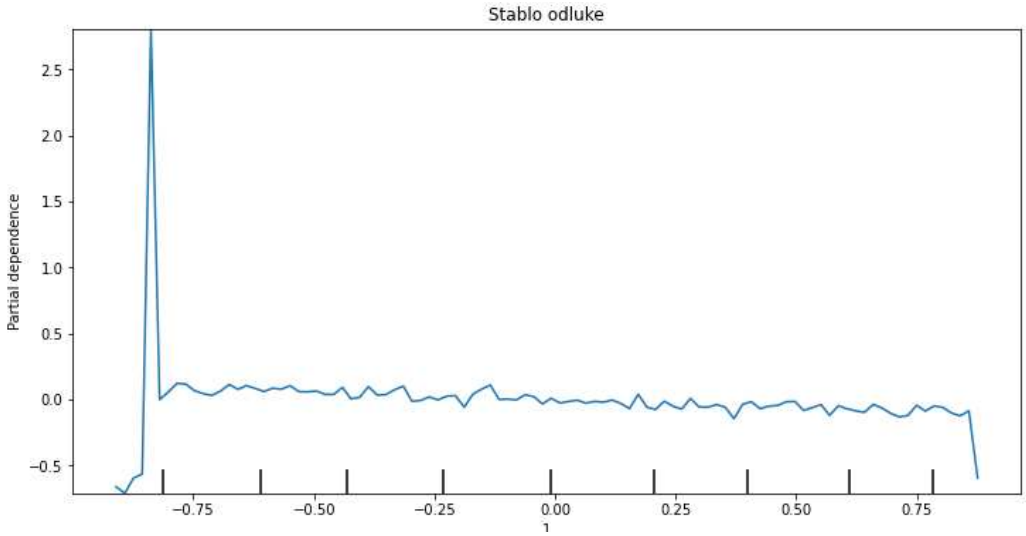
$$\epsilon \overset{iid}{\sim} N(0,1),\ X_1, X_2, X_3 \overset{iid}{\sim} U(-1,1)$$

Nad tim podacima istreniramo Stablo odluke, te prikažemo Graf parcijalne ovisnosti za istrenirani model. PDP sugerira da X_2 u prosjeku ne doprinosi značajno predviđanjem



Slika 3.3: Scatter-plot graf x_2 značajke i y

Y. U svjetlu Figure.3, ovaj je zaključak očito pogrešan. Jasno je da X_2 doprinosit iznosu Y, nego dolazi do svoje vrsnog poništavanja vrijednosti te bi mogli doći do krivog zaključka, o utjecaju značajke na predviđanje.



Slika 3.4: Graf djelomične ovisnosti za Stablo odluke sa generiranim uzorkom

3.2 Grafovi pojedinačnog uvjetnog očekivanja

Graf pojedinačnog uvjetnog očekivanja (Individual Conditional Expectation - ICE) je alat za vizualizaciju modela, dobivenog bilo kojim algoritmom nadziranog učenja. ICE umjesto crtanja prosječne ciljane vrijednosti modela crta N pojedinačnih krivulja uvjetnog očekivanja.

Neka je $\{(x_S^{(i)}, x_C^{(i)})\}_{i=1}^N$ skup ulaznih podatka, gdje je S skup značajki čiji odnos sa ciljanom funkcijom modela proučavamo, te neka je $\hat{f}$ aproksimacija ciljane funkcije koja je rezultat algoritma za učenje, tada je pojedinačno uvjetno očekivanje prikazujem za $i \in \{1, 2, ..., N\}$

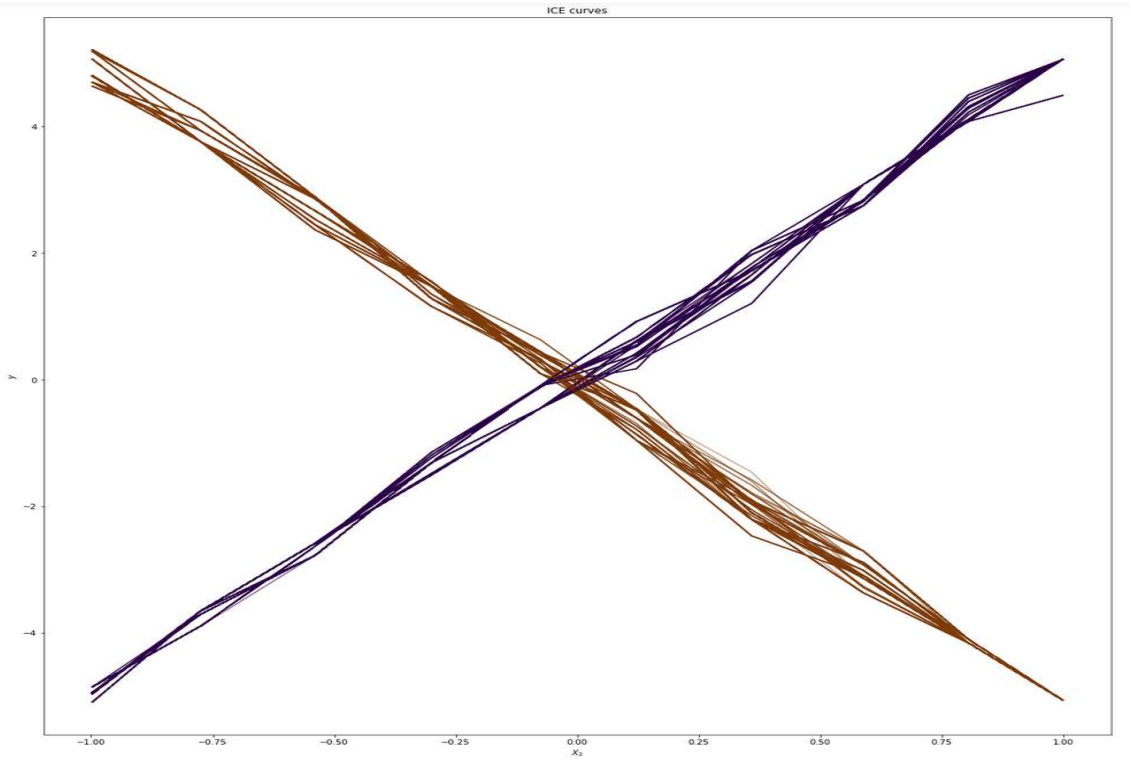
$$\widehat{f_S^{(i)}}(x_S) = \hat{f}(x_S, x_C^{(i)})$$

te metoda grafova pojedinačnog očekivanja crta N funkcija pojedinačnog uvjetnog očekivanja za svaki $x_C^{(i)}$, $i \in \{1, 2, ..., N\}$. Svaka funkcija $\widehat{f_S^{(i)}}$ prikazuje uvjetni odnos između x_S i $\hat{f}$ uz fiksnu vrijednost x_C.

PDP prikazuje prosjek svih ICE krivlji. Ako se ICE metoda primjeni na model, u primjeru za **Pozitivnu\negativnu koreliranost značajki** dobivamo sliku s koje uočavamo da točnije prikazuje odnos ciljane varijable i značajke.

Prednosti ove metode su intuitivnost te mogućnost otkrivanje kompleksnijih odnosa, dok su mane mogućnost prikazivanja više značajki od jedne, još uvijek se zadržava problem

korelacije te je nekada teško vidjeti prosjek.



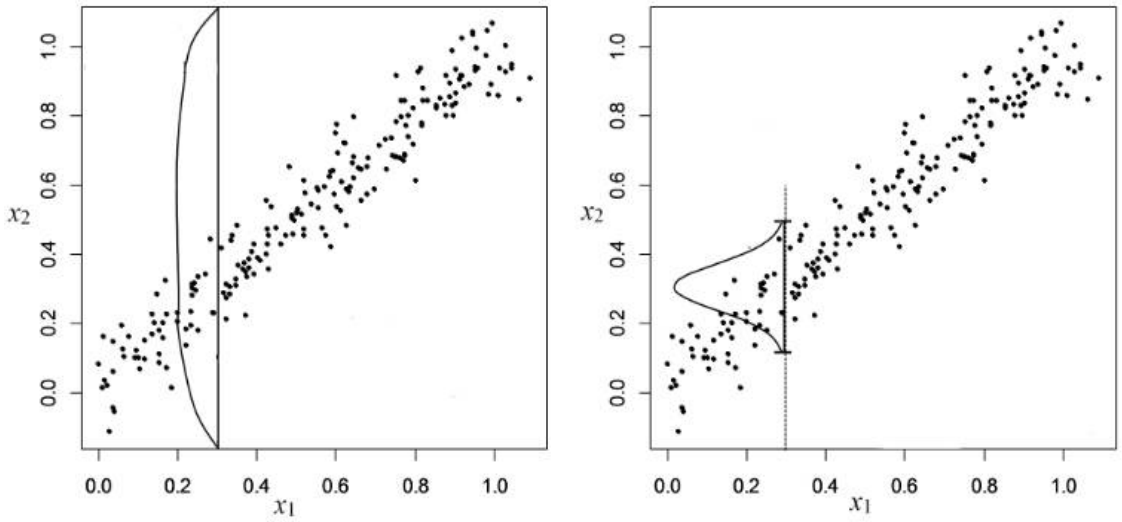
Slika 3.5: ICE metoda primjenjena na slučajno generiranim podacima

3.3 Graf akumuliranih lokalnih učinaka (ALE)

Jedan od problema kod *PDP*-a loša je interpretabilnost u slučaju postojanja korelacije između značajki koje promatramo i onih koje ne promatramo. ALE grafovi prikazuju nam cjelokupno ponašanje modela s obzirom na ulaznu varijablu, te rješava problem s postojanjem korelacija između značajki ili interakcija koje bi mogle utjecati na pouzdanost Grafova djelomične ovisnosti. Kao Grafovi djelomične ovisnosti, ALE grafovi pokazuju oblik - tj. linearnost, monotonost - odnos između predviđanja i ulaznih varijabli, čak i za vrlo složene modele.

Neka je skup $\{x^{(i)}, y^{(i)}\}_{i=1}^{N}$ skup za treniranje, gdje je $x^{(i)} = (x_1^{(i)}, x_2^{(i)}, ..., x_p^{(i)})$ je vektor ulaznih značajki, dok je $y^{(i)}$ ciljana vrijednost, na njemu treniramo model $\hat{f}$. Naš je cilj vizualizirati i razumjeti ovisnost $\hat{f}(x) = \hat{f}(x_1, x_2, ..., x_p)$ istreniranog modela o pojedinim

značajkama, kao i "efekte interakcije" niskog reda među parovima značajki. Prije strogih definicija prikazujemo ovo kroz jednostavan primjer $p = 2$.



Slika 3.6: Prikaz marginalne i uvjetne marginalne distirbucije uzorka $P(X_2|X_1 = 0.3)$

Neka su X_1, X_2 uniformno distribuirane slučajne varijable duž pravca $x_2 = x_1$ kao što vidimo na Slici 3.6. Na lijevoj slici je marginalna distribucija prema kojoj se računa $\widehat{f_{x_S}}$ koja se koristi u PDP-u definirana u Definiciji 3.1.1, dok desna slika prikazuje uvjetnu distribuciju $P(X_2|X_1 = 0.3)$. Iz toga razloga uvodimo novu funkciju koju zovemo **Akumuliranim glavnim lokalnim učinkom od** X_1 definiranu na sljedeći način:

$$f_{1,ALE}(x_1) = \int_{x_{min,1}}^{x_1} \mathbb{E}[f^1(X_1, X_2)|X_1 = z_1]dz_1$$
$$= \int_{x_{min,1}}^{x_1} \int p_{2|1}(x_2|z_1) f^1(z_1, x_2)dx_2dz_1$$

gdje je $f^1(x_1) = \frac{\partial f(x_1,x_2)}{\partial x_1}$ predstavlja lokalni učinak x_1 na f u toči (x_1, x_2), $x_{min,1} = \min_{x_1^{(i)}}\{x_1^{(i)}|i \in \{1, 2, .., N\}\}$. Funkciju $f_{1,ALE}(x_1)$ interpretira se kao akumulirani lokalni učinka od x_1, na sljedeći način, računamo uvjetno očekivanje funkcije f_1 lokalnog učinka te integriramo po svim vrijednostima od $x_{min,1}$ do x_1.

Funkcije ALE glavnog učinka i ALE učinka drugog reda

Sada definiramo ALE glavne učinke funkcije za svaku značajku i ALE učinke drugog reda za svaki par značajki.

Grafovi lokalnih akumuliranih učinaka su grafikoni procjenitelja ovih funkcija, a procjenitelje definiramo u sljedećem odjeljku. Funkcije lokalnih akumuliranih učinaka nisu predviđene za vizualizaciju učinaka trećeg i viših redova, jer je učinke visokog reda teško interpretirati i obično nisu prevladavajući kao glavni i učinci drugog reda. Uvodimo notaciju za ovu sekciju p_j marginalna distribucija značajke $j \in \{1, 2, ..., p\}$ na domeni $S_j = [x_{min,j} - \epsilon, x_{max_j} + \epsilon]$, $0 < \epsilon < 1$ gdje su $x_{min,j} = \min_{x_j^{(i)}}\{x_j^{(i)} | i \in \{1, 2, .., N\}\}$ $x_{max,j} = \max_{x_j^{(i)}}\{x_j^{(i)} | i \in \{1, 2, .., N\}\}$. Sada za $K \in \mathbb{N}$ i $j \in \{1, 2, .., p\}$ neka je $P_j^K = \{z_{k,j}^K : k = 0, 1, ..., K\}$ particija S_j u K intervala t.d je $z_{0,j}^K = x_{min,j}$ i $z_{K,j}^K = x_{max,j}$, također definiramo $\delta_{j,k} = max\{|z_{k,j}^K - z_{k-1,j}^K| : k = 1, 2..., K\}$ što predstavlja finoću particije. Još za svaki $x \in S_j$ definiramo $k_j^K(x) = k$, gdje je $x \in definicijageneralizacijajefunkcije\mathrm{f}_{1,ALE}(\Delta)$ iz gornjeg primjera gdje f nije nužno diferencijabilna i za bilo koji $p \geq 2$. Generalizacija bitno zamjenjuje derivaciju i integral sa ograničenim konačnim sumama.

