**TEHNIČKO VELEUČILIŠTE U ZAGREBU**

**STRUČNI STUDIJ RAČUNARSTVA**

Kristijan Vrabec

Dubinska analiza podataka u svrhu generiranja kviz pitanja

DIPLOMSKI RAD br. I 969

Zagreb, rujan, 2018.

**TEHNIČKO VELEUČILIŠTE U ZAGREBU**

**STRUČNI STUDIJ RAČUNARSTVA**

Kristijan Vrabec

JMBAG: 0246051246

Dubinska analiza podataka u svrhu generiranja kviz pitanja

DIPLOMSKI RAD br. I 969

Povjerenstvo:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Zagreb, rujan, 2018.

A close up of text on a white background

Description generated with high confidence

**Sažetak**

U ovom dokumentu opisana je dubinska analiza podataka i tehnologije u kojima se ona obavlja kao i strojno učenje i pripadajuće tehnologije. Opisan proces izrade praktičnog rada u programskom jeziku Python, označavanje rečenica NER oznakama, generiranja pitanja iz tih rečenica i prikaz na korisničkom sučelju razvijenom u programskom jeziku C#.

**Sadržaj**

[1 Uvod 1](#_Toc531977284)

[2 Dubinska analiza podataka 2](#_Toc531977285)

[2.1 Proces rudarenja 2](#_Toc531977286)

[2.1.1 Prikupljanje podataka 3](#_Toc531977287)

[2.1.2 Filtriranje/selekcija podataka 4](#_Toc531977288)

[2.1.3 Normalizacija podataka 4](#_Toc531977289)

[2.1.4 Obrada podataka 5](#_Toc531977290)

[2.1.5 Analiza podataka 6](#_Toc531977291)

[2.1.6 Izvlačenje informacija 6](#_Toc531977292)

[2.2 Tehnologije dubinske analize podataka 7](#_Toc531977293)

[2.3 Primjene dubinske analize podataka 8](#_Toc531977294)

[2.3.1 Rudarenje teksta 8](#_Toc531977295)

[2.3.2 Medicina 8](#_Toc531977296)

[2.3.3 Procjena rizika 8](#_Toc531977297)

[2.3.4 Marketing 8](#_Toc531977298)

[3 Proces strojnog učenja 9](#_Toc531977299)

[3.1 Strojno učenje 9](#_Toc531977300)

[3.2 Vrste strojnog učenja 10](#_Toc531977301)

[3.3 Koraci strojnog učenja 12](#_Toc531977302)

[3.3.1 Prikupljanje podataka 12](#_Toc531977303)

[3.3.2 Priprema i prilagodba podataka za strojno učenje 13](#_Toc531977304)

[3.3.3 Kreiranje modela 14](#_Toc531977305)

[3.3.4 Treniranje modela 17](#_Toc531977306)

[3.3.5 Evaluacija modela 18](#_Toc531977307)

[3.3.6 Mijenjanje parametara ovisno o rezultatima 18](#_Toc531977308)

[4 Tehnologije strojnog učenja 20](#_Toc531977309)

[4.1 Lokalne tehnologije strojnog učenja 20](#_Toc531977310)

[4.1.1 TensorFlow 20](#_Toc531977311)

[4.1.2 Keras 22](#_Toc531977312)

[4.1.3 Caffe 22](#_Toc531977313)

[4.1.4 Usporedba biblioteka i programskih okvira 22](#_Toc531977314)

[4.1.5 Python 23](#_Toc531977315)

[5 Zaključak 24](#_Toc531977316)

[Literatura 25](#_Toc531977317)

**Kratice i oznake**

**Slike**

[Slika 1 Dijagram procesa dubinske analize podataka 2](#_Toc531968139)

[Slika 2 Primjer HTML koda sa "nebitnim" podacima 4](#_Toc531968140)

[Slika 3 Primjer POS označavanja 5](#_Toc531968141)

[Slika 4 Nadzirano učenje 10](#_Toc531968142)

[Slika 5 Nenadzirano učenje 10](#_Toc531968143)

[Slika 6 Primjer nepotpunih/netočnih podataka 12](#_Toc531968144)

[Slika 7 Podjela početnog seta podataka na trening i test set 13](#_Toc531968145)

[Slika 8 Primjer regresijske linije Udjel poreza u BDP u usporedbi s BDP po glavi 14](#_Toc531968146)

[Slika 9 Graf stope netočnosti po vremenu 16](#_Toc531968147)

[Slika 10 Utjecaj stope učenja na stopu netočnosti 18](#_Toc531968148)

[Slika 11 Primjer TensorBoard vizualizacije 20](#_Toc531968149)

**Tablice**

[Tablica 1 Usporedba programskih jezika 22](#_Toc531968150)

# Uvod

Dubinska analiza podataka je u današnje vrijeme postala dio svakodnevnice. Posvuda u našoj okolini se nalaze podaci dobiveni dubinskom analizom te ih koristimo kako bi iz velike količine podataka mogli izvući nama potrebne informacije. U ovom diplomskom radu opisati će se proces i tehnologije dubinske analize podataka te proces i tehnologije strojnog učenja.

Proći će se kroz proces dubinske analize podataka. Objasnit će se koraci dubinske analize podataka, od prikupljanja podataka, filtriranja nebitnih podataka, normalizacije do pronalaženja uzoraka i izvlačenja korisnih informacija.

Proučit će se pojam i vrste strojnog učenja te će se ukratko objasniti koraci koji su neophodni u strojnom učenju. Od prikupljanja podataka, prilagodbe tih podataka za strojno učenje, kreiranja modela i treniranja istog, do evaluacije i uglađivanja parametara kako bi krajnji rezultat bio što točniji i precizniji.

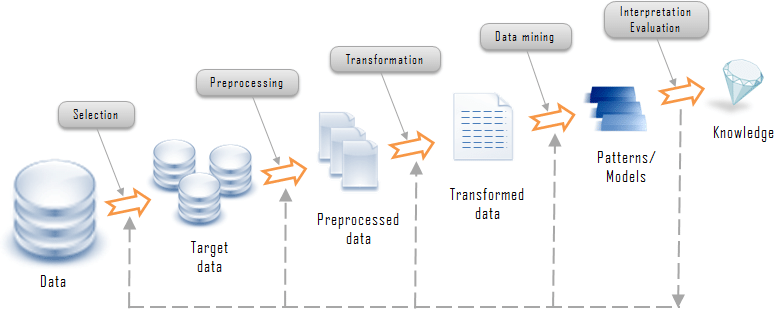
Proučit će se i usporediti najpopularnije tehnologije za strojno učenje te će se objasniti prednosti i nedostaci pojedinih tehnologija. Također, usporedit će se i nekoliko najpopularnijih programskih jezika za strojno učenje te na kraju objasniti odabir programskog jezika Python za izradu praktičnog djela.

