Metode strojnog učenja za klasifikaciju cijena stolnih računala

Menčik, Dominik

Sveučilište Josipa Jurja Strossmayera u Osijeku

Ekonomski fakultet u Osijeku

Diplomski studij Poslovna informatika

Dominik Menčik

Metode strojnog učenja za klasifikaciju cijena stolnih računala

Diplomski rad

Osijek, 2022.

Sveučilište Josipa Jurja Strossmayera u Osijeku

Ekonomski fakultet u Osijeku

Diplomski studij Poslovna informatika

Dominik Menčik

Metode strojnog učenja za klasifikaciju cijena stolnih računala

Diplomski rad

Kolegij: Sustavi poslovne inteligencije

JMBAG: 0010220977

e-mail: dmencik@efos.hr

Mentor: prof. dr. sc. Slobodan Jelić

Osijek, 2022.

Josip Juraj Strossmayer University of Osijek

Faculty of Economics in Osijek

Graduate study

Dominik Menčik

Machine learning methods for classifying desktop prices

Graduate paper

Osijek, 2022.

IZJAVA

O AKADEMSKOJ ČESTITOSTI, PRAVU PRIJENOSA INTELEKTUALNOG VLASNIŠTVA, SUGLASNOSTI ZA OBJAVU U INSTITUCIJSKIM REPOZITORIJIMA I ISTOVJETNOSTI DIGITALNE I TISKANE VERZIJE RADA

1. Kojom izjavljujem i svojim potpisom potvrđujem da je diplomski rad isključivo rezultat osobnoga rada koji se temelji na mojim istraživanjima i oslanja se na objavljenu literaturu. Potvrđujem poštivanje nepovredivosti autorstva te točno citiranje radova drugih autora i referiranje na njih.
2. Kojom izjavljujem da je Ekonomski fakultet u Osijeku, bez naknade u vremenski i teritorijalno neograničenom opsegu, nositelj svih prava intelektualnoga vlasništva u odnosu na navedeni rad pod licencom *Creative Commons Imenovanje – Nekomercijalno – Dijeli pod istim uvjetima 3.0 Hrvatska.* (cc) BY-NC-SA
3. Kojom izjavljujem da sam suglasan/suglasna da se trajno pohrani i objavi moj rad u institucijskom digitalnom repozitoriju Ekonomskoga fakulteta u Osijeku, repozitoriju Sveučilišta Josipa Jurja Strossmayera u Osijeku te javno dostupnom repozitoriju Nacionalne i sveučilišne knjižnice u Zagrebu (u skladu s odredbama Zakona o znanstvenoj djelatnosti i visokom obrazovanju, NN br. 123/03, 198/03, 105/04, 174/04, 02/07, 46/07, 45/09, 63/11, 94/13, 139/13, 101/14, 60/15).
4. izjavljujem da sam autor/autorica predanog rada i da je sadržaj predane elektroničke datoteke u potpunosti istovjetan sa dovršenom tiskanom verzijom rada predanom u svrhu obrane istog.

Ime i prezime studenta/studentice: Dominik Menčik

JMBAG: 0010220977

OIB: 32456433400

e-mail za kontakt: dmencikk@gmail.com

Naziv studija: Poslovna informatika

Naslov rada: Metode strojnog učenja za klasifikaciju cijena stolnih računala

Mentor/mentorica diplomskog rada: prof.dr.sc. Slobodan Jelić

U Osijeku, 2022. godine

Potpis Dominik Menčik

Sažetak

Metode strojnog učenja koriste se i primjenjuju godinama u raznim područjima poznatima čovjeku. Od medicine i obrazovanja sve do masovne proizvodnje i primjene u pametnim tvornicama. U ekonomiji također imamo primjera primjene umjetne inteligencije, npr. u kretanju cijena dionica, prognoziranju rizika, predviđanju cijena pojedinih proizvoda i slično. Ova primjena umjetne inteligencije je veoma česta u našem okruženju. Organizacije svakim danom pokušavaju poboljšati primjenu ove umjetne inteligencije da bi na što brži i efikasniji način predvidjele i dobile odgovore kretanja cijena u realnom vremenu.

U ovome radu bit će prikazane povijesna i teoretska podloga strojnog učenja. Prikazana je i objašnjena osnovna podjela strojnog učenja te je objašnjeno duboko učenje. Objasnit će se modeli poput logičke regresije, k-najbližih susjeda i neuronskih mreža, te će se pokušati utvrditi koja je metoda najbolja prilikom predviđanja cijena računala. Za analizu i modeliranje projekta koristit će se programski alat Python.

Ključne riječi: strojno učenje, k-najbližih susjeda, logička regresija, neuronske mreže, duboko učenje

Abstract

The methods of machine learning are being used and are applied in various areas known to man. From medicine and education to mass production and application in smart factories. In economy, too, are examples of application of artificial intelligence in smart factories i.e., the fluctuation of the stock market, risk prognosis, predicting prices of certain products etc. This application of artificial intelligence is commonly used in our environment. Companies try to enhance the application of this particular AI so that they would predict and find the answers to price fluctuation in real-time more efficiently.
This paper will showcase the historic and theoretic background of machine learning. Machine learning is also shown and explained on a basic level, but also deep learning will be touched upon. Models such as logistic regression, k-nearest neighbour algorithm and neural network will be described and this paper will try to find out which method is most suited for predicting PC prices. Python, a high-level programming language, will be used for the analysis and project modelling. Keywords: machine learning, k-nearest neighbour algorithm, logistic regression, neural network, deep learning

Keywords: Machine learning, k-nearest neighbors, logical regression, neural network, deep learning

Sadržaj

1. Uvod

Strojno učenje je jedna od najpoznatijih i najrelevantnijih tema današnjice, ona predstavlja granu umjetne inteligencije (engl. *Artificial Intelligence*) i računalne znanosti koja koristi podatke da bi sustavima omogućila učenje na temelju iskustva, te isključuje primjenu eksplicitnog programiranja. Osnova strojnog učenja je sustav koji samostalno rješava poslovne probleme na temelju velikih količina podataka. Tvrtke pomoću strojnog učenja mogu zamijeniti ljudsku snagu i zapravo poboljšati neke sposobnosti koje određena ljudska snaga ne bi mogla ostvariti.

Primjenu strojnog učenja možemo vidjeti svugdje u našoj okolini. Filmovi, serije, kupovina *online,* pametni telefoni, razne društvene mreže, sve se temelji na jednom dijelu strojnog učenja. Najpopularniji primjeri strojnog učenja zapravo su Amazon i Netflix. Amazon prikuplja naše podatke, naše ponašanje na njihovoj stranici i tako predviđa koji proizvodi bi nam se svidjeli, te nam olakšava kupnju. U današnjem modernom poslovanju podatak je ključan resurs kojim organizacije raspolažu prilikom donošenja poslovnih odluka. Algoritmi strojnog učenja „uče“ pomoću svih informacija te se modeli s vremenom i povećanjem broja ponavljanja unaprjeđuju i daju sve bolje rezultate.

Velikim razvojem tehnologije u posljednjih nekoliko godina dolazi do sve veće potrebe za raznim sustavima strojnog učenja. Stvaraju se različite metoda kako bi se olakšalo korištenje i razumijevanje velikih količina podataka. Zbog sve više konkurenata koji su spremni ostvariti prednost pomoću nekih vrsta umjetne inteligencije, ostali subjekti moraju se prilagoditi poslovanju da bi držali korak za njima. Bit uvođenja strojnog učenja je da se na što efikasniji i brži način donose odluke koje će zagarantirati uspjeh na tržištu.

1.1. Struktura i cilj rada

U prvom dijelu rada objašnjavaju se umjetna inteligencija, njezina povijest i prednosti. U drugom dijelu rada objašnjavaju se vrste i modeli strojnog učenja. U trećem dijelu dolazi do objašnjenja ključnih algoritama koji će se koristiti u radu, poput logističke regresije, te je objašnjen program koji će se koristiti prilikom izrade projekta. U posljednjem dijelu ovog rada objašnjen je projekt koji će biti implementirat u program Python i dobiveni rezultati.

Cilj rada je doći do odgovora kako se kreću cijene računala. Kreirat će se više modela pomoću kojih će se predviđati cijene. Podaci iz svakog modela bit će analizirani te će se donijeti zaključak koji je model najefikasniji, tj. koji model daje najbolje rezultate. Na kraju rada bit će navedene prednosti i nedostaci ovog istraživanja kao i preporuke za poboljšanje.

2. Teoretska podloga i prethodna istraživanja

2.1. Povijest umjetne inteligencije

Prije uvoda u umjetnu inteligenciju i strojno učenje trebamo znati što je to podatkovna znanost. Podatkovna znanost (engl. *Dana Science*) je znanost o podacima. Ona se bavi velikim količinama podataka koristeći određeni alata za njihovo razumijevanje i donošenje poslovnih odluka. Zauzvrat, ovi sustavi generiraju uvide koje analitičari i poslovni korisnici mogu pretvoriti u opipljivu poslovnu vrijednost.

Pojam strojnog učenje prvi put se spominje 1950. godine. Arthur Samuel iz tvrtke IBM je razvio program koji će omogućiti korisniku igranje dame. Budući da su to bile 1950-e godine, A. Samuel je imao na raspolaganju računalo s izrazito malom memorijom. Njegov dizajn uključivao je funkciju bodovanja koristeći položaje figura na ploči. Pomoću te funkcije pokušao je izmjeriti šanse za pobjedu svake strane. Program odabire sljedeći potez koristeći MinMax strategiju koja se kasnije razvija u MinMax algoritam. 1952. je poboljšao program, gdje je program pamtio pozicije koje su već bile odigrane. Te godine je prvi put zabilježen izraz „strojno učenje“. (Keith, 2021)

1957. godine Frank Rosenblatt na Sveučilištu Cornell konstruirao je model interakcije moždanih stanica Donalda Hebba s naporima strojnog učenja Arthura Samuela i stvorio perceptron. Perceptron je u početku planiran kao stroj koji će imati mogućnost prepoznavanja slika, kasnije je pretvoren u računalni program koji će biti dostupan za ostale strojeve te imati mogućnost prepoznavanja slika. S obzirom na tadašnju ograničenu tehnologiju, nije mogao prepoznati mnoge vrste vizualnih uzoraka (kao što su lica), uzrokujući odugovlačenje istraživanja neuronske mreže do ranih 1990-ih godina.

1967. godine kreiran je već poznatiji algoritam K-najbližih susjeda. Jedan je od najranijih algoritama korištenih u pronalaženju rješenja za problem prodavača u pronalaženju najučinkovitije rute. Upravo se taj algoritam smatra početkom osnovnog prepoznavanja uzoraka. Marcelo Pelilo dobiva zasluge za KNN algoritam.

Umjetna inteligencija i strojno učenje krajem 1970-ih godina idu odvojenim putovima. Istraživači umjetne inteligencije napuštaju istraživanje neuronskih mreža, te se to smatra točkom razdvajanja te dvije znanstvene discipline. Dok je strojno učenje bilo usredotočeno na algoritme, umjetna inteligencija se više usmjerila na korištenjem logičkih pristupa temeljenih na znanju. Sve do tada strojno učenje se smatralo kao program obuke za umjetnu inteligenciju.

