UNIVERZITET U BANJOJ LUCI
ELEKTROTEHNIČKI FAKULTET

Ognjen Moravac

MAŠINSKO UČENJE I POSLOVNO ODLUČIVANJE

Diplomski rad

Banja Luka, mart 2021.

Tema: MAŠINSKO UČENJE I POSLOVNO ODLUČIVANJE

Komisija: doc. dr Miloš Ljubojević, predsjednik
prof. dr Zoran Đurić, mentor
Aleksandar Keleč, ma, član

Kandidat:
Ognjen Moravac

UNIVERZITET U BANJOJ LUCI

ELEKTROTEHNIČKI FAKULTET

KATEDRA ZA RAČUNARSTVO I INFORMATIKU

Predmet: INTERNET PROGRAMIRANJE

Tema: MAŠINSKO UČENJE I POSLOVNO ODLUČIVANJE

Zadatak: Uvod. Poslovno odlučivanje. Tradicionalni alati za poslovno odlučivanje. Mašinsko učenje. Algoritmi mašinskog učenja. Mogućnosti upotrebe mašinskog učenja u svrhu poslovnog odlučivanja. Realizacija jednostavnog sistema poslovnog odlučivanja na bazi mašinskog učenja.

Mentor: Zoran Đurić

Kandidat: Ognjen Moravac (1113/16)

Banja Luka, mart 2021.

SADRŽAJ

Uz rad je priložen CD.

1. UVOD

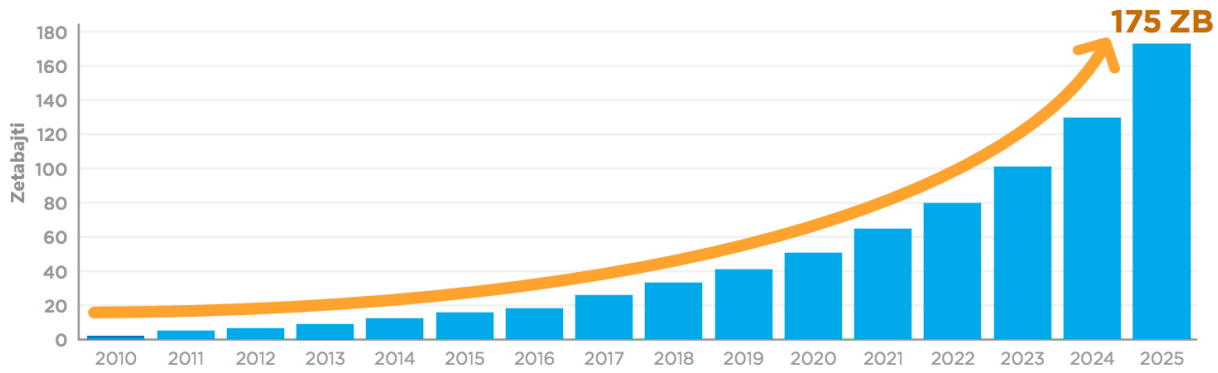
Podatak je činjenica prikazana u formalizovanom obliku. Kroz istoriju ljudi su koristili razne formate i simbole za zapisivanje podataka o raznim pojavama, dešavanjima i svakodnevnim aktivnostima. Potreba za podacima je jasna: bilo je nemoguće pamtiti sve relevantne činjenice i postojala je potreba za dijeljenjem takvih činjenica na brz i efikasan način. Pored toga, podaci su omogućavali dobar uvid i efikasnu analizu prošlih i trenutnih dešavanja. Akumulacijom podataka i njihovom analizom uočeno je da se mnogi procesi ponavljaju i da postoje određeni šabloni na osnovu kojih je moguće predvidjeti tok budućih dešavanja [1].

Kao i u mnogim drugim oblastima, upotreba podataka u poslovnim organizacijama je velika, jer poslovne organizacije bilježe sve relevantne činjenice o poslovanju i na osnovu toga planiraju svoje buduće korake. U skorijoj istoriji, prije aktivne upotrebe računarskih sistema u organizacijama, svi interni i eksterni tokovi u organizaciji, od nabavki, narudžbi, vođenja evidencije o transakcijama do vođenja evidencije o skladištu, bili su u papirnoj formi. Posljedica toga bilo je sporo odvijanje poslovnih procesa, velika potreba za radnom snagom koja je obrađivala podatke i slab uvid u generalno poslovanje i stanje organizacije [2].

Razvoj računarskih sistema i postepeno uvođenje takvih sistema u organizacije dovodi do digitalizacije podataka. To znači da se podaci nalaze u okviru baze podataka informacionog sistema na jednom ili više elektronskih medija. Ovaj način skladištenja i obrade podataka je, pored toga što je automatizovao i ubrzao razne procese unutar organizacije, stvorio uslove za izradu kvalitetnijih izvještaja koji omogućavaju rukovodiocima da prate stanja u organizaciji i na osnovu njih donose odgovarajuće odluke. Naglim razvojem interneta, stvorili su se uslovi za elektronsku komunikaciju između poslovnih organizacija, njihovih klijenata i distributera, što za posljedicu ima dodatno ubrzanje poslovnih procesa, kao i povećanje tržišta, a samim tim i povećanje obima podataka. Pored toga, i podaci koji nisu generisani u okviru kompanije postaju dostupni preko raznih servisa i veb sajtova. Dodatno, danas se sve više primjenjuje tehnologija koja omogućava različitim uređajima da budu upravljivi i dostupni preko interneta, koja se naziva *internet of things*.

Sve ovo dovodi do eksponencijalnog rasta količine podataka, a ogromna količina digitalnih podataka danas se popularno naziva *big data*. Jedno istraživanje procjenjuje da će totalna količina digitalnih podataka do 2025. godine dostići veličinu od 175 zetabajta [3].

Na slici 1.1 prikazan je eksponencijalan rast digitalnih podataka.



Slika 1.1 – Eksponencijalan rast digitalnih podataka [3]

Na osnovu toga, ogroman potencijal „leži“ u nagomilanim podacima. Da bi se ti podaci iskoristili na optimalan način, tj. da bi se izdvojila znanja i korisne informacije koje bi otkrile „skrivene“ veze među podacima i koje bi omogućile predviđanje bliže budućnosti, uvodi se tehnologija mašinskog učenja (eng. *machine learning* - *ML*). Glavni razlozi zbog kojih se ova tehnologija koristi su sljedeći [4]: težnja ka minimizaciji ljudskih napora prilikom analize podataka (ovo se manifestuje automatizovanim kreiranjem modela mašinskog učenja), dostupnost velikog obima i raznovrsnosti podata (*big data*), jeftina i snažna procesna moć, te pristupačni memorijski kapaciteti. Informacija o budućim dešavanjima predstavlja ogromnu stratešku prednost, koja opravdava investicije i zainteresovanost mnogih organizacija u primjeni i integraciji sistema mašinskog učenja u svojim okruženjima.

U nastavku rada izložen je princip i mogućnosti primjene sistema mašinskog učenja u cilju automatizacije poslovnih procesa i sticanja strateške prednosti na osnovu koje se donose optimalne odluke za dobrobit organizacije, a i šireg okruženja. U poglavlju „Poslovno odlučivanje“ opisana je podjela i princip upravljanja poslovnom organizacijom. Prikazana je tipična arhitektura informacionih sistema koji se koriste u okviru takvih organizacija, te primjena tradicionalnih alata iz domena poslovne inteligencije (eng. *business inteligence*) koji se koriste u cilju što bolje analize podataka i praktične pomoći u procesu donošenja odluka. Poglavlje „Mašinsko učenje“ definiše sam pojam, paradigmu, način funkcionisanja i podjelu mašinskog učenja. „Upotreba mašinskog učenja u oblasti poslovnog odlučivanja“ je poglavlje koje opisuje mašinsko učenje iz praktičnog ugla organizacije, prikazuje tipičnu arhitekturu sistema u okviru koje se primjenjuje mašinsko učenje i opisuje najčešće aplikacije u domenu poslovnog odlučivanja. Takođe, objašnjava razliku između tradicionalnih tehnika korištenih u poslovnoj inteligenciji i pristupu koji koristi principe mašinskog učenja. U poglavlju „Realizacija jednostavnog sistema poslovnog odlučivanja na bazi mašinskog učenja“ demonstriran je praktičan primjer upotrebe mašinskog učenja u cilju procjene vrijednosti nekretnina. Ovakav sistem ilustruje princip mašinskog učenja i konkretizuje znanje izloženo u ostalim poglavljima. Na kraju, u zaključku, dat je osvrt na primjenu mašinskog učenja u poslovnom okruženju i iznesene su prognoze za buduće trendove.

2. POSLOVNO ODLUČIVANJE

Cilj svake profitne organizacije jeste da ostvari što veću dobit. Dobit predstavlja razliku između prihoda i rashoda. Prihod predstavlja novac koji organizacija dobija prodavanjem proizvoda ili usluga koje nudi. Rashod predstavlja troškove organizacije kao što su plate, marketing, nabavka sirovina, plaćanje poslovnog prostora, itd. Iako i neprofitne organizacije teže ka efikasnosti i poboljšavanju poslovnih procesa, profitne organizacije su primarno orijentisane ka ostvarivanju dobiti, što ih čini „agilnijim“ i „osjetljivijim“, te više ulažu u svaki vid optimizacije i mogućeg napretka.

Iako primarna tema ovog rada nije menadžment i preduzetništvo, potrebno je da se izlože osnovni pojmovi funkcionisanja organizacije, kako bi se stekla šira slika i stvorila podloga za shvatanje potrebe za sistemima koji koriste tehnike mašinskog učenja.

2.1. STRUKTURA I ORGANIZACIJA PREDUZEĆA

Srž svakog preduzeća jeste proizvod ili usluga koju ono nudi. Da bi se čitav proces od proizvodnje do prodaje odvijao na dobar način, preduzeću je potreban neko ko dizajnira i inovira, neko ko proizvodi, neko ko informiše klijente o proizvodu/usluzi, neko ko vodi računa o finansijama, neko ko nadgleda poslovanje, itd. Isto tako, kao i u većini zajednica, potrebna je hijerarhija koja će uvesti red u taj proces.

2.1.1. FUNKCIONALNE OBLASTI U OKVIRU ORGANIZACIJE

Da bi poslovni procesi funkcionisali što *efektnije*[1] i *efikasnije*[2], organizacije su podijeljene na razne sektore/oblasti koje se bave različitim aktivnostima i aspektima u organizaciji. Na taj način organizacija zapošljava ljude različitih profila i talenata koji doprinose u oblasti za koju su specijalizovani.

Organizacije, u generalnom slučaju, sadrže sljedeće funkcionalne oblasti [5]:

- proizvodnja,
- prodaja i marketing,
- finansije i računovodstvo i
- ljudski resursi.

[1] Efektivnost označava ostvarenje ciljeva postavljenih poslovnim procesom.
[2] Efikasnost označava ostvarenje cilja uz minimalan trošak resursa.

Proizvodnja predstavlja centralni dio nekog preduzeća. U okviru nje se kreiraju proizvodi i usluge od kojih organizacija ostvaruje prihode. Prodaja i marketing je oblast u okviru koje se vrši informisanje klijenata o proizvodima/uslugama, segmentacija klijenata, određivanje cijene i obavljanje transakcija vezanih za prodaju. Finansije i računovodstvo se bave praćenjem svih internih i eksternih transakcija, kao i planiranjem budžeta i odlučivanjem o načinu pribavljanja potrebnog kapitala. Ljudski resursi predstavljaju oblast koja se bavi praćenjem zaposlenih, njihovih afiniteta, produktivnosti, stimulacijom i odgovarajućom kompenzacijom, najčešće u vidu novčanih bonusa ili povišene plate.

Navedene oblasti predstavljaju generalizavanu podjelu preduzeća i nisu sastavni dio svakog preduzeća. Zavisno od veličine preduzeća, kao i djelatnosti kojom se bavi, određene oblasti mogu biti *outsource*[3]-ovane ili uopšte nepostojeće, a sa druge strane preduzeće može sadržati i neke druge oblasti koje nisu navedene, poput oblasti istraživanja i razvoja, IT oblasti, itd.

2.1.2. RUKOVOĐENJE ORGANIZACIJOM

Pojedinci koji planiraju, organizuju, vode ili jednom rječju donose odluke u okviru organizacije nazivaju se rukovodioci, odnosno menadžeri. U generalnom slučaju, organizacije se sastoje od tri nivoa rukovođenja (menadžmenta) [5]: visokog nivoa rukovođenja, srednjeg nivoa rukovođenja i niskog nivoa rukovođenja. Rukovodioci visokog nivoa zaduženi su za strateške odluke i dugoročne planove. Oni odgovaraju na pitanja kao što su: „Koje je ciljno tržište?“, „Koja je tržišna niša[4]?“, „Da li je potrebno uvesti novi sektor u okviru organizacije?“, itd. Planiranje implementacije u skladu sa strateškim odlukama, odnosno donošenje taktičkih odluka koje sadrže više specifičnih detalja vezanih za funkcionalnu oblast organizacije, odgovornost je rukovodilaca srednjeg nivoa. Rukovodioci niskog nivoa ili operativni rukovodioci nadgledaju dnevne aktivnosti i donose rutinske odluke, odnosno odluke koje se odnose na poznate probleme sa unaprijed utvrđenim načinom rješavanja.

[3] Outsource je termin koji označava izvršavanje određenih poslovnih procesa izvan okvira organizacije, često u sklopu neke druge organizacije.

[4] Tržišna niša predstavlja fokusirani, manji dio nekog tržišta, odnosno podgrupu nekog tržišta.

Na slici 2.1 prikazana je vertikalna (po hijerarhijskom nivou) i horizontalna (po funkcionalnim oblastima) organizacija menadžmenta u preduzeću.

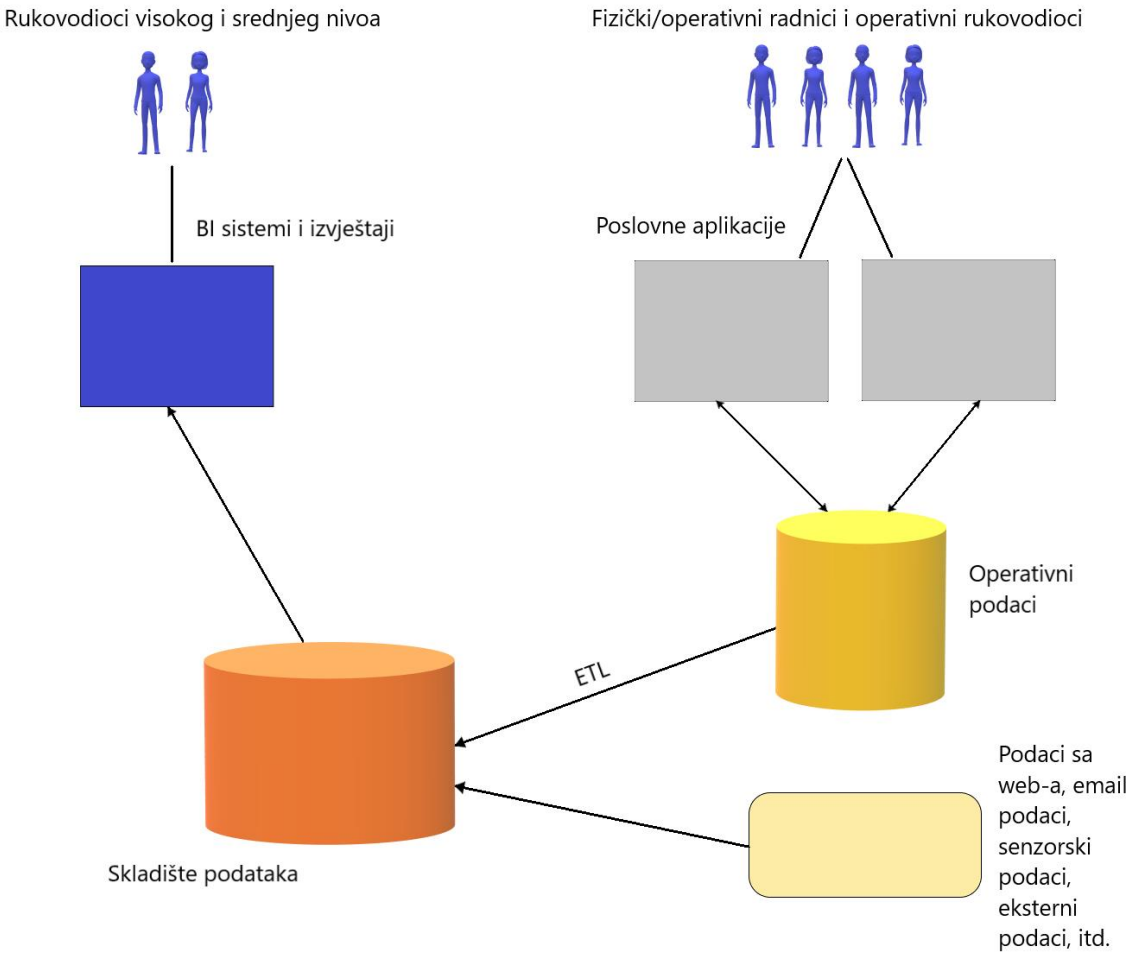
Slika 2.1 – Horizontalna i vertikalna organizacija rukovođenja [5]

2.2 INFORMACIONI SISTEMI

Da bi rukovodioci bili u stanju da donesu dobre odluke, potrebni su znanje i prave informacije. Znanje se dobrim dijelom stiče iz iskustva, ali može i da se „izvuče“ iz podataka, ako se oni pažljivo analiziraju. Metode za izdvajanje takvih znanja iz podataka koriste se u sklopu discipline, koja se naziva nauka o podacima (eng. *data science* – *DS*) i koja je opisana u narednim poglavljima. Za precizne, pravovremene i dostupne informacije o stanju u organizaciji, kao i potencijalna znanja koja se nalaze u podacima, potrebna je infrastruktura koja će omogućiti efikasan unos, skladištenje i pribavljanje podataka. Takvu infrastrukturu danas predstavljaju moderni informacioni sistemi.