Definicija 3.3.1 (Necentrirani ALE glavni učinak)**.** *Neka je $j \in 1, 2, ., , , p$ bilo koja značajka, i pretpostavimo da postoji niz particija $(P_j^K : K = 1, 2, ..)$ takav da je $\lim_{K\to\infty} \delta_{j,K} = 0$. Ako sljedeći limes postoji i neovisan je određenom nizu particija $(P_j^K : K = 1, 2, ..)$ ([3] u Dodatku A navedeni su dovoljni uvjeti za postojanje i jedinstvenost ovog limesa), definiramo funkciju* ***Necentriranog ALE glavnog učinaka*** *za značajku X_j*

$$g_{j,ALE}(x_j) = \lim_{K\to\infty} \sum_{k=1}^{K} \mathbb{E}[f(z_{k,j}^K, X_{\setminus j}) - f(z_{k-1,j}^K, X_{\setminus j}) | x_j \in < (z_{k-1,j}^K), z_{k,j}^K]]$$

Sljedeći teorem, čiji je dokaz također u [3], navodi da je za diferencijabilnu f funkciju, necentrirani ALE glavni učinak X_j ima ekvivalentan, ali jednostavniji oblik

Teorem 3.3.2 (Necentrirani ALE glavni učinak za diferencijabilnu funkciju f)**.** *Neka je $f^j(x_j, X_{\setminus j}) = \frac{\partial f(x_j, X_{\setminus j})}{\partial x_j}$ označavaju parcijalnu derivaciju $f(x)$ po x_j kada ta derivacija postoji. Nastavno na Definiciju. 1 pretpostavimo:*

- *$f(x_j, X_{\setminus j})$ je diferencijabilna u x_j na S_j*
- *$f^j(x_j, X_{\setminus j})$ je neprekidna u x_j na S_j*
- *$E[f^j(x_j, X_{\setminus j}) | X_j = z_j]$ je neprekidana u z_j na S_j*

tada za svaki $x_j \in S_j$ vrijedi:

$$g_{j,ALE}(x_j) = \int_{x_{min,j}}^{x_j} E[f^j(x_j, X_{\setminus j}) | X_j = z_j] dz_j$$

Glavni (centrirani) ALE učinak za X_j, označen sa $f_{j,ALE}(x_j)$, definiran je kao $g_{j,ALE}(x_j)$, ali centriran tako da $f_{j,ALE}(X_j)$ ima očekivanje nula u odnosu na marginalnu distribuciju X_j.

$$\begin{aligned} f_{j,ALE}(x_j) &= g_{j,ALE}(x_j) - E[g_{j,ALE}(X_j)] \\ &= g_{j,ALE}(x_j) - \int p_j(z_j) g_{j,ALE}(z_j) dz_j \end{aligned}$$

Dalje definiramo ALE učinak drugog reda. Za svaki par indeksa $\{j,l\} \subset \{1,2,.,,,p\}$, neka $X_{\setminus\{j,l\}}$ označava vektor svih značajki osim j-te i l-te odnosno , $X_{\setminus\{j,l\}} = (X_k : k = 1,2,.,,,p; k \neq j, k \neq l)$. Općenito f, ne centrirani ALE učinak drugog reda od $\{X_j, X_l\}$ definiran je slično kao ne centrirani glavni učinak, s tim što zamjenjujemo 1-dimenzionalne konačne razlike 2-dimenzionalnim konačnim razlikama drugog reda na 2-D rešetki koja je kartezijanski produkt 1-D particija S_j i S_l, a zbroj je preko 2-D rešetke.

Definicija 3.3.3. *[Necentrirani ALE učinak drugog reda] Neka je $\{j,l\} \in \{1,2,.,,,p\}$ bilo koji podskup značajki, te pretpostavimo da postoje nizovi particija $(P_j^K : K = 1,2,..)$ $(P_l^K : K = 1,2,..)$ takav da $\lim_{K\to\infty} \delta_{j,K} = 0$ i $\lim_{K\to\infty} \delta_{l,K} = 0$. Ako sljedeći limes postoji i neovisan je određenom nizovima particija , definiramo funkciju* ***Necentriranog ALE učinaka drugog*** *za značajke X_j i X_l s istim oznakama napomenutim u ovoj sekciji*

$$h_{\{j,l\},ALE}(x_j, x_l) = \lim_{K\to\infty} \sum_{k=1}^{k_j^K(x_j)} \sum_{m=1}^{k_j^K(x_l)} \mathbb{E}[\Delta(K,k,m; X_{\setminus\{j,l\}}) | x_j \in (z_{k-1,j}^K, z_{k,j}^K], x_l \in (z_{k-1,l}^K, z_{k,l}^K]]$$

gdje je

$$\Delta_f(K,k,m; X_{\setminus\{j,l\}}) = [f(z_{k,j}^K, z_{m,l}^K, X_{\setminus j}) - f(z_{k-1,j}^K, z_{m,j}^K, X_{\setminus j})] - [f(z_{k,j}^K, z_{m,l}^K, X_{\setminus j}) - f(z_{k-1,j}^K, z_{m-1,j}^K, X_{\setminus j})]$$

razlika funkcije $f(\mathbf{x}) = f(x_j, x_l, \mathbf{x}_{\setminus\{j,l\}})$ s obzirom na (x_j, x_l) preko ćelija, $(z_{k-1,j}^K, z_{k,j}^K] \times (z_{m-1,l}^K, z_{m,l}^K]$, kartezijskog produkta 2-D rešetke particija P_j^K i P_l^K .

Slično kao što smo imali teorem za diferencijalnu funkciju f za ALE- glavni učinak, imamo i teorem za

Teorem 3.3.4 (Necentrirani ALE učinak drugog reda za diferencijabilnu funkciju f)**.** *Neka je $f^{\{j,l\}}(x_j, x_l, X_{\setminus\{j,l\}}) = \frac{\partial f(x_j, x_l, X_{\setminus\{j,l\}})}{\partial x_j \partial x_l}$ označava parcijalnu derivaciju drugog reda $f(x)$ sa obzirom na x_j i x_l kada ona postoji. Nastavno na Definiciju. 3 pretpostavimo:*

- *$f(x_j, x_l, X_{\setminus\{j,l\}})$ je diferencijabilna po varijablama x_j i x_l*
- *$f^{\{j,l\}}(x_j, x_l, X_{\setminus\{j,l\}})$ je neprekidna u $(x_j, x_l, X_{\setminus\{j,l\}}) \in S$*

- $E[f^{\{j,l\}}(x_j, x_l, X_{\setminus\{j,l\}})|X_j = z_j, X_l = z_l]$ *je neprekidana u* $(z_j, z_l) \in S_j \times S_l$ *tada za svaki* $(x_j, x_l) \in S_j \times S_l$,

$$h_{\{j,l\},ALE}(x_j, x_l) = \int_{x_{min,j}}^{x_j} \int_{x_{min,l}}^{x_l} E[f^j(x_j, x_l, \boldsymbol{X}_{\setminus\{j,l\}})|X_j = z_j, X_l = z_l]dz_l dz_j$$

Centrirani ALE učinak drugog reda za značajke X_j, X_l označavamo sa $f_{\{j,l\},ALE}(x_j, x_l)$ definira se kao $h_{\{j,l\},ALE}(x_j, x_l)$ koji je "dvostruko-centriran" tako da ima 0 srednju vrijednost sa obzirom na marginalne distribucije (X_j, X_l) te su glavni učinci X_j i X_l također 0. Posljednji zahtjev ispunjen je oduzimanjem ALE glavna učinka za X_j i X_l. Izvod i objašnjenje mogu biti pronađeni u [3].

$$g_{\{j,l\},ALE}(x_j, x_l) = h_{\{j,l\},ALE}(x_j, x_l) - \int_{x_{min,j}}^{x_j} \int p_{l|j}(z_l|z_j) \frac{\partial h_{\{j,l\},ALE}(z_j, z_l)}{\partial z_j} dz_j dz_l$$

$$- \int_{x_{min,j}}^{x_j} \int p_{j|l}(z_j|z_l) \frac{\partial h_{\{j,l\},ALE}(z_j, z_l)}{\partial z_l} dz_j dz_l$$

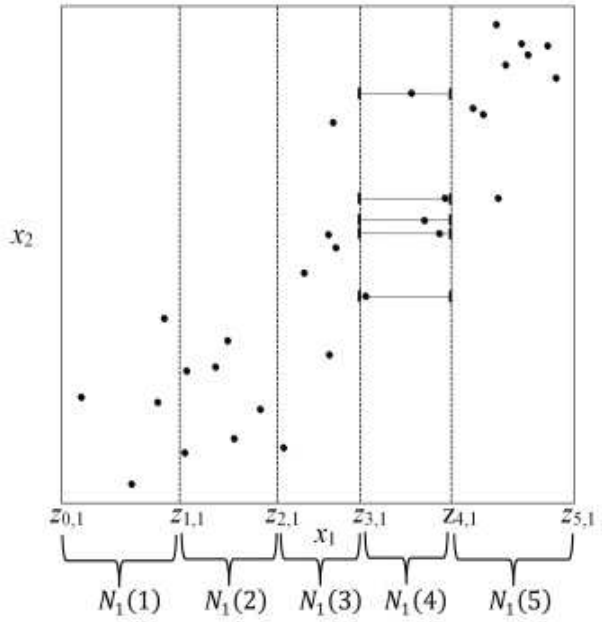
$$f_{\{j,l\},ALE}(x_j, x_l) = g_{\{j,l\},ALE}(x_j, x_l) - \int \int p_{l,j}(z_l, z_j) g_{\{j,l\},ALE}(z_j, z_l) dz_j dz_l$$

Procjenitelji za $f_{j,ALE}(x_j)$ **i** $f_{\{j,l\}ALE}(x_j, x_l)$

Sada ćemo se fokusirati na procjenu funkcija ALE učinka prvog i drugog reda, jer su one najčešće i korisne za vizualizaciju u Grafovima akumuliranih lokalnih učinaka, te nam omogućavaju interpretaciju modela. Procjenitelj za $\widehat{f_{j,ALE}}$ dobiven je korištenjem podataka koje imamo, stoga niz particija zamjenjujemo *Definicija* 3.3.1. i 3.3.3. fiksnom particijom uzorka $\{x_{i,J} : i = 1, 2, ..., n\}$ te uvjetno očekivanje mijenjamo sa srednjom vrijednošću uzorka $x_{i,\setminus J} : i = 1, 2, ..., n$ uz uvjet da $x_{i,J}$ ulaze u istoj ćeliju particije. Još specificiramo, za svaku $j \in \{1, 2, ..., p\}$ neka je $\{N_j(k) = (z_{k-1,j}, z_{k,j}]$|k=1,2,...K$\}$, gdje je K je broj ćelija u nekoj fiksiranoj dovoljno finoj particiji uzorka $\{x_{i,J} : i = 1, 2, ..., n\}$, te $n_j(k) = |N_j(k)|$. Najčešće se u procijeni *ALE* funkcija uzimamo da je $z_{k,j} = \frac{k}{K} * \underset{x_j}{maxS}_j + (1 - \frac{k}{K}) \underset{x_j}{minS}_j$. Slika. 3.7 prikazuje na primjeru gdje je dimenzija ulaznog prostora $p = 2$ gdje je $j = 1$ i $K = 5$. Sa $k_j(x)$ označavamo za vrijednost značajke x_j označavamo indeks i t.d. $x \in (z_{i-1,j}, z_{i,j}]$

Sada za općeniti p procjenjujemo glavni efekt funkcije $f_{j,ALE}$ za prediktor X_j prvo računamo ne centrirani efekt $g_{j,ALE}$ iz *Definicije* 3.3.1.

$$\widehat{g}_{j,ALE}(x) = \sum_{k=1}^{k_j(x)} \frac{1}{n_j(k)} \sum_{\{i:x_{i,j} \in N_j(k)\}} [f(z_{k,j}, x_{i,\setminus j}) - f(z_{k-1,j}, x_{i,\setminus j})]$$



Slika 3.7: Prikaz particije za svojstvo X_1 kod računanja procijenitelja za ALE

za svaki $x \in (z_{0,j}, z_{K,j}]$. Ne centrirani ALE glavni učinak smo centrirali sa oduzimanjem procjenu očekivanja $\mathbb{E}[g_{j,ALE}(X_j)]$

$$\widehat{f}_{j,ALE(x)} = \widehat{g}_{j,ALE}(x) - \frac{1}{n}\sum_{i=1}^{n}\widehat{g}_{j,ALE}(x_{i,j})$$

Slično definiramo i procjenitelja za za ALE učinak drugog reda za par značajki (X_j, X_l) particioniramo ulazni uzorak tih 2 varijabli $\{x_j^{(i)}, x_l^{(i)} | i = 1, 2, ..., N\}$ u mrežu od K^2 ćelija dobivene kartezijskim produktom individualnih 1-dimenzionalnih particija skupova S_j, S_l. Neka su k, m brojevi između 1 i K koji definiraju pravokutnik na kartezijevom produktu particija, te k se odnosi na svojstvo X_j dok se m odnosi na svojstvo X_l. U analogiji sa $N_j(k)$ i $n_j(k)$ definiramo $N_{\{j,l\}}(k, m) = N_j(k) \times N_l(m) = (z_{k-1,j}, z_{k,j}] \times (w_{m-1,j}, w_{m,j}]$, dok je $n_{\{j,l\}}(k, m) = |N_{\{j,l\}}(k, m)|$.