Kasnije je koncept strojnog učenja prenamijenjen za rješavanje praktičnih problema i pružanje različitih usluga. (Keith, 2021)

Strojno učenje danas se smatra jednim od najvećim napredaka u tehnologiji. Od mogućnosti prepoznavanja lica, prepoznavanja govora sve do programa koji imaju mogućnost samostalno donijeti poslovnu odluku. Danas strojno učenje pronalazi veliku primjenu u auto-industriji. Najpoznatiji primjer su Teslini automobili, koji pomoću strojnog učenja pamte ceste te omogućuju vozilima da sama voze, bez čovjekove pomoći. Strojno učenje potaknulo je novi niz koncepata i tehnologija, uključujući nadzirano i nenadzirano učenje, nove algoritme za robote, internet stvari, analitičke alate i još mnogo toga. (Keith, 2021)

Glavna područja primjene strojnog učenja: (Keith, 2021)

- **Analiza podataka o prodaji** (engl. *Analyzing sales data*): pojednostavljenje podataka.
- **Mobilna personalizacija u stvarnom vremenu** (engl. *Real-time mobile personalization*): marketinški pristup koji pruža korisnicima točne i primamljive sadržaje (*e-mail*, *push*-obavijesti, SMS poruke).
- **Otkrivanje prijevare** (engl. *Fraud detection*): zaštita privatnih informacija i podataka.
- **Preporuke proizvoda** (engl. *Product recommendations*): preporuka novih proizvoda na temelju naših preferencija.
- **Sustavi upravljanja učenjem** (engl. *Learning management systems*): programi za donošenje odluka.
- **Dinamično određivanje cijena** (engl. *Dynamic pricing*): fleksibilne cijene koje se mijenjaju u realnom vremenu ovisno o potražnji i ponudi na tržištima.
- **Obrada prirodnog jezika** (engl. *Natural language processing*): *chatbotovi*.
-

2.2. Umjetna inteligencija

„Umjetna inteligencija propituje jednu od konačnih zagonetki. Kako je moguće da spor, maleni mozak, biološki ili elektronički, može percipirati, razumjeti i predviđati svijet, te manipulirati svijetom mnogo većim i mnogo kompleksnijim nego što je on? Kako da izgradimo nešto s takvim svojstvima?“ (Stuart Russell, 1995)

Umjetna inteligencija (engl. *Artificial intelligence*) omogućuje strojevima da uče iz iskustva kao čovjek. Polazi iz pretpostavke da strojevi mogu razmišljati i donositi kompleksne odluke kao normalno funkcionalna osoba. Umjetna inteligencija upravo to omogućuje strojevima. Strojevi imaju mogućnost prilagođavanja novim ulazima iz okoline, uče iz njih te obavljaju zadatke.

Današnji primjeri umjetne inteligencije kreću se od računala za igranje šaha, programa za prepoznavanje govora do samovozećih automobila. Ovakvi programi funkcioniraju na temelju obrade velikih količina podataka koje strojevi primaju iz okoline.

Umjetnu inteligenciju možemo podijeliti na 4 vrste: (Hintze, 2016)

- reaktivni strojevi
- umjetna inteligencija s ograničenom memorijom
- teorija uma
- samosvijest.

Reaktivni strojevi su zapravo najosnovniji principi umjetne inteligencije, sposobni su sami koristiti svoju inteligenciju da percipira i reagira na svijet ispred sebe. Ovakvi strojevi ne mogu pohraniti prošla iskustva, tj. ne mogu donositi odluke u realnom vremenu na temelju prošlih iskustava. Najpoznatiji primjer ove vrste umjetne inteligencije je IBM-ovo superračunalo za igranje šaha. Računalo Deep Blue može prepoznati sve figurice na šahovskoj ploči, znati njihova kretanja i prepoznati moguće poteze koje protivnik može poduzeti. Upravo zbog toga je ovo računalo uspjelo pobijediti prvaka u šahu Garija Kasparova. Mana ovoga sustava je to što ne može pamtiti poteze koji su se dogodili u prošlosti, tako da nema mogućnost prilagodbe. Ovakvi strojevi uvijek će se isto ponašati, pojavili se neki događaj jedanput ili više od 100 puta. Ovakvi strojevi ne mogu funkcionirati izvan specifičnih zadataka koji su im dodijeljeni i lako ih je prevariti. (Hintze, 2016)

Ograničena memorija je složenija od reaktivnih strojeva. Ova vrsta umjetne inteligencije ima sposobnost pohranjivanja prošlih podataka i donošenja odluka na temelju prošlih događaja. Prilikom kreiranja ove umjetne inteligencije važno je slijediti 6 vrlo bitnih koraka: (Anon., 2021)

- Moraju se kreirati podaci o obuci.
- Mora se kreirati model strojnog učenja.
- Model mora biti sposoban predviđati.

- Model mora moći primati povratne informacije od ljudi ili okoline.
- Povratne informacije moraju biti pohranjene kao podaci.
- Koraci se moraju ponoviti kao ciklus.

U ovoj vrsti umjetne inteligencije razlikuju se 3 bitna modela:

- Učenje s pojačanjem: ovi modeli uče i pokušavaju pružiti što bolja predviđanja nakon mnogo ciklusa ponavljanja.
- Dugotrajno pamćenje: ova vrsta modela koristi podatke iz prošlosti, te na temelju toga daje predviđanja. Ovi sustavi više vrednuju novije informacije jer smatraju da informacije iz daleke prošlosti nisu jednako važne kao i nove informacije iako ih i dalje koriste prilikom predviđanja.
- Evolucijske generativne mreže: ovaj model razvija se svakom novom odlukom. Model se može usporediti s čovjekovom evolucijom, odnosno ovaj model svakom novijom odlukom postaje sve točniji i precizniji.

Jedna od važnijih pretpostavki umjetne inteligencije je taj da čovjek može imati osjećaje prema nečemu dok stroj to ne može. Svako živo biće može razviti osjećaj o stvarima iz svoje okoline i donositi vlastite odluke na temelju tih osjećaja, dok stroj ne može. Da bi strojevi ikada mogli „živjeti“ među ljudima, morat će razviti sposobnost razumijevanja naših osjećaja kako bi mogli donijeti prave odluke. Najbolji način da donesu najpogodnije odluke u takvim situacijama je da sami razviju i razumiju vlastite osjećaje. To je temeljna pretpostavka ove vrste umjetne inteligencije, omogućiti strojevima da razviju vlastite osjećaje i da donesu odluke na temelju tih osjećaja. Ova umjetna inteligencija smatra se budućnošću, ali još uvijek nemamo te znanstvene i tehnološke sposobnosti ostvariti ju.

Samosvijest bi bio zadnji korak u finaliziranju umjetne inteligencije. Ovo se smatra najtežim korakom jer istraživači trebaju prvo razumjeti što je to svijest te kako omogućiti stroju da ostvari vlastitu svijest. Primjer ove teorije je kada osoba zna da želi nešto uraditi, npr. kada čekamo u redu u banci i čujemo po glasu osobe iza nas da je uznemirena. Upravo to ova teorija želi da naučimo nekog robota. Ovaj korak dolazi nakon što se ostvari teorija uma, tj. ova teorija je zapravo proširenje teorije uma.

2.3. Prednosti i rizici umjetne inteligencije

Znamo da umjetna inteligencija donosi velike prednosti i pogodnosti poput auta bez potrebe za vozačem, preporuka na raznim *web*-stranicama te čak i sustava i strojeva koji će nam pomagati i odgovarati na naša zdravstvena pitanja. Trebamo si postaviti pitanje je li zapravo AI siguran ili trebamo provesti određene sigurnosne protokole da bi se zaštitili. Još jedno veoma važno pitanje koje si trebamo postaviti je što će se dogoditi ako uspijemo razviti savršen sustav umjetne inteligencije, što će biti s ljudima.

Stručnjaci smatraju da se mogu dogoditi svi scenariji s razvojem umjetne inteligencije: (Anon., 2016)

- Umjetna inteligencija je programirana da učini nešto razorno: sustavi umjetne inteligencije namijenjene ratovanju. Stručnjaci smatraju da ovakvi sustavi ako dođu u pogrešne ruke mogu dovesti do velikih katastrofa. Najveći problem ovakvih sustava je to što bi oni bili dizajnirani tako da ih je teško isključiti. Trebamo biti oprezni prilikom dizajniranja ovakve tehnologije.
- Umjetna inteligencija je programirana da učini nešto korisno: ovi sustavi bi uvelike unaprijedili život na zemlji, poboljšali bi naš okoliš te radili ono što ljudi ne mogu.

Najveći strah kada se spomene pojam umjetne inteligencije je taj hoće li ona ugroziti radna mjesta. Danas AI ne ugrožava radna mjesta jer postoji mnogo radnih pozicija koje AI još uvijek ne može uspješno odraditi.

Iako postoje neke negativne strane umjetne inteligencije, ne mogu se ignorirati njezini veliki potencijali. AI svakim se danom sve više razvija, unaprjeđuje i olakšava naš život na Zemlji. Potrebno je da razumijemo njene pogodnosti i mane.

„Uzimajući u obzir sve prednosti i nedostatke umjetne inteligencije ovisi o korisniku i njihovoj perspektivi. AI i robotika poboljšat će način na koji razmišljamo, način na koji istražujemo i način na koji živimo.“ (Anon., 2019)

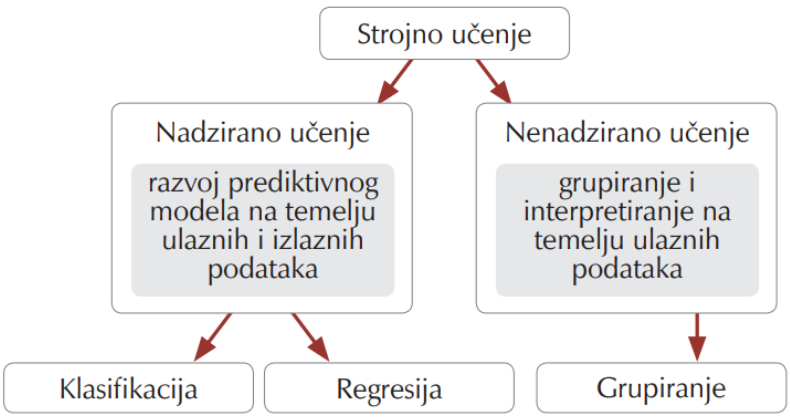
3. Strojno učenje

Strojno učenje je jedan oblik umjetne inteligencije odnosno grana umjetne inteligencije. Strojno učenje omogućava strojevima da imitiraju ponašanje čovjeka, odnosno daje strojevima mogućnost učenja bez prisutnosti čovjeka. Najbolju definiciju strojnog učenja dao je Tom M. Mitchell: računalni program uči iz iskustva E s obzirom na neku klasu zadataka T i mjeru učinka P, ako se njegova izvedba u zadacima u T, mjerena pomoću P, poboljšava s iskustvom E.