Današnji informacioni sistemi sastoje se iz grupe poslovnih aplikacija (eng. *enterprise applications*), koje se koriste u različitim funkcionalnim oblastima i koje su povezane na isti sistem za upravljanje bazama podataka (eng. *database management system* - DBMS). To znači da su podaci iz različitih domena integrisani na jednom mjestu, što, osim toga što smanjuje redundantnost i nekonzistentnost, olakšava i samu analizu podataka. Pored baze podataka koja sadrži aktivne podatke, informacioni sistemi često upotrebljavaju i skladište podataka (eng. *data warehouse*) koje sadrži i aktivne (operativne) i istorijske podatke, kao i podatke iz eksternih izvora.

Na slici 2.2 prikazana je uproštena arhitektura informacionog sistema. Prikazani su sistemi i aplikacije koji se koriste, način na koji su međusobno povezani i grupe korisnika koji ih koriste.



Slika 2.2 – Tipična arhitektura informacionog sistema na visokom nivou apstrakcije

2.3 POSLOVNA INTELIGENCIJA

Poslovna inteligencija (eng. *business intelligence* - *BI*) predstavlja zasebnu oblast, često u okviru IT oblasti, koja se bavi praćenjem trendova i izvlačenjem zaključaka iz podataka, odnosno, obrađivanjem podataka u cilju dobijanja relevantnih informacija. Takve informacije se dalje koriste u rukovođenju prilikom donošenja strateških i taktičkih odluka. Moderni *BI* sistemi koriste kombinovane tehnike i često primjenjuju i *DS* disciplinu u svojim okruženjima. U tekstu ispod opisana je tradicionalna *BI*, koja se primjenjivala prije integracije sa *DS* disciplinom.

BI se bavi „poznatim nepoznanicama“, za razliku od mašinskog učenja i *DS*-a koji se bave „nepoznatim nepoznanicama“, što znači da u sklopu *BI* postoji skup definisanih poslovnih pravila i formula koji se primjenjuju na podatke. Dakle, *BI* koristi poznate metode da dobije nepoznate odgovore. Zbog toga, *BI* je ograničena znanjem na osnovu koga su definisana poslovna pravila. *BI* daje sveobuhvatan pogled na stanje organizacije i prikazuje ključne indikatore performansi (eng. *key performance indicator* - *KPI*) u vidu *dashboard*[5]*-a*. Drugim riječima, domen *BI* je najčešće deskriptivna analitika [6]. Deskriptivna analitika se bavi interpretacijom trenutnog stanja i istorijskih dešavanja.

Na slici 2.3 prikazane su vrste analitike i pitanja na koje daju odgovore.

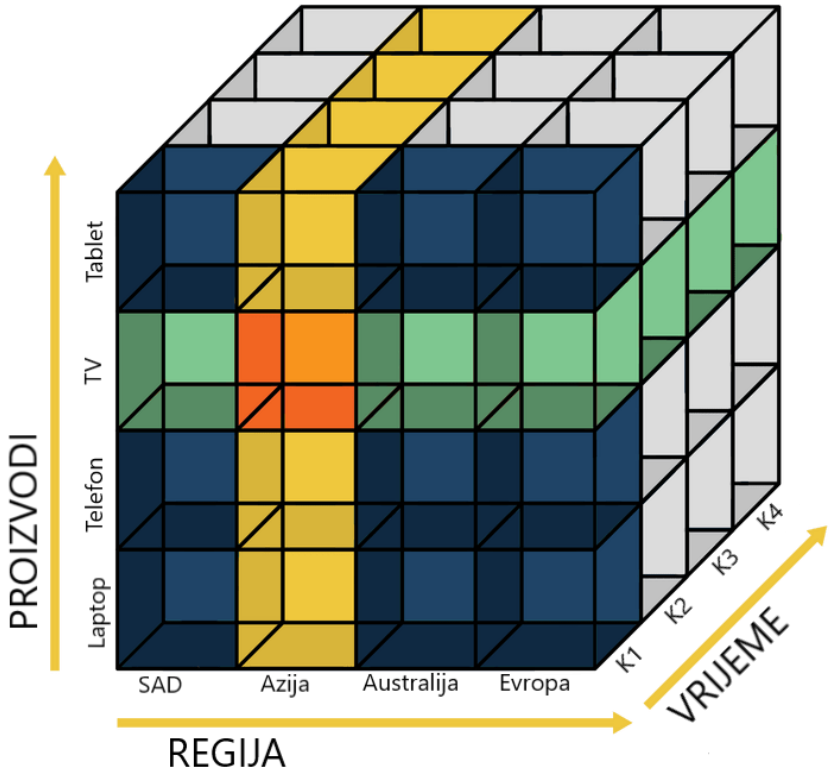
Slika 2.3 – Vrste analitike i njihov značaj [7]

[5] Dashboard predstavlja kontrolnu tablu na kojoj su izložene ključne informacije vezane za trenutno stanje.

Alati koje *BI* koristi su najčešće vrlo kompleksni i sveobuhvatni izvještaji (eng. *reports*) kreirani *SQL* upitima nad relacionom bazom podataka. Pored toga, dosta česta je i upotreba *OLAP*[6] (eng. *OnLine Analytical Processing*) alata koji omogućava multidimenzionalnu analizu podataka. Podaci koje ovi alati koriste su strukturirani podaci[7], što predstavlja ograničenje budući da se dosta podataka koji se prikupljaju, pored osnovnih transakcionih podataka iz baze (*web*, *email*, senzori, itd.), nalaze u nestrukturiranom ili polu-strukturiranom formatu.

OLAP koristi multidimenzionalnu bazu podataka koja je kreirana iz više relacionih baza. Podaci unutar ovakve baze organizovani su u specifičnom formatu u obliku kvadra. Ovo omogućava jednostavnu analizu istih podataka iz ugla različitih dimenzija.

Na slici 2.4 predstavljen je primjer organizacije podataka u *OLAP* sistemu. Glavna tema je broj prodanih stavki, a dimenzije su proizvodi, regija i vrijeme. Tako na primjer, broj prodanih TV uređaja u Aziji u prvom kvartalu predstavlja sadržaj ćelije označene narandžastom bojom.



Slika 2.4 – Primjer organizacije podataka u OLAP sistemu [8]

[6] URL zvaničnog sajta na kojem se nalazi projekat: https://olap.com.

[7] Strukturirani podaci su podaci koji imaju definisanu strukturu koja je definisana u šemi relacione baze podataka. Pored strukturiranih podataka, postoje i nestrukturirani, npr. slike ili video zapisi, i polustrukturirani, npr. tekstualni dokumenti.

Zbog svoje predefinisanosti i unaprijed poznatih pravila, *BI* je dosta kruta i ograničena. Usljed toga, ona nije u mogućnosti da dâ odgovore na sva relevantna pitanja čiji odgovori „leže“ u podacima. Kao odgovor na to, javlja se *DS* disciplina koje prevazilazi ta ograničenja. To ne znači da tehnike tradicionalne *BI* postaju suvišne. One su itekako potrebne, ali za širu sliku i kvalitetna predviđanja potrebna su drugačija rješenja.

3. MAŠINSKO UČENJE

Tradicionalni razvoj softvera bazira se na eksplicitnom znanju ljudi, na osnovu koga se ručno definišu pravila, u vidu pisanja koda. Drugačija paradigma u programiranju koja pronalazi svoju višestruku primjenu, iako postoji već duže vremena, postaje sve popularnija zahvaljujući povećanom obimu podataka, te jeftinijim memorijskim kapacitetima i procesnom moći. Riječ je o mašinskom učenju. Ova paradigma bazirana je na podacima, na osnovu kojih mašina[8] sama uči, koristeći određene algoritme, kako da modeluje problem koji se želi riješiti u vidu šablona koji se pojavljuju u podacima, bez eksplicitnog programiranja. Dakle, laički rečeno, mašina stiče iskustvo obrađujući velike količine istorijskih podataka i na osnovu njega oblikuje svoje ponašanje. Mašinsko učenje je podcjelina jedne šire oblasti zvane vještačka inteligencija (eng. *artificial intelligence - AI*).

Tradicionalni razvoj softvera i mašinsko učenje ne predstavljaju jedno drugom konkurenciju. Određeni domeni u kojima se primjenjivalo tradicionalno programiranje postepeno prelaze na upotrebu mašinskog učenja, dok oblasti poput razvoja infrastrukture, toka podataka, poslovne logike i grafičkog interfejsa ostaju u domenu tradicionalnog razvoja softvera.

3.1. OBLASTI PRIMJENE

Mašinsko učenje pronalazi svoju primjenu u različitim oblastima. Neke od tih oblasti su sljedeće:

- računarski vid,
- prepoznavanje govora,
- obrada prirodnog jezika,
- detekcija anomalija i
- prediktivno modelovanje.

Računarski vid (eng. *computer vision*) predstavlja oblast u računarstvu u kojoj se analiziraju vizuelni podaci u cilju detektovanja i identifikovanja objekata, prepoznavanja lica, prepoznavanja rukopisa, i sl. Prepoznavanje govora (eng. *speech recognition*) je oblast u okviru koje računar analizira zvuk i prepoznaje ljudski govor. Obrada prirodnog jezika (eng. *natural language processing*) je oblast koja se bavi prepoznavanjem, segmentacijom i obradom teksta napisanog prirodnim ljudskim jezikom. Detekcija anomalija (eng. *anomaly detection*) je oblast u okviru koje se detektuju anomalije, odnosno izuzeci u odnosu na „normalne“ podatke. Prediktivno modelovanje (eng. *predictive modelling*), odnosno prediktivna analitika, je oblast koja se bavi predviđanjem određenih budućih vrijednosti na osnovu istorijskih podataka.

[8] Mašina u ovom kontekstu može da označava računar, ugrađeni računarski sistem, softver, i sl.

U okviru poslovnog odlučivanja, mašinsko učenje se koristi u sklopu prediktivnog modelovanja, obrade prirodnog jezika, detekcije anomalija, kao i jednog dijela deskriptivne analitike.

3.2. TIPOVI MAŠINSKOG UČENJA

U okviru oblasti mašinskog učenja postoji mnogo različitih tipova problema i načina na koji se ti problemi rješavaju. Na osnovu toga, postoji mnogo različitih algoritama mašinskog učenja. Algoritam je skup definisanih koraka ka rješavanju nekog problema i predstavlja centralni dio mašinskog učenja. Na osnovu algoritma, mašina zna kako da nauči odgovarajuće šablone koji se javljaju u podacima i na osnovu njih modeluje svoje ponašanje. Svi algoritami pripadaju jednom od tri generalna tipa mašinskog učenja: nadgledano (eng. *supervised*), nenadgledano (eng. *unsupervised*) i pojačano (eng. *reinforcement*) učenje [9].

3.2.1. NADGLEDANO UČENJE

Nadgledano učenje predstavlja najpopularniji tip učenja [10]. Ovaj tip učenja funkcioniše tako što model „uči“ funkciju mapiranja između ulaznih varijabli, koji se nazivaju atributi (eng. *features*) i izlazne varijable, koja se naziva labela, odnosno ciljna varijabla (eng. *target*). Atribut je drugi naziv za nezavisnu varijablu, a labela predstavlja zavisnu varijablu koju je potrebno predvidjeti. Učenje funkcije mapiranja naziva se treniranje, a skup podataka nad kojim algoritam trenira jeste trening skup. Svaki pojedinačan podatak iz trening skupa sastoji se i od atributa i od labele, što znači da u procesu treniranja algoritam zna kako da modifikuje model – zbog toga se i naziva nadgledano učenje. Budući podaci neće sadržati labelu, ona predstavlja nepoznatu koju će model biti u stanju da predvidi na osnovu atributa. Matematički definisano, neka X predstavlja uređen niz (vektor) atributa, y labelu, a f funkciju mapiranja koja predstavlja istreniran model nad podacima posredstvom određenog algoritma. Tada izraz $y = f(X)$ definiše predviđanje.

Tamo gdje je poznata funkcionalna zavisnost između atributa i labele nema potrebe za mašinskim učenjem, jer se labela dobija jednostavnim uvrštavanjem atributa u formulu i njenim izračunavanjem. Ovaj tip učenja se primjenjuje tamo gdje je nepoznata funkcionalna zavisnost između atributa i labele, tj. tamo gdje ne postoji tačna formula funkcije mapiranja. Često i ne postoji 100% precizna funkcionalna zavisnost, jer su mnogi problemi koji se rješavaju društvene prirode, što znači da uvijek postoji određeni procenat slučajnosti izazvan ljudskim ponašanjem, ali postoje približne aproksimacije.

Na slici 3.1 prikazan je minijaturni skup podataka o pacijentima i srčanim oboljenjima. „Bol u grudima“, „Blokirane arterije“ i „Starost“ predstavljaju atribute, a „Srčano oboljenje“ predstavlja labelu, odnosno ciljnu varijablu čija vrijednost se želi predvidjeti.

Bol u grudima	Blokirane arterije	Starost	Srčano oboljenje
Ne	Da	66	Da
Ne	Ne	62	Ne
Ne	Da	38	Ne
Da	Ne	73	Da
Da	Ne	78	Da

Slika 3.1 – Primjer skupa podataka

Postoje dvije klase problema koje se rješavaju algoritmima nadgledanog učenja, to su regresija i klasifikacija.

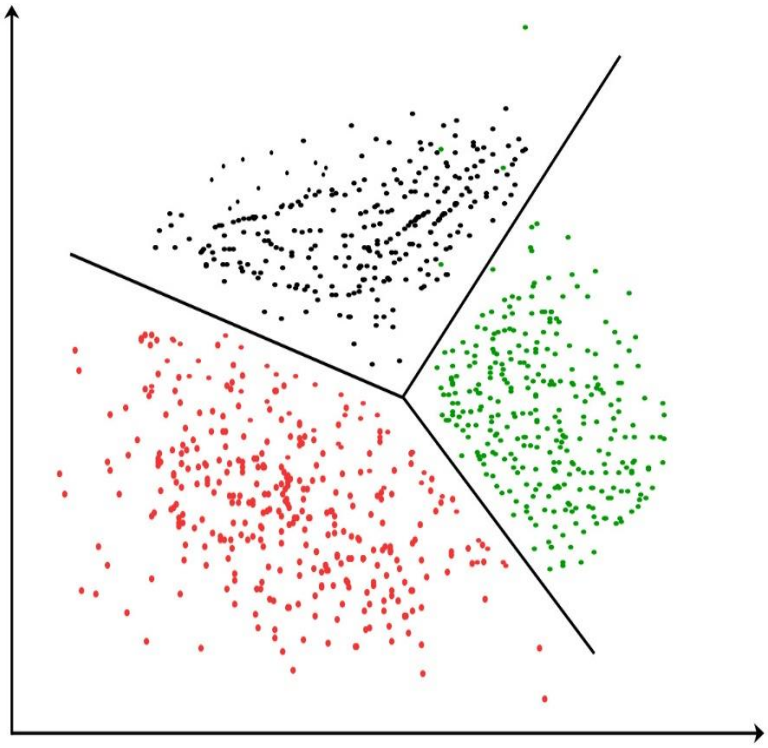
Regresija predstavlja klasu problema u kojima labela predstavlja kontinualnu vrijednost. To znači da labela može biti bilo koji realan broj. Primjer problema koji se rješava regresijom je predviđanje prodaje određenog proizvoda u određenom vremenskom trenutku. Algoritmi koji se koriste za regresiju su: linearna regresija (eng. *Linear Regression*), polinomska regresija (eng. *Polynomial Regression*), stablo odlučivanja (eng. *Decision Tree*), slučajna šuma (eng. *Random Forest*), algoritmi gradijentnog pojačavanja (eng. *Gradient Boosting*), neuronska mreža (eng. *Neural Network)*.