Definiramo sada $\widehat{h}_{\{j,l\}}$ ne centriranu verziju ALE drugog reda za značajke X_j i X_l:

$$\widehat{h}_{\{j,l\}}(x_j, x_l) = \sum_{k=1}^{k_j(x_j)}\sum_{m=1}^{k_l(x_l)}\frac{1}{n_{\{j,l\}}(k, m)}\sum_{\{i|X_{i,\{j,l\}}\in N_{\{j,l\}}(k,m)\}}\Delta_f^{\{j,l\}}(K, k, m; \mathbf{x}_{i,\setminus\{j,l\}})$$

$$\Delta_f(K, k, m; \mathbf{x}_{i,\setminus\{j,l\}}) = [f(z_{k,j}, z_{m,l}, \mathbf{x}_{i,\setminus\{j,l\}}) - f(z_{k-1,j}, z_{m,j}, \mathbf{x}_{i,\setminus\{j,l\}})]$$

$$-[f(z_{k,j}, z_{m,l}, \mathbf{x}_{i,\setminus\{j,l\}}) - f(z_{k-1,j}, z_{m-1,j}, \mathbf{x}_{i,\setminus\{j,l\}})]$$

Za procjenu $f_{\{j,l\},ALE}(x_j, x_l)$ još trebamo izračunati procjenu od $g_{\{j,l\},ALE}(x_j, x_l)$

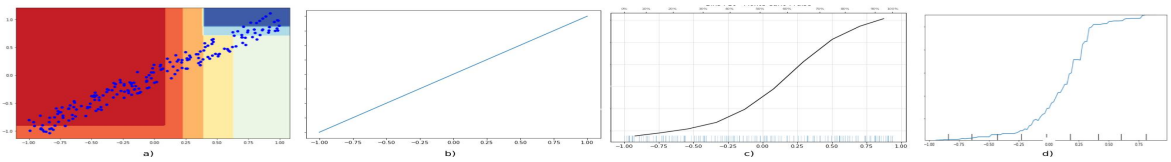
$$g_{\{j,l\},ALE}(x_j, x_l) = \widehat{h}_{\{j,l\},ALE}(x_j, x_l) - \sum_{k=1}^{k_j(x_j)} \frac{1}{n_j(k)} \sum_{\{i:x_{i,j}\in N_j(k)\}} [\widehat{h}_{\{j,l\},ALE}(z_{k,j}, x_{i,l}) - \widehat{h}_{\{j,l\},ALE}(z_{k-1,j}, x_{i,l})]$$

$$- \sum_{m=1}^{k_l(x_l)} \frac{1}{n_l(m)} \sum_{\{i:x_{i,l}\in N_l(k)\}} [\widehat{h}_{\{j,l\},ALE}(z_{i,j}, x_{m,l}) - \widehat{h}_{\{j,l\},ALE}(z_{i,j}, x_{m-1,l})]$$

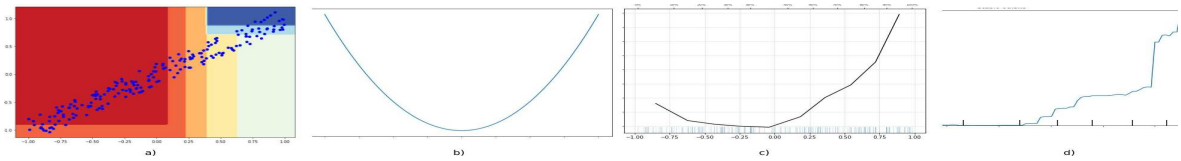
$$\hat{f}_{\{j,l\},ALE}(x_j, x_l) = \hat{g}_{\{j,l\},ALE}(x_j, x_l) - \frac{1}{n} \sum_{i=1}^{n} \hat{g}_{\{j,l\},ALE}(x_{i,j}, x_{i,l})$$

Primjeri.

Vratimo se primjer iz uvoda o ALE-metodi gdje smo imali 2. varijable X_1 i X_2 dakle $p = 2$, te slučajne varijable prate uniformnu distribuciju po segmentu $x_2 = x_1$ s nezavisnom varijablom $N(0, 0.005)$ dodanom oba prediktora. Definiramo i varijablu cilja $Y = X_2^2 + X_1$



Slika 3.8: Prikazujemo na slici **a)** prikazuje korelaciju značajki, te odluke stabla regresije istreniranog na podacima **b)** prikazuje pravi odnos između značajke x_1 i y c) prikazuje ALE-graf i **d)**Prikazuje PDP značajke x_1 i y



Slika 3.9: Prikazujemo na slici **a)** prikazuje korelaciju značajki, te odluke stabla regresije istreniranog na podacima **b)** prikazuje pravi odnos između značajke x_2 i y c) prikazuje ALE-graf između značajke x_2 i y i **d)**Prikazuje PDP značajke x_2 i y

3.4 Lokalno interpretabilna model-agnostična objašnjenja

Sada predstavljamo Lokalno interpretabilne model-agnostičnu metodu (Locally Interpretable Model-angostic Explonation- LIME). Opći cilj LIME-A je identificirati interpretabilni model preko "*interpretabilne reprezentacije*" koji je lokalno vjeran klasifikatoru.

Prije nego što krenemo, treba razjasniti da je za objašnjenje koje konstruiramo važno razdvojiti značajke i interpretabilnu reprezentaciju podataka. Interpretabilni modeli trebaju koristiti interpretabilnu reprezentaciju podataka, bez obzira na to koje značajke koristi algoritam za učenje. Kao primjer uzmimo klasifikaciju teksta, interpretativna reprezentacija koristi riječi dok model koristi vjerojatno koristi kompleksnije oblik podataka kao što su tzv. *word-embedding*. U ovom odjeljku označavamo $x \in R^d$ kao originalnu reprezentaciju instance koju objašnjavamo, dok $x' \in \{0,1\}^{d'}$ binarnu vektor za interpretacijsku reprezentaciju. Neka je model koji objašnjavamo $f : \mathbb{R}^d \to \mathbb{R}$

U ovom odjeljku definiramo objašnjenje kao model $g \in G$ gdje je G klasa Samo-objašnjivih modela kao što su linearni modeli, stabla odluke itd. koja lokalno aproksimiraju funkciju modela f. Domena modela g je $\{0,1\}^{d'}$ objašnjivi model prima informaciju o prisustvu elementa interpretabilne reprezentacije podatka ili ne, npr. 1 signalizira da slika za koju želimo interpretaciju sadrži psa, dok 0 signalizira njegovu odsutnost. Sa $\Omega(g)$ označavamo mjeru kompleksnosti objašnjenja $g \in G$, za Stabla odluke Ω može biti dubina stabla, dok za linearnu regresiju to može biti, broj ne nula težina. Sa $\pi_x(z)$ označavamo udaljenost između z i x, te tako njome definiramo lokalnost oko x. Konačno neka je $\mathcal{L}(f, g, \pi_x)$ mjera koliko je g ne točan u svojoj aproksimaciji f dok je $\Omega(g)$ njegova složenost dovoljno niska da je shvatljiv ljudima ljudima. Objašnjenje proizvedeno od LIME-a je dobiveno sljedećim načinom:

$$\xi(x) = \underset{g \in G}{\operatorname{argmin}} \mathcal{L}(f, g, \pi_z) + \Omega(g) \tag{3.1}$$

Ovakva formulacija je korištena s različitim objašnjenjima g iz različitih obitelji funkcija G, različitim funkcijama vjernosti $\mathcal{L}$ i mjerama složenosti Ω.

Opišimo ovu metodu na konkretnom primjeru neka je klasa G klasa linearnih modela takva da $g(z') = w_g \cdot z'$. Koristimo lokalno π_x otežanu sumu kvadrata kao $\mathcal{L}$, te neka je procjena udaljenosti $\pi_x(z) = exp(-D(x,z)^2)/\sigma^2$ gdje je D neka funkcija udaljenosti (Euklidska za slike ili kosina sličnost za tekst), dok je σ unaprijed definirana konstanta:

$$\mathcal{L}(f, g, \pi_x) = \sum_{z, z' \in \mathcal{Z}} \pi_x(z)(f(z) - g(z'))^2$$

gdje je $\mathcal{Z}$ slučajno generiran poduzorak oko x koji se sastoji od N točaka za neki $\epsilon > 0$

$\mathcal{Z} = \{(z, z')|z$ se nalazi u okolini od x zadanoj s $\pi_x(z)$, z' interpretabilna reprezentacija od $z\}$

.

Bilo koji izbor interpretabilnih reprezentacija i familije funkcija G ima neke svojstvene nedostatke. Prvo, dok se model tretira kao crna kutija, određeni skupovi samo-objašnjivih modela G neće biti dovoljno moćni da objasne određena ponašanja. Treba napomenuti da naš izbor za G (linearni modeli) znači da ako je temeljni model izrazito ne linearan na mjestu predviđanja možda neće G neće biti vjerodostojno objašnjenje. Različite implementacije ovoga algoritma imaju različite načine konstrukcije ovog skupa $\mathcal{Z}$, implementacija koju ćemo mi koristiti u primjeru je dostupna na linku LIME (Implementacija dolazi od autora ove metode) uzima za konstantu $\sigma = 0.75 * \sqrt{d}$ gdje je d broj ulaznih značajki. Za ulaznu vrijednost x iz skupa za treniranje generiramo lokalni skup instanci na kojem ćemo trenirati samo-objašnjiv model(u ovoj implementaciji to je ridge linearna regresija) na N' uzoraka $\{x'^{(i)}|i \in \{1, 2, ..., N'\}\}$. Svaka značajki $x_j'^{(i)}$ generirana iz distribucije $N(x_j, \sigma_j)$. σ_j uzoračka varijanca izračunata za značajku j na originalnom skupu za treniranje.

Primjer.

Za demonstraciju LIME metode koristimo skup podataka za procjenu rizika kod revizija tvrtki, gdje se ispituju mnogi čimbenici iz različitih područja poput prošlih evidencija ureda za reviziju, revizije paragrafa, izvješća o uvjetima okoliša, čvrstog sažetaka reputacije, izvješća o tekućim izdanjima, evidencije vrijednosti i dobiti, evidencije o gubicima, pratećih izvještaja itd. Dubinski intervjui s revizorima procjenjuju se važni faktori rizika i izračunava se njihova vjerojatnost postojanja iz sadašnjih i prošlih podataka. Treniramo model *Random Forest* te na njega primjenjujemo LIME implementaciju sa spomenutog linka te dobivamo sljedeće rezultate.

Sa slike 3.10 vidimo svojstva koje su najvažnija te njihov doprinos u donošenju odluke.

Navodimo još jedan interesantan primjer LIME metode u klasifikaciji teksta istrenirali smo *Support vector machine* klasifikator na skupu podataka tekstova iz Scikit-learn 20newsgroups, gdje izabiremo koristiti samo tekstove iz sljedečih područja:

'alt.atheism', 'soc.religion.christian', 'comp.graphics', 'sci.med'.

Nakon treniranja modela, knjižnicom *eli5* istražujemo zbog kojih riječi je klasificirao određen tekst kao takav na slici ispod vidimo objašnjenje zašto tekst klasificiramo kao medicinski tekst te prema kojim riječima je donio taj zaključak.