Na kraju će se opisati postupak izrade praktičnog dijela ovog diplomskog rada sa objašnjenim primjerima koda i slikama aplikacije.

# Dubinska analiza podataka

Dubinska analiza podataka je proces otkrivanja zanimljivih i korisnih uzoraka i odnosa u velikim količinama podataka. Kombinira znanja iz statistike i umjetne inteligencije sa bazama podataka kako bi analizirali velike digitalne kolekcije podataka.[[1]](#footnote-1) Dubinska analiza podataka usko je povezano sa strojnim učenjem. Podaci dobiveni dubinskom analizom često se koriste kao ulazni set podataka za strojno učenje. Podaci za dubinsku analizu mogu biti bilo što, od komentara i objava sa društvenih mreža, raznih tekstova i književnih djela do običnog niza brojeva.

## Proces rudarenja



Slika 1 Dijagram procesa dubinske analize podataka

Proces dubinske analize podataka je dugotrajan i ne postoji univerzalno rješenje. Svaki set podataka zahtjeva poseban način obrade, no može se podijeliti na nekoliko segmenata, a to su redom:

* Prikupljanje podataka
* Filtriranje/selekcija podataka
* Normalizacija podataka
* Obrada podataka
* Analiza podataka
* Izvlačenje znanja

### Prikupljanje podataka

Proces rudarenja podataka započinje pronalaženjem odgovarajućeg izvora i prikupljanjem podataka s njega. Što je izvor čišći, tj. što manje netočnih i nebitnih riječi ima, to će proces dubinske analize podataka biti točniji i učinkovitiji. Potrebno obratiti pozornost na privatnost podataka, tj. jesmo li autorizirani koristiti te podatke, te ako jesmo, u koje svrhe ih smijemo koristiti. Neki od mogućih izvora podataka su:

* Publikacije (knjige, časopisi, novine)
* Ankete (telefonske, papirnate, online)
* Internet (web stranice, podatkovni setovi)

Publikacije su najsigurniji izvori podataka, pošto netko vodi računa da su podaci koji se nalaze u knjigama i novinama uglavnom gramatički i pravopisno točni. U takve izvore podataka ulaze i stare knjige koje se digitaliziraju kako bi se očuvala djela autora i kulturna baština te se kasnije javno objavljuju u edukativne svrhe. Najbolji primjer takvih starih digitaliziranih djela se nalazi u NLTK[[2]](#footnote-2) biblioteci u Python programskom jeziku, gdje se nalaze djela poput „Moby Dick“-a.

Ankete su dobri izvori podataka ako se traži nečije mišljenje ili stajalište o nečemu. Ispunjavaju ih ljudi te se ti podaci kasnije koriste kako bi se npr. unaprijedila neka usluga ili proizvod. Problem kod takvog izvora podataka je nepotpunost podataka, jer korisnici uglavnom ne ispune anketu na željeni način ili daju lažne podatke što utječe na krajnji rezultat.

Na kraju, Internet je najdostupniji i najlakši izvor podataka, no i najnesigurniji. Nitko ne garantira da su podaci preuzeti sa interneta točni i potpuni. Kao primjer se može pogledati stranica „Wikipedia“, koja je uglavnom glavni izvor podataka i informacija za većinu ljudi. Iako postoje ljudi koji se brinu za ispravnost sadržaja, mogućnost da svaki korisnik upisuje podatke i informacije na stranici ostavlja veliku mogućnost da ti podaci neće biti u potpunosti točni. No postoje razne stranice koje nude gotove setove podataka za razne potrebe, poput „UCI“, „Kaggle“, „Google“… Setovi podataka sa takvih izvora su puno sigurniji odabir pošto su podaci uglavnom već formatirani i spremni za čitanje. Te podatke je netko pripremio i obradio kako bi se koristili u svrhu edukacije i raznih istraživanja, te se kasnije javno objavio. Takvi podaci uglavnom sadrže vrlo malo grešaka.

### Filtriranje/selekcija podataka

Nakon prikupljanja, potrebno je selektirati željene podatke. Ako je prikupljen gotov set podataka, odnosno podaci su strukturirani, takvi setovi podataka su često multifunkcionalni i mogu se koristiti u različite namjene. Stoga je potrebno iz tog seta izvući potrebne podatke.

Ako pak ne postoji struktura podataka, što je slučaj kod prikupljanja podataka koji nije podatkovni set, tada je potrebno kreirati vlastitu strukturu podataka prema kojoj će se selektirati podaci. Ovaj korak se najviše dotiče sa web stranicama. Preuzimanje podataka s web stranica uključuje preuzimanje HTML[[3]](#footnote-3) koda stranice, te je potrebno izbaciti nebitne elemente iz HTML koda, poput skripti i meta podataka. Potrebno je selektirati samo podatke koji imaju neki značaj.



Slika 2 Primjer HTML koda sa "nebitnim" podacima

### Normalizacija podataka

Mnoge riječi imaju isto značenje, isto se čitaju, no drugačije se pišu. Primjerice, riječ „Coca-Cola“ možemo napisati na nekoliko različitih načina: „Coca-Cola“, „Coca-cola“, „Coca Cola“, „Coca cola“, “Cola“ … Sve te riječi imaju isto značenje, odnosno fonološki sve su riječi iste, no grafološki gledano, možemo primijetiti očigledne razlike u vidu velikog i malog slova, te spojnice (crtica koji spaja dvije riječi). Ovo je sam jedan od primjera gdje normalizacija podataka povećava točnost krajnjeg rezultata. Takvi se podaci često moraju ručno tražiti i ispravljati. Također, u gornjem primjeru možemo vidjeti i razliku velikog i malog slova, čime „Coca cola“ neće biti jednako „Coca Cola“. Taj se problem rješava tako da se sve riječi svedu na mala slova.

Dijakritički znakovi[[4]](#footnote-4) također predstavljaju problem. Na primjer „Luka Modrić“ se često može vidjeti da kao „Luka Modric“, što predstavlja problem. Zahtjeva dodatnu potrošnju vremena pri spremaju podataka tako što se mora paziti na kodiranje (eng. *encoding*) znakova. Ovaj se problem rješava tako da se zamjene dijakritički znakovi sa najbližim srodnim znakom.

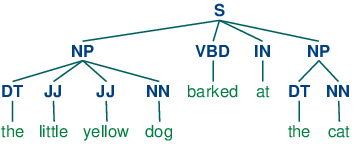
Uklanjanje *stopword*-a[[5]](#footnote-5) još je jedna metoda za normalizaciju podataka. Tako se može ukloniti velik dio podataka koji ne bi imali utjecaj na konačan rezultat, te se time dodatno poboljšati konačni rezultat.