Klasifikacija predstavlja klasu problema u kojima labela predstavlja odgovarajuću kategoriju iz skupa već definisanih kategorija. Primjer problema koji se rješava klasifikacijom je klasifikacija klijenta banke na visokorizičnog i niskorizičnog prilikom odobrenja kredita. Algoritmi koji se koriste za klasifikaciju su: logistička regresija (eng. *Logistic Regression*), stablo odlučivanja, slučajna šuma, metod potpornih vektora (eng. *Support Vector Machines*), neuronska mreža.

3.2.2. NENADGLEDANO UČENJE

Za razliku od nadgledanog učenja, u ovoj vrsti učenja nema nikakvog oblika „mentorstva“ u vidu labela koje su unaprijed poznate kod testnih podataka. To znači da ne postoji ni treniranje. Ova vrsta učenja primjenjuje se kada je potrebno da se podaci na određeni način okaratkerišu i potencijalno podijele na određene grupe. To može biti u vidu otkrivanja grupa sličnih podataka, odnosno klasterizacija (eng. *clustering*) ili u vidu učenja asocijativnih pravila (eng. association rule learning) između podataka.

Na slici 3.2 vizuelno je prikazana klasterizacija.



Slika 3.2 – Vizuelni primjer klasterizacije [11]

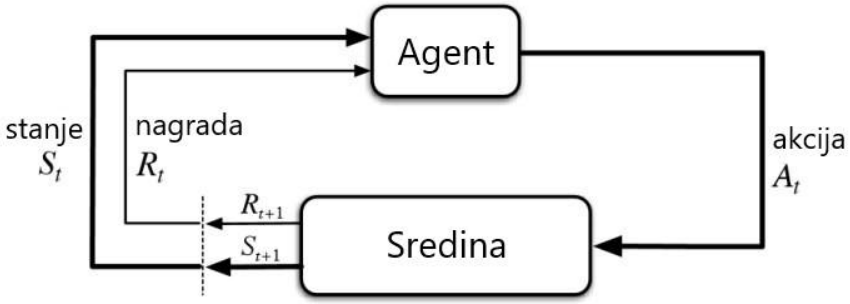
Klasterizacija je metoda koja razdvaja skup podataka na grupe u kojima se nalaze slični podaci, a između kojih su prilično različiti podaci. Algoritmi koji se koriste za klasterizaciju su: k sredina (eng. *K-means*), mješavina normalnih raspodjela (eng. *Mixture of Gaussians*), hijerarhijska klasterizacija (eng. *Hierarchical Clustering*).

Učenje asocijativnih pravila predstavlja metodu u okviru koje se podaci iz nekog skupa dovode u asocijativnu vezu, ali ne na osnovu sličnosti kao kod klasterizacije. Klasičan primjer koji ilustruje ovu metodu je analiza tržišne korpe, tj. analiza veza između artikala koji se najčešće zajedno kupuju. Algoritmi koji se koriste za učenje asocijativnih pravila su: Apriori algoritam, Eclat algoritam, itd.

3.2.3. POJAČANO UČENJE

Pojačano učenje je kategorija mašinskog učenja koja se bavi obukom inteligentih agenata u realnom vremenu. Inteligentni agent je zaseban entitet koji interaguje sa sredinom u kojoj se nalazi (može biti realna ili simulirana sredina), na primjer, automobil koji posjeduje autonomni režim vožnje predstavlja inteligentnog agenta. Za razliku od prethodna dva tipa učenja čije se učenje zasniva na treniranju modela nad podacima koji sadrže labelu, odnosno pronalasku i grupisanju podataka iz nekog skupa podataka, pojačano učenje zasniva se na učenju kroz povratnu informaciju za svaku akciju koju agent izvrši prilikom interakcije sa sredinom, u vidu nagrade i kazne.

Na slici 3.3 prikazana je šema interakcije inteligentnog agenta i sredine u kojoj se nalazi.



Slika 3.3 – Interakcija agenta i sredine [12]

Postoje dva generalna pristupa u okviru pojačanog učenja, i to pojačano učenje bazirano na modelu i pojačano učenje bez modela. U svakoj od ove dvije grupe postoji više algoritama koji se koriste za obučavanje agenata. Algoritmi koji se koriste u okviru pojačanog učenja su: gradijent politike (eng. *Policy Gradient*), *C51*, modeli svijeta (eng. *World Models*), *AlphaZero*.

3.3. PROCES UČENJA

Pošto većina praktične primjene mašinskog učenja pripada kategoriji nadgledanog učenja, ovdje je izložen generalni proces učenja koji važi za sve algoritme koji pripadaju ovoj kategoriji.

3.3.1. FAZE MAŠINSKOG UČENJA

Proces učenja, odnosno treniranje modela do faze u kojoj je model spreman za *produkciju*[9] sastoji se od više faza. Te faze mogu se grupisati u četiri osnovne [13]:

1. priprema podataka,
2. treniranje modela,
3. evaluacija modela i
4. postavljanje modela na produkciono okruženje

Priprema podataka je faza u okviru koje se vrši manipulacija nad podacima. Vrši se prikupljanje podataka, njihov pregled i analiza i čišćenje podataka. Prikupljanje podataka znači da svi relevantni podaci treba da se skupe na jedno mjesto i da budu uniformni, odnosno u istom formatu. Osnovni pregled podataka i analiza je faza u kojoj treba da se ispita i upozna struktura podataka, treba da se provjeri da li postoje abnormalne vrijednosti, da li nedostaju određene vrijednosti i treba da se utvrdi distribucija vrijednosti. Nakon toga dolazi do izbacivanja odgovarajućih podataka (podaci koji sadrže abnormalne vrijednosti, dupli podaci, podaci koji su u manjini i sl.), ili do popunjavanja podataka koji nedostaju u okviru faze čišćenja.

Treniranje modela je faza u okviru koje se određuje cilj učenja, vrši selekcija atributa i inženjering atributa, vrši podjela podataka na trening, validacioni i testni skup, bira algoritam, podešavaju hiperparametri[10] algoritma i vrši samo treniranje. Cilj učenja predstavlja labela, odnosno vrijednost koja se želi predvidjeti. Često podaci sadrže stotine različitih atributa od kojih su mnogi redundanti, a neki čak loše opisuju podatak, odnosno nisu relevantni. Imajući pored toga na umu i brzinu treniranja, koja zavisi od algoritma, hiperparametara, obima podataka i broja atributa, poželjno je koristiti samo one atribute koji su relevantni. Izbacivanje nepotrebnih atributa naziva se selekcija atributa (eng. *feature selection*). Pored toga, često postoje atributi koji grupno daju određenu relevantnu informaciju o podatku, ali pojedinačno nemaju mnogo smisla. Zbog toga se vrši kreiranje novih atributa na osnovu više pojedinačnih, koji bolje opisuju podatak. Taj proces se naziva inženjering atributa (eng. *feature engineering*). Da bi se stekla slika o rezultatima koje algoritam postiže, potrebno je skup podataka podijeliti na trening, validacioni i testni skup. Trening skup predstavlja skup podataka nad kojim se model trenira, dok testni skup predstavlja skup podataka na osnovu koga se evaluiraju rezultati predviđanja modela. Validacioni skup predstavlja skup podataka na osnovu kojeg algoritam „zna“ kada treba da zaustavi treniranje modela, da se performanse modela ne bi pogoršale. Više riječi o tome se nalazi u sekciji

[9] Produkcija predstavlja aktivnu upotrebu aplikacije u realnom okruženju.

[10] Hiperparametri predstavljaju određene parametre nekog algoritma čijim se podešavanjem u određenoj mjeri mijenja rad algoritma.

„Underfitting i overfitting“. Većina podataka se uzima za trening skup, često i do 90%, dok se ostali podaci koriste kao validacioni i testni podaci. Izbor algoritma je faza u okviru koje se sagledava struktura podataka, cilj učenja, brzina učenja i preciznost. U zavisnosti od tih faktora i postavljenih ciljeva bira se algoritam učenja. Mnogi algoritmi imaju mogućnost dodatne konfiguracije u vidu podešavanja određenih hiperparametara koji su više-manje specifični za svaki algoritam. Tako npr. neki algoritmi omogućavaju podešavanje broja iteracija, brzine učenja, rano zaustavljanje i sl. Faza treniranja predstavlja samo učenje modela koristeći specifikovani algoritam i trening podatke. Ova faza je potpuno automatizovana, tj. mašina vrši sva izračunavanja i obradu sve dok ne postigne određenu preciznost, ne prođe kroz sve specifikovane iteracije, ili, ukoliko nakon određenog broja iteracija ne dođe do poboljšanja rezultata. Preciznost je detaljnije obrađena u nastavku ove sekcije, ali bitno je naznačiti da ona opisuje koliko dobro model predviđa target varijablu. Iteracija predstavlja jedan ciklus izvršavanja u okviru algoritma u kome se svi, ili dio trening podataka, obrađuju u cilju poboljšanja preciznosti.

Evaluacija modela je faza u kojoj se definišu različite metrike, evaluiraju modeli i porede alternative. Da bi se model evaluirao, potrebno je da postoji testni skup podataka. Taj skup predstavlja podatke koji su potpuno nepoznati modelu. Na osnovu tog skupa, kreira se skup koji sadrži sve podatke bez labela i skup koji sadrži samo labele. Skup sa testnim podacima bez labela se proslijedi modelu koji izvrši predviđanje za svaki podatak iz skupa. Tada se porede sve predviđene labele sa pravim labelama. Da bi bilo jasno koliko je model precizan, odnosno koliko dobro vrši predviđanje, potrebno je da se definišu odgovarajuće metrike. Metrika predstavlja način evaluacije i kao rezultat daje konkretan broj. Postoji više metrika kao što su srednja kvadratna greška (eng. *mean squared error – MSE*), srednja apsolutna greška (eng. *mean absolute error – MAE*), koeficijent korelacije (R2), F1-mjera, itd. Sve one daju broj koji opisuje koliko je model precizan, ali ne mjere sve isti aspekt. Npr. *MAE* je metrika koja se koristi u regresiji i mjeri srednju apsolutnu grešku, što znači da se saberu sva apsolutna odstupanja predviđene vrijednosti od prave vrijednosti i podijele sa brojem testnih podataka. Sa druge strane *MSE*, koja se isto koristi u regresiji, mjeri srednje kvadratno odstupanje predviđene od prave vrijednosti, što znači da se time naglašavju velike greške, a zanemaruju sitne greške. Najčešće je potrebno koristiti kombinaciju metrika da bi rezultati koji se dobiju evaluacijom bili relevantni. Nakon što se evaluira model, često se vraća na fazu treniranja gdje se vrši drugačiji izbor atributa, algoritama, hiperparametara, sa ciljem da se pokuša poboljšati preciznost. Dakle, faze treniranja i evaluacije modela nisu sekvencijalne, odnosno one najčešće predstavljaju nelinearan proces.

Postavljanje modela na produkciono okruženje je završna faza. U okviru nje se optimalan model izabran u prethodnim fazama trenira nad 100% podataka. Nakon toga model se postavlja na odgovarajuću infrastrukturu (najčešće u okviru odgovarajućih aplikacija) i koristi za predviđanje novih podataka. U toku ove faze stalno se nadgledaju performanse modela i, ukoliko je potrebno, vrše poboljšanja.

3.3.2. UNDERFITTING I OVERFITTING

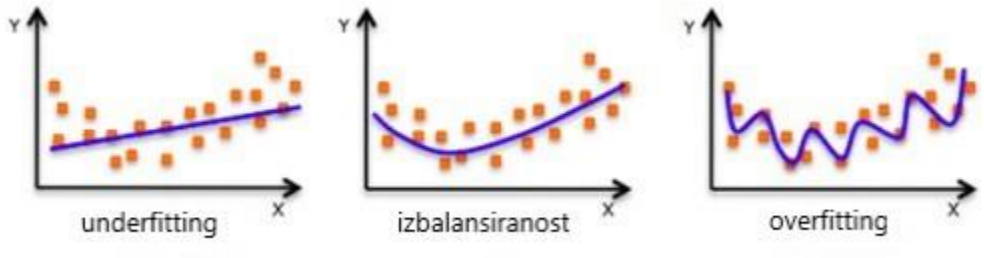
Greške modela javljaju se iz više razloga. Prvo, kao što je već rečeno, kod većine problema ne postoji 100% precizna funkcija mapiranja između ulaza i izlaza. Greške takođe nastaju iz nedovoljno iskustva, tj. nedovoljnog obima podataka nad kojim je model istreniran. Pored toga postoje još dva faktora koji utiču na greške. To su *underfitting* i *overfitting*.

Underfitting je termin koji označava preveliku generalizaciju modela, tj. nedovoljno naučenih šablona među podacima. Obično je to vrlo pojednostavljen model. Ovakav model pravi velike greške i u trening i u testnim podacima.

Overfitting je termin koji označava model koji je pored šablona između podataka inkorporirao u sebe i slučajnost, odnosno šum koji se javlja u podacima. Takav model je previše prilagođen trening podacima i zbog toga pravi veoma male greške u trening podacima, ali velike u testnim podacima, koji su modelu nepoznati.

Dakle, da bi modeli bili optimalni, tj. da bi pravili najmanju grešku, potrebno je da ne budu prejednostavni, ali i da ne budu suviše kompleksni.

Na slici 3.4 prikazana je razlika između *underfitting*-a, *overfitting*-a i izbalansiranog modela.



Slika 3.4 – Vizuelni prikaz underfitting-a i overfitting-a [14]

3.4. PRIMJERI ALGORITAMA

Postoji mnogo različitih algoritama i njihovih podjela koje su već navedene. Neki od najpoznatijih predstavnika nadgledanog učenja koji imaju podršku i za regresione i klasifikacione probleme su stablo odlučivanja, slučajna šuma, algoritmi gradijentnog pojačavanja [15], kao i neuronska mreža [10]. Njihov princip rada i generalne karakteristike izloženi su ispod.