Primjena strojnog učenja uvelike se razvila od onog poznatog primjera A. Samuela i njegove igre dame. Danas imamo široku primjenu strojnog učenja, ovo su neka od područja njegove primjene: (Selig, 2022)

- Sigurnost podataka: modeli strojnog učenja mogu predvidjeti potencijalne opasnosti ranjivosti podataka, te na temelju prošlih iskustava predviđaju te sprječavaju da dođe do curenja podataka.
- Financije: banke koriste ovakve programe za predviđanje kreditne sposobnosti prilikom odobravanja kredita, također se koriste i za automatizaciju korisničke podrške.
- Maloprodaja: jedan od poznatijih primjera primjene strojnog učenja je maloprodaja. Programi uče o navikama potrošača te tako predviđaju proizvode za svakog potrošača pojedinačno.
- Obrada prirodnog jezika: raspoznavanje govora.

Na Slici 1. vidi se osnovna podjela strojnog učenja koja će biti detaljnije objašnjena u nastavku.



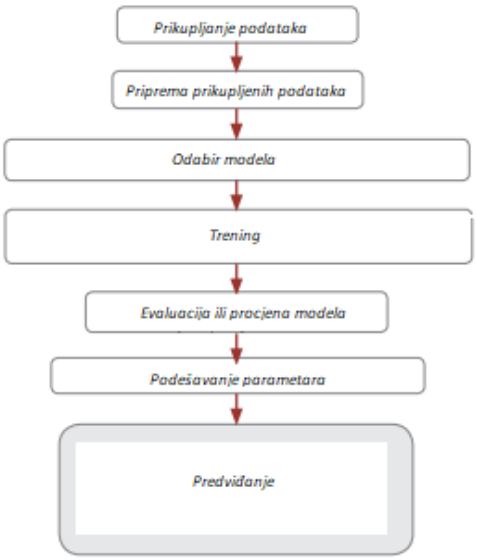
Slika 1. Osnovna podjela strojnog učenja

Izvor: https://hrcak.srce.hr/file/382926

Kako se podaci nastavljaju širiti, potreba za ovakvom tehnologijom je sve veća. Da bi bolje objasnili primjenu strojnog učenja, u sljedećem poglavlju bit će objašnjeni koraci procesa strojnog učenja.

3.1. Proces strojnog učenja

Proces strojnog učenja mnogi smatraju veoma kompleksnim. U ovom poglavlju proces strojnog učenja pokušat će se objasniti u 7 jednostavnih koraka. Kako bi se olakšao proces razumijevanja strojnog učenja koristit će se jednostavan primjer kruške i jabuke. (Martinez, 2020). Slika 2. prikazuje dijagram postepenih koraka strojnog učenja.



Slika 2. Koraci u strojnom učenju

1. korak: prikupljanje podataka

Prilikom razvoja svakog modela potrebni su nam podaci. Podaci su ujedno jedni od najvažnijih resursa današnjice i bez njih modeli strojnog učenja ne bi funkcionirali. Kao što je bilo navedeno, koristit ćemo primjer jabuka i krušaka. Da bi naš program ispravno funkcionirao, moramo pripremiti što više različitih primjera jabuka i krušaka, koje će varirati od boje prema veličinama. Veoma je bitno da odmah u početku imamo dovoljno podataka da naš stroj može prepoznati o kojem podatku, u ovom slučaju voćki, se radi. Ovaj korak je veoma važan i bitno je da se odmah u početku ne dogode greške koje će dovesti do kvara našeg programa.

2. korak: priprema prikupljenih podataka

Nakon što smo odradili prvi korak, sljedeći zadatak nam je pripremiti sve te podatke. Ispitat ćemo naša dva skupa da ne bi došlo do pristranosti određene vrste voća. Ključni zadatak ove faze bit će minimiziranje potencijalnih pristranosti u našim skupovima podataka. Drugi veoma bitan zadatak bit će raščlanjivanje skupova na dva dijela. Veći dio (80 %) bi se koristio za obuku modela, dok se manji dio (20 %) koristi za potrebe evaluacije. Ova faza radi se zbog toga da ne dođe do korištenja istih skupova podataka za evaluaciju. Glavni zadatak ove faze je dobra pripremljenost podataka jer time povećavamo učinkovitost programa.

3. korak: odabir modela

Nakon što su nam svi podaci spremni za korištenje, red je na odabiru modela. Prilikom odabira modela moramo posvetiti veliku pozornost pošto su neki modeli namijenjeni za rad s tekstualnim podacima, slikama ili samo brojkama.

4. korak: trening

Za ovaj korak se smatra da je najvažniji od svih 7 koraka. Obukom naših podataka u ovom modelu pokušavamo pronaći obrasce i napraviti predviđanja. Ovaj korak zahtijeva mnogo strpljenja. Očekivano je da dođe do nekoliko neuspjelih rezultata prije negoli dođe do uspješnih rezultata. Naš model uči u ovoj fazi te s vremenom on postaje sve točniji u predviđanju naših podataka.

5. korak: evaluacija ili procjena modela

Korak procjene modela dolazi nakon uspješno obavljenog treninga modela. Prilikom procjene modela ne koristimo isti skup podataka kao i na treningu. Dok smo na treningu koristili skup od 80 % podataka, u procjeni modela koristimo onaj manji, od samo 20 % podataka. Ovaj korak nam omogućuje da vidimo kako će se model ponašati s podacima koje još nije vidio. Pomoću ovog modela vidimo kako bi se on ponašao u stvarnom svijetu gdje ne zna podatke. Na kraju ovaj korak nam pomaže da znamo ako smo ostvarili konačni cilj ovog projekta.

6. korak: podešavanje parametara

Sljedeći korak dolazi na red ako je evaluacija bila uspješna. Nakon što smo izradili svoj model, na red je došla provjera ako možemo na bilo koji način poboljšati naš model. To ostvarujemo pomoću parametara odnosno varijabli koje smo ugradili u naš model. Parametre smo definirali u treningu, prilikom ovog koraka vraćamo se na fazu treninga gdje ispitujemo dodatne

mogućnosti kako bi poboljšali naš model i učinili naše predviđanje točnijim. Nakon što smo zadovoljni s našim parametrima idemo na sljedeći finalni korak.

7. korak: predviđanje

Predviđanje je finalni korak u ovom procesu koji dolazi na red kada je sve prije toga uspješno odrađeno. U ovom koraku dobivamo odgovore na naša pitanja, tj. rezultate naših predikcija. Kada dođemo do 7. koraka, smatramo da je naš model spreman za praktičnu uporabu i spreman je samostalno predviđati.

3.2. Vrste i modeli strojnog učenja

Strojno učenje može se podijeliti na 3 glavne vrste: nadzirano strojno učenje, nenadzirano strojno učenje, polunadzirano i podržano ili ojačano strojno učenje.

3.2.1. Nadzirano strojno učenje

Nadzirano učenje podskup je strojnog učenja. Algoritmi nadziranog učenja rade, tj. uče na temelju primjera. Ako želimo da naš program odredi nalazi li se na slici stol ili stolica, moramo unijeti podatke, tj. slike o stolicama i stolovima kako bi naš program mogao odrediti razliku između te dvije slike. Naš program uči na temelju tih podataka te kasnije može predvidjeti što se nalazi na određenoj slici. Znači, kada korisnik unosi ulazne podatke u model, sustav se prilagođava kako bi predvidio ishode i preciznije klasificirao podatke unakrsnom provjerom valjanosti.

Nakon što smo unijeli sve potrebne podatke, dolazi faza takozvanog treninga. Trening je prva faza učenja našeg programa. Tijekom treninga algoritam će tražiti uzorke u podacima koji su u korelaciji sa željenim rezultatima. Nakon treninga algoritmu dajemo nove ulaze, testni skup nudi testne podatke iz stvarnog svijeta, potrebno je program pripremiti za rad u stvarnom svijetu gdje će svakodnevno dobivati nove, prije neviđene podatke. Na kraju, cilj ovog modela je da predvidi oznaku novih podataka koji mu dolaze, tako da može razlikovati dvije slike. (Yufeng, 2017)

Primjenu nadziranog učenja možemo uočiti u mnogim aplikacijama jer modeli nadziranog učenja uvelike unaprjeđuju aplikacije. Primjeri primjene nadziranog učenja uključuju: (Balodi, 2019)

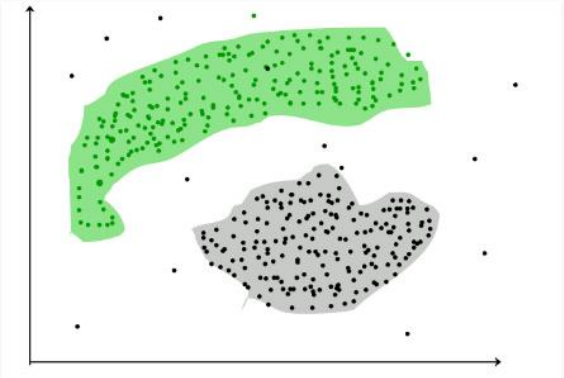
- Analizu osjećaja: ovi algoritmi strojnog učenja omogućuju organizacijama da lakše upravljaju golemim količinama podataka pri identifikaciji značenja određenog teksta prilikom pružanja informacija o raspoloživosti određenog proizvoda.
- Preporuke: ovi algoritmi strojnog učenja daju preporuku kupcima na osnovi njihovih prošlih aktivnosti. Ovaj model koriste mnogi prodavači, ali ga je i Netflix integrirao u svoj model. Netflix pomoću ovog primjera algoritma preporuča svojim korisnicima filmove i serije na temelju njihovih sviđanja, tj. na temelju prošlih aktivnosti na stranici.
- Prepoznavanje slika: pomoću ovih algoritama strojnog učenja imamo mogućnost prepoznati i izolirati slike i objekte na slikama i videozapisima.
- Filtriranje neželjene pošte: uvođenjem nadziranih klasifikacijskih algoritama pomoću strojnog učenja korisnici mogu uspješno organizirati neželjenu poštu. Pomoću ovog modela moguće je otkriti neželjeni virus, zlonamjerni softver ili čak otkrivanje štetnih URL-ova.

Nadzirano učenje razlikuje dvije podvrste, klasifikaciju i regresiju, o kojima će biti više rečeno u daljnjim poglavljima.