Slika 3.10: Prikaz rezultate LIME metode na skup podataka u reviziji tvrtki

y=alt.atheism (probability **0.007**, score **-4.872**) top features		**y=comp.graphics** (probability **0.035**, score **-3.270**) top features		**y=sci.med** (probability **0.942**, score **3.781**) top features		**y=soc.religion.christian** (probability **0.016**, score **-4.105**) top features	
Contribution[?]	**Feature**	**Contribution**[?]	**Feature**	**Contribution**[?]	**Feature**	**Contribution**[?]	**Feature**
+0.847	people	+1.068	using	+1.379	effects	+0.803	people
+0.373	of people	+0.644	available	+0.830	therapy	+0.308	it is
+0.309	and error	+0.579	help	+0.759	brain	+0.250	error but
+0.295	is any	+0.146	and all	+0.578	people using	+0.226	serotonin questions
+0.280	it is	+0.122	antidepressant it	+0.478	help	+0.224	hang in
+0.262	a stimulating	+0.118	set	+0.323	chemistry	+0.204	antidepressant it
+0.259	without restating	+0.116	and error	+0.297	questions	+0.196	have answers
+0.258	someday a	+0.112	the thread	+0.268	lot	+0.186	available and
+0.256	have answers	+0.111	here zoloft	+0.223	conditions	+0.166	people using
+0.249	trial and	+0.106	help manage	+0.199	medications	+0.165	someday a
+0.249	that antidepressant	+0.104	be available	+0.155	antidepressant	+0.160	brain chemistry
+0.234	is trial	+0.098	restating the	+0.104	the	+0.160	the many
+0.231	the thread	+0.091	using the	+0.085	zoloft is	+0.140	all the
+0.192	manage other	+0.085	questions will	+0.066	there	+0.134	thread going
+0.186	and all	+0.085	trial and	+0.065	side	+0.129	of people
+0.178	thread going	+0.084	if it	+0.059	zoloft	+0.124	is trial
+0.167	chemistry set	+0.084	other conditions	+0.056	all	+0.118	unfortunate
+0.167	people using	+0.083	a lot	+0.052	that	+0.115	are a
+0.165	going here	+0.082	the side	+0.046	is a	+0.108	stimulating antidepressant
+0.165	questions will	+0.080	going here	+0.046	the serotonin	+0.103	going here
+0.162	the side	+0.076	there are	+0.043	here	+0.103	therapy is
+0.160	the many	+0.069	the serotonin	+0.043	are	+0.101	trial
+0.155	is a	+0.066	have answers	+0.040	maybe	+0.100	without restating
+0.153	conditions hang	+0.066	a brain	+0.037	it is	+0.097	but if
+0.152	all the	+0.062	there maybe	+0.035	of	+0.094	manage other
+0.151	but if	+0.059	is trial	+0.031	many medications	+0.089	questions
+0.143	set will	+0.057	people using	+0.029	in there	+0.082	a stimulating

Slika 3.11: LIME interpretacija klasifikacije teksta

3.5 SHAP vrijednosti

Shapley-jeva objašnjenja su tehnike Nobelovca Lloyd Shapley, s vjerodostojnom teorijskom podrškom iz ekonomije i teorije igara.

Shapley-eva objašnjenja objedinjuju pristupe kao što su LIME i LOCO te ističu lokalne doprinose varijabli završnom predviđanju modela. Shapley također stvara dosljedne mjere globalne varijable važnosti. SHAP skraćeno za (SHapley Additive exPlanations) dodjeljuje svakoj značajki vrijednost važnosti za određeno predviđanje.

Koncept Shapley vrijednosti (SHAP) prvobitno je razvijen kako bi procijenio važnost pojedinog igrača u timu. Imao je za cilj distribuirati ukupni dobitak ili isplatu između igrača. Shapley vrijednosti dodjeljuju poštenu ili razumnu nagradu svakom igraču i predstavljaju jedinstveni rezultat koji posjeduje sljedeća svojstva: lokalna točnost (aditivnost) ,nula efekt i konzistentnost (simetrija). U kontekstu modela za predviđanje, Shapley-jeve vrijednosti mogu se gledati kao poštena ili razumna raspodjela važnosti značajke s obzirom na određeni model i predviđanje. Značajke doprinose predviđanju modela različitom veličinom i predznakom, što je sve opisano pomoću Shapley-jevih vrijednosti.

Za intuiciju ove metode uvodimo jednostavan primjer - linearne regresije:

$$f(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_n x_n$$

gdje je $x = (x_1, ..., x_n) \in [0, 1]^n$

zanima nas kako vrijednost određene značajke utječe na predviđanje. Za to mogli bi koristiti sljedeći izraz

$$\phi_i(x) = \beta_i x_i - \beta_i E[X_i] \tag{3.2}$$

što se u literaturi još i naziva situacijskom važnošću $X_i = x_i$. Situacijska važnost je razlika između onoga što značajka doprinosi kada je vrijednost značajke x_i i što se očekuje da će pridonijeti. Ako je situacijska važnost pozitivna, onda značajka ima pozitivan doprinos (povećava predviđanja za ovu konkretnu instancu), ako je negativna, značajka ima negativan doprinos (smanjuje predviđanje) i ako to je 0, nema doprinosa.

Računanje doprinosa značajke na ovom primjeru je jednostavno pošto koristimo poznati model i značajke međusobno ne djeluju. Stoga je doprinos neke vrijednosti značajke ,x_i isti u svim slučajevima, bez obzira na vrijednosti ostalih značajki. U želji za pronalaskom model-agnostičkog načina, međutim, model je nepoznat i ne postoje druge pretpostavke osim one da model preslikava neki poznati prostor ulaznih vrijednosti u poznatu kodomenu(izlazni prostor). Ta su ograničenja potrebna da metoda bude općenita. Neki načini rješavanja ovog problema su računanje doprinosa na sljedeći način

$$\phi_i(x_1, x_2, ..., x_n) = f(x_1, x_2, ..., x_n) - E[f(x_1, x_2, .., X_i, .., x_n)$$

ovaj izraz je razlika između predviđanja za instancu $x = (x_1, ...x_n)$ i očekivanog predviđanje za isti slučaj i-te značajke. U praksi ovaj izraz nije teško aproksimirati (ili izračunati, ako je domena značajke konačna) trebamo posumirati vrijednosti i-te značajke, dok su vrijednosti ostalih ulaznih značajki ostaju fiksne te uzeti srednju vrijednost izračunate sume. Uz to ako je model aditivan, ovaj izraz je zapravo ekvivalentan 3.2. Međutim, kada model nije aditivan, što je čest slučaj u primjenama ovaj izraz daje izrazito loša rješenja. Uzmimo za primjer logičku funkciju *ILI*, neka je $f(x_1, x_2) = x_1 \vee x_2$ gdje su obije značajke dobivene iz uniformne distribucije [0, 1]. Kada računamo doprinos prve značajke u predviđanju $f(1, 1) = 1$ dobivamo da je doprinos x_1 zapravo 0 što je netočno za logičku funkciju *ILI*.

Uvodimo sljedeće oznake za ovu sekciju. Neka je $\mathcal{X} = [0, 1]^n$ prostor ulaznih značajki te neka je Y kodomena modela. Neka je $\{(x_1^{(i)}, x_2^{(i)}, ..., x_n^{(i)}, y^{(i)})\}_{i=1}^{M}$ skup podataka koji nam je na raspolaganju te na kojem je model treniran. Funkcija $f : \mathcal{X} \to \Re$ predstavlja model koji koristimo za predviđanje. Uvodimo oznaku

$$f_Q(x) = \mathbb{E}[f|X_i = x_i, \forall i \in Q]$$

, gdje je $Q \subseteq S = \{1, 2, 3, ..., n\}$ je podskup značajki. Za prazan skup ovaj izraz je srednja vrijednost funkcije f, odnosno $f_{\{\}}(x) = E[f]$. Definiramo doprinos modelu nekih značajki iz skupa Q:

$$\Delta_Q(x) = f_Q(x) - f_{\{\}(x)}$$

sada definiramo doprinos značajke i

Definicija 3.5.1. *Neka je f model predviđanja, te neka je $i \in S = \{1, 2, ..., n\}$ gdje je n broj značajki tada definiramo doprinos $i-te$ značajke*

$$\phi_i(x) = \sum_{Q \subseteq S \setminus \{i\}} \frac{|Q|!(|S| - |Q| - 1)!}{|S|!}(\Delta_{Q \cup \{i\}}(x) - \Delta_Q(x))$$

Aproksimacija Shapley vrijednosti

Računanje Shapley vrijednosti preko Definicije 3.5.1. je eksponencijalno vremenski složeno, što čini ovaj način ne isplativim za praktične primjene sljedeća aproksimacija se koristi za izračun Shapley-evih vrijednosti

$$\widehat{\phi}_i(x) = \frac{1}{n!} \sum_{O \in \pi(n)} (\Delta_{Pre^i(O) \cup \{i\}}(x) - \Delta_{Pre^i(O)}(x)) \tag{3.3}$$

gdje $\pi(n)$ je skup uređenih permutacija značajki $\{1, 2, 3, .., n\}$ i $Pre^i(O)$ je skup svih značajki koje prethode i u permutaciji $O \in \pi(n)$.

Δ-izraze računamo na sljedeći način, gdje je p neka distribucija gdje su značajke međusobno ne zavisno distribuirane

$$\Delta_Q(x) = f_Q(x) - f_{\{\}}(x)$$

$$= \sum_{w \in \mathcal{X} \text{ t.d } \forall i \in S\,(w_i = x_i)} p(w) f(w) - \sum_{w \in \mathcal{X}} p(w) f(w)$$

$$= \sum_{w \in \mathcal{X}} p(w)(f(w_{[w_i = x_i, i \in S]}) - f(w))$$

gdje $w_{[w_i = x_i, i \in S]}$ označava instancu w, kojoj vrijednosti značajki $i \in S$ mijenjamo s vrijednostima x_i

Računamo li sada aproksimaciju Shaplyeve vrijednosti Δ izrazi u jednadžbi (3.3) dobivamo sljedeći izraz:

$$\widehat{\phi}_i(x) = \frac{1}{n!} \sum_{O \in \pi(n)} \sum_{w \in \mathcal{X}} p(w)\Big(f(w_{[w_j = x_j, j \in Pre^i(O \cup \{i\})]}) - f(w_{[w_j = x_j, j \in Pre^i(O)]})\Big) \tag{3.4}$$

Označimo za svaki $i \in S$,

$$V_{i,O,w \in \mathcal{X}} = \Big(f(w_{[w_j = x_j, j \in Pre^i(O \cup \{i\})]}) - f(w_{[w_j = x_j, j \in Pre^i(O)]})\Big)$$

Sada opisujemo algoritam računanja aproksimacije Shaply vrijednosti. Ako uzimamo slučajni uzorak s ponavljanjima veličine m, vjerojatnosna distribucija p postaje diskretna uniformna, $p(w) = \frac{1}{m}$. Također pri odabiru jednog od m uzoraka, odaberemo i jednu permutaciju $O \in \pi(n)$. Zatim računamo potrebne vrijednosti s odabranim podacima te ove korake

iteriramo m puta. Algoritam je prikazan dolje.

Algorithm 1: Algoritam aproksimacije Shaply vrijednosti

Aproksimacija i-tog doprinosa značajke za model f, instancu $x \in X$, uz slučajno izvlačenje m uzoraka s ponavljanjem iz X

Inicijalizacija: $\phi_i(x) \leftarrow 0$
for *i:=1 do m* **do**
  Odaberemo slučajnu permutaciju $O \in \pi(n)$
  Odaberemo slučajnu instancu $w \in X$

  $\vec{b}_1 \leftarrow$ vektor koji sadržava vrijednosti x za vrijednosti iz O prije i uključeno, te w za vrijednosti nakon
  $\vec{b}_2 \leftarrow$ vektor koji sadržava vrijednosti x za vrijednosti iz O prije i isključen, te w za vrijednosti nakon

  $\phi_i(x) \leftarrow \phi_i(x) + f(\vec{b}_1) - f(\vec{b}_2)$
$\phi_i(x) \leftarrow \frac{\phi_i(x)}{m}$

KernelSHAP

LIME koristi samo-objašnjive modele za lokalnu aproksimaciju f, gdje se lokalno mjeri kroz pojednostavljeni binarni ulazni prostor opisan u prethodnoj sekciji, ako kao samo-objašnjivi model koristimo model linearne regresije tada govorimo o LinerLime-u. Izraz formuliran kao u jednadžbi 3.1 ne podsjeća baš na Shaply vrijednosti kojima se bavimo u ovoj sekciji, no uz pažljivo odabrane funkciju greške L, težinsku jezgru $\pi_{x'}$ te regularizacijski izraz Ω iz 3.1 možemo dobiti Shaplyeve vrijednosti.

Objašnjivi modeli često koriste pojednostavljene ulazne vrijednosti, kao što smo koristili u LIME-u. Koje preslikavaju originalne ulazne vrijednosti u interpretabilnu binarnu reprezentaciju.

Teorem 3.5.2. *(Kernel Shapley) Ako u jednadžbu 3.1 uvrstimo sljedeću funkciju greške L, težinsku jezgru $\pi_{x'}$ te regularizacijski izraz Ω, tada će težine linearnog modela g dobivene minimizacijom funkcije greške L uz regulacijski izraz Ω bit će Shapley izrazi*

$$\Omega(g) = 0$$

$$\pi_{x'}(z') = \frac{(M-1)}{(M|z'|)|z'|(M-|z'|)}$$

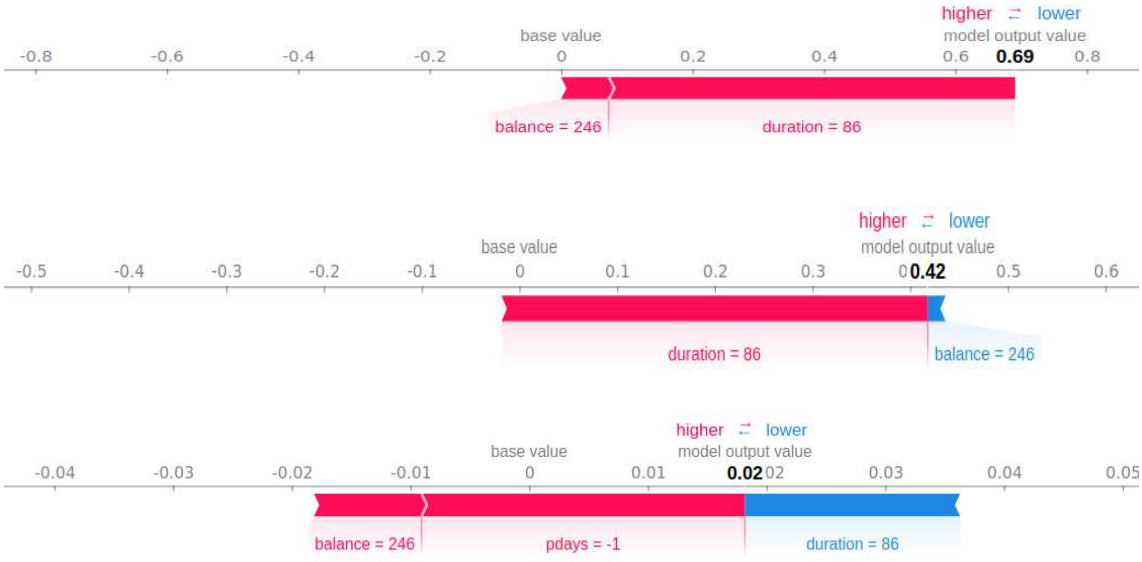
$$L(f, g, \pi_{x'}) = \sum_{(z,z')\in Z} [f(h_x(z')) - g(z')]^2 \pi_x(z')$$

Radi linearnosti funkcije g te činjenica da se $L = \sum_{(z,z')\in Z}[f(h_x(z'))\sqrt{\pi_x(z')} - g(z')\sqrt{\pi_x(z')}]^2$ može prikazati kao kvadratna funkcija greške, KernelShapley-ve vrijednosti mogu biti aproksimirane običnom linearnom regresijom, odnosnom pronalaskom težina koje minimiziraju funkcije greške L. Ovo je računarski efikasniji način pronalaska SHAP vrijednosti.