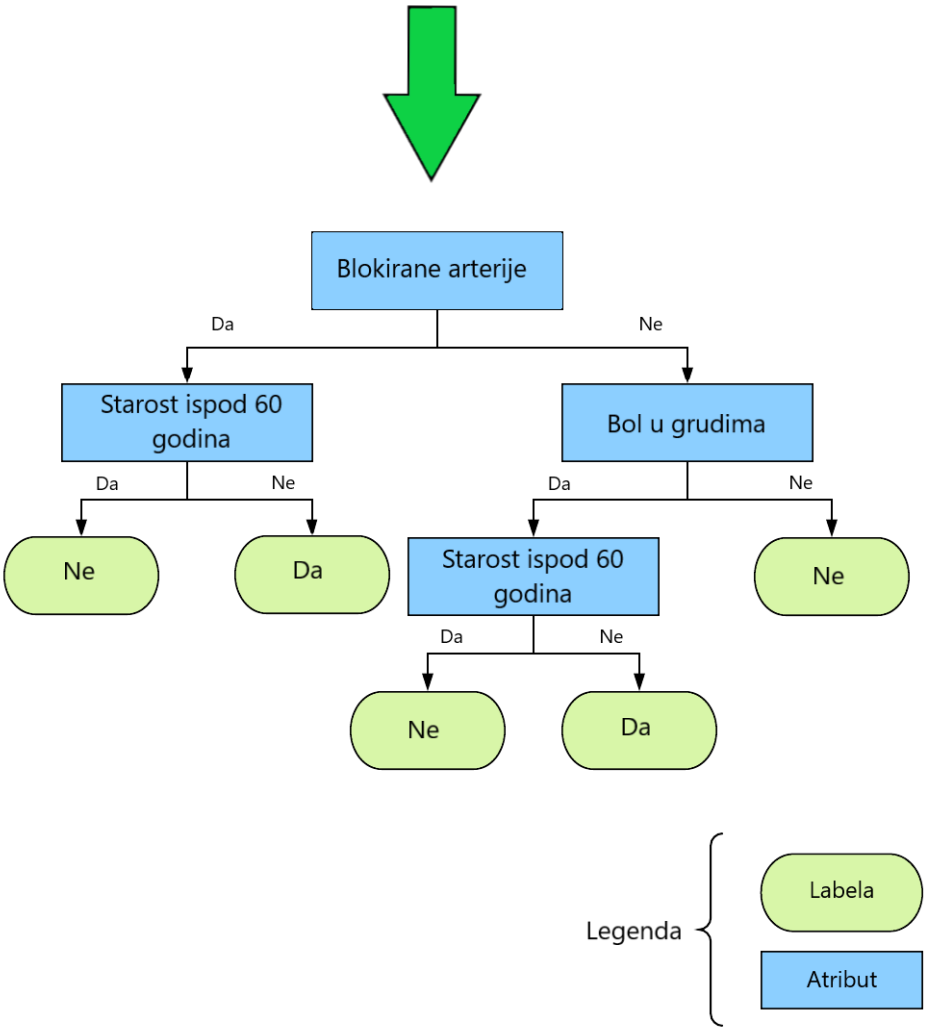
3.4.1. STABLO ODLUČIVANJA

Jedan od osnovnih algoritama mašinskog učenja jeste stablo odlučivanja. Kao što sam naziv kaže, ovaj algoritam funkcioniše kao stablo odlučivanja. Struktura stabla sastoji se od čvorova i listova. Čvor predstavlja uslov na osnovu koga se stablo dijeli na dvije grane (binarno stablo), gdje lijeva grana predstavlja potvrdan odgovor na uslov, a desna odričan. List predstavlja kraj grane i sadrži predviđenu labelu. Ukoliko se radi o regresionom problemu, listovi sadrže kontinualne vrijednosti, a ukoliko se radi o klasifikacionom problemu, listovi sadrže diskretne vrijednosti.

Stablo se konstruiše od vrha ka dnu. Na vrhu se nalazi korjeni čvor. Svaki uslov u okviru čvora vezan je za određeni atribut. Pri konstrukciji stabla, za svaki čvor algoritam bira optimalan atribut i vrijednost koja najbolje segmentira podatke, odnosno koja ima najmanju vrijednost funkcije gubitka (eng. *loss function*). Postoji više funkcija gubitka, odnosno funkcija koje izračunavaju koliko dobro određeni čvor segmentira podatke, tj. koji čvor pravi najmanju grešku u preciznosti. Jedna od najpoznatijih je *Gini Impurity [16]*. Na osnovu navedenog proizilazi činjenica da se na vrhu stabla nalaze čvorovi koji sadrže bitnije atribute.

Na slici 3.5 prikazano je jednostavno stablo odlučivanja konstruisano nad minijaturnim skupom podataka o srčanim oboljenjima prikazanim u sekciji o nadgledanom učenju. Stablo na osnovu atributa „Bol u grudima“, „Blokirane arterije“ i „Starost“ predviđa da li osoba ima srčano oboljenje.

Bol u grudima	Blokirane arterije	Starost	Srčano oboljenje
Ne	Da	66	Da
Ne	Ne	62	Ne
Ne	Da	38	Ne
Da	Ne	73	Da
Da	Ne	78	Da



Slika 3.5 – Primjer stabla odlučivanja konstruisanog nad skupom podataka [16]

Pored osnovnog principa rada, postoje određeni hiperparametri koji omogućavaju različite konfiguracije algoritma. Moguće je mijenjati funkciju gubitka, maksimalnu dubinu stabla, maksimalan i minimalan broj listova, maksimalan broj atributa koji se koriste i sl.

Iako je jedan od najjednostavnijih algoritama, stablo odlučivanja daje prilično dobre rezultate i nije računski zahtjevan.

3.4.2. SLUČAJNA ŠUMA

Slučajna šuma pripada porodici takozvanih ansambl algoritama (eng. *ensemble learning*). Ansambli predstavljaju algoritme koji se sastoje od više osnovnih algoritama, kakav je npr. stablo odlučivanja. Slučajna šuma se, konkretno, sastoji od više stabala odlučivanja. Da bi se postigla raznovrsnost među stablima, slučajna šuma koristi tehniku zvanu *bagging*. Ova tehnika sastoji se od dvije faze koje se nazivaju *bootstraping* i *aggregating*.

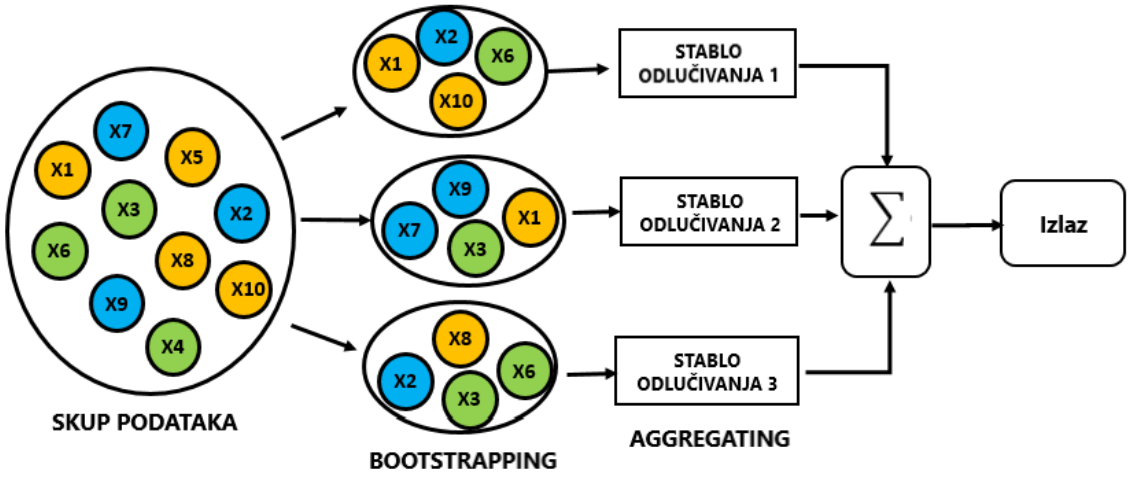
Bootstrapping – prije konstrukcije stabla kreira se novi skup podataka koji je iste veličine kao i originalni trening skup podataka i koji se sastoji od slučajno izabranih podataka iz originalnog skupa. To znači da se u novom *bootstrap* skupu mogu nalaziti i redundantni podaci, a sa druge strane, neki od podataka iz trening skupa uopšte se neće pojaviti u *bootstrap* skupu.

Aggregating – faza konstrukcije stabla nad *bootstrap* skupom podataka. Ključna stvar ove faze predstavlja slučajan izbor atributa koji se „takmiče“ za segmentaciju u okviru svakog čvora. Broj atributa koji se takmiče je manji od ukupnog broja atributa.

Nakon kreiranja odgovarajućeg broja stabala, algoritam pravi predviđanja tako što svaki od stabala odlučivanja napravi svoje predviđanje. Ako je klasifikacioni problem u pitanju, dolazi do glasanja tako što se uzme *mode*[11] vrijednost od svih predviđenih vrijednosti. Ako je regresioni problem u pitanju, uzima se aritmetička sredina svih predviđenih vrijednosti.

[11] Mode vrijednost predstavlja srednju vrijednost koja se izračuna tako što se prebroje pojedinačne vrijednosti i izabere ona vrijednost koja je imala najviše ponavljanja.

Na slici 3.6 prikazan je princip rada slučajne šume.



Slika 3.6 – Vizuelni prikaz slučajne šume [17]

Kao i većina algoritama, slučajna šuma sadrži dosta konfigurabilnih hiperparametara koji uključuju parametre iz stabla odlučivanja plus druge, kao što je broj stabala odlučivanja, broj poslova koji se izvršavaju paralelno i sl.

Dobre strane slučajne šume su bolja preciznost koja je rezultat odsustva *overfitting*-a koji je svojstven stablu odlučivanja i mogućnost paralelnog izvršavanja algoritma, jer procesi *bagging*-a mogu da se izvršavaju paralelno, tj. ni jedno stablo odlučivanja u okviru algoritma nije zavisno od nekog drugog stabla odlučivanja. Loša strana je zahtjevno izračunavanje koje može trajati prilično dugo ako se model ne trenira na distribuiranim sistemima.

3.4.3. ALGORITMI GRADIJENTNOG POJAČAVANJA

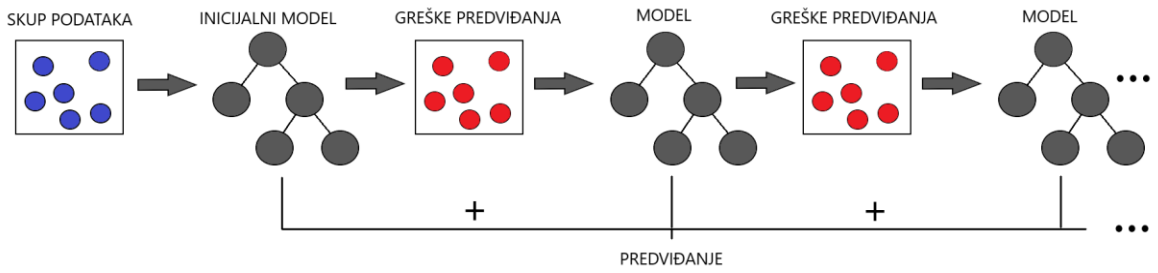
Algoritmi gradijentnog pojačavanja takođe pripadaju porodici ansambl algoritama. Metoda gradijentnog pojačavanja izgrađuje ansambl tako što dodaje model po model, iterativno, pri čemu se svaki od modela obučava tako da što bolje nadomjesti slabosti tekućeg skupa modela, odnosno da ga pojača. Optimizacioni algoritam koji se koristi da minimizuje grešku modela (funkciju gubitka) naziva se *gradient descent*. Metoda gradijentnog pojačavanja je dizajnirana tako da podrži različite konfiguracije i za regresione i klasifikacione probleme. Princip rada najčešće konfiguracije za regresioni tip problema je sljedeći [18]:

1. kao inicijalni model uzima se aritmetička sredina svih labela,
2. u svakoj iteraciji izračunaju se greške predviđanja postojećeg ansambla za svaku instancu trening skupa podataka (eng. *residuals*) i na osnovu atributa u skupu

podataka trenira se novi model (koristi se stablo odlučivanja) koji predviđa te greške, te se doda u postojeći ansambl,

3. ovako kreiran ansambl se dalje koristi za predviđanje novih podataka.

Ponavljanje procesa treniranja novog modela koji predviđa greške postojećeg ansambla za određene podatke i njegovo dodavanja u ansambl, vrši se onoliko puta koliko je specifikovano hiperparametrom koji određuje broj modela u ansamblu ili do trenutka kada se predviđanja ansambla ne poboljšavaju u odnosu na validacioni skup podataka. Inicijalni model predviđa vrijednost labele, dok svaki sljedeći model predviđa greške postojećeg ansambla. Finalno predviđanje se izračuna tako što se saberu vrijednosti svih modela. Na slici 3.7 prikazan je princip rada metode gradijentnog pojačavanja.



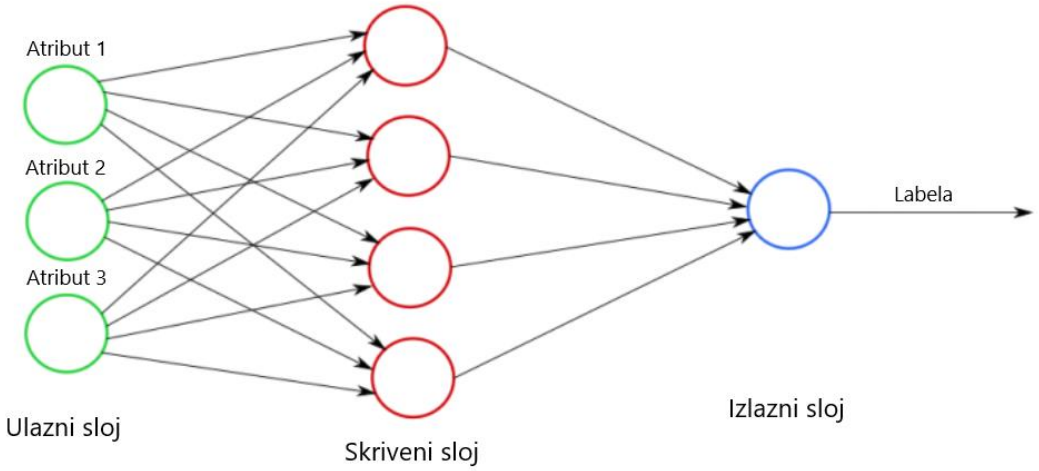
Slika 3.7 – princip rada metode gradijentnog pojačavanja

Postoji više varijanti (algoritama) koji koriste metodu gradijentnog pojačavanja. Neki od najpoznatijih su [19]: *GBM*, *XGBoost*, *LightGBM i CatBoost*. Ovi algoritmi sadrže veliki broj hiperparametara od kojih postoji nekoliko zajedničkih kao što su brzina učenja, maksimalan broj stabala odlučivanja i sl.

Algoritmi gradijentnog pojačavanja predstavljaju jedne od najpreciznijih algoritama mašinskog učenja, koji sa druge strane nisu previše računski zahtjevni. Njihova prednost je preciznost i brzina. Nedostatak ovih algoritama jeste dostizanje zasićenja, odnosno maksimalne preciznosti koja se ne može popraviti povećavanjem broja podataka.

3.4.4. NEURONSKA MREŽA

Jedan od najpoznatijih i najbitnijih algoritama mašinskog učenja jeste neuronska mreža [10]. Danas čak postoji grana mašinskog učenja koja se naziva duboko učenje (eng. *deep learning*) i koja se bavi isključivo proučavanjem i implementacijom neuronskih mreža. Ovaj algoritam nastao je po ugledu na neurone i njihovu međusobnu povezanost u živim organizmima. Tipična struktura neuronske mreže prikazana je na slici 3.8.



Slika 3.8 – Struktura neuronske mreže [20]

Neuron se sastoji od aktivacione funkcije, ulaza i izlaza. Ulazi predstavljaju izlazne vrijednosti neurona iz prethodnog sloja pomnožene odgovarajućim težinskim faktorima koji označavaju jačinu veze između neurona. Aktivaciona funkcija transformiše ulaznu vrijednost dobijenu zbirom ulaza. Izlaz predstavlja izlaznu vrijednost neurona. Sloj označava grupu neurona u istom nivou koji nisu međusobno povezani. Sve neuronske mreže sastoje se iz ulaznog, skrivenog i izlaznog sloja. Ulazni sloj sadrži onoliko neurona koliko postoji atributa i ti atributi predstavljaju njihove ulaze. Skriveni sloj se može sastojati od proizvoljnog broja slojeva i proizvoljnog broja neurona u njima. Izlazni sloj sadrži onoliko neurona koliko postoji izlaza. U regresiji, to je najčešće jedan neuron, dok u klasifikaciji broj zavisi od broja kategorija koje predstavljaju domen labele. Izlazi neurona koji pripadaju izlaznom sloju predstavljaju labele.

Neuronska mreža funkcioniše na osnovu dva principa: *feed forward* i *back propagation*. *Feed forward* znači da se predviđanje vrši transformacijom od ulaznog sloja ka izlaznom. Dakle, vrijednosti atributa dolaze na ulazni sloj i dalje se transformišu kroz neuronsku mrežu dok ne dobiju svoj finalni oblik u vidu izlaza iz neurona izlaznog sloja. *Back propagation* predstavlja korigovanje težinskih faktora mreže u cilju smanjenja greške predviđanja. To se postiže posredstvom funkcije gubitka i transformacijom težinskih faktora koja se vrši od izlaznog sloja ka ulaznom.