3.2.2. Nenadzirano učenje

Nenadzirano učenje veoma se razlikuje od onoga koje je nadzirano. Kod ovoga učenja nisu poznati izlazni podaci. Znači, naš program mora sam otkriti što mu se prikazuje. Učenje bez nadzora koristi podatke koji nisu označeni ni klasificirani. Velika je razlika između nenadziranog i nadziranog učenja, gdje smo u početku definirali skupove s podacima. U nenadziranom učenju program mora sam doći do rezultata bez smjernica. Znači, program nenadziranog učenja grupirat će sve podatke prema nekim sličnostima ili razlikama bez prijašnjih smjernica ili bez prijašnje obuke podataka. Npr. stroj neće znati razliku između jabuke i kruške, ali ih može razlikovati prema nekim sličnostima ili možda razlikama. Jabuka je manjeg i okruglijeg oblika od kruške, prema tome će i program uočiti razliku između te dvije slike i moći će donijeti odluku koja slika što predstavlja. Krajnji zadatak ovog strojnog učenja je zapravo otkrivanje kakav je izlazni podatak, odnosno od kakve je strukture, te ga čini idealnim rješenjem za istraživačku analizu podataka, segmentaciju kupaca i prepoznavanje slika. Ovaj model strojnog učenja općenito se dijeli na: (Gatto, 2021)

- Grupiranje: predstavlja metodu grupiranja objekata u iste skupine. U grupama će biti isti podaci te se ne smiju poklapati sličnosti podataka s ostalom grupom. Na Slici 3. možemo vidjeti primjer grupiranja, gdje su isti podaci združeni u grupama.



Slika 3. Grupiranje podataka

Izvor: https://www.geeksforgeeks.org/clustering-in-machine-learning/

Ne postoje kriteriji za dobro grupiranje, ono ovisi o korisniku. Korisnik određuje kriterije koji zadovoljavaju njegovu potrebu koju želi ostvariti programom.

Glavni problem grupiranja je otkriti inherentna grupiranja u našim podacima. Ova vrsta nenadziranoga strojnog učenja je vrlo važna jer određuje grupiranje među neobilježenim podacima. (Mishra, 2017)

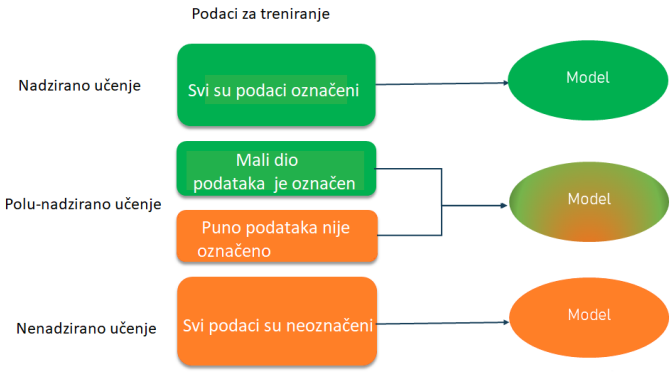
- Smanjenje dimenzionalnosti: ova vrsta strojnog učenja odnosi se na smanjenje broja ulaznih varijabli prilikom modeliranja. Smanjenjem dimenzionalnosti smanjuju se i parametri te dolazi do jednostavnije strukture modela strojnog učenja. Poželjno je imati što jednostavnije modele koji će se dobro generalizirati, što rezultira dobivanjem ulaza sa što manje varijabli.

3.2.3. Polunadzirano učenje

Polunadzirano učenje je zapravo kombinacija nadziranoga i nenadziranoga učenja. Polunadzirano učenje koristi malo označenih podataka i mnogo neoznačenih podataka za treniranje modela strojnog učenja. Ovim načinom rada model polunadziranog učenja pruža prednosti nadziranog i nenadziranog učenja. Velika prednost ovog modela je što ne moramo koristiti velik broj označenih podataka prilikom izrade modela. Polunadzirano učenje funkcionira na sljedeći način: (Anon., 2022)

- Prilikom treniranja koristimo model s malom količinom označenih podataka. Ovaj način uvježbavanja modela koristi se i u nadziranom učenju. Uvježbavamo model do trenutka kada smo zadovoljni s našim rezultatima.
- Drugi korak je da rezultate kombiniramo sa skupom koji ima označene podatke prilikom obuke i predviđanja izlaza.
- Treći korak je povezivanje oznaka iz označenih podataka s oznakama ostvarenih u prethodnom koraku.
- Četvrti korak je povezivanja podataka u označenim podacima o treningu s ulazima u neoznačenim podacima.
- Posljednji korak je treniranje modela na isti način kao što smo to radili na početku u prvom koraku s označenim skupom. Cilj ovog koraka je smanjiti pogrešku i poboljšati točnost modela. (Anon., 2020)

Na Slici 4. vidimo razliku između podataka koji se koriste kod nadziranog, nenadziranog i polunadziranog učenja.



Slika 4. Razlike između primjene podataka kod vrsta strojnog učenja

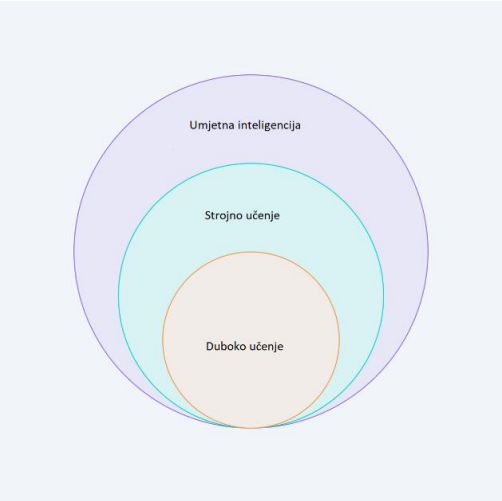
Izvor: https://www.altexsoft.com/blog/semi-supervised-learning/

Najbolji primjer polunadziranog učenja je klasifikator tekstualnih dokumenata. Ovaj primjer je idealan za polunadzirano učenje jer primjenjuje malo označenih podataka. Polunadzirano učenje najbolje radi zbog upotrebe malih količina označenih tekstualnih dokumenata dok i dalje klasificira veliku količinu neobilježenih tekstualnih dokumenata prilikom obuke podataka.

3.3. Duboko učenje

Duboko učenje je podskup strojnog učenja. Jedna od bitnih karakteristika dubokog učenja je da se ono u potpunosti temelji na neuronskim mrežama. Neuronska mreža nastoji oponašati ljudski mozak. Više o neuronskim mrežama bit će obrađeno u nastavku ovog rada. Prijašnji algoritmi koji su bili spomenuti u ovom radu su linearni, dok se duboko učenje temelji na složenijim te kompleksnijim algoritmima.

Na Slici 5. prikazan je odnos između umjetne inteligencije, strojnog učenja i dubokog učenja.



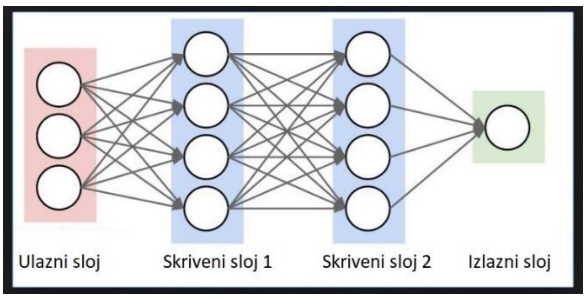
Slika 5. Odnos između umjetne inteligencije, strojnog učenja i dubokog učenja

Izvor: https://assets-global.website-files.com/5d7b77b063a9066d83e1209c/6178a71fecbc21865e972aa9_ldfyEGEJwZmBmylddpfMQ8e30wCHe6O241XJWBZWufNli3Ab36yX9RZiB8bmjCs-IZIaxVEFGjpFWzyuT00gBscJf2OM1JT-BlPKOH9J2nlmnVrOfe1GFAsV6ojZLTjvwv66ISTH.png

Programi dubokog učenja rade na način kao i što čovjek uči. Duboko učenje sastoji se od 4 sloja međusobno povezanih neurona ili čvorova. Svaki od tih čvorova ima svoju funkciju, obično se sastoji od linearne funkcije koja mapira informacije. (Barla, 2022)

Neuronska mreža sastoji se od ulaznog sloja, 2 skrivena sloja i jednog izlaznog sloja. Ulazni sloj služi za preuzimanje informacija i prosljeđivanje tih informacija prema skrivenim slojevima. U skrivenom sloju dolazi do upotreba linearnih funkcija. Pomoću tih funkcija algoritam koristi podatke koje je dobio od ulaznog sloja. Ovi slojevi nazivaju se skrivenim jer ne znamo parametre u svakom od slojeva. Skriveni slojevi dodaju parametre podacima da bi transformirali podatke, od kojih svaki daje različite rezultate. Izlazni sloj onda dobiva sve te podatke i na temelju njih predviđa. (Barla, 2022)

Na Slici 6. vidimo prikazana sva 4 sloja.



Slika 6. Prikaz slojeva dubokog učenja

Izvor: https://www.i2tutorials.com/hidden-layers-in-neural-networks/

Dok ova metoda ima brojne prednosti poput eliminiranja nepotrebnih troškova te je lako identificirati nedostatke koje je inače teško otkriti, ima i svoje nedostatke. Jedan od nedostataka ovog učenja je to što treba velike količine podataka, te je oprema za obuku tj. računalo jako skupa. Mogućnosti strojnog učenja i dubokog učenja svakim danom postaju sve veće. Od primjena u zabavi do velikih tvornica koje se sve više automatiziraju i sve je veća potreba za primjenom strojeva. Strojno učenje je poboljšalo naš život i nastavlja ga poboljšavati.

4. Metodologija rada

4.1. Algoritmi strojnog učenja

Najbitniji dio strojnog učenja su njegovi algoritmi. Prilikom rješavanja problema strojnog učenja ne možemo koristiti uvijek iste algoritme. Algoritmi se razlikuju po načinu na koji rješavaju problem te po vrsti problema za koji su namijenjeni. Prilikom odabira dobrog algoritma moramo paziti na sljedeće bitne stavke: točnost, vrijeme treninga, količinu podataka, parametre i još mnogo toga. Stoga je vrlo bitno odabrati pravi algoritam da bi zadovoljili vlastite potrebe.

Vrste regresijske analize: (Mali, 2021)

- linearna regresija
- logistička regresija
- polinomska regresija
- *ridge* regresija
- LASSO regresija.

4.2. Linearna regresija

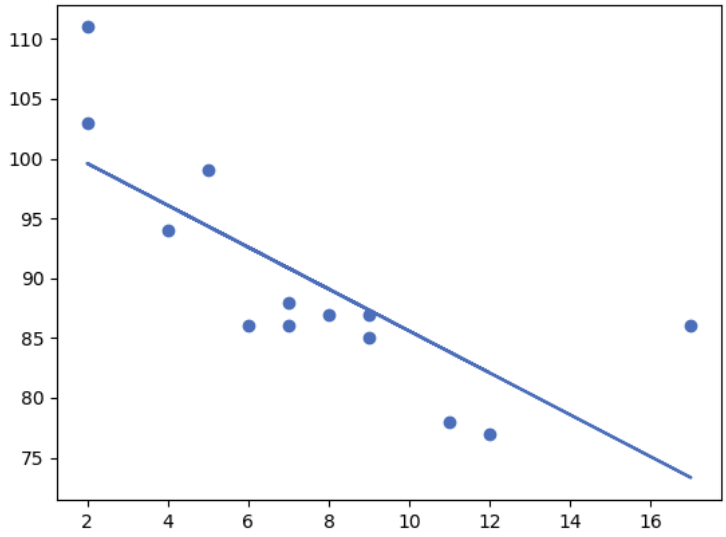
Linearna regresija je osnovna vrsta regresije. Ova vrsta regresije je jedan od modela strojnog učenja koji koristimo prilikom predviđanja podataka. Linearna regresija sastoji se od dviju varijabla, nezavisne i zavisne. Ova vrsta regresije broji samo jednu nezavisnu varijablu, ukoliko dođe do više nezavisnih varijabli to prestaje biti linearna regresija te se pretvara u višestruku linearnu regresiju.