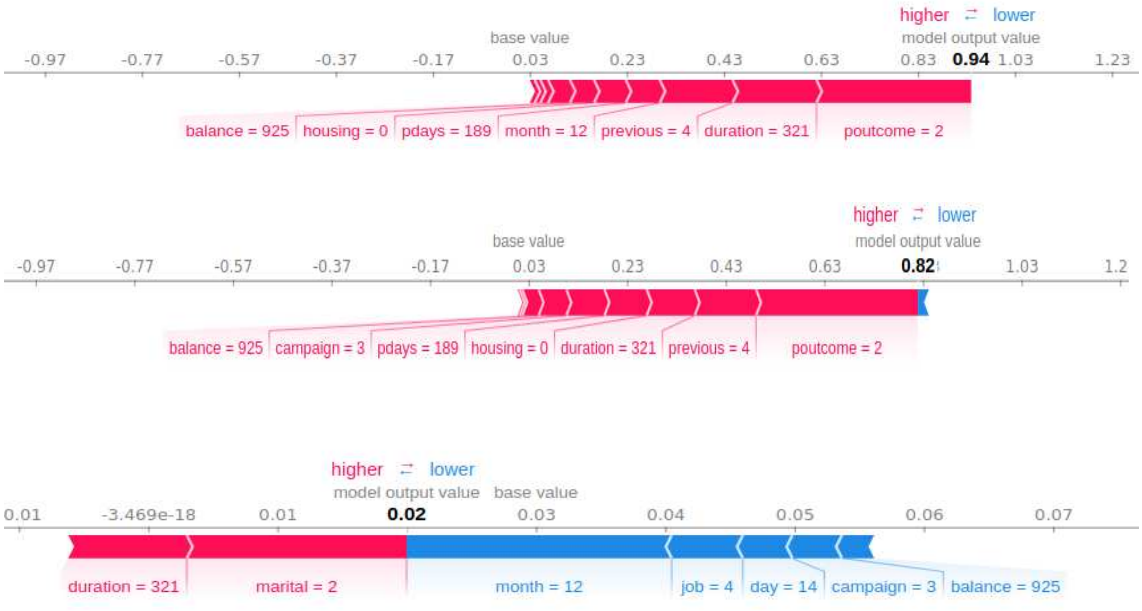
Primjeri.

Koristimo podatke iz Bank Marketing Data Set(izvor). Podaci su povezani s izravnim marketinškim kampanjama portugalske banke. Marketinške kampanje temeljile su se na telefonskim pozivima. Često je bilo potrebno više kontakata istom klijentu kako bi klijentov konačni odgovor bio (bankovni oročeni depozit) ('da') ili ne ('ne') vezano za pretplatu koju promovira marketinška kampanja. Cilj klasifikacije je predvidjeti hoće li se klijent pretplatiti (varijabla y).

Trenirali smo 3 različita klasifikatora, KNN, Neuronske mreže i Random Forest-a iz knjižnice sklearn, te smo koristili implementaciju KernelShap-a dostupnu na linku (Link). Te dobivamo sljedeća objašnjenja predviđanju. Za tri različite instance pokazujemo objašnjenje za istrenirane modele.

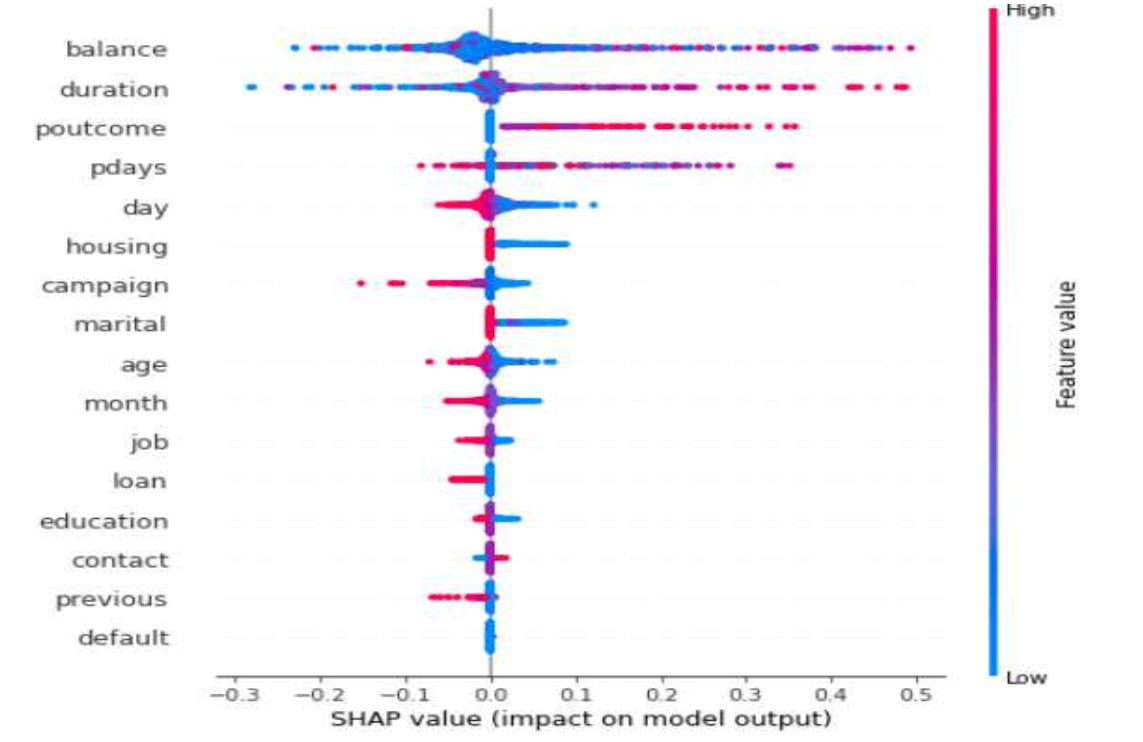


Slika 3.12: KernelShap objašnjenja određenih instanci za model KNN



Slika 3.13: KernelShap objašnjenja određenih instanci za model RF

Slika 3.14: KernelShap objašnjenja određenih instanci za model MLP



Slika 3.15: Jačina utjecaja svojstva na predviđanje MLP modela

Poglavlje 4

Model-specifične metode

4.1 Interpretacija Dubokih neuronskih mreža

Duboke arhitekture pokazale su vrhunske rezultate u raznim primjenama, posebno s području računalskog vida.Iz toga se javlja potrebna za kvalitativnom usporedbom rješenja različitih naučenih dubokih arhitektura

Konvolucijska neuronska mreža

Prije nego što krenemo s objašnjenjima recimo nekoliko riječi o samome modelu, pošto su interpretacijske metode koje ćemo promatrati model-specifične.

Za $\mathbf{x} \in \mathbb{R}^{H \times W \times D}$ kažemo da je tenzor trećeg reda, sadrži $H \cdot W \cdot D$ elementa indeksiranih pomoću (i, j, d) t.d. $0 \leq i \leq H, 0 \leq j \leq W, 0 \leq d \leq D$. Drugi način gledanja na tenzor trećeg reda je da ga tretiramo kao da sadrži D kanala matrica, odnosno svaki kanal sačinjava jednu $H \times W$ matricu. Ovaj pojam uvidom zato što se digitalne slike pohranjuju kao tenzori trećeg reda, odnosno matrice 3 kanala. Ako je slika u standardnom *RGB* formatu svaki kanal matrice predstavlja redom crvenu, zelenu i plavu boju. Za kompjuterski vid te pronalaženje obrazaca na slikama u boji koriste se najčešće KNN ili konvolucijske neuronske mreže. Konvolucijske mreže, poznate i pod nazivom CNN-ovi, specijalizirane su vrste neuronske mreže za obradu podataka koje ulazni podaci imaju topologiju nalik mrežama. Primjeri uključuju vremenske nizove, koji se mogu smatrati 1D mrežom koja uzima uzorke u pravilnim vremenskim intervalima, te slikovnim podacima, koje možemo zamisliti kao 2D rešetka piksela. Konvolucijske mreže su iznimno uspješne u praktičnim primjenama. Naziv "konvolucijska neuronska mreža" označava da mreža koristi matematičku operaciju koja se zove konvolucija. Konvolucija je specijalizirana vrsta linearnih operacija. Konvolucijske mreže su jednostavno neuronske mreže koje koriste konvoluciju umjesto općeg množenja matrica, barem u jednom od njihovih slojeva.

Sada ne formalno navodimo ideju konvolucije, neka su f i g neke realne funkcije, konvolucija $f \star g(t) = \int f(a)g(t-a)da$, te u ako su f i g funkcije na diskretnim skupovima tada je njihova konvolucija $fg(t) = \sum_{a=-\infty}^{+\infty} f(a)g(t-a)$ Funkcija f se u primjenama naziva ulaz, dok funkcija g je jezgra.

U primjenama strojnog učenja ulaz je obično višedimenzionalni niz podataka odnosno tenzor, dok jezgra je obično višedimenzionalni niz parametara koji se mijenjaju ("uče")) algoritmom učenja. Svaki element ulaza i jezgre mora biti pohranjen odvojeno, te pretpostavljamo da su ove funkcije svugdje jednake nuli, osim na konačnom skupu točaka za koje pohranjujemo vrijednosti. To znači da u praksi možemo implementirati beskonačni zbroj kao zbrajanje preko konačnog broja elementa niza.

Sada uvodimo i konvoluciju za 2-dimenzionalne funkcije. Neka I predstavlja matricu ulaznog podatka te neka je K 2-dimenzionalna jezgra tada definiramo konvoluciju na sljedeći način

$$(I \star K) = \sum_m \sum_n I(m,n)K(i-m, j-n)$$

.

Valja napomenuti da su konvolucije komutativne te se $(K \star I)$ češće implementira u knjižnicama strojnog učenja, no većina knjižnica implementira sljedeći oblik konvolucije

$$(I \star K)(i,j) = \sum_m \sum_n I(i+m, j+n)K(m,n)$$

.

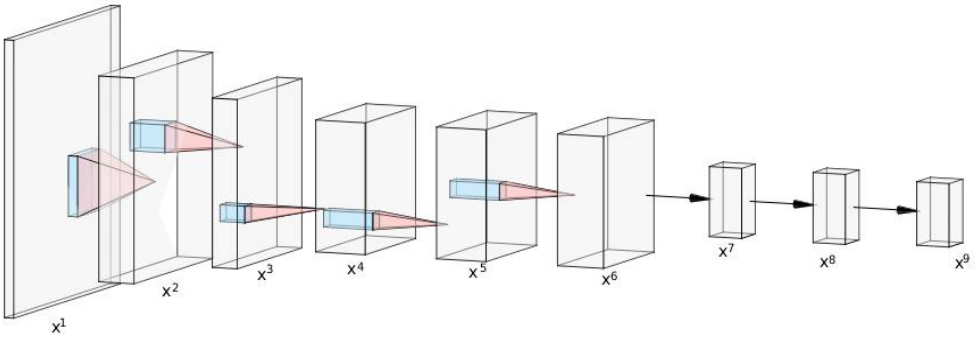
Nakon što smo dobili intuiciju o konvoluciji, definirajmo konvolucijski sloj, istoimene neuronske mreže. Navodimo najjednostavniju verziju konvolucijskog sloja.