### Obrada podataka

Nakon filtriranja i normalizacije, dolazi obrada podataka. Podaci se obrađuju ovisno o svrsi u koju se koriste. Pošto se u ovom diplomskom radu govori o generiranju pitanja, fokus će biti na obradi potrebnoj za tu svrhu. Kao ulazni podatak će se uzeti velika količina teksta.

Za početak, tekst je potrebno podijeliti na rečenice. Zvuči kao vrlo jednostavan zadatak, tražimo rečenične znakove koji označavaju kraj rečenice, no i nije sve tako jednostavno. Jedan od problema su kratice, koje se često pišu s točkom. Jedno od rješenja može biti kreiranje liste kratica te ignoriranje tih riječi. Također, postoje razne biblioteke, poput već spomenutog NLTK, koje će takve stvari odraditi.

Rečenice je zatim potrebno podijeliti na riječi, koje je zatim označiti (eng. *tagging*) POS (eng. *Part-of-speech*) oznakama. Riječi se označavaju po tome koja su vrsta riječi (imenica, glagol,…). Tako se dobivaju odnosi između riječi u rečenici.



Slika 3 Primjer POS označavanja

### Analiza podataka

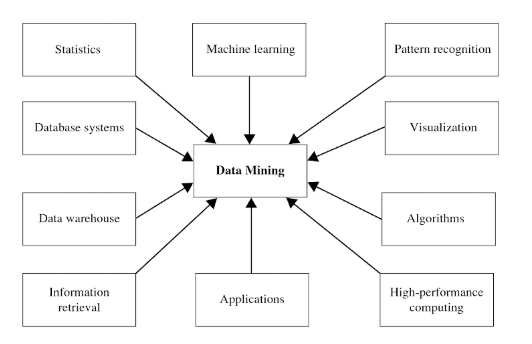
### Izvlačenje znanja

## Tehnologije dubinske analize podataka

Dubinska analiza podataka je integracija više tehnologija, a neke od bitnijih tehnologija su:

* Statistika
* Strojno učenje
* Baze i skladišta podataka
* Izvlačenje informacija

Uglavnom se ne koristi samo jedna, nego kombinacija tih tehnologija.



Slika 4 Tehnologije dubinske analize podataka

### Statistika

Statistika proučava prikupljanje, analizu, interpretaciju i prezentaciju podataka. Statistika doprinosi dubinskoj analizi podataka statističkim modelom, setom matematičkih funkcija koje opisuju ponašanje objekata u određenoj klasi na temelju nasumičnih varijabli i njihovih pridruženih raspodjela vrijednosti. Koriste se kako bi se kreirao model podataka i klasa te mogu biti rezultat dubinske analize podataka, isto kao što može biti i ulazni set za dubinsku analizu podataka. Primjerice, može se kreirati statistički model za oštećene (eng. *corrupted*) i nedostajuće podatke te pomoći otkriti i sanirati te podatke.

### Strojno učenje

Dubinska analiza podataka je jako blizak pojam sa strojnim učenjem. Glavna zamisao je da se trenira model raznim pravilima uzorcima dobivenim dubinskom analizom te primjena tih pravila na rješavanje problema. Kombinira se sa statističkim tehnikama kako bi se razvile sofisticirane statističke tehnike za dubinsku analizu podataka. Detaljnije o strojnom učenju u sljedećem poglavlju.

### Baze i skladišta podataka

Baze podataka su organizirane kolekcije podataka. Kako je za dubinsku analizu podataka potrebna velika količina ulaznih podataka, baze podataka su idealno rješenje, zbog svoje optimizacije, skalabilnosti i sposobnosti za procesuiranje velikih količina podataka. Također, dubinska analiza podataka se može koristiti u bazama podataka kako bi proširile mogućnosti koje baze podataka nude. Skladišta podataka mogu sadržavati podatke iz više iz više različitih izvora, te se koriste za analizu podataka i kreiranje izvještaja.

### Izvlačenje informacija

Izvlačenje informacija (eng. *Information retrievel*) je znanost traženja dokumenata nestrukturirane prirode (uglavnom tekst ili multimedija) koji zadovoljava informacija u velikoj kolekciji (bazi podataka na računalu). Razlika između baza podataka i izvlačenja informacija je u dvije stvari:

* Baze očekuju strukturirane ulazne podatke
* Upiti su formirani po ključnim riječima iz teksta, ne u nekom jeziku (SQL[[6]](#footnote-6))

Tipično izvlačenje informacija uključuje model vjerojatnosti, u kojem se ulazni tekstni dokument prezentira kao set riječi koje kreiraju jezični model, gustoći određenih riječi u tekstu. Sličnost između dva dokumenta je rezultat sličnosti njihovih modela.

## Primjene dubinske analize podataka

Dubinska analize podataka može se koristiti u mnogim područjima, od proizvodnje i marketinga, medicinskih istraživanja, istraživanja prijevara do analize tržišta za velike trgovačke lance.

### Rudarenje teksta

Dubinske analize podataka se može primijeniti na različite vrste tekstualnih dokumenata, koji predstavljaju nestrukturirane podatke, kako bi se klasificirali članci, knjige, dokumenti, elektronička pošta i web stranice. Dobar primjer su web stranice, koje sadrže, osim teksta, i različite HTML elemente (<div>, <body>, …) i entitete (&nbsp;, &euro;…). Razne online aplikacije (email klijenti) već imaju ugrađene filtere neželjene („*spam*“) pošte koji rade na principu dubinske analize podataka teksta.

### Medicina

Iskoristivost dubinske analize podataka je neprocjenjiva u medicini. Omogućuje rano otkrivanje bolesti na temelju kliničkih testova i istraživanja, smanjuje troškove liječenja te poboljšava skrb za pacijenta.

### Procjena rizika

Analiza rizika je vrlo bitan segment svih velikih događanja, putovanja, ulaganja. Na primjer, turističke agencije koriste procjenu rizika kako bi svojim putnicima priuštili sigurna putovanja. To rade na način da rudare podatke o broju kriminalnih djela, vremenskim uvjetima, prirodnim katastrofama i slično te tako rangiraju turističke destinacije.

### Marketing

Dubinska analiza podataka na području marketinga uvelike je poboljšala rezultate marketinških kampanja. Neki od bitnijih segmenata u području marketinga su analiza tržišta, analiza potrošačke košarice, predviđanje stope pozitivne reakcije na marketinške kampanje, analiza ponašanja kupaca u trgovini i sl. Također, danas je vrlo popularna analiza povijesti pretraživanja interneta preko koje se korisniku prikazuju reklame u ovisnosti o pojmovima koje je pretraživao.