„Zadaća regresije je pronaći liniju između svake vrijednosti prediktora i predikcije, a linija gdje je udaljenost između vrijednosti osi X i Y najmanja, izabere se model predikcije.“ (Marco, 2018)

Formula za linearnu regresiju je sljedeća: $Y = a + b\mathrm{X}$

gdje je X varijabla nezavisna, a Y varijabla zavisna, b predstavlja nagib pravca, „a“ je presjek (vrijednost y kada je x = 0).

Na Slici 7. prikazan je model linearne regresije.



Slika 7 Graf linearne regresije

Postoje 4 pretpostavke povezane s modelom linearne regresije: (Anon., 2020)

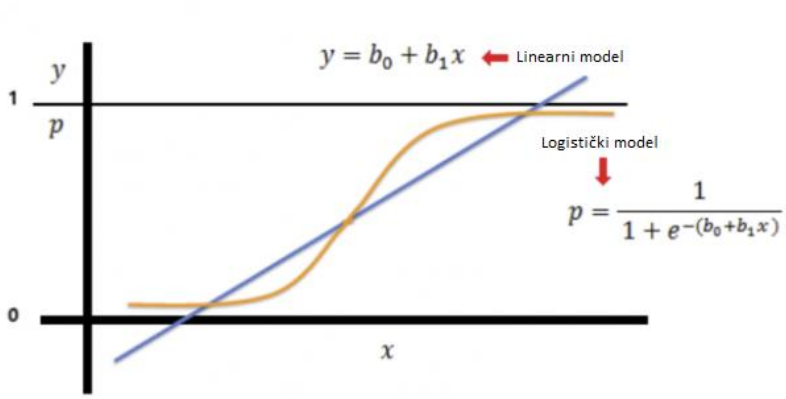
1. Linearnost: odnos između zavisne i nezavisne varijable mora biti linearan. Kada dijagram raspršenja slijedi linearan uzorak, zaključujemo da je linearnost ispunjena.
2. Homoskedastičnost: konstantna varijanca pogrešaka. Odnosi se na 3 uvjeta: (Nau, 2020)
    A. u odnosu na vrijeme kada se promatra vremenska serija
    B. u odnosu na finalna predviđanja
    C. u odnosi na bilo koju neovisnu varijablu.
3. Neovisnost: promatranja su neovisna jedna o drugima. Ne postoji korelacija, odnosno korelacija bi trebala biti približno 0. Ne bi trebalo izgledati da postoji odnos.
4. Normalnost: fiksna vrijednost nezavisne varijable, uvijek zavisna varijabla normalno raspoređena.

Zbog svoje jednostavnosti, brzine treniranja i vrlo razumljive matematičke formule ova vrsta regresija je primjenjiva na različitim područjima poslovnog okruženja, od medicine do društvenih znanosti i poslovanja. Iako ova vrsta regresije ima veliku primjenu, ima i svoje nedostatke. Linearna regresija je jako osjetljiva na odstupanja. Pretpostavlja da uvijek postoji linearan odnos između varijabli, što se u praksi pokazalo da nije uvijek točno. (Rout, 2020)

4.3. Logistička regresija

Logistička regresija jedan je od popularnijih algoritama nadziranog strojnog učenja. Logistička regresija je vrsta statističke metode koja se koristi prilikom izgradnje modela strojnog učenja. Logistička regresija, poznata i kao logistička funkcija, sadrži vrijednosti između nula i jedan.

Logistička regresija dobiva se pomoću takozvane sigmoidne funkcije, također poznate kao logistička funkcija. Sigmoidna funkcija dobiva oblik slova „S“, upravo zbog toga logistička funkcije ne poprima oblik pravca kao kod slučaja linearne funkcije. (Vijay Raghavan, 2016)

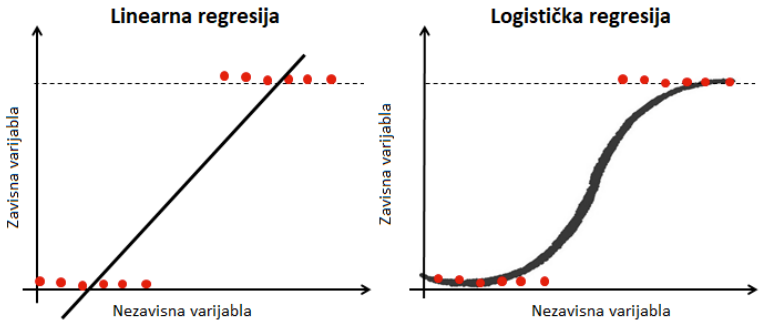


Slika 8. Logistička regresija

Izvor: https://vitalflux.com/linear-vs-logistic-regression-differences-examples/

Na Slici 9. imamo prikazanu logističku funkciju baziranu na sigmoidnoj funkciji. Pomoću te slike možemo vidjeti da je vrijednost logističke funkcije između početka i kraja između 0 i 1. Ako bi ovaj model dao vrijednost 0.5, rezultat modela bio bi 1 ili „točno“, dok ispod 0.5 rezultat glasi 0 ili „netočno“.

Na Slici 10. prikazana je grafička usporedba između linearne i logističke regresije.



Slika 9. Usporedba između linearne i logističke regresije

Izvor: https://www.analyticsvidhya.com/blog/2020/12/beginners-take-how-logistic-regression-is-related-to-linear-regression/

Ova regresija ima svoje sličnosti s linearnom regresijom, dok se linearna regresija fokusira na predviđanje vrijednosti kontinuiranih varijabli, logistička regresija najviše se primjenjuje pri rješavanju klasifikacijskog problema. Koristi se kod binarne klasifikacije za predikciju binarnih vrijednosti poput (0 ili 1) ili točno i netočno. Ova vrsta algoritma također se može koristiti kod višerazredne klasifikacije. (Kumar, 2022)

4.4. Algoritam K-najbližih susjeda

KNN algoritam jedan je od najpoznatijih algoritama strojnog učenja pri predviđanju podataka. „Klasifikator k-najbližih susjeda jednostavna je, ali učinkovita nadaleko poznata metoda u rudarenju podataka i strojnom učenju.“ (Maillo, Ramirez, Triguero, & Herrera, (Maillo, 2017))

Ovaj algoritam prvi se put pojavio 1951. godine kada je bio korišten za zadatak klasifikacije uzorka. Algoritam se od 1970-ih godina koristi kao neparametarska tehnika pri statističkoj procjeni i prepoznavanju uzorka.

K pokušava predvidjeti svaku ispravnu klasu svake podatkovne točke, odnosno ovaj algoritam radi na pretpostavci da svaka podatkovna točka koja je blizu jedna drugoj pripada istoj klasi. Drugim riječima, ovo je algoritam za kvalificiranje podatkovnih točaka na temelju sličnosti. (Christopher, 2021)

KNN algoritam koristi se za regresiju i za kvalifikaciju, obično se koristi kao klasifikacijski algoritam, radeći na pretpostavci da se slične točke mogu naći jedna blizu druge. Ovaj algoritam pokušava predvidjeti vrijednost novih točaka podataka pomoću sličnosti značajki.

Formula KNN algoritma glasi:

$$d=\sqrt{((x2-x1)^2+(y2-y1)^2)}$$

$$d(x, x') = \sqrt{(x_1 - x'_1)^2 + \ldots + (x_n - x'_n)^2}$$

Ova formula nam služi prilikom određivanja udaljenosti između dvije točke. Da bismo bolje objasnili ovu formulu, koristit ćemo sljedeći primjer na temelju udaljenosti između dvije točke.

Podaci zadatka: (Anon., 2021)

Tablica 1. Podaci zadatka

Ime	Broj godina	Spol	Vrsta sporta
Ajay	32	0	Nogomet
Mark	40	0	Ništa
Sara	16	1	Kriket
Zaira	34	1	Kriket
Sachin	55	0	Ništa
Rahul	40	0	Kriket
Pooja	20	1	Ništa
Smith	15	0	Kriket
Laxmi	55	1	Nogomet
Michael	15	0	Nogomet

Ako želimo izračunati udaljenost između Ajaya i Angeline, unose se podaci u sljedeću formulu te dobivamo rezultat:

$d=\sqrt{((broj\ godina2-broj\ godina1)^2+(spol2-spol1)^2)}$

$d=\sqrt{((5-32)^2+(1-0)^2)}$

d=√729+1

d=27.02

Dobivamo podatak 27.02 koji predstavlja našu udaljenost.

Ostali rezultati zadatka:

Tablica 2. Podaci zadatka

Ajay	27.02
Ocjena	35.01
Sara	11.00
Zaira	9.00
Sachin	50.01
Rahul	35.01
Pooja	15.00
Smith	10.00
Laxmi	50,00
Michael	10.05

Zbog toga što je vrijednost k za Angelinu 3, zaključujemo da su Angelini najbliži Zaira, Smith i Michael.

Prednost ovog algoritma je njegova jednostavnost i lakoća implementacije. Nema obuke i dodavanjem novih podataka model se prilagođava te nema potrebe za ponovnim pokretanjem modela. Najveća prednost ovog modela je njegova mogućnost kroz primjenu u klasifikaciji i regresiji. Zbog svog načina rada na temelju bilježenja pozicija svih instanci dolazimo do njegove mane, a to je sporost modela. Povećanjem broja podataka KNN model postaje sve sporiji. Model se muči kada radi s velikim skupovima podataka. (Anon., 2019)

4.5. Neuronske mreže

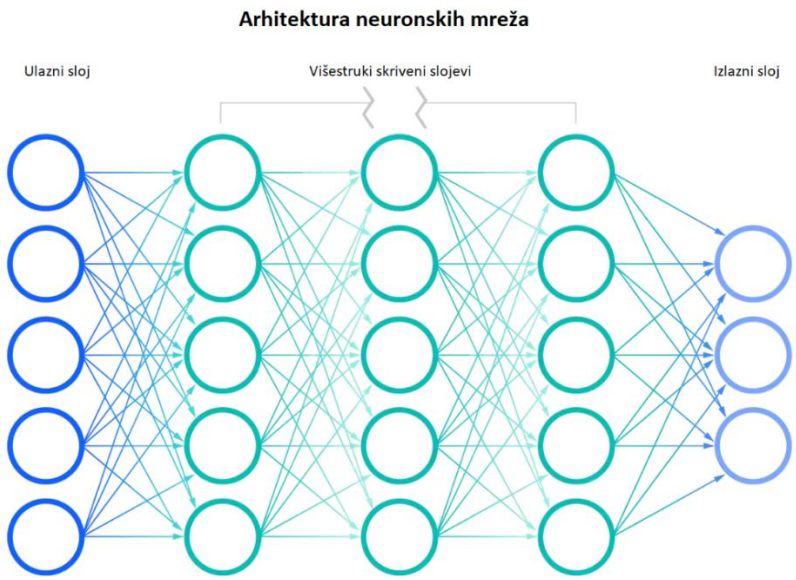
Neuronska mreža je metoda strojnog učenja koja je inspirirana ljudskim mozgom. Neuronske mreže smatraju se najpouzdanijom metodom strojnog učenja. Ova metoda koristi grupu povezanih jedinica, takozvanih neurona koji su organizirani u slojeve i koji nalikuju ljudskom mozgu. Upravo zbog toga metoda neuronskih mreža rješava složenije probleme s većom točnošću od prijašnjih metoda koje smo spomenuli u prethodnim poglavljima.