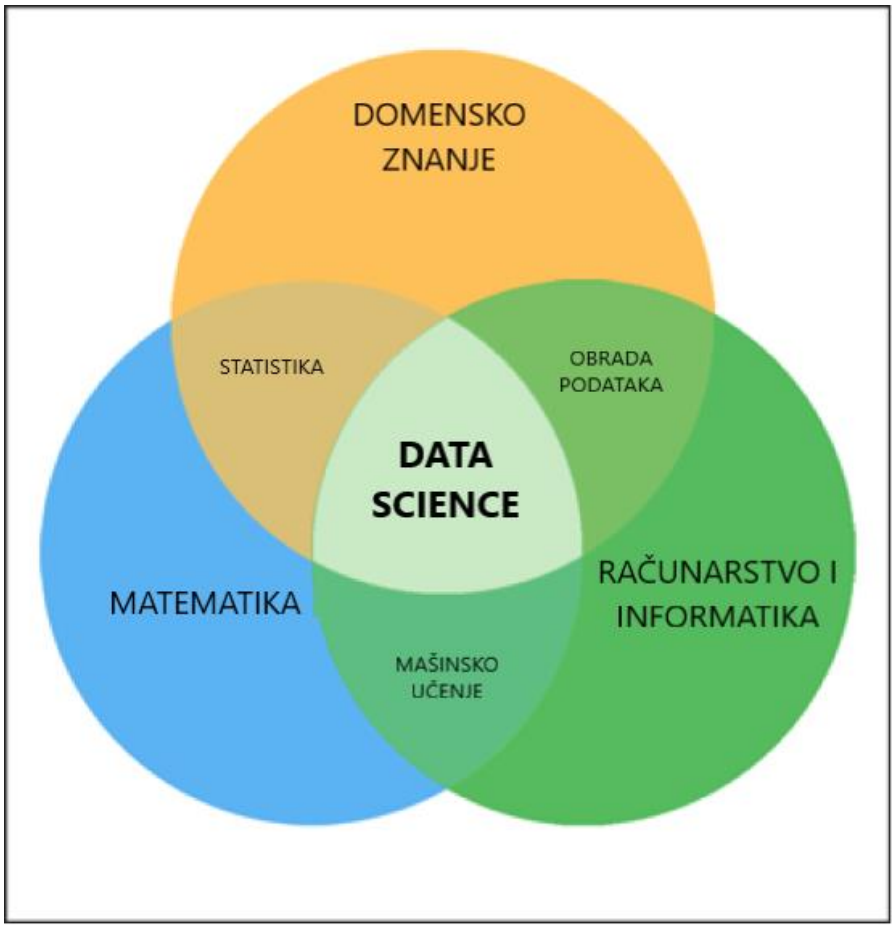
Neuronske mreže su postale popularne zbog toga što su, sa jedne strane, vrlo precizne i dobro modeluju nelinearne probleme, a sa druge, stalno im se poboljšava preciznost uvođenjem novih trening podataka. Mnogi tradicionalni algoritmi mašinskog učenja dostignu zasićenje, tj. ne mogu da poprave svoju preciznost čak ni uvođenjem novih trening podataka, što nije slučaj sa neuronskom mrežom. Nedostatak neuronskih mreža u odnosu na većinu tradicionalnih algoritama jeste taj što dosta sporije uče i treba im dosta više trening podataka da bi dostigle solidnu preciznost.

4. UPOTREBA MAŠINSKOG UČENJA U OBLASTI POSLOVNOG ODLUČIVANJA

Pored ogromne primjene koju mašinsko učenje pronalazi u različitim poljima, postoji značajan uticaj mašinskog učenja i u domenu poslovnog odlučivanja. Mašinsko učenje u poslovnom domenu najčešće je dio discipline koja se naziva nauka o podacima (eng. *data science* - *DS*). *DS* je širok pojam koji može da varira od domena i nivoa primjene u različitim organizacijama. Cilj *DS*-a dosta se preklapa sa ciljem BI-a u smislu da pruža podršku strategijskom i taktičkom nivou menadžmenta. Pristup je različit, tradicionalni *BI* koristi poznata pravila za ekstrakciju informacija iz skupa podataka, dok *DS* koristi pristup vođen podacima (eng. *data driven*), dobrim dijelom potpomognut mašinskim učenjem, da otkrije informacije u podacima za koje ne postoje jasna ili poznata pravila. Dakle, *BI* i *DS* predstavljaju komplementarne discipline koje zajedno daju potpunu sliku o trenutnim i budućim dešavanjima u okviru organizacije. Pored procesa izdvajanja informacija, *DS* često automatizuje različite procese u kojima je potreban određen nivo kognitivnog faktora, tako da rukovodioci i analitičari koji su se bavili organizacijom takvih procesa mogu da se bave drugim aktivnostima, jer sistem automatski obavlja te procese.

Domen *DS*-a je najčešće prediktivna analitika. Prediktivna analitika se bavi predviđanjem određenih vrijednosti na bazi ostalih i u sklopu prediktivne analitike koriste se algoritmi mašinskog učenja koji pripadaju kategoriji nadgledanog učenja. Pored prediktivne analitike, česta primjena *DS*-a je u otkrivanju asocijativnih pravila između podataka i podjeli podataka na grupe, na osnovu sličnosti, koje nisu uočljive „golim“ okom.

Na slici 4.1 prikazana je interdisciplinarnost *DS*-a. Domensko znanje je potrebno da bi se razumjela priroda podataka i da bi se postavila relevantna pitanja (pitanja na koja je moguće dati odgovor posredstvom podataka). Matematika predstavlja bitan faktor u razumijevanju rada algoritama mašinskog učenja i odabiru istih prilikom kreiranja modela. Pored toga, statističko znanje je neophodno da bi se izvršila korektna analiza podataka. Računarstvo i informatika je nezaobilazan dio *DS*-a jer se čitav proces odvija na računarima. Stoga je potrebno znanje na osnovu kojeg će se kreirati odgovarajuća infrastruktura za pribavljanje podataka, njihovu manipulaciju, te integraciju modela sa produkcionim sistemima.



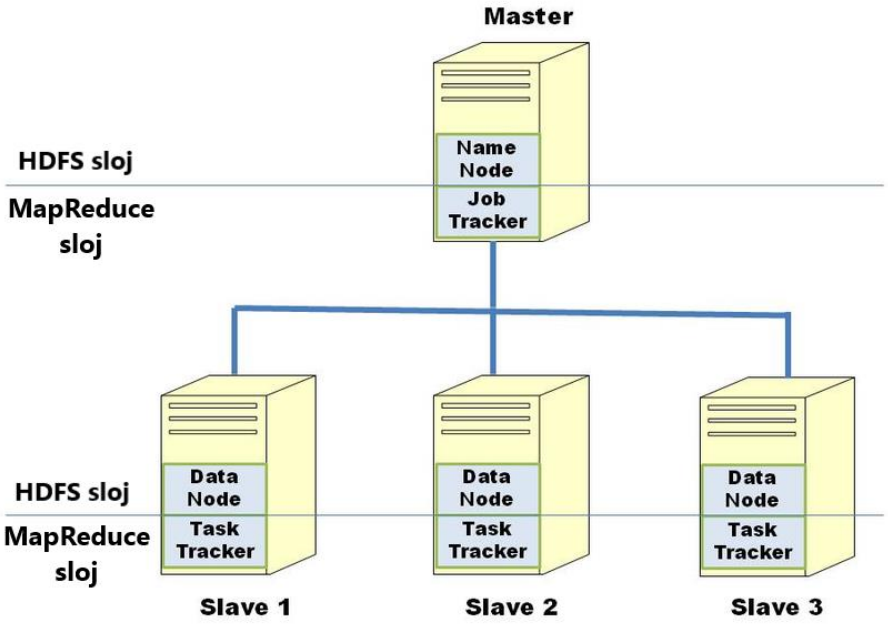
Slika 4.1 – Presjek DS-a i ostalih oblasti [21]

4.1. PLATFORMA

DS obično koristi ogromnu količinu podataka (*big data*). *Big data* predstavlja podatke koji često dolaze sa različitih izvora i u različitim formatima. Obrada i skladištenje podataka ovog obima i njihova priprema za kreiranje modela mašinskog učenja najčešće se obavlja paralelno na većem broju mašina i domen je oblasti koja se naziva *data engineering*.

Tipična arhitektura distribuiranog sistema[12] korištenog u *big data* okruženju sastoji se od distribuiranog skladišta podataka i sistema za paralelnu obradu podataka nad tim skladištem. Popularno rješenje otvorenog koda (eng. *open source*)[13] koje se primjenjuje u mnogim vodećim IT organizacijama jeste *Apache Hadoop*[14] [22]. To je *framework*[15] koji objedinjuje sistem za paralelno skladištenje podataka (*Hadoop Distributed File System* - HDFS), sistem za paralelnu obradu podataka (*MapReduce*), sistem za upravljanje resursima i raspoređivanjem (*YARN*) i dodatne druge servise. Distribuirani sistem koji koristi ovakvu arhitekturu naziva se klaster (eng. *cluster*). Klasteri se obično sastoje iz jednog (nekad i više) *master* čvora i većeg broja *slave* čvorova. *Master* čvor je glavna mašina koja upravlja radom i sinhronizacijom ostalih mašina koje predstavljaju *slave* čvorove i čiji je zadatak izvršavanje odgovarajućih procesa.

Uproštena arhitektura *Hadoop* klastera prikazana je na slici 4.2. *Name Node* je glavni proces HDFS-a koji upravlja *Data Node* procesima, a *Job Tracker* je glavni proces *MapReduce*-a koji upravlja *Task Tracker* procesima.



Slika 4.2 – Arhitektura Hadoop klastera [23]

[12] Distribuirani sistem je sistem čije su komponente mapirane na različitim mašinama koje su povezane odgovarajućom mrežom i koje međusobno komuniciraju i koordinišu akcije.

[13] Open-source termin označava besplatan softver otvorenog koda koji je moguće vidjeti i modifikovati, za razliku od komercijalnog softvera.

[14] URL zvaničnog sajta na kojem se nalazi projekat: https://hadoop.apache.org.

[15] Framework označava skup različitih integrisanih softverskih modula i biblioteka koji često sadrže elemente kontrole toka i koji predstavljaju parcijalnu aplikaciju, koja rješava određeni problem, koju je moguće prilagoditi specifičnoj potrebi.

Pored *Apache Hadoop framework*-a, postoje razna druga rješenja koja su u većoj ili manjoj mjeri kompatibilni sa *Hadoop*-om. Trenutno dosta popularno rješenje, koje obrađuje podatke brže nego *Apache Hadoop*, jeste *Apache Spark*[16]. *Spark* je *framework* koji međurezultate obrade podataka čuva u *RAM* memoriji, za razliku od *Hadoop*-a koji sve čuva na disku. *Spark* se često integriše sa *Hadoop* klasterom, jer kao izvor podataka za obradu koristi *HDFS*. Pored toga, *Spark* se može integrisati i sa drugim izvorima podataka poput relacionih baza podataka, *Redis*[17] baze podataka i sl. Pored ova dva *framework*-a, u *big data* ekosistemu koriste se i razna druga rješenja. Neka od najpoznatijih koji rješavaju drugačije aspekte problema u odnosu na *Hadoop* i *Spark* su *Apache Hive*[18] (*framework* koji prevodi *SQL* upite u *MapReduce task*-ove, često se integriše sa *Hadoop*-om), *Apache Kafka*[19], *Apache Samza*[20], itd.

Navedena rješenja su potrebna da bi se velika količina podataka iskoristila na efikasan način u realnom vremenu.

4.2. OBLASTI PRIMJENE I TIPOVI APLIKACIJA

Tipične aplikacije *DS*-a na strateškom i taktičkom nivou rukovođenja su sljedeće:

1. Segmentacija klijenata,
2. Sprečavanje odliva klijenata,
3. Predviđanje prodaje,
4. Poboljšanje kvaliteta,
5. Procjena rizika i
6. Finansijsko modelovanje.

Opis aplikacija i privilegija koje organizacije ostvaruju njihovom implementacijom dat je u nastavku.

4.2.1. SEGMENTACIJA KLIJENATA

Segmentacija klijenata (eng. *customer targeting*) predstavlja podjelu klijenata na grupe koje su slične u odgovarajućim segmentima relevantnim za marketing. To omogućava organizacijama da usmjere personalizovane ponude u smislu različitih proizvoda različitim klijentima tako da se vjerovatnoća konverzije[21] maksimizuje. Na ovaj način organizacija ostvaruje bolju komunikaciju sa klijentima, smanjuje troškove marketinga i povećava dobit.

[16] URL zvaničnog sajta na kojem se nalazi projekat: https://spark.apache.org.
[17] URL zvaničnog sajta na kojem se nalazi projekat: https://redis.io.
[18] URL zvaničnog sajta na kojem se nalazi projekat: https://hive.apache.org.
[19] URL zvaničnog sajta na kojem se nalazi projekat: https://kafka.apache.org.
[20] URL zvaničnog sajta na kojem se nalazi projekat: http://samza.apache.org.
[21] Konverzija je pojam koji označava da je klijent izvršio određenu akciju koja je predstavljala cilj marketinga. To može biti npr. online kupovina proizvoda.

4.2.2. SPREČAVANJE ODLIVA KLIJENATA

Sprečavanje odliva klijenata (eng. *churn prevention*) predstavlja aplikaciju *DS*-a koja predviđa koji klijenti, kada i zašto raskidaju svoje veze sa organizacijom. Ovaj fenomen može da bude prilično skup, budući da je zadržavanje postojećeg klijenta mnogo jeftinije od „dobijanja" novog. Ova aplikacija omogućava organizacijama da djeluju preventivno na zadržavanju klijenata prije nego što bude prekasno.

4.2.3. PREDVIĐANJE PRODAJE

Predviđanje prodaje (eng. *sales forecasting*) analizira istoriju prodaje odgovarajućeg proizvoda/servisa, godišnje sezone, kretanje događaja i sličnih proizvoda/servisa na tržištu i sl., da predvidi realističnu potražnju za proizvodom/servisom. Može da se primijeni na kratkoročna, srednjeročna i dugoročna predviđanja. Na osnovu rezultata ovakve DS aplikacije, organizacije mogu da predvide prihode i da optimalno alociraju resurse.

4.2.4. POBOLJŠANJE KVALITETA

Poboljšanje kvaliteta(eng. *quality improvement*) bavi se analizom tržišta, kao i preferencijama klijenata u cilju dizajniranja što kvalitetnijeg proizvoda/usluge. Ovakva aplikacija omogućava organizacijama da dizajniraju kvalitetnije proizvode/usluge koji će maksimizirati potražnju a, samim tim, i dobit.

4.2.5. PROCJENA RIZIKA

Procjena rizika (eng. *risk assessment*) predstavlja identifikaciju različitih tipova rizika u organizaciji. Tako npr. jedna od aplikacija može da predviđa rizik davanja kredita određenom klijentu u banci. Na osnovu aplikacija u ovom domenu, organizacije mogu da minimizuju rizike koji često rezultuju gubitkom sredstava i umanjivanjem dobiti.

4.2.6. FINANSIJSKO MODELOVANJE

Finansijsko modelovanje (eng. *financial modeling*) je tip *DS* aplikacije koja predviđa ekonomske tokove na višem nivou. Npr. ovakve aplikacije mogu da predvide kada će doći do rasta/pada cijena određenih sirovina. Na osnovu toga organizacije mogu da nabave odgovarajuće sirovine u najboljem trenutku, sa minimalnim rashodom. Pored toga, ovakve aplikacije svoju primjenu pronalaze u organizacijama i kada je riječ o novim ulaganjima ili povratu investicije.

5. REALIZACIJA SISTEMA ZA PROCJENU VRIJEDNOSTI NEKRETNINA NA BAZI MAŠINSKOG UČENJA

Cilj praktičnog dijela rada bio je razvoj konkretnog sistema na bazi mašinskog učenja koji pronalazi svoju primjenu u poslovnom okruženju koje se u određenom obimu bavi nekretninama. U ovom poglavlju opisan je proces razvoja pomenutog sistema. Ovakav sistem pronalazi svoju dominantnu primjenu u organizacijama koje se bave trgovinom nekretnina, ali i u organizacijama poput banaka koje nude kredite i koje u mnogim slučajevima kao pokriće stavljaju hipoteku na neku nekretninu. Krajnji cilj ovakvog sistema predstavlja povećanje dobiti koju bi organizacija ostvarila njegovom primjenom.