# Proces strojnog učenja

U ovom poglavlju će se proći kroz proces strojnog učenja. No da bi razumjeli proces strojnog učenja, prvo valja razumjeti pojam sam strojnog učenja.

## Strojno učenje

Da bi razumjeli što je strojno učenje, prvo je potrebno znati sljedeće pojmove: stroj, inteligencija, umjetna inteligencija te učenje. Stroj (ili uređaj) jest skup dijelova povezanih u jednu logičnu cjelinu s ciljem izvođenja određene operacije. Za primjer strojnog učenja, kao stroj se može uzeti računalo. Računalo samo po sebi nema sposobnost samostalnog donošenja odluka, iako su računala u današnje vrijeme došla vrlo blizu tom koraku, odnosno podarena im je umjetna inteligencija.

Proučimo definiciju inteligencije. Inteligencija je mentalna karakteristika koja se sastoji od sposobnosti za učenje iz iskustva, prilagodbe na nove situacije, razumijevanja i korištenja apstraktnih pojmova i korištenja prethodnih znanja za snalaženje u novoj okolini u kojima ne pomaže stereotipno nagonsko ponašanje. Prema definiciji, primjena stečenih znanja i iskustava u novim okolinama je glavna karakteristika inteligencije. Ako inteligenciju promatramo sa stajališta računala, možemo je objasniti na primjeru prepoznavanja slika, odnosno objekata na slikama. Testnom subjektu predočena je određenu količinu uzoraka nekog objekta, npr. slike mačaka, te nakon određenog perioda taj bi subjekt trebao imati mogućnost prepoznati traženi objekt na novom uzorku, nekom iz kojeg nije učio te ga prvi puta vidi. Kada računalo ima tu mogućnost, onda kažemo da posjeduje umjetnu inteligenciju. Umjetna inteligencija (UI) je naziv koji pridajemo svakom neživom sustavu koji pokazuje sposobnost snalaženja u novim situacijama (inteligenciju). Inteligencija se stječe učenjem novih pojmova i znanja te je učenje ključno za stjecanje i povećanje inteligencije.

Definicija učenja, prema rječniku, je:

* Dobivanje znanja učenjem, iskustvom
* Učenje preko informacija ili promatranjem
* Pamćenje
* Dobivanje instrukcija
* Utvrđivanje

Ta definicija vrijedi za ljude, dok za računala vrijede samo zadnje tri točke. Prve dvije točke ne mogu biti primijenjene na računala jer je nemoguće utvrditi da li je računalo samostalno prikupilo informacije i naučilo, ne možemo ga pitati ili očitati informacije. Zadnje tri točke su dosta trivijalne i jednostavne i na takav način se mogu izvršiti na računalu.

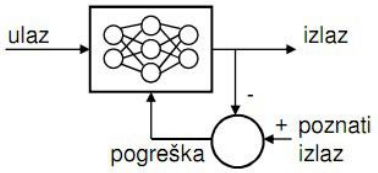
Strojno učenje je grana umjetne inteligencije koje često koristi statističke tehnike kako bi računalu dalo mogućnost da uči iz podataka, bez da je eksplicitno programirano. Strojno učenje se temelji na algoritmima koji imaju mogućnost učiti i dati predikciju rezultata temeljenu na nekoj vrsti podataka. Takvi algoritmi daju rezultate temeljene na prijašnjim ulaznim podacima te balansiraju svoje parametre ovisno o novim ulazima, ne slijede strogo zadana pravila. Dobar primjer bio bi email filter, koji ima početni ulazni set podataka i ključne riječi koje traži u tekstu, te kasnije prilagođava svoj korpus ovisno o korisničkim unosima. Nove email-ove, koji su označeni kao „*spam*“, privlači kroz razne algoritme i uspoređuje s drugima te tako pronalazi nove ključne riječi i dodaje ih u svoj korpus. Iako takvi algoritmi rade sa velikom točnošću, i dalje nisu savršeni i mogu pogriješiti.

## Vrste strojnog učenja

Postoji nekoliko vrsta strojnog učenja:

* Nadzirano učenje
  + Polu-nadzirano učenje
  + Aktivno učenje
  + Učenje pojačavanjem
* Nenadzirano učenje

U nadziranom učenju dostupan je skup uzoraka za učenje i za svaki uzorak je poznat rezultat. Tijekom procesa učenja izvode se iteracije ponavljanja istih uzoraka sve dok izlaz ne odgovara očekivanim rezultatima.

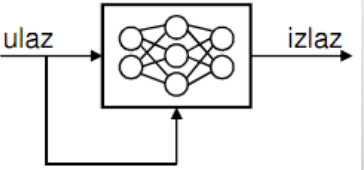


Slika 5 Nadzirano učenje

Polu-nadzirano učenje se temelji na nadziranom učenju, no uzorci često nisu potpuni, odnosno svi uzorci za učenje sadrže ulazne parametre, ali samo neki sadrže i izlazne.

Aktivno učenje je specijalan slučaj polu-nadziranog učenja u kojem algoritam zahtjeva unose korisnika za određeni ulaz kako bi se dobio željeni izlaz za nove uzorke.

Nenadzirano učenje je slično kao i nadzirano učenje, no za skup uzoraka ne postoje poznati rezultati koji bi se uspoređivali sa izlazima. Tijekom učenja mreža na različite ulazne uzorke reagira na različite načine te tako stvara internu reprezentaciju ulaznih podataka. Ova vrsta učenja se najčešće koristi kod grupiranja podataka te prepoznavanja sličnosti.



Slika 6 Nenadzirano učenje

Učenje pojačavanjem je hibridni način koji uključuje aspekte nadziranog i nenadziranog učenja. Neuronska mreža prima uzorke za učenje za koje nema očekivanog rezultata, ali je za svaki izlaz poznato je li on ispravan ili neispravan za zadani ulaz, što je postignuto povratnom vezom.

## Koraci strojnog učenja

Strojno učenje je grana umjetne inteligencije koja se još uvijek uvelike razvija i otkrivaju se novi i efikasniji načini kako bi proces bio što učinkovitiji. No, iako se strojno učenje još uvijek razvija, postoje određeni koraci koji će se uvijek morati slijediti, a to su redom:

* Prikupljanje podataka
* Priprema i prilagodba podataka za strojno učenje
* Kreiranje modela
* Treniranje modela
* Evaluacija modela
* Mijenjanje parametara ovisno o rezultatima

U daljnjem tekstu se opisuje proces nadziranog strojnog učenja.