Ova vrsta strojnog učenje funkcionira na sljedeći način. Kao što ljudski mozak funkcionira pomoću moždanih stanica, takozvanih neurona koji tvore složenu međusobno povezanu mrežu i komuniciraju međusobno, na sličan način funkcionira i ova metoda strojnog učenja. Neuronska mreža kod strojnog učenja sastoji se od takozvanih čvorova ili umjetnih neurona te oni zajedno tvore računalni program namijenjen za rješavanje matematičkih problema. (Lončarić, n.d.), (Anon., n.d.)

Arhitektura neuronskih mreža sastoji se od 3 sloja: (Anon., n.d.)

- Ulazni sloj: služi za primanje podataka, obradu, analiziranje, te ih onda kategorizira i prosljeđuje na sljedeći sloj.
- Skriveni sloj: prima podatke od ulaznog sloja ili od drugih skrivenih slojeva. Umjetne neuronske mreže poznate su po tome što imaju jako puno skrivenih slojeva, ti slojevi primaju i analiziraju podatke od drugih skrivenih slojeva. Na kraju obrade tih podataka prosljeđuju ih na sljedeći sloj, tj. izlazni sloj.
- Izlazni sloj: posljednji je sloj u arhitekturi umjetne neuronske mreže. Izlazni sloj ove metode strojnog učenja razlikuje se od ostalih slojeva zbog toga što može imati više izlaznih čvorova. Više izlaznih čvorova omogućava ovoj metodi da riješi problem klasifikacije s više klasa.

Na Slici 11. prikazana je arhitektura umjetnih neuronskih mreža.



Slika 10. Arhitektura neuronske mreže

Izvor: https://www.ibm.com/cloud/learn/neural-networks

Neuronske mreže mogu se podijeliti u sljedeće skupine: (Nduati, 2020)

- Perceptron: jedna od najranijih i najjednostavnijih vrsta neuronskih mreža. Ovaj model razdvaja podatke u dvije različite klasifikacije. Perceptron se koristi za složenije probleme kao što je prepoznavanje glasa. Mana ovog modela je to što zbog njegove složenosti dolazi do dugotrajne obrade podataka.
- Umjetne neuronske mreže unaprijed (*Feed Forward Neural Network*): podaci se kreću u jednom smjeru između ulaznih i izlaznih čvorova. Podaci se kreću unaprijed kroz čvorove i nemaju mogućnost vraćanja unatrag kroz iste čvorove. Ovo čini ovaj model vrlo jednostavnim. Najčešća primjena ovog modela je za pojednostavljene primjere klasifikacije.
- Rekurentna neuronska mreža (engl. *Recurrent Neural Network*): ova vrsta neuronskih mreža koristi podatke vremenskih serija. Rekurentna mreža ima mogućnost pohrane prošlih informacija, kako bi mogli upotrijebiti prošle podatke. Ova vrsta najviše se koristi za prepoznavanje govora, jedan od najpoznatijih primjera je Appleov pomoćni asistent Siri.
- Modularna neuronska mreža (engl. *Modular Neural Network*): sastoji se od niza neovisnih mreža koje rade pojedinačno. Svaka od tih mreža ima zasebne ulaze za rješavanje vlastitih zadataka koji se na kraju koriste za rješavanje cijelog problema

mreže. Prednost ovog modela je što se procesi dijele na nezavisne komponente. Time se povećava brzina računanja i složeni problemi sustava rješavaju se na brži način.

- Neuronska mreža radijalne baze (engl. *Radal Basis Function Neural Network*): ova vrsta neuronskih mreža veoma se razlikuje u arhitekturi od ostalih neuronskih mreža. Radijalna neuronska mreža sastoji se od ulaznog, skrivenog i izlaznog sloja. Ova vrsta mreže ima i može imati samo jedan skriveni sloj. Sve računanje odvija se u skrivenom sloju, ulazni sloj služi samo za primanje podataka.

Neke od prednosti neuronskih mreža su: pohranjivanje informacija na cijeloj mreži, a ne samo u bazi podataka (nestanak nekih informacija neće spriječiti funkcioniranje mreže); sposobnost rada s nepotpunim znanjem; podaci mogu proizvesti izlaz s nepotpunim informacijama; sposobnost učenja i donošenja odluka; sposobnost paralelne obrade. Neuronske mreže imaju sposobnost obavljanja više poslova u isto vrijeme. Dok ova metoda strojnog učenja ima većinom pozitivne značajke, moramo spomenuti i one negativne. Ovisna je o hardveru, umjetne neuronske mreže zahtijevaju procesore s paralelnom procesorskom snagom, zbog toga način rada ove metode uvelike ovisi o našem računalu. Nije strogo definirana struktura neuronskih mreža, prilikom odabira pravilne strukture često je presudno iskustvo istraživača. (Mijwil, 2017)

U današnje vrijeme neuronske mreže sve se više razvijaju. Prednosti neuronskih mreža sve se više povećavaju, dok se njihove mane svakim danom sve više eliminiraju. Korištenje ovih mreža sve se više povećava, te one postaju dio našeg života, a i naša budućnost.

5. Alati za izgradnju modela strojnog učenja

Prilikom početka izrade programa za strojno učenje bitno je da odaberemo pravi softver. Postoje mnogi softveri koji imaju mogućnost izrade modela za strojno učenje, moramo znati koji softver najviše odgovara našim potrebama. Moramo pripaziti na sljedeće kriterije prilikom odabira softvera za izradu programa: (Boog, 2022)

- Korisničko sučelje: jednostavnost i preglednost programa, jesu li podaci prikazani tako da ih možemo razumjeti.
- Upotrebljivost: mogućnost tutoriala za lakše razumijevanje programa. Mogućnost programa da radi s ostalim programskim jezicima.
- Integracija: mogućnost uvoza raznih biblioteka kao što su Java, Hadoop, Keras, Pytorch i Scikit-learn.
- Cijena: je li naš program besplatan za upotrebu za neke pogodnosti, ili moramo platiti cijelu verziju programa.

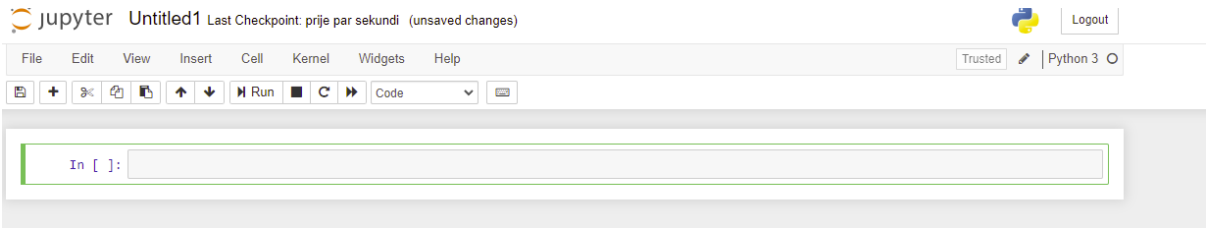
Neki od najpoznatijih softvera za strojno učenje su: IBM-ov paket strojnog učenja otvorenog koda, Microsoftov Azure program za strojno učenje, te ukoliko je riječ o malim tvrtkama ili pojedincima koji žele tek početi raditi sa strojnim učenjem, veoma dobar izbor je softver Anaconda.

5.1. Python

Python je programski jezik koji se koristi za izradu *web*-stranica i softvera pomoću strojnog učenja. Python je jedan od besplatnih jezika opće namjene. Zbog svoje raznolike primjene, jedan je od najpopularnijih programskih jezika današnjice. Primjena Pythona: (Anon., 2022)

- analiza podataka i strojno učenje
- *web*-razvoj
- automatizacija ili skriptiranje
- testiranje softvera i izrada prototipa
- svakodnevni zadaci.

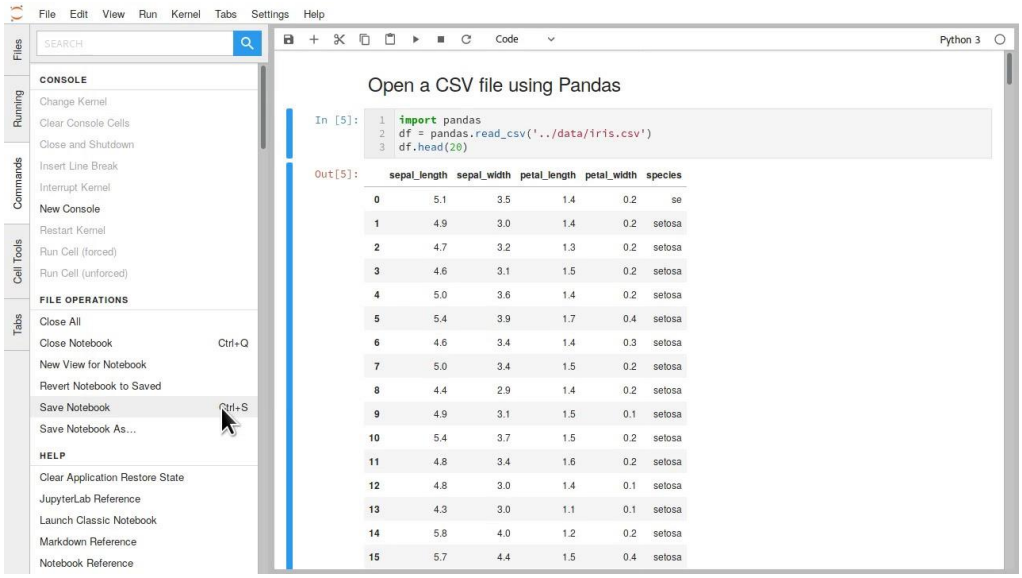
Neka od najpopularnijih okružja Pythona su JupyterLab i Notebook. Jupyter Notebook i Lab nisu uključeni u Python, pa ih je potrebno instalirati. Jedan od programa koji nam nudi Jupyter Notebook je Anaconda. Slika 12. prikazuje izgled Jupyter Notebooka pomoću Anaconde.