Definicija 4.1.1. *Neka je $\boldsymbol{x}^l \in \mathbb{R}^{(H^l \times W^l \times D^l)}$ ulaz l-tog sloja, te H^l, W^l, D^l te neka je $\boldsymbol{f} \in \mathbb{R}^{H \times W \times D^l \times D}$ tenzor 4-reda gdje koristimo indekse $0 \leq i \leq H, 0 \leq j \leq W, 0 \leq d^l \leq D^l$ i $0 \leq d \leq D$ da bismo odredili konkretan element tenzora. Tada je konvoluciju sloja x^l i tenzora f definirana na sljedeći način:*

$$y_{i^{l+1}, j^{l+1}, d} = \sum_{i=0}^{H-1} \sum_{j=0}^{W-1} \sum_{j=0}^{D^l-1} \boldsymbol{f}_{i,j,d^l,d} \cdot \boldsymbol{x}^l_{i^{l+1}+i, j^{l+1}+j, d^l}$$

Sloj u kojem se provodi konvolucija između ulaznog sloja x^l i tenzora f naziva se konvolucijski sloj

Arhitektura CNN sagrađena je od slojeva kao što je prikazano na slici 4.1. slojevi nad slojevima računamo konvoluciju na prethodno opisan način, te ostale vrste slojeva (pooling, potpuno povezane ili ReLu) koje računamo na drugačije načine, te na kraju za zadnji sloj vezano o kojoj vrsti problema je riječ, koristimo prigodne funkcije (npr.softmax) da bi izlazan CNN prilagodili podacima za treniranje.



Slika 4.1: Prikaz arhitekture Konvolucijske neuronske mreže

Vizualizacija vidljivosti klase za sliku

Sada ćemo predstaviti vizualizacijsku metodu koja za rezultat daje, objašnjenje koji dijelovi slike odnosno pikseli najviše utječu na predviđanje, odnosno na vidljivost klase za sliku. Ako je zadana slika I_0, klasa c te klasifikacijska Konvolucijska neuronska mrežu C. Sa S_C označavamo rezultantnu funkcija Konvolucijske neuronske mreže.

Pretpostavimo za početak da je rezultantna funkcija S_C linearna odnosno postoje w_C i b_C t.d vrijedi

$$S_C(I) = w_C^T I + b_C$$

gdje je I jednodimenzionalna reprezentacija vektora. U ovome slučaju jednostavno je uočiti da je važnost piksela zadana težinama w_C. Međutim rezultantna funkcija konvolucijske neuronske mreže daleko je od linearne u I, tako da zaključivanje iz prethodnog primjera ne može biti izravno upotrebljeno. Uz zadanu sliku I_0 možemo aproksimirati $S_C(I)$ s linearnom funkcijom pomoću Taylorovog razvoja prvog reda:

$$S_C(I) \approx w^T I + b$$

gdje je w derivacija S_c s obzirom na sliku I (u vektorskoj reprezentaciji)

$$w = \frac{\partial S_c}{\partial I}(I_0) \tag{4.1}$$

Slika mape vidljivosti dobiva se na sljedeći način, uz zadanu sliku I_0 sa m-redaka, n-stupaca te klasom c. Vidljivost klase je matrica $M \in \mathbb{R}^{m \times n}$ izračunata na sljedeći način. Prvo derivacija w iz jednadžbe 4.1 je pronađena kroz back-propagation. Ako se radi o crno bijeloj slici broj kanala, odnosno slojeva ulazne slike, je 1 stoga $M_{i,j} = |w_{h(i,j)}|$, gdje je $h(i, j)$ indeks elementa w koji odgovara pikselu na mjestu (i, j) ulazne slike. Slike s više kanala

na ulazu npr. RGB računati će svoju mapu vidljivosti na sljedeći način $M_{i,j} = max_c w_{h(i,j,c)}$ gdje su c ulazni kanali, a $h(i, j, c)$ odgovara pikselu na mjestu (i, j) kanala c.

Usmjereni Backpropagation

U prethodnom odjeljku vidjeli smo da računanje karte vidljivosti koristimo Backpropagation kroz CNN, te dobivene vrijednosti koristimo za prikaz važnosti određenih piksela u klasifikaciji slike. Iako jednostavan i donekle precizan ovaj pristup moguće je poboljšati s metodom koju nazivamo **Usmjereni backpropagation**. Ova metoda jednaka je običnom prolasku unatrag, osim što kod prolaska unatrag gradijent računamo jedino na drugačiji način koji pomaže u preciznosti. Naime, kada je aktivacijska funkcija sloja CNN funkcija *RELU*, gradient je izračunat na sljedeći način ako f_i^{l+1} označava specifičnu i-tu jedinicu sloja f^{l+1} , dok je y^l izlaz konvolucije prijašnjeg sloja.

$$f_i^{l+1} = relu(y^l)$$

Obični back-propagation računa gradijent na sljedeći način:

$$(f_i^l > 0)\frac{\partial S_c}{\partial f_i^{l+1}}$$

u usmjerenom Backpropagationu gradijent se računa na sljedeći način:

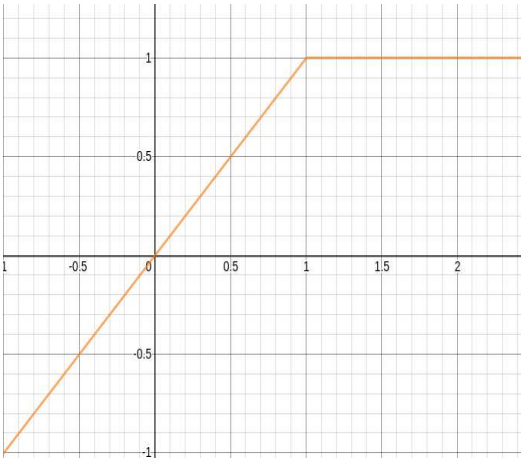
$$(f_i^l > 0)(R_i^{l+1} > 0)\frac{\partial S_c}{\partial f_i^{l+1}}$$

Gdje vrijedi $R_i^{l+1} = \frac{\partial S_c}{\partial f_i^{l+1}}$.Ovo sprječava povratni tok negativnih gradijenata, što odgovara neuronima koji smanjuju aktivaciju jedinica višeg sloja, koje želimo vizualizirati.

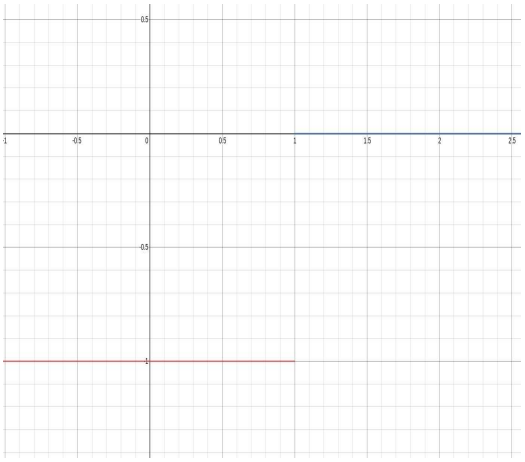
Integrirani Backpropagation

Za interpretabilnost linearnih modela, za postizanje interpretabilnosti koriste se koeficijenti modela i vrijednosti značajki kako bi se obrazložilo predviđanje. Gradijenti (izlaza s obzirom na ulazne varijable) prirodan su analogon koeficijentima modela dubokih neuronskih mreža, a samim tim i umnožak gradijenta i vrijednosti značajke razumno polazište za interpretabilnost ovih modela.

Definicija 4.1.2. *Neka je* $F : \mathbb{R}^n \to [0, 1]$ *funkcija, koja predstavlja duboku neuronsku mrežu, te neka je* $x = (x_1, x_2, .., x_n) \in \mathbb{R}^n$*, te neka je* x' *referenta vrijednost ulaznog podataka. Pripisivanje s ulaznom vrijednosti* x *s obzirom na* x' *je* $A_F(x, x') = (a_1, a_2, ..., a_n) \in \mathbb{R}^n$ *gdje je* a_i *doprinosit značajke* x_i *u predviđanju* $F(x)$*. Način odabira pripisivanja nazivamo metodom pripisivanja.*



Slika.4.1 Aktivacijska funkcija



Slika.4.1 Gradijent aktivacijske funkcije

Na primjer, u dubokoj neuronskoj mreži za prepoznavanje objekta, metoda pripisivanja nam govori koji su pikseli slike odgovoran za odabir određene klasifikacije.

Navodimo dva svojstva koja tražimo od metode pripisivanja. Metoda pripisivanja zadovoljava svojstvo **Osjetljivosti** ako za svaki ulaz x koji se za samo jednu značajku razlikuje od referentne vrijednosti vektora x', te imaju različita predviđanja tada svojstvo u kojem se razlikuju mora imati doprinos različit od 0.

Primjer ne zadovoljavanja ovog svojstva je sljedeći.

Pretpostavimo da je $f(x) = 1 - ReLu(1 - x)$ izlazna funkcija modela, ako za referentnu vrijednost uzmemo $x' = 0$, te za ulaznu vrijednost $x = 2$, jasno je da $f(0) = 0$, dok $f(2) = 1$ što bi značilo da postoji razlika. Uzmemo li da je metoda pripisivanja gradijent, tada vidimo da je doprinos pridružen $x = 2$, vrijednost 0. Ne zadovoljavanje osjetljivosti uzrokuje da se metoda pripisivanja bez razloga obazire na nevažna svojstva.

Sljedeće svojstvo koje želimo da metoda pripisivanja zadovoljava je svojstvo Neovisnost o implementaciji. Kažemo da su neuronske mreže M_1 i M_2 funkcijski ekvivalentne ako su njihovi izlazi za sve ulazne vrijednosti jednake. Kažemo da je metoda pripisivanja **Neovisna o implementaciji** ako za svake dvije funkcijski ekvivalente mreže sva predviđanja su identična.

Možemo primijetiti da su svi gradijenti neovisni o poretku funkcija u lančanom pravilu da deriviranje. Naime znamo da vrijedi $\frac{\partial f}{\partial g} = \frac{\partial f}{\partial h}\frac{\partial h}{\partial g}$, uzmemo li da je f izlazna vrijednost dok g ulazna vrijednost, dok je h različita unutarnja implementacija. Gradijent $\frac{\partial f}{\partial g}$ može biti izračunat teoretski i bez među funkcije h, čime vidimo da je ovaj gradijent neovisan o implementacijskoj funkciji h. Ako metoda pripisivanja ne uspije zadovoljiti neovisnost o implementaciji, pripisivanja su potencijalno osjetljiva na nevažne aspekte modela.

Sada definiramo metodu koja zadovoljava 2 prethodno navedena svojstva

$$Ig_i(x) := (x_i - x_i') \int \frac{\partial S_c(x' + \alpha(x - x'))}{\partial x_i} d\alpha$$

Referentu vrijednost biramo tako da vrijedi da je $F(x't) \approx 0$. Za sliku ovo svojstvo zadovoljava crna slika.

SmoothGrad

U ovom odjeljku predstavljamo metodu koja zaoštrava mape vidljivosti predstavljene u prethodnim poglavljima. U primjenama navedene metode zaista naglašavaju dijelove razumljive ljudima za interpretaciju. No u isto vrijeme, mape vidljivosti su često vizualno pune šuma (noisy), naglašavajući neke piksele koji se ljudskom oku doimaju slučajni. Bez pokušaja eliminacije šuma, ne možemo znati radi li se o unutarnjem problemu kako mreža donosi odluke, ili je razlog površinske prirode.

Opisat ćemo tehniku SmoothGrada, vrlo jednostavnu tehniku koja se u primjenama često koristi za uklanjanje vizualnog šuma. Ključna ideja SmoothGrada je uzeti sliku čija nas interpretacija zanima, napraviti uzorak sličnih slika s dodanim šumom te zatim uzeti srednju vrijednost od dobivenih mapa vidljivosti. U prijašnjim metodama računali smo gradijent i njegove varijante te smo gradijent funkcije prema ulazu smatrali mjerom koliko će se promijeniti globalna funkcija S_c za promjenu u svakom od piksela x, te se "nadamo" da će te promjene označiti ključne dijelove slike potrebne za donošenje odluke. Jedan od mogućih razloga za šum je da derivacija S_c može oštro fluktuirati na malenim skalama. Drugim riječima, može doći do prividnog šuma koji se vidi na karti vidljivosti zbog nevažnim lokalnim varijacijama u parcijalnih derivacija. S obzirom na ove brze fluktuacije, gradijent S_c će u svakom trenutku bit manje značajan od lokalnog prosjeka vrijednosti gradijenta. To sugerira novi način stvaranja poboljšanih mapa vidljivosti tako, da umjesto zasnivanja vizualizacije izravno na gradijentu $\partial S\, c$, mogli bismo ga temeljiti na prosjeku ∂S_c slučajnih uzoraka s dodanim šumom. Možemo uzeti slučajni uzrok veličine n koji je blizini ulaza x te izračunati prosječne mape vidljivosti, što matematički interpretiramo

$$\frac{\widehat{\partial S_c}(x)}{\partial x} = \sum_{i=1}^{n} \frac{\partial S_c(x + \mathcal{N}(0, \sigma^2))}{\partial x}$$

gdje je $\mathcal{N}(0, \sigma^2)$ označava Gaussovu slučajnu varijablu s varijancom σ. Ovako izračunate gradijente nazivamo **SmoothGrad**