### Prikupljanje podataka

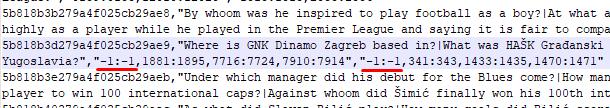
Prikupljanje podataka prvi je korak u procesu strojnog učenja. Što je veća količina podataka koju prikupimo, to će veća biti konačna točnost istreniranog modela. Podaci se mogu prikupljati sa razni izvora:

* Papirnati (knjige, časopisi, novine)
* Ankete (telefonske, papirnate, online)
* Internet

Pri odabiru izvora podataka treba voditi računa da podaci mogu biti nepotpuni ili netočni. Papirnati izvori su najsigurniji izvori podataka, pošto netko vodi računa da su podaci koji se nalaze na papiru uglavnom točni pošto se ne mogu korigirati. Uglavnom se takvim izvorima podataka smatraju stare knjige koje se digitaliziraju kako bi se očuvala djela autora i kulturna baština. Ankete su dobri izvori podataka ako se traži nečije mišljenje ili stajalište o nečemu. Ispunjavaju ih ljudi te se ti podaci kasnije koriste kako bi se npr. unaprijedila neka usluga ili proizvod. Problem kod takvog izvora podataka je nepotpunost podataka, jer korisnici uglavnom ne ispune anketu na željeni način ili daju lažne podatke što utječe na krajnji rezultat. Na kraju, Internet je najdostupniji i najlakši izvor podataka, no i najnesigurniji. Nitko ne garantira da su podaci preuzeti sa interneta točni i potpuni. Kao primjer se može pogledati stranica „Wikipedia“, koja je uglavnom glavni izvor podataka i informacija za većinu ljudi. Iako postoje ljudi koji se brinu za ispravnost sadržaja, mogućnost da svaki korisnik upisuje podatke i informacije na stranici ostavlja veliku mogućnost da ti podaci neće biti točni. No postoje razne stranice koje nude gotove setove podataka za razne potrebe, poput „UCI“, „Kaggle“, „Google“… Setovi podataka sa takvih izvora su puno sigurniji odabir od kreiranja svojeg izvora, naravno ako nam takvi podaci odgovaraju.

### Priprema i prilagodba podataka za strojno učenje

Podaci prikupljeni sa bilo kojeg izvora nisu dobar ulazni set za strojno učenje, potrebno ih je prilagoditi kako bi ih algoritam mogao razumjeti. Za početak, može se proći kroz sve podatke i ukloniti duplikate, jer dupli podaci u setu podataka stvaraju samo smetnju i mogu imati negativan utjecaj na rezultate strojnog učenja. Zatim je potrebno ispraviti greške i nedostatke u podacima. Pravopisne greške i riječi istog značenja, no drugačije napisane, kao npr. automobil i auto, Coca-Cola i Coca Cola i Cola, imaju utjecaj na konačne rezultate te bi ih trebalo ispraviti ako je moguće. Ako podaci ne sadrže neke vrijednost, tj. nepotpuni su, može doći do grešaka u algoritmu za učenje ili algoritam može naučiti krive vrijednosti, stoga je te podatke potrebno ispraviti ili izbaciti kako bi ubrzao i unaprijedio proces učenja. No isto tako je moguće da te vrijednosti neće imati značajan utjecaj na proces učenja, stoga se mogu zanemariti. Vrijednosti koje nedostaju su uglavnom označene sa numeričkom vrijednosti (-1, 0), praznim mjestom ili '-' znakom.

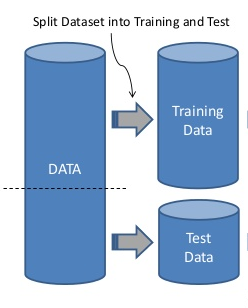


Slika 7 Primjer nepotpunih/netočnih podataka

Nakon uklanjanja duplikata i grešaka, potrebno je pretvoriti podatke u računalu razumljive podatke i normalizirati vrijednosti. To ukratko znači da bi svaki podatak iz ulaznog seta podataka trebao imat jedinstvenu brojčanu vrijednost kako bi ih algoritam za strojno učenje mogao razlikovati. Za pretvaranje ulaznih podataka u brojčane vrijednosti postoje razna gotova rješenja, a ovdje će se opisati postupak pretvaranja teksta u brojčane vrijednosti, pošto se koristi kasnije u praktičnom dijelu.

#### Kreiranje seta podataka

Daljnja prilagodba podataka odvija se tako da se podaci dijele u setove. Jedan se set podataka koristi za učenje i formiranje modela, odnosno treniranje, trening set, dok se drugi set podataka koristi za test modela, testni set. Trening set podataka trebao bi sadržati sve vrijednosti, odnosno ulazne i izlazne podatke, dok bi testni set podatka trebao sadržavat samo ulazne podatke. Testni set podataka ne bi se trebao mijenjati sve dok se ne kreira najbolji model. Koristi se za evaluaciju modela kao set nepoznatih i još ne viđenih podataka. Omjer tih setova u odnosu na sveukupni set podatka je uglavnom 7:3 ili 8:2. Dobra je praksa nasumično spremati podatke u setove podataka, kako bi se izbjeglo „navikavanje“ algoritma na određene vrijednosti.



Slika 8 Podjela početnog seta podataka na trening i test set

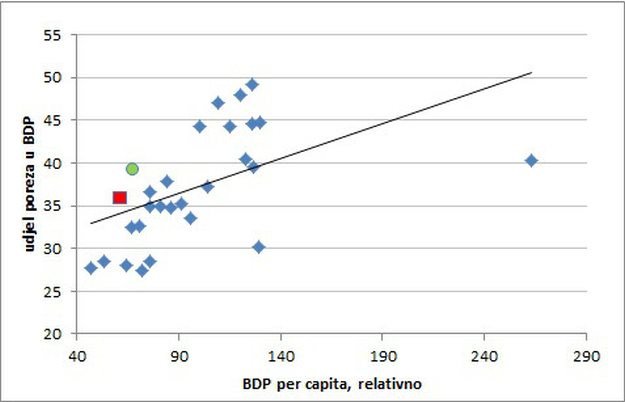
### Kreiranje modela

Kreiranje modela ovisi o ulaznim podacima. Model se kreira prema parametrima ulaznih podataka, odnosno mora imati onoliko ulaza koliko je ulaznih podataka, te isto tako i sa izlazima. Između ulaznog i izlaznog sloja se nalaze slojevi koji sadrže neurone čije vrijednosti zapravo određuju model. Također, moguće je kombinirati više modela tako da izlaz jednog bude ulaz drugog. Postoji mnogo gotovih modela koje su znanstvenici kreirali pomoću svojih istraživanja koji se vrlo lako mogu pronaći i upotrijebiti. Primjerice, postoje gotovi modeli za slikovne tipove podataka, tekstualne, numeričke i sl. Jedan od glavnih dijelova modela je algoritam za strojno učenje. Zapravo taj algoritam određuje vrijednosti modela i dobar algoritam za učenje može uvelike ubrzati proces učenja. Neki od poznatijih algoritama za učenje su:

* Linearna regresija (engleski. *Linear Regression*)
* Stabla odluke (engleski. *Decision Trees*)
* Naivna Bayes-ova klasifikacija (engleski. *Naive Bayes Classification*)

#### Linearna regresija

Linearna regresija je idealna kada su svi podaci numerički. Ovaj algoritam ostvaruje vežu između ulaznih i izlaznih vrijednosti preko najboljeg pravca, koji se još naziva regresijska linija. Najčešće se koristi za procjenu rasta i pada cijena nekretnina, dionica i sl. Vrlo je dobar izbor za statističke procjene, no problem je ako podaci pokazuju nelinearnu zavisnost, tada se regresijska linija nikada neće pronaći.