Slika 11. Jupiter Notebook

JupyterLab je *web*-aplikacija otvorenog koda koja je dizajnirana na primjeru sučelja Jupyter Notebooka. Ova nova verzija je zapravo samostalna *web*-aplikacija koja povezuje Python okruženje s nekoliko Python biblioteka. Prednosti Jupyter okruženja: (Das, 2020)

- Sve pogodnosti su na jednom mjestu: program koji kombinira kod, tekst, slike i videozapise u jedan dokument pomoću jednog sučelja.
- Više platformi: moguće ga je instalirati na Windowsu, Linuxu i macOS.
- Kopiranje i pomicanje ćelija: JupyterLab nam daje mogućnost laganog povlačenja ćelija. Povlačenje ćelija olakšava posao jer možemo jednostavno spustiti ćelije na mjesta gdje su nam potrebne. Također nam omogućuje da kopiramo ćelije iz jedne bilježnice u drugu, što smanjuje vrijeme potrebno za pisanje algoritama.
- Neovisnost o jeziku: mogućnost obrade na nekoliko jezika i pretvaranja u nekoliko formata datoteka kao što su HTML i PDF.
- Interakcije s kodom: koristi paket *ipywidgets* koji omogućuje interaktivnost kod podataka. Zbog toga korisnici mogu uređivati kod i ponovno ga testirati.



Slika 12. Sučelje JupyterLaba

Izvor: https://jupyterlab.readthedocs.io/en/stable/user/interface.html

5.1.1. Python biblioteke

Python ima mnogo biblioteka koje programerima olakšavaju pisanje koda. Možemo primijeniti već postojeće funkcionalnosti bez pisanja koda. Neke od najčešćih biblioteka su scikit-learn, matplotib, numpy, pandas. (Milošević, 2018)

Scikit-learn: ova biblioteka izgrađena je na temelju Numpy i Scypy biblioteka, te se koristi za rad sa složenim podacima. Scikit-learn sadrži mnogo funkcija za strojno učenje uključujući regresiju i klasifikaciju. Radi na način da dobro funkcionira s Numpy i Scipy bibliotekama.

Numpy: kada je u pitanju Python programski jezik, jedna od najbitnijih biblioteka je Numpy biblioteka. Ovu biblioteku koristimo kada radimo s matematičkim operacijama koje uključuju vektore ili matrice, odnosno rad s jednodimenzionalnim ili višedimenzionalnim nizovima.

Matplotblib: služi za crtanje grafova i dijagrama odnosno vizualizaciju podataka. Ova biblioteka je veoma bitna jer omogućava da lakše vidimo podatke, uvijek je lakše donositi zaključke kada vidimo podatke u grafovima nego u linijama teksta.

Pandas: ova biblioteka namijenjena je za analizu i manipulaciju podacima. Sastoji se od dvije strukture podataka, serije i podatkovnih okvira. Podatkovni okviri predstavljaju dvodimenzionalnu strukturu podataka kao dvodimenzionalne tablice koje izgledaju slično kao u Excelu. Serije su jednodimenzionalni nizovi podataka poput stupca u tablici. Popularnost ove biblioteke među programerima dolazi iz mogućnosti korištenja prednosti raznih biblioteka poput Numpy, Scipy, Matplotlib i raznih drugih.

Za potrebe projektnog zadatka koristit će se neke od ovih biblioteka. Sam izgled primjene biblioteka i izgled algoritma prikazat će se u sljedećem poglavlju.

6. Opis istraživanja i rezultati istraživanja

U sljedećim poglavljima provedeno je istraživanje baze podataka koja je preuzeta s repozitorija Kaggle. Izgrađena su tri modela strojnog učenja: logistička regresija, kNN regresija i neuronske mreže. Radi se o višerazrednoj klasifikaciji.

6.1. Opis podataka

Baza podataka sastoji se od podataka vezanih za specifikacije računala. Baza podataka se sastoji od 1304 opservacije i 14 varijabli. Izgledi varijabli prikazani su u Tablici 3.

Tablica 3. Varijable baze podataka

Ime varijable	Opis varijable
company	Tvrtka
TypeName	Vrsta računala
Inches	Veličina ekrana
Ram	Radna memorija
OpSys	Operativni sustav
Weight	Težina
Resolution	Rezolucija
TouchScreen	Ekran na dodir
CpuFreq	Frekvencija procesora
SSDstorage	SSD pohrana
HDDstorage	Hard disk pohrana
OtherStorage	Ostala pohrana
GpuManufacturer	Proizvođač grafičke kartice
PriceCategory	Cjenovna kategorija

Baza podataka nalazi se u Excell datoteci, da bi ju prikazali u programu moramo koristiti sljedeću naredbu:

```
PC = pd.read_csv('laptop_price.csv', encoding="latin-1")
```

Slika 13. Učitavanje baze podataka

Izvor: Autor

Nakon što je baza uspješno učitana, možemo ju prikazati pomoću sljedeće naredbe koja je prikazana Slikom 14.

```
PC.head()
```

Slika 14. Naredba za prikaz podataka

Izvor: Autor

Nakon naredbe PC.head() rezultati su prikazani Slikom 15.

	Company	TypeName	Inches	Ram	OpSys	Weight	Resolution	TouchScreen	CpuFreq	SSDstorage	HDDstorage	OtherStorage	GpuManufacturer	PriceCategory
0	16	0	13.3	8	4	1.37	4096000	0.0	2.3	128.0	0.0	0.0	1	2
1	16	0	13.3	8	4	1.34	1296000	0.0	1.8	0.0	0.0	128.0	1	1
2	14	1	15.6	8	6	1.86	2073600	0.0	2.5	256.0	0.0	0.0	1	0
3	16	0	15.4	16	4	1.83	5184000	0.0	2.7	512.0	0.0	0.0	0	2
4	16	0	13.3	8	4	1.37	4096000	0.0	3.1	256.0	0.0	0.0	1	2

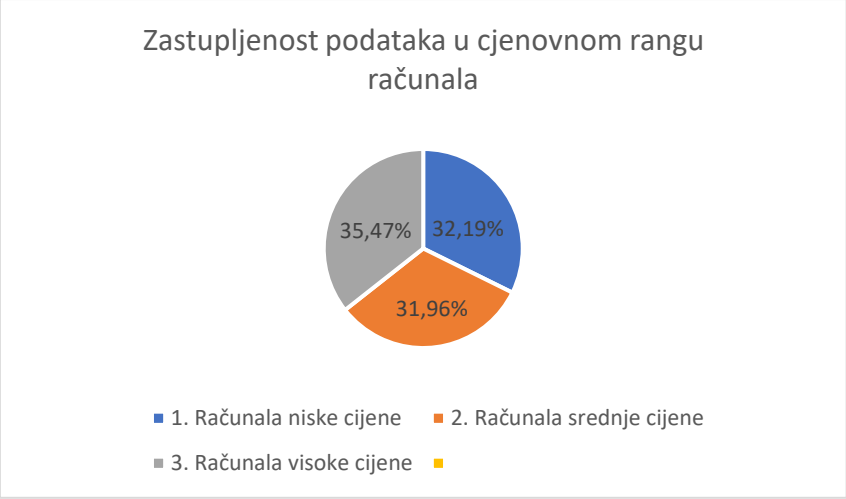
Slika 15. Sadržaj baze podataka

Izvor: Autor

Kao što možemo vidjeti iz Slike 15, varijabla cijena predstavlja kategorijalnu varijablu, podjela te varijable izgleda ovako:

- 0 – računala niske cijene
- 1 – računala srednje cijene
- 2 – računala visoke cijene.

Zastupljenost podataka u pojedinoj kategoriji cijena nije potpuno ravnomjerno podijeljena. No, ne postoje ni velike razlike, tako da je ova baza podataka približno balansirana te će predstavljati podlogu za izgradnju modela za predviđanje. Na Slici 16. prikazana je podjela podataka po cjenovnim kategorijama ove baze podataka.



Slika 16. Zastupljenost podataka u cjenovnom rangu

Izvor: autor

Da bi izvukli deskriptivnu statistiku naše baze podataka koristimo naredbu describe(). Pomoću naredbe describe() možemo izvući tablicu deskriptivne statistike iz cijele baze podataka ili samo iz pojedinih varijabli. Na Slici 17. prikazana je deskriptivna statistika cijele baze podataka.

	Company	TypeName	Inches	Ram	OpSys	Weight	Resolution	TouchScreen	CpuFreq	SSDstorage	HDDstorage	OtherStorage	GpuManufacturer	PriceCategory
count	1303.000000	1303.000000	1303.000000	1303.000000	1303.000000	1303.000000	1.303000e+03	1303.000000	1303.000000	1303.000000	1303.000000	1303.000000	1303.000000	1303.000000
mean	9.272448	1.621642	15.017191	8.382195	1.732924	2.038734	2.168807e+06	0.147352	2.298772	182.750580	413.016113	13.387568	1.170376	1.033001
std	4.673707	1.449209	1.426304	5.084665	1.730955	0.665475	1.391292e+06	0.354593	0.506340	184.821381	514.196838	96.977631	0.647681	0.823792
min	0.000000	0.000000	10.100000	2.000000	0.000000	0.690000	1.049088e+06	0.000000	0.900000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	4.000000	1.000000	14.000000	4.000000	1.000000	1.500000	1.440000e+06	0.000000	2.000000	0.000000	0.000000	0.000000	1.000000	0.000000
50%	7.000000	1.000000	15.600000	8.000000	1.000000	2.040000	2.073600e+06	0.000000	2.500000	256.000000	0.000000	0.000000	1.000000	1.000000
75%	14.000000	3.000000	15.600000	8.000000	1.000000	2.300000	2.073600e+06	0.000000	2.700000	256.000000	1000.000000	0.000000	2.000000	2.000000
max	18.000000	5.000000	18.400000	64.000000	8.000000	4.700000	8.294400e+06	1.000000	3.600000	1000.000000	2000.000000	1000.000000	3.000000	2.000000

Slika 17. Deskriptivna statistika

Izvor: autor

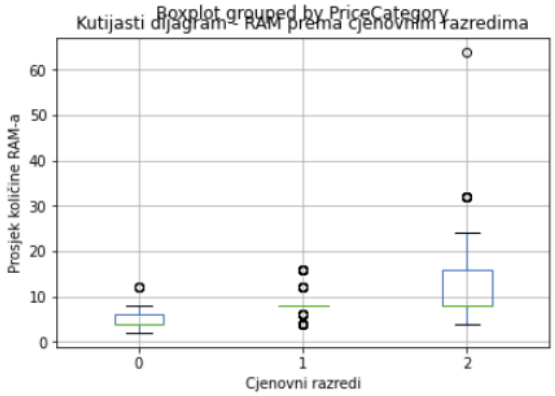
Pomoću deskriptivne statistike vidimo minimum, maksimum, standardnu devijaciju, prosjek i medijane. Da bi lakše vidjeli odnose između varijabli koristit će se kutijasti dijagram. Pomoću kutijastog dijagrama možemo prikazati odnos između minimuma, maksimuma, medijana i donjeg ili gornjeg kvartila. Kod koji koristimo da bi prikazali kutijasti dijagram je sljedeći:

```
PC.boxplot(column="CpuFreq", by="PriceCategory")
plt.title("Kutijasti dijagram - CPU prema cjenovnim razredima")
plt.xlabel("Cjenovni razredi")
plt.ylabel("CPU")
```

Slika 18. Kreiranje kutijastog dijagrama

Izvor: autor

Nakon upisane naredbe sa Slike 18., program pomoću matplotlib biblioteke ispisuje sljedeći dijagram prikazan na Slici 19.