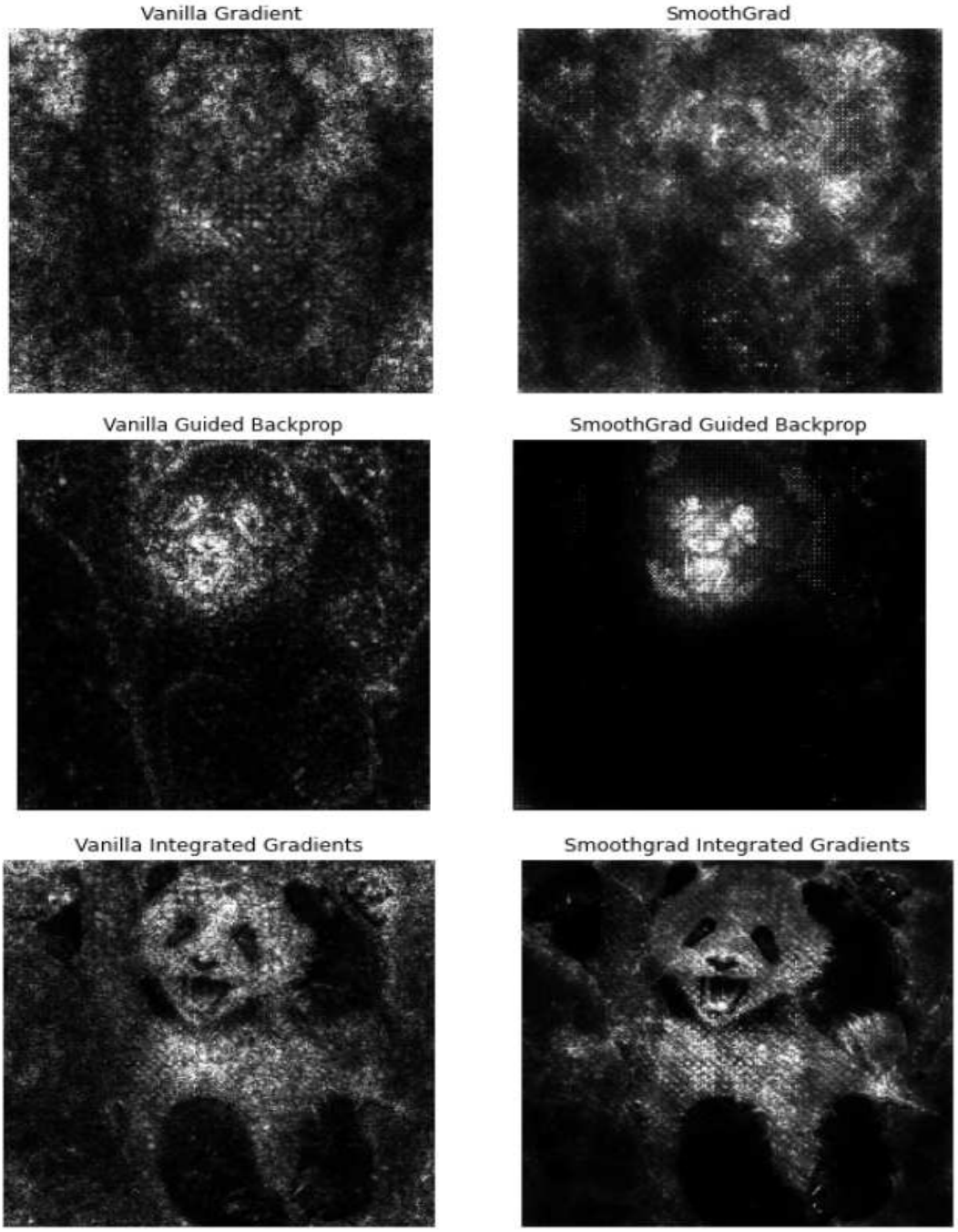
Primjeri.

Za primjer u ovoj sekciju učitali smo pretrenirane CNN, dostupne na linku. Mreže koje smo učitali su InceptionV4 i NASNet-A Large, koji postižu najbolje rezultate među po-

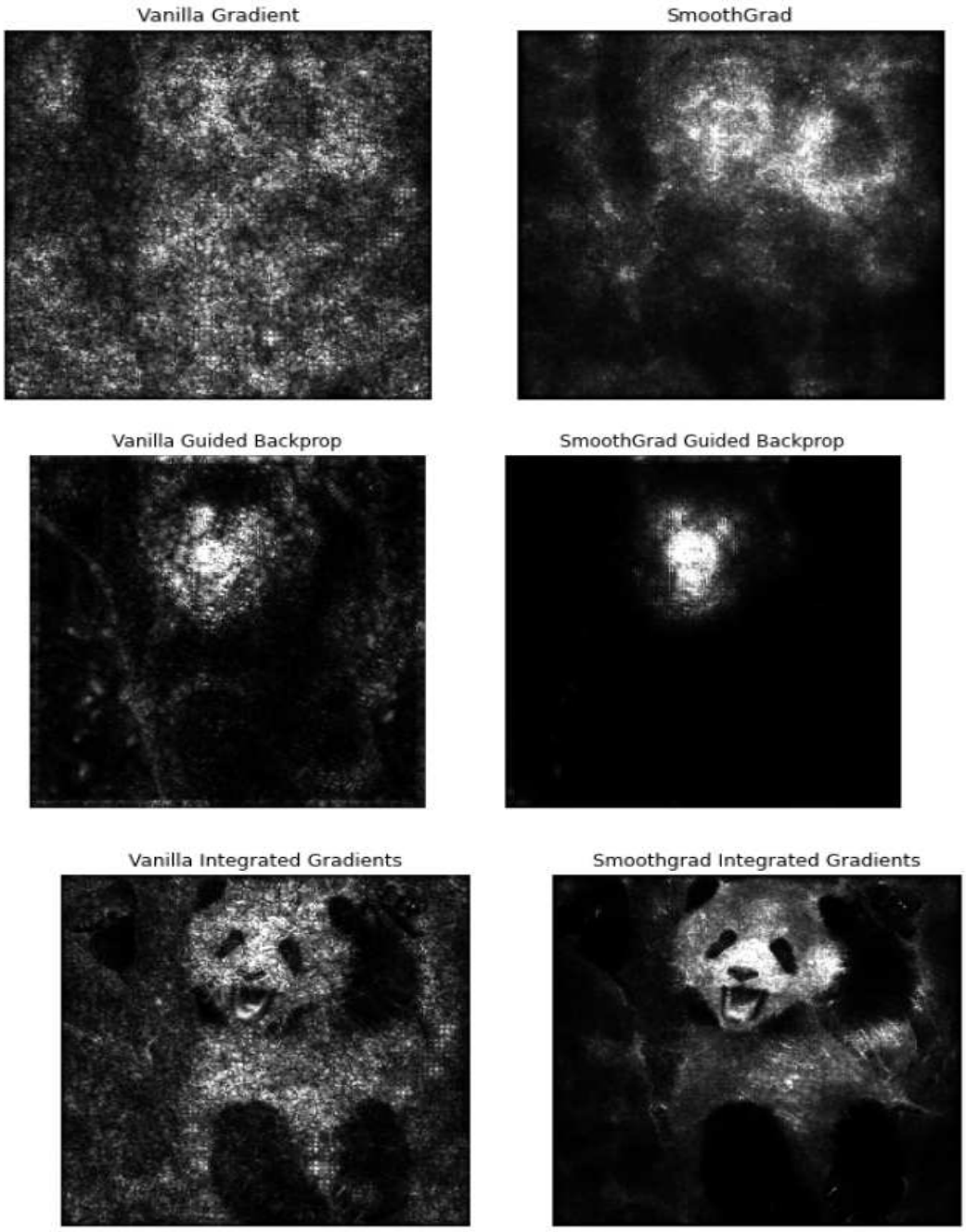
nuđenim pretreniranim modelima. Na modele smo primijenili sve prije definirane metode prikaza vidljivosti, odnosno računa gradijenta *Obični Backopropagation*, *Usmjereni backpropagation*, *Integrirani Backpropagation* te na svaku od tih metoda primjenili SmoothGrad, koristili smo implementaciju ovih algoritama sa sljedećeg linka: saliency. Slika koju koristimo je slika Pande te ju model NASNet-A Large 91.96% u klasu *giant panda, panda, panda bear, coon bear, Ailuropoda melanoleuca*, dok model InceptionV4 sliku klasificira u istu kasu 90.77% ulaznu sliku.

Slika 4.2: Ulazna slika za prikaza mapa vidljivosti

Mape vidljivosti za NASNet-A Large i InceptionV4 neuronske mreže su sljedeći.



Slika 4.3: Mapa vidljivosti za NASNet-A Large računanjem gradijenta pomoću različitih varijaniti Backpropagationa i uz dodatak SmoothGrada



Slika 4.4: Mapa vidljivosti za InceptionV4 računanjem gradijenta pomoću različitih varijaniti Backpropagationa i uz dodatak SmoothGrada

4.2 Interpretabilnost Modela Dubokog učenja za analizu vremenskih nizova

Unatoč zapanjujućim rezultatima modela temeljenih na dubokom učenju u nizu primjena koji uključuju računalni vid, analizu govora, prevoditeljski sustavi itd.,primjenjivost ovih modela visokih performansi je ograničena zbog toga što su crne kutije te im nedostaje objašnjivosti. To se posebno odnosi na domene poput poslovanja, financija, upravljanja prirodnim katastrofama, zdravstva, samostalnih automobila i protuterorizma gdje su razlozi za donošenje određene odluke podjednako važno kao i samo predviđanje. Postoji značajni pokušaji otkrivanja prirode ovih modela temeljenih na dubokom učenju gdje je vizualizacija modela bila najčešća strategija. Gotovo svi predloženi sustavi vizualizacije su fokusirani na sliku gdje se vizualizacija slike izravno može protumačiti (prirodna povezanost sa sličnim izgledom predmeta poput očiju, lica, pasa, automobila itd.). Većina ideja jednako su primjenjive i na vremenske nizove, ali zbog ne intuitivne prirode podataka vremenskih nizova, otežan je izravni prijenos ove ideje za poboljšanje ljudskog razumijevanja modela.

Računanje utjecaja

Želimo razumjeti kako računati utjecaj različitih dijelova mreže na predikciju. Postoje dva različita utjecaja koja se mogu izračunati na temelju bilo kojeg određenog filtera u mreži. Prva procjena proizlazi iz činjenice da ulaz utječe na izlaz određenog filtera u pitanju dok je drugi utjecaj samog filtera na konačni izlaz.

Ulazni utjecaj

Prvi utjecaj potječe od ulaznih podataka. Ova informacija pruža važan uvid u vezu između podataka s ulaza na koje mreža zapravo odgovara izračunavanje svog izlaza. Informacije o dijelovima ulaz koji je bio odgovoran za određeno predviđanje smatra se važnim objašnjenjem u mnogim scenarijima, uključujući domene poput samovozećih automobila, financija i medicine. Važno je napomenuti da možemo izračunati i utjecaj ulaza za svaki filtar zajedno s konačnim izlazom.

Ova se vrijednost može dobiti izračunavanjem gradijenta od sloja l s obzirom na ulazni sloj. Koristimo apsolutnu vrijednost gradijenta kao veličinu koja je relevantna bez obzira na smjer gradijenta. Ulaz označavamo kao a_j^0, što označava $j - ti$ ulazni neuron, dok a^l označava izlaz l-tog sloja, ovaj utjecaj ulaznih podataka može se izračunati pomoću sljedećih jednadžbi.

$$\delta^l_{0,j} = \frac{\partial a^l}{\partial z^l_j} \frac{\partial z^l_j}{\partial a^0_j}$$

$$I^l_{input} = |\delta^l_0|$$

Utjecaj filtera

Druga vrsta utjecaja koja je zanimljiva za proučavati je utjecaj različitih filtera. Ove informacije o važnosti filtera govore koji su filtri bili najutjecajniji za određenog predviđanja. Da bismo izračunali taj utjecaj, računamo gradijent izlaznog sloja L s obzirom na trenutni sloj l. Ovo nam pruža točkovnu procjenu o utjecaju svake vrijednosti filtera na izlaznu aktivaciju a^L. Pošto nam je važan utjecaj filtera uzimamo 1-p normu vektora utjecaja, te sumiramo po svim komponentama, što daje sveukupno dobru procjenu utjecaja filtera.

Računamo ju prema sljedećim jednadžbama.

$$\delta^l_{0,j} = \frac{\partial a^L}{\partial a^l_j} \frac{\partial a^l_j}{\partial z^j_l}$$

$$I^l_{input} = \sum_j |\delta^l_j|$$

Grupiranje filtera

Grupiranje filtera pomaže nam u otkrivanju različitih vrsta filtera prisutnih u mreži kao prikaz raznolikosti koju se postigla tijekom treniranja. Filtre grupiramo prema njihovom aktivacijskom obrascu, odnosno filtri s istim aktivacijskim obrascem u suštini hvataju isti koncept. Ovo grupiranje također je korisno u smanjenju preopterećenje za korisnika u fazi vizualizacije filtera gdje samo najistaknutiji filtri iz svakog klastera trebaju biti vizualizirani.

Kako nas zanima samo sličnost između uzoraka aktivacije, koristimo tehniku koja se često koristi u analizi vremenskih nizova zvanu Dinamičko Time Wrapping. Kodiramo svaki filter preko aktivacijskog vektora $a \in \mathbb{R}^d$. Algoritam prvo kreira matricu udaljenosti između svaka dva aktivacijska vektora $a_m \in R^d$ i $a_n \in \mathbb{R}^d$. Sa $DTW[i, j]$ računamo udaljenost između vektora $a^{1:i}_m$ i $a^{1:j}_n$ s najboljim poravnavanjem.

Računamo ju prema sljedećim jednadžbama.

$$DTW[i, j] = D(a^i_m, a^j_n) + min(\begin{bmatrix} DTW[i-1, j] \\ DTW[i, j-1] \\ DTW[i-1, j-1] \end{bmatrix})$$

$$D(a_m^i, a_n^j) = \|a_m^i - a_n^j\|_2$$

Jednom kada se izračuna DTW matrica, $DTW[d, d]$ može se koristiti kao mjera najmanje moguće udaljenosti za poravnanje dva aktivacijska vektora gdje je d dimenzionalnost aktivacijskih vektori. Stoga ovu udaljenost koristimo za grupiranje aktivacijskih vektora zajedno.