Slika 9 Primjer regresijske linije Udjel poreza u BDP u usporedbi s BDP po glavi

#### Stabla odluke

Stabla odluke, kao i svugdje, rade na principu podjeli pa vladaj. U ovom algoritmu se vrijednosti dijele na dvije ili više homogenih setova. Mora se pronaći najčišći ulazni atribut koji će se pozicionirati kao korijen stabla. Mjera za čistoću ulaznog atributa je informacija i mjeri se u bitovima. Informacija predstavlja očekivanu količinu informacija koja je potrebna da bi se odredilo da li je novi ulazni podatak klasificiran kao istina ili laž. Računa se prema sljedećoj formuli:

A predstavlja atribut na koji se ulazni set dijeli, S je ulazni set podataka, T je set podataka kreiran razdvajanjem atributa A od seta S, p(t) je udio broja elemenata t u setu S. H(S) je entropija seta S, koja predstavlja čistoću podataka u proizvoljnoj classifier kolekciji podataka, dok je H(t) entropija seta t. Entropija se računa prema sljedećoj formuli:

gdje je S je ulazni set podataka, p(c) udio elemenata klase c u setu S. Na kraju se odabire atribut kojem je vrijednost informacije najveća. Ista stvar se ponavlja sa ostale atribute dok se stablo potpuno ne složi.

Stabla odluke se najčešće koriste za klasifikaciju problema. Za razliku od linearnih algoritama, nemaju problema sa nelinearnim vrijednostima.

#### Naivna Bayes-ova klasifikacija

Temeljena na Bayes-ovom teoremu, ova tehnika klasifikacije se temelji na nepovezanosti atributa ulaznog seta podataka. Primjerice, jabuka je okrugla, crvena i veličine oko 10 centimetara u promjeru. Iako te vrijednosti zavise jedna o drugoj, svaka od njih zasebno može doprinijeti odluci algoritma da je ulazni podataka zapravo jabuka, te se zbog toga ova klasifikacija naziva naivnom.

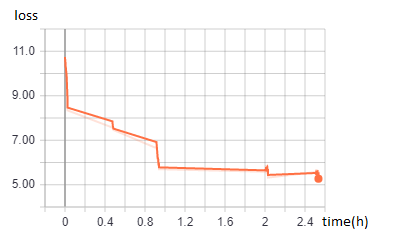
Naivna Bayes-ova klasifikacija idealna je za velike setove podataka. Poznata je po svojoj jednostavnosti te performansama čak i sa vrlo kompleksnim problemima. Izlazna vjerojatnost se računa prema formuli:

gdje je x ulazni podatak, c je atribut podatka, P(c | x) je izlazna vjerojatnost za ulazni podatak c, P(c) udio ulaznog atributa u ukupnom setu podataka dok je P(x) je udio ulaznog podatka u ukupnom setu podataka.

Zbog svoje brzine, ovaj se algoritam često koristi za predviđanje u realnom vremenu. Najčešće se koristi za klasifikaciju teksta i kao „spam“ filter za detektiranje neželjenih poruka i email-ova.

### Treniranje modela

Za treniranje modela koristi se trening set podataka. Treniranje se obavlja tako da se ulazni podaci iz trening seta daju modelu da prilagodi svoje parametre kako bi izlazi odgovarali onima iz trening seta. Krajnji cilj treninga je da model točno predvidi izlaze sa što većom točnošću. Ovo je ujedno i najduži dio procesa strojnog učenja. Ukoliko se koriste veliki setovi podataka, treniranje modela može trajati satima, danima ili čak mjesecima. Treniranje modela završava ili određenim brojem ponavljanja ili kada stopa netočnosti padne ispod određenog broja. Ta ponavljanja se nazivaju epohe. Jedna epoha je jedan prolaz ulaznog seta podataka kroz model, tj. epoha završava kada svi podaci iz ulaznog seta prođu kroz model. Stopa netočnosti je interpretacija koliko dobro je model prilagodio svoje parametre.



Slika 10 Graf stope netočnosti po vremenu

### Evaluacija modela

Nakon treninga modala dolazi evaluacija da se vidi da li je model zadovoljio. Ovaj bi korak trebao biti reprezentacija performansi modela u stvarnom svijetu. Evaluacija modela se obavlja na više načina. Uglavnom se koriste neke metrike kao pokazatelji performansi modela.

Kao prvi način se može gledati stopa netočnosti koja je najbolji pokazatelj koliko su dobro postavljeni parametri modela. Što je niža stopa netočnosti, to je bolji model, osim u slučajevima kada je trening set previše tematski homogen, što može dovesti do „navikavanja“ modela na određene podatke, te loše performanse sa drugim podacima.

Drugi način je evaluacija uspješnosti modela na dosad neviđenim podacima. U tu svrhu se kreira evaluacijski set podataka koji nije tematski povezan sa trening ili test setom. Na taj se način zapravo najbolje testira mreža jer su ti podaci stvarno „neviđeni“, dok je testni set podataka također neviđen, ali je tematski vezan za trening set, iako se evaluacija može obaviti i sa testnim setom, ako je tematski heterogen.

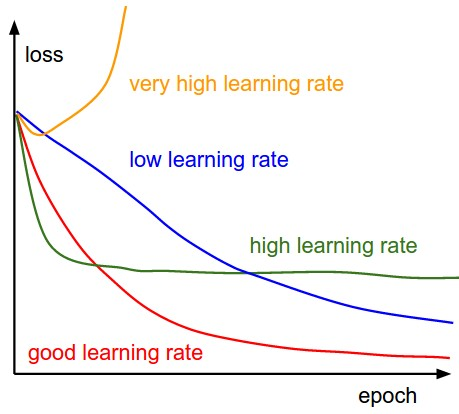
U konačnici, ako evaluacija modela zadovolji kriterije, model je spreman za stvarni svijet. No u većini slučajeva, odnosno gotovo nikad, prvi model nije idealan te zahtjeva male ili velike preinake kako bi rezultati bili što bolji.