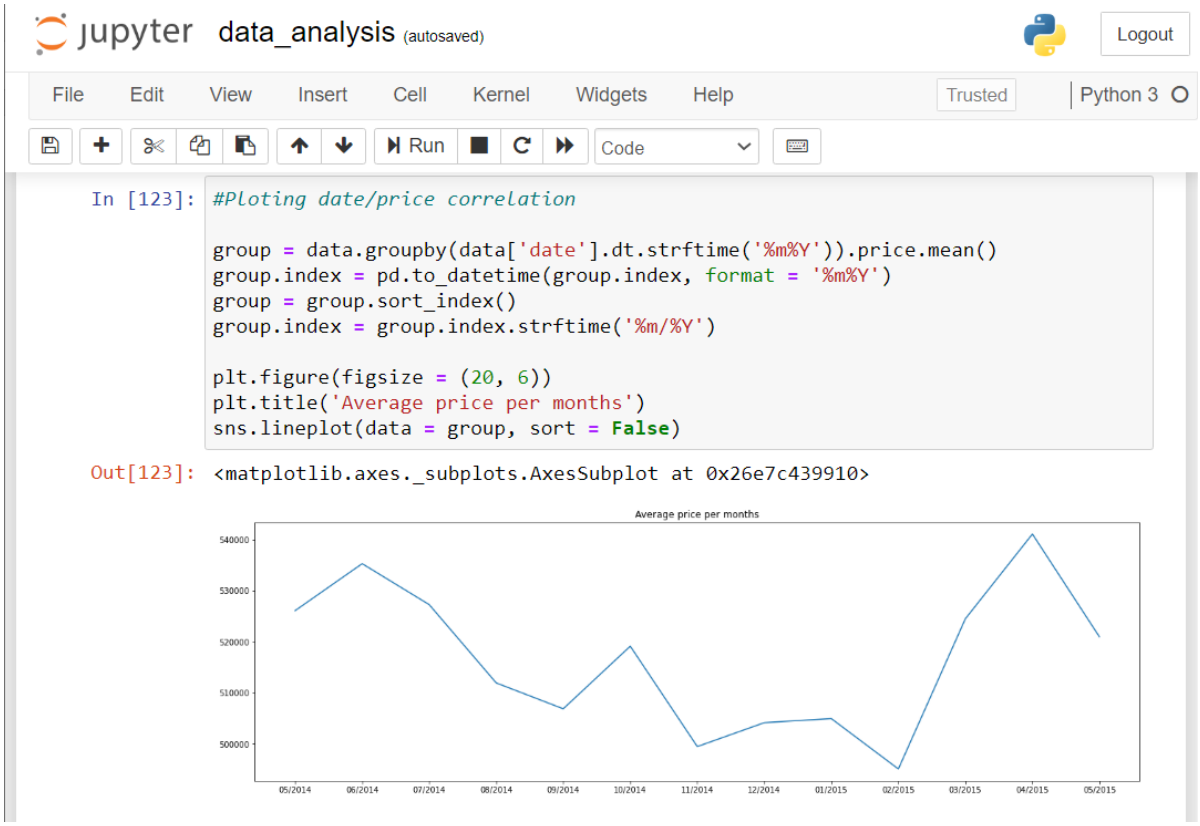
Da bi se procijenila vrijednost nekretnine, potrebno je angažovati agenta koji je dobro upućen u trenutnu ekonomsku situaciju i trendove na tržištu, koji će pažljivo analizirati sva svojstva nekretnine i koji će nakon određenog vremena potrebnog za analizu, donijeti svoju procjenu vrijednosti nekretnine. Pored rashoda koji agent generiše u vidu novčane kompenzacije za svoj rad, agentu je potrebno i određeno vrijeme, koje predstavlja veoma bitan resurs. Automatizovan sistem koji bi na osnovu određenih svojstava nekretnine koji predstavljaju ulaze, u vrlo kratkom roku koji se mjeri sekundama vratio rezultat u vidu predviđene cijene sa prilično velikom preciznošću, napravio bi ogromne uštede i ubrzao bi poslovne procese, pogotovo u organizacijama koje imaju veliki broj transakcija.

5.1. ALATI I RAZVOJNO OKRUŽENJE

Za realizaciju ovog projekta, korišten je programski jezik *Python*[22] koji predstavlja dominantan jezik za realizaciju *DS* i *ML* projekata jer sadrži veliki broj modula i biblioteka koje olakšavaju rad u tom domenu [24]. *Python* je korišten u sklopu razvojnog okruženja koje se naziva *Jupyter Notebook*[23]. Ovo okruženje je dosta praktično jer koristi tzv. *IPython*, odnosno interaktivni *Python*, što omogućava fragmentaciju koda i izvršavanje fragmenata nezavisno jedan od drugoga. Sve varijable čuvaju se na globalnom nivou, tako da svaki fragment može pristupiti varijabli i modifikovati je. Sa druge strane, *Jupyter Notebook* omogućava i formatiranje običnog teksta i iscrtavanje raznih grafikona, te jednostavne konverzije dokumenta u PDF, HTML i druge formate.

[22] URL zvaničnog sajta na kojem se nalazi pomenuti alat: https://www.python.org.
[23] URL zvaničnog sajta na kojem se nalazi pomenuti alat: https://jupyter.org.

Na slici 5.1 prikazan je primjer *Jupyter Notebook* dokumenta.



Slika 5.1 – Primjer Jupyter Notebook dokumenta

5.2. PRIPREMA PODATAKA

Skup podataka (eng. *dataset*) koji je korišten za realizaciju sistema, predstavlja podatke prodanih kuća iz okruga *King County* u okviru države *Washington* u SAD-u, u periodu od maja 2014. do maja 2015. godine[24]. Skup podataka sadrži 21613 zapisa i 21 atribut (kolonu). Da bi se konstruisao model koji dobro predviđa cijene, najprije je potrebno dobro pregledati i analizirati skup podataka na osnovu koga se konstruiše taj model.

[24] URL sajta sa kojeg je preuzet skup podataka: https://www.kaggle.com.

Na slici 5.2 se okvirno vidi struktura podataka, ali potrebna je detaljna analiza o tipovima podataka svih atributa, statistici podataka, podacima koji nedostaju, redundantnim zapisima i podacima koji imaju abnormalne vrijednosti.

	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	...
0	7129300520	20141013T000000	221900.0	3	1.00	1180	5650	1.0	0	
1	6414100192	20141209T000000	538000.0	3	2.25	2570	7242	2.0	0	
2	5631500400	20150225T000000	180000.0	2	1.00	770	10000	1.0	0	
3	2487200875	20141209T000000	604000.0	4	3.00	1960	5000	1.0	0	
4	1954400510	20150218T000000	510000.0	3	2.00	1680	8080	1.0	0	

	view	condition	grade	sqft_above	sqft_basement	yr_built	yr_renovated	zipcode	lat	long	sqft_living15	sqft_lot15
	0	3	7	1180	0	1955	0	98178	47.5112	-122.257	1340	5650
	0	3	7	2170	400	1951	1991	98125	47.7210	-122.319	1690	7639
...	0	3	6	770	0	1933	0	98028	47.7379	-122.233	2720	8062
	0	5	7	1050	910	1965	0	98136	47.5208	-122.393	1360	5000
	0	3	8	1680	0	1987	0	98074	47.6168	-122.045	1800	7503

Slika 5.2 – Prikaz prvih 5 instanci iz skupa podataka

```
id               int64
date            object
price           float64
bedrooms          int64
bathrooms       float64
sqft_living       int64
sqft_lot          int64
floors          float64
waterfront        int64
view              int64
condition         int64
grade             int64
sqft_above        int64
sqft_basement     int64
yr_built          int64
yr_renovated      int64
zipcode           int64
lat             float64
long            float64
sqft_living15     int64
sqft_lot15        int64
```

Slika 5.3 – Pregled tipova atributa

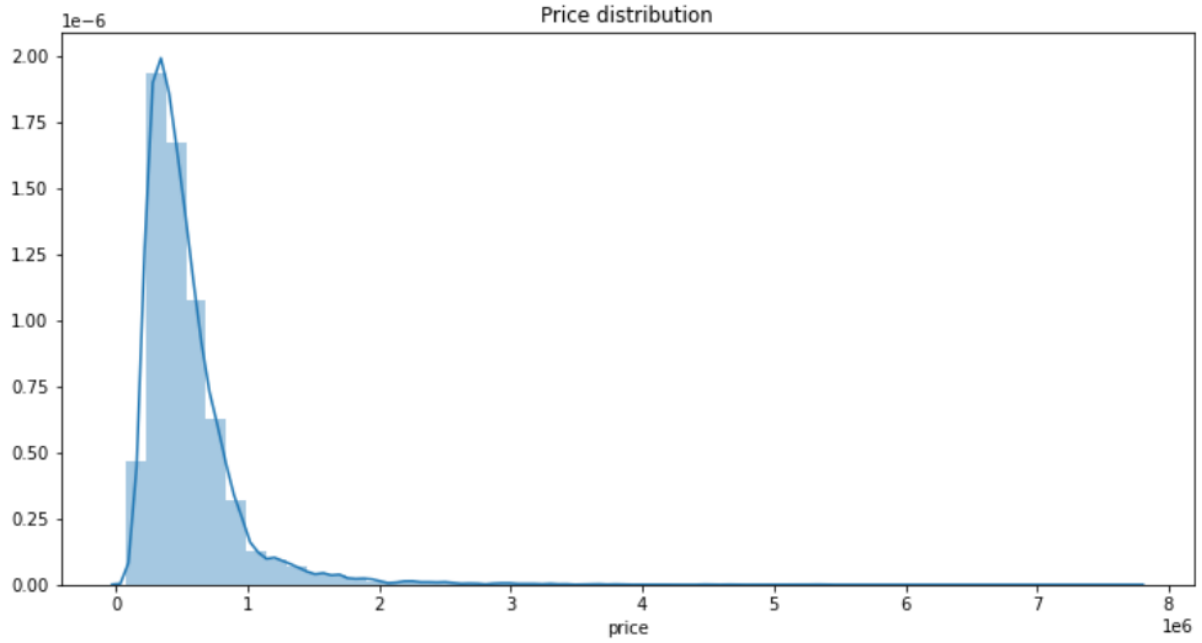
Na slici 5.3 prikazani su svi atributi i njihovi tipovi podataka. *Int64* predstavlja cijeli broj, *float64* predstavlja decimalni broj, a *object* u ovom slučaju predstavlja *string* vrijednost. Opis atributa je sljedeći:

- *id* – jedinstveni identifikujući broj svake kuće,
- *date* – datum prodaje kuće,
- *price* – cijena po kojoj je kuća prodana u američkim dolarima,
- *bedrooms* – broj soba,
- *bathrooms* – broj kupatila (postoje 3 tipa kupatila),
- *sqft_living* – kvadratura kuće u stopama,
- *sqft_lot* – kvadratura placa u stopama,
- *floors* – broj spratova (postoje 2 tipa: 0,5 i 1, gdje 0,5 opisuje visoko potkrovlje, a 1 kompletan sprat),
- *waterfront* – varijabla koja opisuje da li plac „izlazi“ na vodenu površinu,
- *view* – indeks koji opisuje koliko je dobar pogled kuće na skali od 1 do 4,
- *condition* – indeks koji opisuje stanje kuće na skali od 1 do 5,
- *grade* – indeks koji opisuje sveobuhvatni kvalitet i dizajn kuće na skali od 1 do 13,
- *sqft_above* – kvadratura kuće iznad tla u stopama,
- *sqft_basement* – kvadratura podruma u stopama,
- *yr_built* – godina izgradnje kuće,
- *yr_renovated* – godina u kojoj je kuća renovirana,
- *zipcode* – jedinstven broj koji identifikuje područje u kojem se nalazi kuća,
- *lat* – geografska širina,
- *long* – geografska dužina,
- *sqft_living15* – prosječna kvadratura kuće od 15 najbližih kuća u stopama,
- *sqft_lot15* – prosječna kvadratura placa od 15 najbližih kuća u stopama.

Ovaj skup podataka ne sadrži podatke koji nisu potpuni, tj. nema praznih polja. Takođe ne postoje redundantni zapisi. Daljnjom provjerom podataka utvrđeno je da postoji jedan zapis o kući koja ima 33 sobe, a nesrazmjerno malu kvadraturu kuće i jedan zapis o kući koja ima veću kvadraturu kuće iznad tla nego kvadraturu placa, a pritom ima samo jedan sprat. Ta dva zapisa su otklonjena jer predstavljaju abnormalne vrijednosti koje mogu loše uticati na model. Ostali podaci su validni.

Potrebno je analizirati i raspodjelu cijena u ovom skupu podataka da bi se otklonile rijetke vrijednosti odnosno *outlier*-i. *Outlier*-i su podaci koji su u manjini i koji mogu narušiti preciznost modela.

Na slici 5.4 prikazana je distribucija cijena. Većina zapisa sadrži kuće čija se cijena nalazi u intervalu do jednog miliona, a zapisi koji sadrže kuće čija cijena prelazi preko dva miliona su vrlo rijetki, tj. postoji svega 198 zapisa o kućama čija cijena se nalazi u intervalu od 2 do 8 miliona. To praktično znači da nema dovoljno podataka na osnovu kojih bi model naučio kako da predviđa cijene kuća u tom intervalu. Iz tog razloga iz skupa podataka su uklonjeni zapisi o kućama čija cijena prelazi dva miliona.



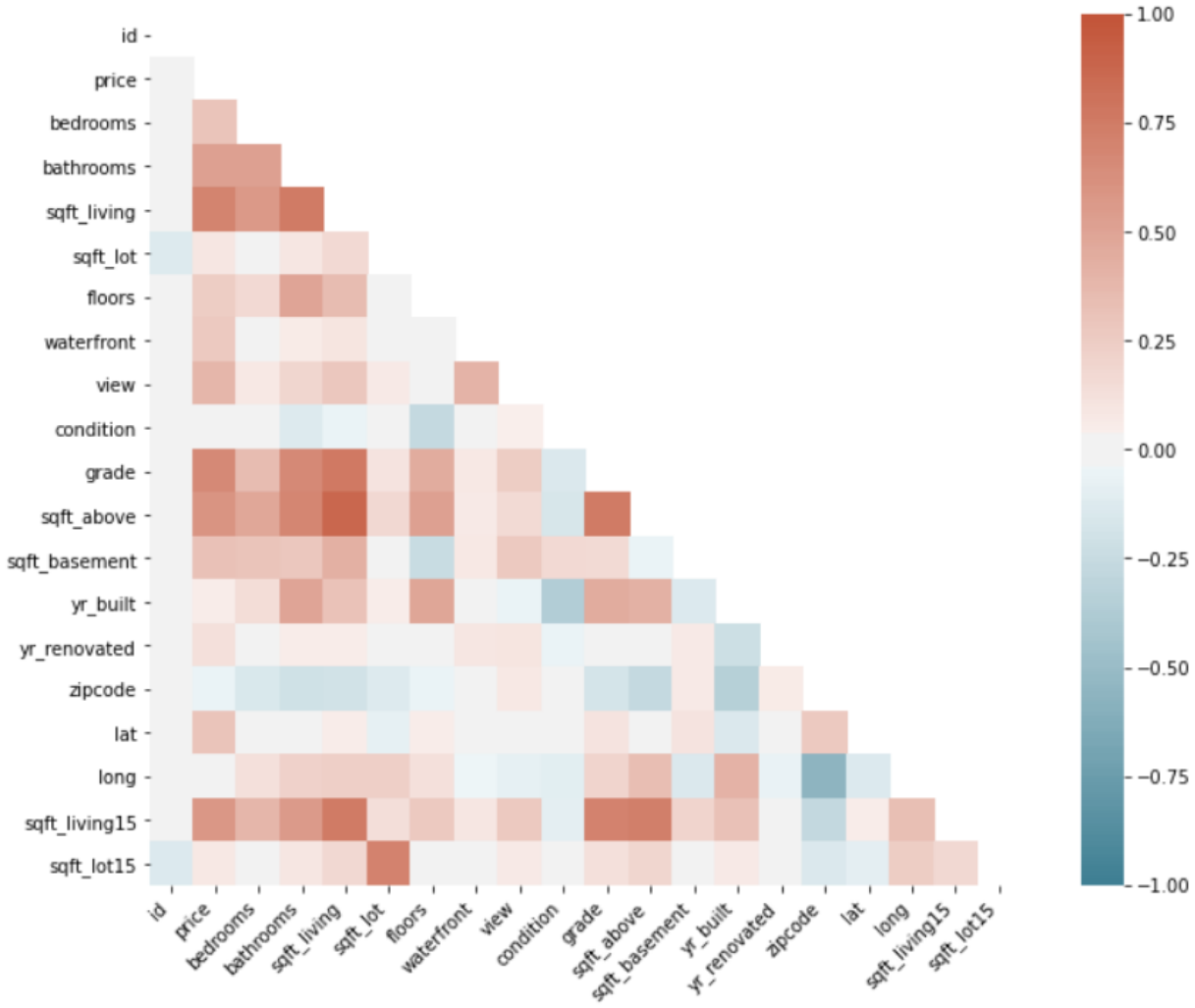
Slika 5.4 – Distribucija cijena kuća

5.3. TRENIRANJE MODELA

Nakon pripreme podataka, slijedi analiza uticaja različitih atributa na cijenu kuće. Ovo je dosta bitno, pogotovo kod podataka koji imaju veliki broj atributa, jer se na osnovu toga vrši biranje atributa koji ulaze u model i kreiranje novih atributa na bazi postojećih o čemu je bilo riječi u poglavlju „Mašinsko učenje“. Jedan od najbržih načina za pregled bitnijih atributa (atributa koji imaju veći uticaj na cijenu) je matrica korelacije, koja pokazuje linearnu korelaciju između svih atributa.