Kad je riječ o grupiranju, K-Means je najčešći izbor za bilo koji problem. Međutim, K-Means djeluje s euklidskom udaljenostom kao metrikom udaljenosti. Prelazak na DTW kao metriku udaljenosti s K-Means rezultira nepouzdanim rezultatima ili čak problemima konvergencije. Stoga izvršavamo hijerarhijsko (aglomerativno) grupiranje pomoću DTW-a. Grupe koje su najbliži jedna drugoj prema definiranoj udaljenosti, kombiniraju se zajedno dajući novu grupu. Udaljenost koja se pokazuje najkorisnija je potpuno povezivanje $d_C L \in \mathbb{R}$. Ova udaljenost se računa tako da se pronalazi sljedeći maksimum.

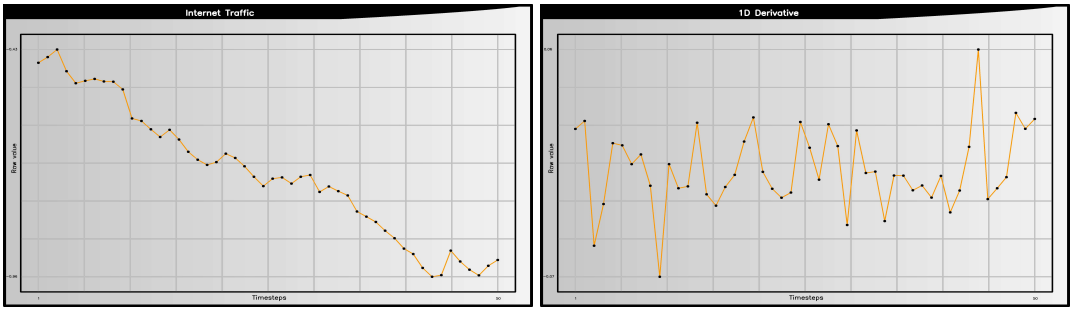
$$d_{CL}(G, H) = \max_{i \in G, j \in H} DTW[i, j]$$

Primjer.

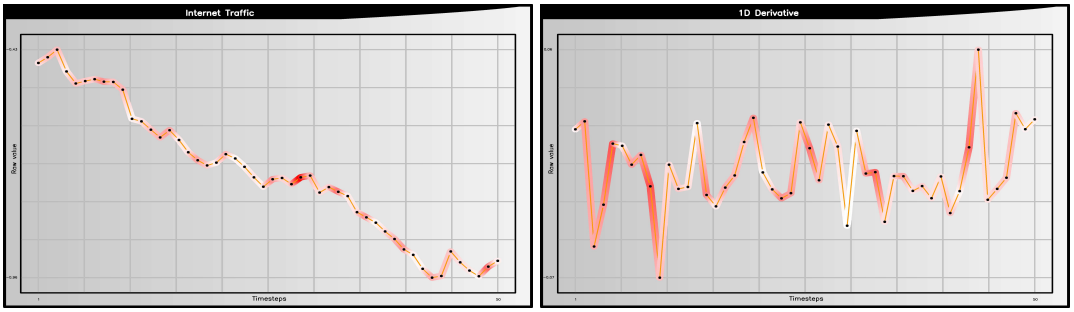
Koristimo skup vremenskih nizova za predviđanje Internetskog prometa, te učitavamo pred treniranu mrežu za predviđanje rezultata. Dostupan na Linku. Za generiranje objašnjenja koristimo servis TS-viz Link, Mreža radi na ulazu veličine 50 vremenskih koraka i sastoji se od tri Konvolucijska, dva Max-pool i jednog Dense sloj. Mreža je treniran koristeći Adam optimizator sa stopom učenja 0,001 za 5000 epoha koje koriste veličinu batch-a od 5 i srednje kvadratne pogreške (MSE) kao funkcija gubitka. Ova je mreža postiže vrhunski rezultat za spomenuti skup podataka. Mreža uzima 2 kanala kao ulaz, značajke ulaza, prvi kanal je originalni ulaz Internetskog prometa, dok je drugi njegova derivacija.

Uz knjižnicu TS-viz implementiranu, prikazujemo instancu iz skupa za testiranje, te promatramo kako model donosi svoju predikciju. Originalna izlazna vrijednost vremenske nizove sa Slike.4.5 je $y = 0.0134653$, dok je predikcijska vrijednost modela sljedeća $\hat{y} = 0.0133846$.

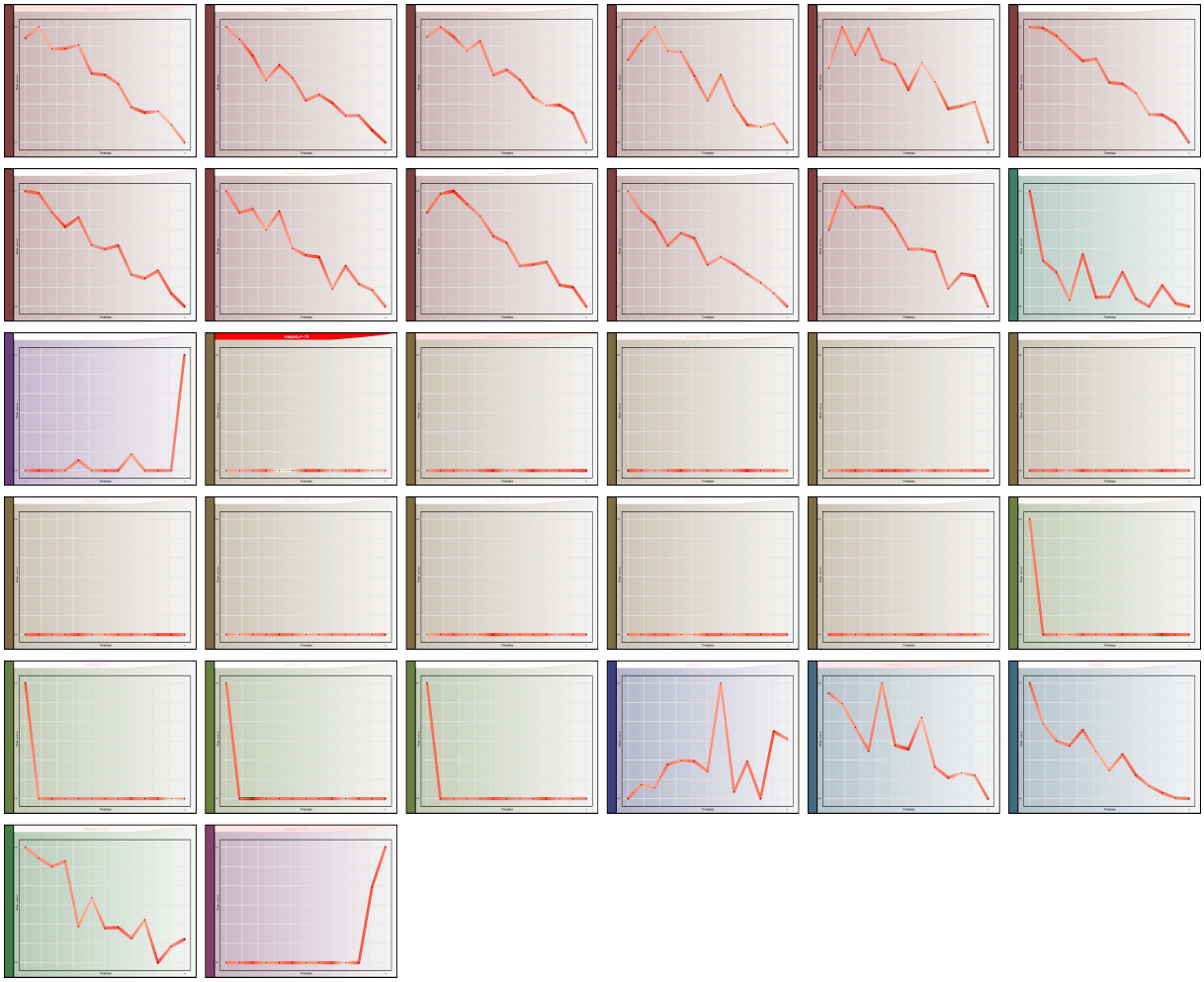
Važnost ulaznih podataka izračunata je na prikazana je na Slici.4.6, dok Slici.4.7 prikazano je grupiranje filtera.



Slika 4.5: Slika lijevo Internetski promet kroz vrijeme, desna slika prikazuje derivaciju ulazne vrijednosti



Slika 4.6: Slika lijevo važnost internetskog prometa kroz vrijeme za predikciju, desna slika prikazuje važnost derivacije za predikciju



Slika 4.7: Filteri mreže klasterirani prema aktivacijskim obrascima

Bibliografija

[1] T.Hastie, R.Tibshirani, J. Friedman. The Elements of Statistical Learning; Springer Series in Statistics; 2009.

[2] Fanaee-T, Hadi, Joao Gama. "Event labeling combining ensemble detectors and background knowledge." Progress in Artificial Intelligence. Springer Berlin Heidelberg, 1–15.

[3] Daniel W. Apley, Jingyu Zhu "Visualizing the Effects of Predictor Variables in Black Box Supervised Learning Models"

[4] D. Gunning , Explainable artificial intelligence (xAI), Technical Report, Defense Advanced Research Projects Agency (DARPA), 2017 .

[5] Friedman, J.H.; Popescu, B.E. Predictive learning via rule ensembles. Ann. Appl. Stat. 2008, 2, 916–954.

[6] Erik Štrumbelj, Igor Kononenko. "Explaining prediction models and individual predictions with feature contributions". In: Knowledge and information systems 41.3 (2014), pp. 647–665

[7] Friedman, J.H. Greedy function approximation: A gradient boosting machine. Ann. Stat. 2001, 29, 1189–1232.

[8] Goldstein, A.; Kapelner, A.; Bleich, J.; Pitkin, E. Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation. J. Comput. Gr. Stat. 2015, 24, 44–65.

[9] Apley, D.W. Visualizing the Effects of Predictor Variables in Black Box Supervised Learning Models. arXiv 2016, arXiv:1612.08468.

[10] Friedman, J.H.; Popescu, B.E. Predictive learning via rule ensembles. Ann. Appl. Stat. 2008, 2, 916–954.

[11] Ribeiro, M.T.; Singh, S.; Guestrin, C. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 1135–1144.

[12] Matthew D. Zeiler; Graham W. Taylor; Rob Fergus ;Adaptive Deconvolutional Networks for Mid and High Level Feature Learning; Dept. of Computer Science, Courant Institute, New York University

[13] P. Cortez, M. Rio, M. Rocha, and P. Sousa, "Multi-scale internet traffic forecasting using neural networks and time series methods," Expert Sys- tems, vol. 29, no. 2, pp. 143–155, 2012.

Sažetak

U ovom radu predstavili smo dio metoda za Objašnjivost umjetne inteligencije, podijelili smo metode prema kategorijama. Objasnili terminologiju korištenu u ovom polju. Za većinu metoda uz teoretsko objašnjenje dali smo i primjere korištenja. Predstavili smo najvažnije Samo-objašnjive metode kao što su Linearna Regresija, Logistička regresija i Stabla odluke. Objasnili zašto ih smatramo samo-objašnjivim. Za Model-Agnostičke metode koje spadaju u Post-Hoc metode objašnjivosti naveli smo Graf djelomične ovisnosti, Grafove pojedinačnog uvjetnog očekivanja, Grafove Akumuliranih lokalnih učinaka, Lokalno interpretabilnih model Agnostičke (LIME), te SHAP vrijednosti.

Sve ove metode, omogućavaju nam određeni uvid u rad bilo kojeg modela. Kod metoda koje su specifične za modele koncentrirali smo se na Konvolucijske Neuronske mreže, te prikazali poboljšanja u mapama vidljivosti. Također smo pokazali Objašnjive metode za CNN koje kao podatke imaju Vremenske nizove.

Summary

In this paper, we have presented some of the methods for Explanatory Artificial Intelligence, we have divided the methods into categories and explained the terminology used in this field. For most methods, in addition to a theoretical explanation, we have given examples of use. We presented the most important Self-Explainable methods that are Linear Regression, Logistic Regression and Decision Trees and explained why we consider them only explainable. For Model-Agnostic methods that belong to the Post-Hoc methods, we have explained the Graph of Partial Dependence, Graphs of Individual Conditional Expectation, Graphs of Accumulated Local Effects, Locally Interpretable Model Agnostic (LIME), and SHAP values.

All these methods allow us to see any model. For model-specific methods, we concentrated on Convolutional Neural Networks, and showed which improvements exist in visibility maps of CNN. We also showed Interpretable methods for time-series CNN.

Životopis

Rođen u Rijeci 16.03.1996. Godine 2014. završavam u Prirodoslovnu gimnaziju u Prirodoslovno-grafičkoj školi Rijeka. Nakon završetka srednje škole upisujem Prirodoslovno-matematički fakultet u Zagrebu. Za vrijeme studiranja radio sam u ugostiteljstvu, te u tvrtkama poput Photomatha, Apis-it, Algebre te sam postao stipendist PBZ-a, gdje trenutno radim u Big data timu.