### Mijenjanje parametara ovisno o rezultatima

Mijenjanje parametara ovisi o rezultatima evaluacije modela. Što su rezultati evaluacije bolji, to se manje promjena mora raditi. Parametri se mijenjaju kako bi se pokušalo dobiti što bolje rezultate u bilo kojem smislu, ili brži trening, ili veću točnost. Parametri koji se mogu mijenjati kako bi se poboljšali rezultati su sljedeći:

* Stopa učenja
* Model
* Algoritam za učenje
* Ulazni set podataka

Ako je model blizu stanja koje tražimo, onda je potrebno fino ugađanje parametara koje se postiže mijenjanjem stope učenja. Stopa učenja je parametar koji kontrolira koliko brzo će algoritam mijenjati svoje vrijednosti u svakom koraku. Što je taj iznos veći, to će algoritam brže učiti, no postoji i problem. Ako je iznos stope učenja prevelik, može doći do izobličenja vrijednosti te postupak učenja može otići u krivom smjeru. Ako je pak stopa učenja premalena, algoritam radi premalene promjene i presporo uči te je negativna posljedica to što proces učenja može predugo trajati. Također, mijenjanje stope učenja može pomoći ukoliko je stopa netočnosti zapela na određenoj vrijednosti. Stoga treba testirati sa raznim vrijednostima i vidjeti koja najbolje odgovara. Uz stopu učenja, mogu se mijenjati i drugi parametri algoritma koji mogu imati utjecaj na proces učenja, koji se mogu pronaći u dokumentaciji algoritma.



Slika 11 Utjecaj stope učenja na stopu netočnosti

Ako mijenjanjem stope učenja i drugih parametara nije uočena veća promjena u rezultatima, vrijeme je za uređivanje modela. Na modelu se rade sitne preinake kako bi se poboljšali rezultati poput dodavanja/oduzimanja slojeva neurona ili mijenjanja postojećih.

Mijenjanje algoritma za učenje malo je drastičniji korak. Pri promjeni algoritma treba voditi računa da odgovara našem modelu, odnosno da može raditi sa tipom podataka kojim mi raspolažemo. Nisu svi algoritmi za sve tipove podataka. Alternativni algoritmi kao i njihova upotreba mogu se pronaći u dokumentaciji programskog jezik algoritma.

Ukoliko niti jedan od prijašnjih koraka nije donio poboljšanje rezultata, promjena ulaznog seta podataka bi mogla. Moguće je da ulazni set podataka nije pravilno procesuiran i očišćen od nepotpunih i netočnih podataka ili su podaci tematski previše slični i mreža se previše „navikla“ na takve podatke.

# Tehnologije strojnog učenja

Već je zaključeno da je strojno učenje vrlo popularna grana umjetne inteligencije, stoga ne čudi interes mnogih velikih kompanija, poput Microsoft-a, Google-a, Amazon-a i sl. Svatko bi želio svoj udio u tehnologiji budućnosti. Velike kompanije nude vrlo popularna rješenja strojnog učenja u oblaku (engleski cloud). Osim rješenja u oblaku, postoje i biblioteke i programski okviri (engleski Framework) koje se izvršavaju na lokalnom računalu.

## Lokalne tehnologije strojnog učenja

Lokalne tehnologije zahtijevaju računalo na kojem će se izvršavati te biblioteke i programski okviri. Također, potrebno je samostalno napisati programski kod u nekom od programskih jezika za iskorištavanje tih biblioteka. Neke od poznatijih biblioteka i programskih okvira su:

* TensorFlow
* Keras
* Caffe

Najbolji programski jezici za strojno učenje su:

* Python
* R
* C/C++
* Java

### TensorFlow

TensorFlow je biblioteka otvorenog koda visokih performansi za numeričke obračune. Razvili su je Google-ovi stručnjaci iz Google Brain tima 2015. godine unutar Google-ove AI organizacije. TensorFlow je jedan od najpopularnijih i najraširenijih sustava za umjetnu inteligenciju i strojno učenje.

Arhitektura TensorFlow-a omogućava korisnicima razvoj i računanje na različitim platformama, od korištenja procesorske snage i snage grafičkih kartica do posebnih čipova za akceleraciju procesa strojnog učenja. Nove verzije TensorFlow-a omogućavaju integraciju umjetne inteligencije na mobilne uređaje, uz klasična stolna računala i servere, te web stranice zbog podrške za JavaScript, programski jezik koji se najčešće koristi na Web stranicama. Uz Python, koji standardni programski jezik za TensorFlow, podržani su još i Java, Go i C.

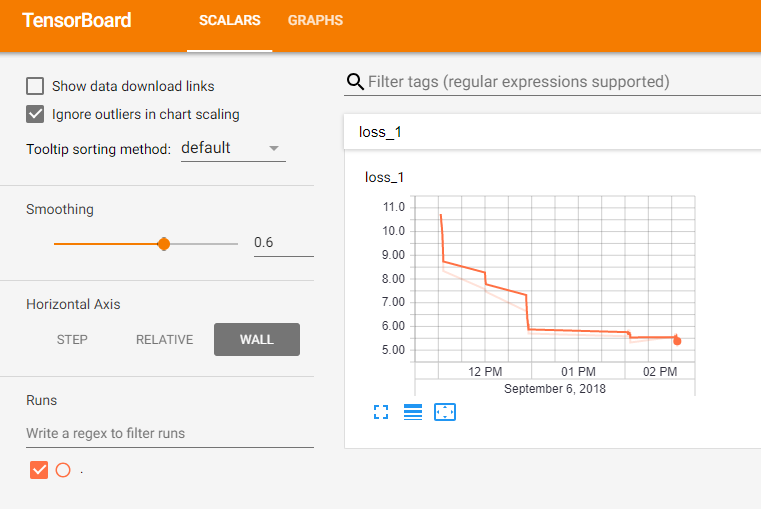
Kako bi se lakše razumjeli, debugirali i optimizirali TensorFlow programi, kreiran je alat za vizualizaciju nazvan TensorBoard. TensorBoard se pokreće preko komandne linije sljedećom komandom:

tensorboard --logdir=putanja\_do\_direktorija

te se predaje parametar –logdir koji je putanja do log direktorija TensorFlow algoritma. Nakon pokretanja, TensorBoard vizualizacija se otvara u Internet pregledniku preko adrese:

localhost:6006 ili ime\_računala:6006

TensorBoard se može koristiti za vizualizaciju grafa stope netočnosti, grafički prikaz modela i sl.