Na slici 5.5 prikazana je matrica korelacije na osnovu koje se vidi međusobna povezanost različitih atributa. Jarko crvena boja označava veliku pozitivnu korelaciju (ako se vrijednost jednog atributa povećava, onda se i vrijednost drugog atributa povećava), svijetle nijanse i bijela boja označavaju malu ili nikakvu korelaciju, dok jarko plava boja označava veliku negativnu korelaciju (ako se vrijednost jednog atributa povećava, vrijednost drugog atributa se smanjuje). Očigledno je da postoji velika pozitivna linearna korelacija između atributa *bathrooms*, *sqft_living*, *grade*,

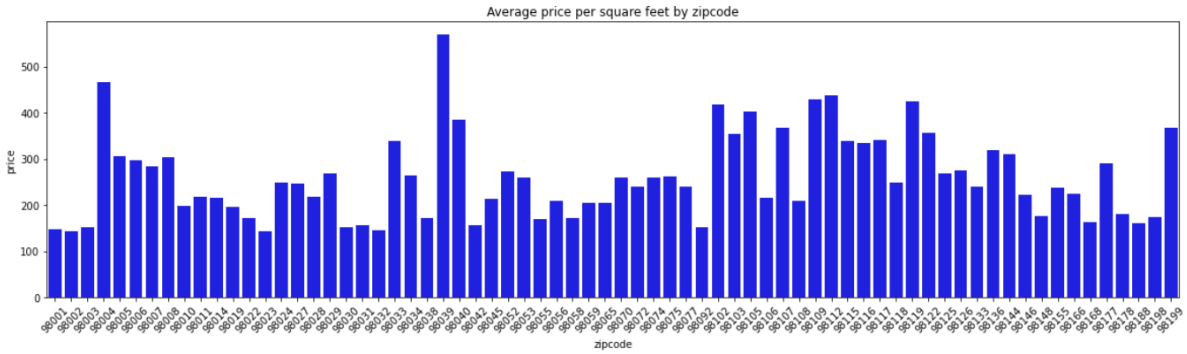
sqft_above, *sqft_living15* i *price*. Ovo je donekle i intuitivno, osim *sqft_living15* atributa za koji je neobično da ima toliko veliku pozitivnu korelaciju sa cijenom. Međutim, vidi se da *sqft_living15* atribut ima veliku korelaciju sa *sqft_living* atributom što objašnjava njegovu korelaciju sa cijenom. Očigledno je i da uticaj ostalih atributa na cijenu nije zanemarljiv, osim *condition*, *yr_built*, *zipcode*, *long* i *sqft_lot15* atributa. Međutim, mala linearna korelacija sa cijenom ne znači da oni zapravo nisu bitni. Radi se o tome da možda postoji neki drugi oblik funkcionalne zavisnosti između tih atributa i cijene koji nije linearan. Npr., neobično je da lokacija bude nebitna. Ali ovdje se radi o tome da je lokacija predstavljena sa 3 atributa: *zipcode*, *lat* i *long*. *Lat* i *long* se ne mogu posmatrati izolovano i to objašnjava malu korelaciju sa cijenom. Kada se analizira distribucija cijena kuća prodanih u različitim okruzima označenim *zipcode* atributom, vidi se da su u većini okruga prodavane i skuplje i jeftinije kuće, odnosno da je distribucija neravnomjerna, tako da je vrlo teško povezati takav atribut sa cijenom.



Slika 5.5 – Matrica korelacije između atributa

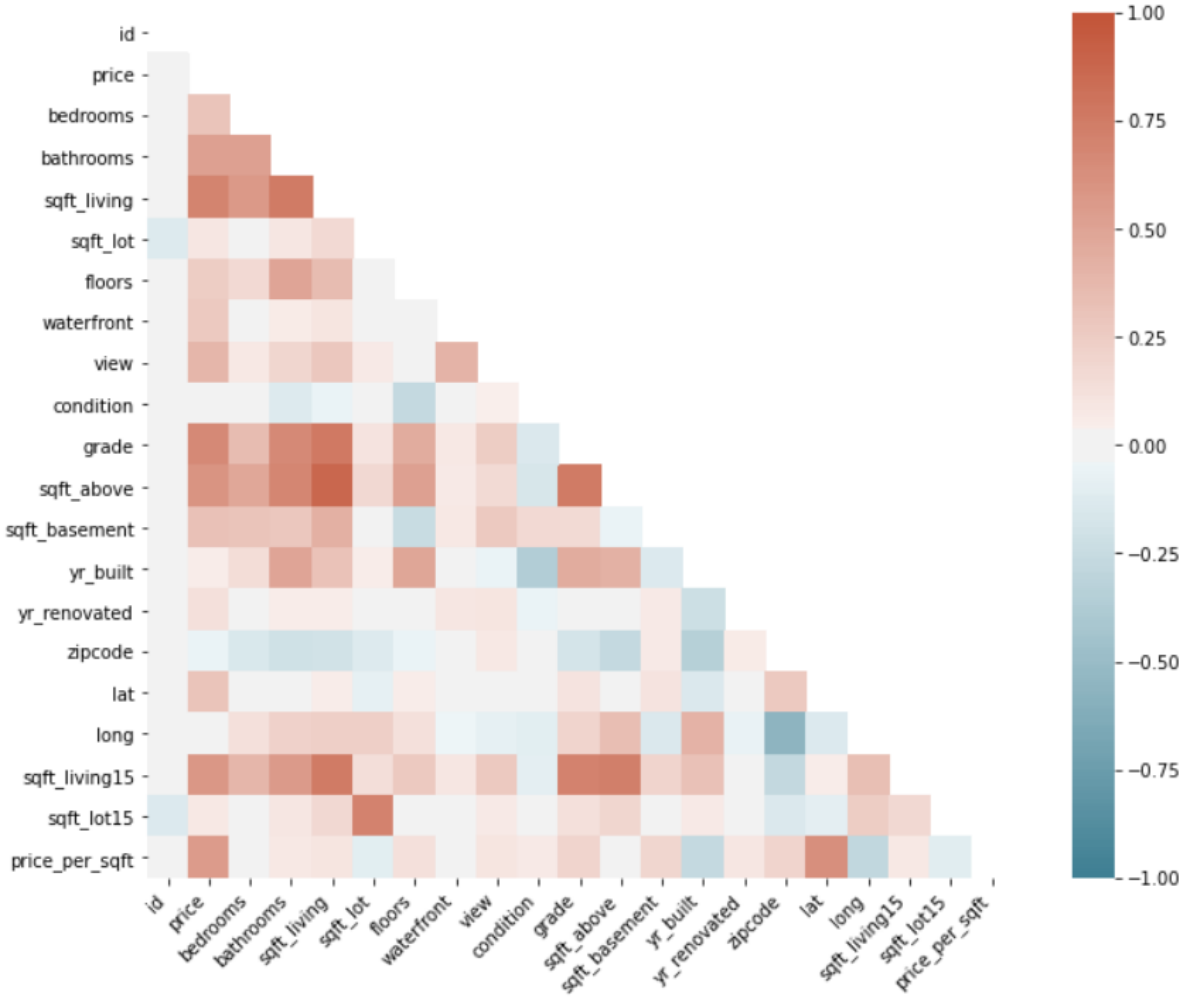
Iako sofisticiraniji algoritmi mašinskog učenja mogu prilično dobro da detektuju uticaje ovih atributa, mogao bi se postići značajan pozitivan uticaj na preciznost modela kreiranjem novog atributa koji će bolje predstavljati korelaciju između cijene i lokacije. Ideja je da se za svaki okrug označen *zipcode* atributom izračuna prosječna cijena prodanih kuća i prosječna kvadratura kuća i da se te dvije vrijednosti podijele. Na taj način dobija se novi atribut koji ugrubo predstavlja cijenu kvadrata u određenom okrugu. Neka se taj atribut naziva *price_per_sqft*.

Sa slike 5.6 se vidi da cijena kvadrata varira u različitim okruzima, a kada se na mapi provjere ti okruzi, vidi se da su skuplji okruzi oni koji se nalaze na periferiji grada *Seattle*-a, dok se jeftiniji okruzi nalaze na prilično velikoj distanci od grada. Nakon kreiranja novog atributa, iscrtana je ponovo matrica korelacije na kojoj se sad jasno vidi velika pozitivna korelacija između tog atributa i cijene.



Slika 5.6 – Prosječna cijena kvadrata po okrugu

Na slici 5.7 ponovo je prikazana matrica korelacija, na kojoj se ovoga puta nalazi korelacija između atributa *price_per_sqft* i cijene.



Slika 5.7 – Matrica korelacije između atributa na kojoj se vidi uticaj price_per_sqft atributa na cijenu

Još jedan bitan aspekt predstavlja datum prodaje. Jasno je da cijena varira vremenom zbog različitih faktora kao što je generalna ekonomska situacija ili potražnja. Međutim, pošto se u skupu podataka nalaze samo kuće koje su prodane u okviru jedne godine, postoji velika vjerovatnoća da šabloni koji se potencijalno mogu izvući iz podataka predstavljaju slučajnost. Iz tih razloga *date* atribut je otklonjen. Pored toga, otklonjen je i *id* atribut, jer je on dodijeljen nasumično i nema nikakvu povezanost sa cijenom.

Sada slijedi podjela podataka na trening, validacioni i testni skup, izbor algoritma mašinskog učenja, podešavanje hiperparametara, te njegovo treniranje. Prvo je potrebno razdvojiti pripremljeni skup podataka na skup podataka koji sadrži samo labele (cijene) i skup podataka koji sadrži ostale atribute. Nakon toga, potrebno je podijeliti oba skupa na trening,

validacioni i testni skup. 20% podataka čini testni skup podataka, dok od preostalih 80% podataka, 10% čini validacioni skup, a 70% trening skup. Broj podataka u trening skupu iznosi 15411, u validacionom 1713 i u testnom 4282. Na slici 5.8 prikazan je dio koda koji vrši pomenuto razdvajanje skupa podataka.

Na slici 5.8 prijazan je primjer koda koji dijeli originalni skup podataka na trening, validacioni i testni skup.

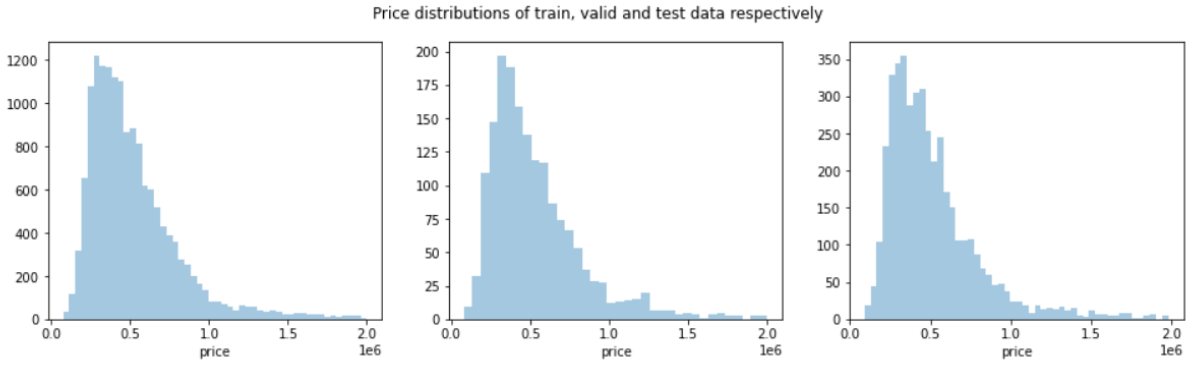
```
y = house_data.price
X = house_data.drop(columns = ['price'])

X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size = 0.2,
                                                    random_state = 1)
X_train, X_valid, y_train, y_valid = train_test_split(X_train, y_train,
                                                      test_size = 0.1,
                                                      random_state = 1)
```

Slika 5.8 – Primjer koda koji dijeli originalni skup podataka na trening, validacioni i testni skup

Da bi treniranje i evaluacija modela bila relevantna, bitno je da sva tri skupa imaju sličnu distribuciju cijena. Kada bi npr. cijena većine kuća u testnom skupu podataka bila preko jednog miliona, onda evaluacija modela ne bi utvrdila preciznost predviđanja cijena za kuće ispod jednog miliona.

Sa slike 5.9 se vidi da je distribucija cijena u sva 3 skupa veoma slična.



Slika 5.9 – Distribucija cijena u trening, validacionom i testnom skupu respektivno

Novi atribut, *price_per_sqft*, je u okviru analize uticaja atributa na cijenu kreiran na osnovu kompletnog skupa podataka. To znači da je kreiran i na osnovu testnih i validacionih podataka. Takav atribut je onda inkorporiorao u sebe i modelu nepoznate podatke, što može rezultovati boljom preciznošću modela koja je neobjektivna. Ovaj problem se naziva *Data Leakage*. Da bi se to izbjeglo ovaj atribut ponovo je kreiran samo nad testnim skupom podataka.

Na slici 5.10 prikazan je primjer koda koji kreira *price_per_sqft* atribut

```
# New feature price_per_sqft will be created for each part of the county(zipcode)
X_train_integrated = X_train.copy()
X_train_integrated['price'] = y_train

zipcode_average_price = X_train_integrated.groupby('zipcode').price.mean()
zipcode_average_sqft_living = X_train_integrated.groupby('zipcode').sqft_living.mean()

zipcode_price_per_sqft = zipcode_average_price.copy()

i = 0
while i < zipcode_average_price.size:
    zipcode_price_per_sqft.iloc[i] = zipcode_average_price.iloc[i] / zipcode_average_sqft_living.iloc[i]
    i += 1

X_train['price_per_sqft'] = X_train.apply(lambda row: zipcode_price_per_sqft[row.zipcode], axis = 1)
X_valid['price_per_sqft'] = X_valid.apply(lambda row: zipcode_price_per_sqft[row.zipcode], axis = 1)
X_test = X_test.copy()
X_test['price_per_sqft'] = X_test.apply(lambda row: zipcode_price_per_sqft[row.zipcode], axis = 1)
```

Slika 5.10 – Primjer koda koji kreira price_per_sqft atribut nad trening skupom podataka

Sada je sve spremno za izbor algoritma, konfiguraciju hiperparametara i treniranje modela. Pošto je labela kontinualna vrijednost, to znači da problem koji se rješava pripada regresiji.

Proces izbora algoritma, optimalnih hipermarametara i evaluacija modela je, kao što je već rečeno, nelinearan proces. U toku ove faze, vršilo se eksperimentisanje sa slučajnom šumom, *XGBoost*, *LightGBM* i neuronskom mrežom. Evaluacijom, a i generalnim performansama u vidu brzine treniranja, *LightGBM* se pokazao kao najbolji algoritam za ovaj skup podataka. Rezultati evaluacije se nalaze u sekciji „Evaluacija modela“.

Neki od bitnijih hiperparametara koji su korišteni za prilagođavanje rada *LightGBM* algoritma su sljedeći:

- *boosting* (*dart*) – tip metode gradijentnog pojačavanja (*dart* označava tip u kojem se izbacuje određeni procenat stabala odlučivanja iz modela radi smanjenja *overfitting*-a, podrazumijevani procenat iznosi 10%),
- *learning_rate* (0,06) – brzina učenja, odnosno parametar koji određuje uticaj pojedinačnog stabla odlučivanja na kompletan model,
- *num_boost_round* (5000) – broj iteracija,
- *feature_fraction* (0,8) – procenat atributa koji ulaze u fazu kreiranja stabla odlučivanja i
- *max_bin* (500) – broj diskretnih vrijednosti atributa (*LightGBM* pretvara kontinualne vrijednosti u diskretne radi povećanja brzine treniranja i smanjenja korištenja memorije [25]).

Kod korišten za treniranje modela prikazan je na slici 5.11.

```
# LightGBM - Core interface

categorical_features = ['condition', 'view', 'grade', 'waterfront']
train_data = lightgbm.Dataset(X_train, label = y_train,
                              categorical_feature = categorical_features,
                              params = {'max_bin': 500})
validation_data = lightgbm.Dataset(X_valid, label = y_valid,
                                   categorical_feature = categorical_features,
                                   params = {'max_bin': 500})

parameters = {
    'metric': ['mae', 'rmse'],
    'learning_rate': 0.06,
    'feature_fraction': 0.8,
    'boosting': 'dart',
    'drop_rate': 0.1
}

model = lightgbm.train(parameters,
                       train_data,
                       num_boost_round = 5_000,
                       valid_sets = validation_data,
                       early_stopping_rounds = 5,
                       categorical_feature = categorical_features,
                       verbose_eval = False)
```

Slika 5.11 – Primjer koda koji definiše algoritam, parametre algoritma i trenira model

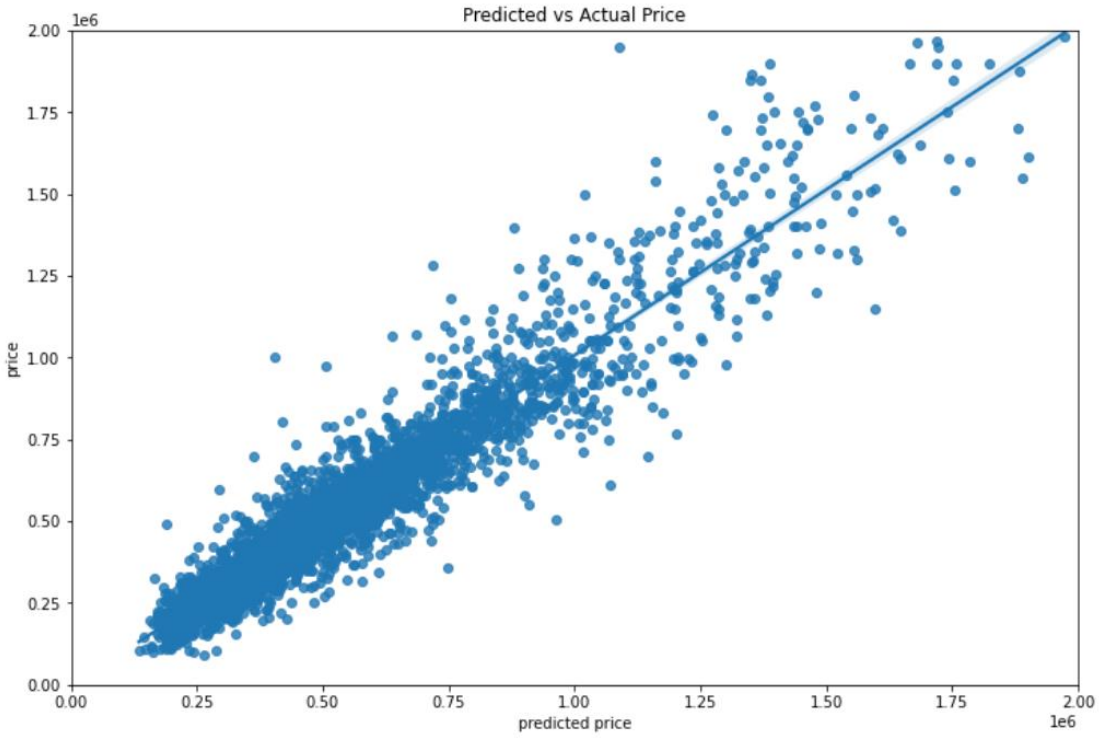
5.4. EVALUACIJA MODELA

Za evaluaciju modela korištene su sljedeće 3 metrike: *RMSE* (eng. *root mean squared error*), *MAE* i R2. *RMSE* predstavlja kvadratni korijen prosječne kvadratne greške predviđene vrijednosti u odnosu na pravu, *MAE* predstavlja prosječnu apsolutnu grešku predviđene vrijednosti u odnosu na pravu, a R2 predstavlja koeficijent korelacije predviđenih i pravih vrijednosti. Evaluacijom pomenutih modela treniranih u prethodnoj fazi, pokazuje se da se najbolji rezultati dobijaju primjenom *LightGBM* algoritma.

Rezultati su prikazani u tabeli 5.1.

ALGORITAM	RMSE	MAE	R2
Slučajna šuma	94937,85	60975,65	0.89
XGboost	90336,46	58883,90	0.90
LightGBM	85135,26	54719,74	0,91
Neuronska mreža	114853,46	74465,07	0,84

Tabela 5.1 – Rezultati evaluacije modela

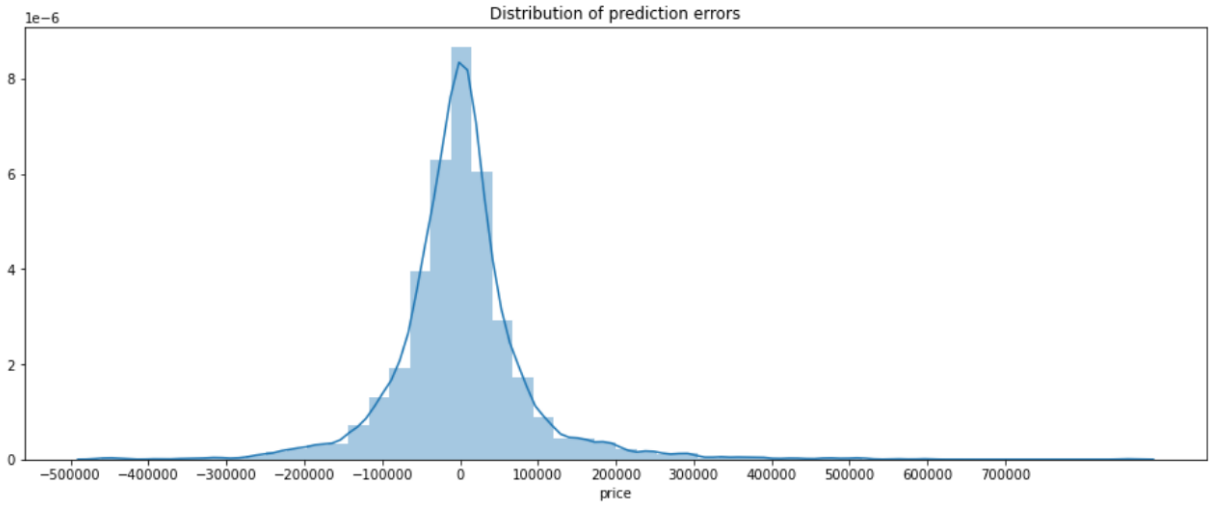


Slika 5.12 – Korelacija predviđenih i stvarnih cijena

Na slici 5.12 prikazana je korelacija predviđenih cijena i stvarnih cijena testnih podataka. Što su tačke bliže liniji to je predviđanje tačnije. Očigledno je da je model dosta precizan kada se radi o kućama čija je cijena ispod jednog miliona, za razliku od kuća čija je cijena preko jednog miliona. Razlog za to je manjak podataka o kućama čija je vrijednost preko jednog miliona.

Pored vizuelnog prikaza korelacije bitno je i da se prikaže distribucija grešaka da bi se utvrdila vjerovatnoća greške koja će se pojaviti u nekom opsegu.

Sa slike 5.13 vidi se da distribucija grešaka prati normalnu raspodjelu i da su greše skoncentrisane oko nule. Standardna devijacija iznosi 85127,017 što znači da će oko 70% grešaka nastalih u predviđanju cijene biti manje ili jednake 85127,017 dolara.



Slika 5.13 – Distribucija grešaka

5.5. ZAVRŠNA RAZMATRANJA

Cilj ovog poglavlja bila je demonstracija sistema koji koristi tehnike mašinskog učenja u cilju automatizacije poslovnih procesa. Ovakav sistem prilično dobro ilustruje koncept i zadovoljava definisani cilj. Iako preciznost koja je postignuta zadovoljava demonstrativnu namjenu sistema, za produkciono okruženje potrebno je dodatno poraditi na optimizaciji hiperparametara vezanih za sam algoritam i potencijalno na dodatnom inženjeringu atributa. Zavisno od algoritma, i skaliranje podataka, kao i kodovanje kategoričkih podataka *One Hot Encoding*[25] metodom, može pozitivno da utiče na preciznost.

[25] One Hot Encoding predstavlja tip kodovanja kategoričkih podataka gdje se kreira onoliko atributa koliko postoji kategorija, a oznaka da li odgovarajući red sadrži tu kategoriju ili ne označava se sa 1 ili 0 u odgovarajućem atributu.

6. ZAKLJUČAK

Mašinsko učenje nije novi termin, razni algoritmi i principi koji se primjenjuju u okviru oblasti mašinskog učenja bili su razvijeni sredinom prošlog vijeka. Međutim, sve masovnija primjena mašinskog učenja, zbog dva resursa koja su veoma dostupna danas: veliki podaci i jeftina procesna moć, postaje današnja stvarnost i novi veliki trend u IT oblasti. *Data science*, disciplina koja inkorporira mašinsko učenje u cilju automatizacije kognitivnih procesa i donošenja boljih odluka u poslovnom okruženju, danas je tek u fazi začetka. Veliki tehnološki giganti u IT industriji, kao što su *Google* i *Facebook*[26], aktivno primjenjuju *DS* tehnike i doprinose razvoju i poboljšanju te oblasti [26], [27]. Sve više organizacija prepoznaje prednosti ove discipline, a mnogi *startup*[27]-i svoje poslovne procese već od samog početka usklađuju sa *DS* sistemima [28].

Da bi primjena *DS*-a bila efektivna, potrebno je obezbijediti glavni resurs koji ona koristi: podatke. Neophodno je da podaci budu kvalitetni i tačni. Dakle, u samom startu bitno je da se razvije dobra infrastruktura koja će prikupljati što potpunije i relevantnije podatke. Otežavajuća okolnost u primjeni *DS*-a kod postojećih organizacija jeste inertnost ka uvođenju promjene. Mnoge organizacije koje kroz istoriju bilježe solidno poslovanje često nisu sklone ka promjeni svojih metoda i načina i sa nepovjerenjem „gledaju“ na *DS*, dok sa druge strane žele da i dalje budu konkurentne na tržištu i da maksimizuju dobit. Promjena je uvijek teška i u početku nailazi na mnoge opstrukcije, ali na duže staze rezultati koji se postižu daleko nadmašuju i zasjenjuju poteškoće koje su bile sastavni dio procesa uvođanja promjene.

Još jedna bitna napomena: nije sve za svakoga. Upravo iz razloga što je *DS* u trendu, mnoge organizacije ne žele da zaostaju za tim i to je njihov glavni razlog za težnjom ka implementaciji takvih sistema. Problem je što organizacije prvo treba da sagledaju da li im je *DS* zaista potreban i koje ciljeve žele da ostvare posredstvom te discipline. Mnoga mala preduzeća uopšte nemaju potrebu za takvim sistemima jer, niti imaju dovoljno prihoda da pokriju troškove takvih sistema, niti raspolažu velikim brojem podataka iz razloga što posluju u manjem obimu. *DS* sistemi svoju primjenu uglavnom pronalaze u srednjim i velikim preduzećima, jer takve organizacije raspolažu većim obimom podataka koji se mogu iskoristiti za obučavanje prediktivnih modela na osnovu kojih će organizacija generisati veći prihod od rashoda potrebnog za integraciju i implementaciju takvog sistema.

Pored svega navedenog, ono što se već sada nazire, a i prognozira, jeste sve veća upotreba *DS* i *ML* sistema u poslovnom okruženju, ali i šire, koja će u bliskoj budućnosti bilježiti eksponencijalan rast [29].

[26] URL-ovi zvaničnih sajtova pomenutih organizacija: https://www.google.com, https://www.facebook.com.
[27] Startup predstavlja kompaniju u začetku.

LITERATURA

[1] „The Age of Predictive Analytics: From Patterns to Predictions“, https://www.priv.gc.ca/media/1753/pa_201208_e.pdf.

[2] „Eight old ways of working that would seem crazy today“, https://www.telegraph.co.uk/business/ready-and-enabled/8-old-ways-working-seem-crazy-today.

[3] Reinsel, David, John Gantz, and John Rydning, "Data Age 2025", https://www.seagate.com/files/www-content/our-story/trends/files/idc-seagate-dataage-whitepaper.pdf.

[4] „Machine Learning, What it is and why it matters“, https://www.sas.com/en_be/insights/analytics/machine-learning.html.

[5] Kenneth C. Laudon and Jane P. Laudon, „Management Information Systems“, https://repository.dinus.ac.id/docs/ajar/Kenneth_C.Laudon,Jane_P_.Laudon_-_Management_Information_Sysrem_13th_Edition_.pdf.

[6] „4 Types of Data Analytics and How to Apply Them“, https://www.michiganstateuniversityonline.com/resources/business-analytics/types-of-data-analytics-and-how-to-apply-them.

[7] „Analytics 101 – The Four Types of Analytics and Their Uses“, https://infogovworld.com/ig-topics/analytics-101-four-types-analytics-their-uses.

[8] „What is the definition of OLAP“, https://olap.com/olap-definition/.

[9] Mladen Nikolić, Anđelka Zečević, „Mašinsko učenje“, http://ml.matf.bg.ac.rs/readings/ml.pdf.

[10] „Supervised and Unsupervised Machine Learning Algorithms“, https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms.

[11] „Clustering in Machine Learning“, https://www.geeksforgeeks.org/clustering-in-machine-learning/.

[12] „Reinforcement Learning algorithms – an intuitive overview“, https://smartlabai.medium.com/reinforcement-learning-algorithms-an-intuitive-overview-904e2dff5bbc.

[13] „4 Stages of the Machine Learning (ML) Modeling Cycle“, https://www.linkedin.com/pulse/4-stages-machine-learning-ml-modeling-cycle-maurice-chang.

[14] „Model Fit: Underfitting vs. Ovefitting“, https://docs.aws.amazon.com/machine-learning/latest/dg/model-fit-underfitting-vs-overfitting.html.

[15] „Commonly used Machine Learning Algorithms (with Python and R Codes)“, https://www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms.

[16] „Introduction to decision tree”, https://xzz201920.medium.com/introduction-to-decision-tree-97380cbfce3a.

[17] „A Trip to Random Forest...“, https://medium.com/greyatom/a-trip-to-random-forest-5c30d8250d6a.

[18] „Gradient Boost Part 1: Regression Main Ideas“, https://statquest.org/gradient-boost-part-1-regression-main-ideas.

[19] „4 Boosting Alghorithms You Should Know – GBM, XGBoost, LightGBM & CatBoost“, https://www.analyticsvidhya.com/blog/2020/02/4-boosting-algorithms-machine-learning.

[20] „How does a neural network make predictions?“, https://towardsdatascience.com/how-does-a-neural-network-make-predictions-6740663a63cb.

[21] „Data Science Advisory“, https://www.shellypalmer.com/data-science.

[22] „Usage of Hadoop and Microsoft Cloud in Big Data Analytics“, https://www.researchgate.net/publication/330637915_Usage_of_Hadoop_and_Microsoft_Cloud_in_Big_Data_Analytics.

[23] „Architecture of the Hadoop cluster“, https://www.researchgate.net/figure/Architecture-of-the-Hadoop-cluster_fig1_264103202.

[24] „2020 Kaggle Data Science & Machine Learning Survey“, https://www.kaggle.com/paultimothymooney/2020-kaggle-data-science-machine-learning-survey.

[25] https://lightgbm.readthedocs.io/en/latest/Features.html#optimization-in-speed-and-memory-usage.

[26] Valliappa Lakshmanan, „Data Science on the Google Cloud Platform“, https://www.academia.edu/42737273/Data_Science_on_the_Google_Cloud_Platform_Implementing.

[27] „Applied Machine Learning at Facebook: A Datacenter Infrastructure Perspective“, https://research.fb.com/wp-content/uploads/2017/12/hpca-2018-facebook.pdf.

[28] „Top 25 Machine Learning Startups To Watch In 2020“, https://www.forbes.com/sites/louiscolumbus/2020/04/26/top-25-machine-learning-startups-to-watch-in-2020.

[29] California University of Pennsylvania, „Data science: one of the fastest growing occupations“, https://www.calu.edu/academics/undergraduate/bachelors/data-science/jobs-career-salaries.aspx.