Slika 12 Primjer TensorBoard vizualizacije

### Keras

Kao i TensorFlow, Keras je također biblioteka otvorenog koda izdana 2015. godine. Kreiran je kao dopuna drugim bibliotekama i programskim okvirima poput TensorFlow-a. Keras se fokusira na korisničkoj interpretaciji, modularnosti i proširivosti. Kreiran je sa fokusom da omogući što kraći proces od ideje do rezultata.

Minimaliziran je broj korisničkih akcija za učestale slučajeve te vraća razumljive i poduzetne povratne informacije u slučaju grešaka. Slično kao i TensorFlow, može koristiti procesorsku snagu i snagu grafičkih kartica.

### Caffe

Službeno izdan 2017. godine, Caffe (Convolutional Architecture for Fast Feature Embedding) je programski okvir za strojno učenje koji se fokusira na brzinu i modularnost. Kao i prijašnji algoritmi, također je otvorenog koda, napisan u C++ sa Python sučeljem. Caffe svoju popularnost najviše može zahvaliti zajednici koja zajedno sa Berkeley AI Research (BAIR), istraživačkim laboratorijem za umjetnu inteligenciju, koji je razvio Caffe, i dalje razvija ovaj programski okvir.

Caffe se najviše koristi za klasifikaciju i segmentaciju slika, zvuka i ostalih multimedija. Na stranici <http://demo.caffe.berkeleyvision.org/> postoji i demo projekt koji reprezentira mogućnost Caffe programskog okvira.

### Usporedba biblioteka i programskih okvira

TensorFlow je daleko najpopularnija biblioteka od svih te ima najveću zajednicu. Zbog konstantnog razvijanja, TensorFlow ima budućnost ispred sebe i novi korisnici dolaze svakodnevno. Postoji i mnogo kvalitetne dokumentacije i literature. Jako je širok obujam problema koje TensorFlow može riješiti, te ima jako dobar vizualizacijski alat TensorBoard. Također, dostupan je i na mobilnim uređajima te na Webu.

Jednostavnost je glavna značajka Keras-a. Postoji velika podrška zajednice i dostupno je mnogo gotovih rješenja. Zbog svoje jednostavnosti i sučelja idealan je za učenje i početnike u strojnom učenju. Loša stvar je to što ima slabu podršku za treninge velikih obujama koji se obavljaju na više računala. Postoje i problemi u performansama u odnosu na TensorFlow.

Caffe je idealan za velike količine slikovnih podataka za slikovnu klasifikaciju, gdje je još uvijek bolji od TensorFlow-a. Također, postoji velik broj gotovih modela za računalnu viziju koje je moguće preuzeti i iskoristiti. Problemi nastaju kada treba dodati novi sloj u model, pošto je biblioteka napisana u C++, potrebno je mijenjati stvari. Isto tako, nije idealno za stvarni svijet pošto je još u razvijanju.

### Python

Python je daleko najpopularniji programski jezik za strojno učenje.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Programski jezik | Brzina | Efektivnost učenja | Podrška zajednice | Efektivnost | Spremnost za izdavanje | Biblioteke | Suma |
| Python | - | + | + | + | + | + | 5 |
| R | + | + | + | + | - | - | 4 |
| Octave | + | + | - | + | - | - | 3 |
| Matlab | + | + | + | - | - | - | 3 |

Tablica 1 Usporedba programskih jezika

# Zaključak

Područje strojnog učenja je vrlo zanimljivo i još je uvijek u razvoju. Ne možemo ni zamisliti što se sve može napraviti sa strojnim učenjem, koliko se stvari može automatizirati. Tek se u zadnjih nekoliko godina tržište počelo zanimati za proizvode sa umjetnom inteligencijom i dan danas taj interes samo raste. Google i Microsoft ulažu ogromne resurse u umjetnu inteligenciju jer su uvidjeli potencijal tog područja.

Rudarenjem podataka mogu se iz naočigled nasumičnih i nebitnih podataka izvući informacije ili iz hrpe podataka izvući neka informacija. U primjeru gore je pokazano kako je iz hrpe recenzija moguće izvući je li opće mišljenje dobro ili loše o određenoj marki automobila. Također je moguće, kasnije, pretražiti što se korisnicima ne sviđa te tako promijeniti te stvari u budućim verzijama automobila.

# Literatura

**Wikipedia**

<https://hr.wikipedia.org/wiki/Strojevi> (30.6.2018.)  
<https://hr.wikipedia.org/wiki/Inteligencija> (30.6.2018.)  
<https://hr.wikipedia.org/wiki/Umjetna_inteligencija> (30.6.2018.)  
<https://en.wikipedia.org/wiki/Machine_learning> (1.7.2018.)  
<https://hr.wikipedia.org/wiki/Rudarenje_podataka> (1.7.2018.)  
<https://en.wikipedia.org/wiki/Bayesian_network> (1.7.2018.)  
<https://en.wikipedia.org/wiki/Directed_acyclic_graph> (2.7.2018.)

**Knjige**

Data Mining, Third Edition, Ian H. Witten, Eibe Frank, Mark A. Hall, 2011.  
Data Mining: Technologies, Techniques, Tools, and Trends, Bhavani Thuraisingham, 1999

**Publikacije**

Open Source Data Mining Programs: A Case Study on R, Muhammet Sinan Başarslan, Fatih Kayaalp, Travanj 2018.   
Introduction to Information Retrieval - Stanford NLP, Online edition, 2009

**TEHNIČKO VELEUČILIŠTE U ZAGREBU**

**POLITEHNIČKI SPECIJALISTIČKI DIPLOMSKI STRUČNI STUDIJ**

**Specijalizacija elektrotehnika**

Kristijan Vrabec

JMBAG: 0246051246

Dubinska analiza podataka u svrhu generiranja kviz pitanja

DIPLOMSKI RAD br. I 969

Povjerenstvo:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Zagreb, rujan 2018.

1. Enciclopedia Britannica, <https://www.britannica.com/technology/data-mining> (1.7.2018.) [↑](#footnote-ref-1)
2. NLTK – Natural Language Toolkit – biblioteka u programskom jeziku Python za obradu teksta [↑](#footnote-ref-2)
3. HTML - HyperText Markup Language – prezentacijski jezik za izradu web stranica [↑](#footnote-ref-3)
4. Dijakritički znakovi - znakovi sa preglasima, kvačicama i sl. [↑](#footnote-ref-4)
5. Stopword - riječi koje nemaju utjecaj na značenje teksta [↑](#footnote-ref-5)
6. SQL - Structured Query Language – strukturirani jezik za upite, koristi se u bazama podataka [↑](#footnote-ref-6)