Slika 19. Kutijasti dijagram RAM-a prema cjenovnim razredima

Izvor: autor

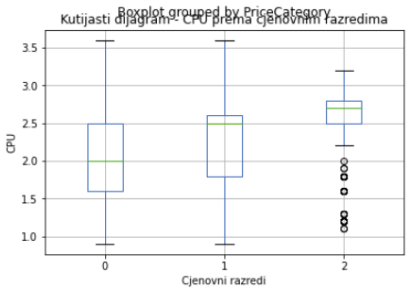
Kutijasti dijagram na Slici 19. prikazuje razlike između aritmetičkih sredina varijable RAM-a prema cjenovnim razredima koja predstavlja izlaznu varijablu, odnosno prosjek količine RAM-a. Pomoću dijagrama možemo bolje uočiti pokazatelje te pojedine vrijednosti koje možda nećemo uočiti na tablici. Imamo mogućnost prikaza i ostalih varijabli ako se modificira kod. Na Slici 20. je kod čiji je bio cilj prikazati odnos između varijable CPU i izlazne varijable cjenovnih razreda.

```
PC.boxplot(column="CpuFreq", by="PriceCategory")
plt.title("Kutijasti dijagram - CPU prema cjenovnim razredima")
plt.xlabel("Cjenovni razredi")
plt.ylabel("CPU")
```

Slika 20. Kreiranje kutijastog dijagrama

Izvor: autor

Slika 21. prikazuje kutijasti dijagram koji se dobije nakon upisa koda kao što je na Slici 20.



Slika 21. Kutijasti dijagram CPU-a prema cjenovnim razredima

Izvor: Autor

Kutijasti dijagram na Slici 21. prikazuje razliku aritmetičke sredine varijable CPU i cjenovnih razreda, koja je u ovom kontekstu izlazna varijabla. Varijabla CPU je numerička varijabla koja se sastoji od podataka koji predstavljaju brzine procesora (brzine su u GHz). Iz kutijastog dijagrama možemo zaključiti sljedeće kod računala s niskom cijenom: raspon brzine procesora kreće se od 1.0 GHz do 3.6GHz; crta zelenom bojom označava medijan, medijan označava srednju točku podataka te dijeli naše podatke na dva dijela; polovica podataka je veća ili jednaka promatranoj vrijednosti dok je druga polovica manja ili jednaka promatranoj vrijednosti; konkretno na ovom grafu možemo zaključiti da kod jeftinih računala (prikazana brojem 0) naša srednja vrijednost iznosi 2.0, dok donji kvartil iznosi 1.6, a gornji kvartil 2.5. Ovo bi interpretirali na sljedeći način:

- Q1 – 25 % podataka statističkog skupa ima vrijednost 1.6 ili manju, dok 75 % podataka ima vrijednost 1.6. ili veću.
- Q3 – 75 % podataka statističkog skupa ima vrijednost 2.5 ili manju, dok 25 % podataka ima vrijednost 2.5 ili veću.

Ako želimo vidjeti razlike između kategorija, možemo to prikazati pomoću naredbe crosstab(). Pomoću crosstab() naredbe program će nam ispisati sve opservacije iz pojedinih varijabli koje pripadaju drugoj varijabli. Slika 22. prikazuje primjer naredbe crosstab() na primjeru vezanom za kategorije cijena računala i ako naše računalo ima ekran osjetljiv na dodir.

```
pd.crosstab(PC["PriceCategory"], PC["TouchScreen"])
```

TouchScreen	0.0	1.0
PriceCategory		
0	384	37
1	376	42
2	351	113

Slika 22. Kod i kontigencijska tablica varijabli PriceCategory i TouchScreen

Izvor: autor

Iz tablice je vidljivo da su podaci kod monitora koji nemaju ekran s dodirom približno jednako raspršeni kod svih cjenovnih rangova, dok kod monitora koji su osjetljivi na dodir postoji velika razlika kod skupljih računala. Slikom 23. prikazana je tablica gdje su se uspoređivale varijable kategorija cijena i frekvencije procesora.

```
pd.crosstab(PC["PriceCategory"], PC["CpuFreq"])
```

CpuFreq	0.90	1.00	1.10	1.20	1.30	1.44	1.50	1.60	1.80	1.90	...	2.40	2.50	2.60	2.70	2.80	2.90	3.00	3.10	3.20	3.60
PriceCategory																					
0	1	1	48	0	0	12	10	72	6	0	...	31	88	0	18	0	5	14	0	0	3
1	3	0	3	3	1	0	0	52	44	0	...	12	131	18	60	39	0	0	0	0	2
2	0	0	2	12	5	0	0	10	28	2	...	9	74	58	88	126	16	5	3	1	0

Slika 23. Kod i kontigencijska tablica varijabli PriceCategory i CpuFreq

Izvor: autor

Pomoću ove tablice prikazano je koji procesor se koristi kod određenih cjenovnih rangova. Tablica je drugačije strukturirana nego prijašnja, te je moguće bolje vidjeti određenu frekvenciju i njezinu primjenu kod drugačijih cjenovnih rangova. Sljedeći kod koji je bio korišten u ovom istraživanju je kod za klasifikaciju podataka s neskalarnim vrijednostima. Primjer koda prikazan je na Slici 24.

```
X = PC.iloc[:, :-1]
y = PC.iloc[:, -1]
print(X.shape)
print(y.shape)
X.head()
```

```
(1303, 13)
(1303,)
```

	Company	TypeName	Inches	Ram	OpSys	Weight	Resolution	TouchScreen	CpuFreq	SSDstorage	HDDstorage	OtherStorage	GpuManufacturer
0	16	0	13.3	8	4	1.37	4096000	0.0	2.3	128.0	0.0	0.0	1
1	16	0	13.3	8	4	1.34	1296000	0.0	1.8	0.0	0.0	128.0	1
2	14	1	15.6	8	6	1.86	2073600	0.0	2.5	256.0	0.0	0.0	1
3	16	0	15.4	16	4	1.83	5184000	0.0	2.7	512.0	0.0	0.0	0
4	16	0	13.3	8	4	1.37	4096000	0.0	3.1	256.0	0.0	0.0	1

Slika 24. Kod i tablica za klasifikaciju podataka s neskalarnim vrijednostima

Izvor: autor

6.2. Izgradnja modela

Da bismo izgradili model, potrebno je učitati klasifikatore koji će se koristiti prilikom predviđanja. Koristit će se 3 klasifikatora koja učitavamo iz scikit-learn biblioteke: logistička regresija, kNN model i neuronske mreže. Na Slici 25. prikazan je kod za učitavanje potrebnih klasifikatora.

```
from sklearn.model_selection import KFold

from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.neural_network import MLPClassifier
```

Slika 25. Učitavanje klasifikatora

Izvor: autor

Radi postizanja veće točnosti modela korištena je slojevita unakrsna validacija. Cilj unakrsne validacije je koristiti samo podatke o obuci. Cross_val_score procjenjuje očekivanu točnost modela na podacima izvan treninga dobivenim iz istog procesa. Prednost unakrsne validacije je u tome što ne moramo odvajati podatke i možemo trenirati model na svim dostupnim podacima. Na Slici 26. prikazan je kod za unakrsnu validaciju, korištena je desetorostruka validacija tako da svaki sloj sadrži 130 opservacija.

```
from sklearn.model_selection import cross_val_score
kfold = KFold(n_splits=10)
```

Slika 26. Unakrsna validacija, podjela na 10 slojeva

Izvor: autor

Nakon ovoga koda imamo sve potrebno da krenemo izraditi model za predviđanje. Na Slici 27. možemo vidjeti kod za kreiranje predviđanja pomoću logističke regresije, navedenim kodom dobivamo točnost po pojedinom sloju i prosječnu točnost između svih 10 slojeva.

```
logreg = LogisticRegression(multi_class="auto", solver="lbfgs")
scores = cross_val_score(logreg, X, y, cv=kfold)
print("Cross validation scores {}".format(scores))
avg = format(scores.mean())
print(f"Average: {avg}")
```

Slika 27. Kod za model strojnog učenja pomoću logičke regresije

Izvor: autor

Nakon toga je kreiran model kNN (k-najbližih susjeda). Na Slici 28. prikazan je kod za izgradnju kNN modela. Prvo se izračunava optimalni k, koji se mora uzeti u obzir prilikom izgradnje ovog modela. Poslije toga se upisuje kod za predviđanje pomoću kNN-a, s time da je uzet u obzir najoptimalniji k.

```
from sklearn.model_selection import GridSearchCV

knn = KNeighborsClassifier()
paramGrid = {"n_neighbors":np.arange(1,25)}
knnGscv = GridSearchCV(knn, paramGrid, cv=10)
knnGscv.fit(X, y)
knnGscv.best_params_
```

```
{'n_neighbors': 3}
```

```
knn = KNeighborsClassifier(n_neighbors=3)
scoresN = cross_val_score(knn, X, y, cv=kfold)
print(f"Cross-validation scores {scoresN}")
print(f"Prosječni score {np.mean(scoresN)}")
```

Slika 28. Kod za optimalni k i za izgradnju modela kNN

Izvor: autor

Iz Slike 28. vidimo da je naš optimalni k 3, nakon toga uvrštavamo taj broj u kod prilikom izgradnje kNN modela. Kao i u logističkoj regresiju i u ovom modelu dobivamo točnost po pojedinom sloju od njih ukupno 10 i također na kraju izračunavamo prosječnu vrijednost između svih 10 slojeva. Nakon kNN modela gradimo posljednji model u ovom projektu, a to su neuronske mreže. Na Slici 29. prikazan je model neuronskih mreža.

```
nn = MLPClassifier(hidden_layer_sizes=(100,), random_state=0, solver="adam", learning_rate="adaptive", max_iter=100000000000)
scoresNN = cross_val_score(nn, X, y, cv=kfold)
print(f"Cross-validation scores: {scoresNN}")
print(f"Prosječni score: {np.mean(scoresNN)}")
```

Slika 29. Kod za kreiranje modela strojnog učenja pomoću neuronskih mreža

Izvor: autor

Kao što je prikazano na Slici 29., izgrađen je model strojnog učenja pomoću neuronskih mreža. Skriveni sloj neuronske mreže sastoji se do 100 elemenata, veličinu elemenata u skrivenom sloju definiramo kodom: hidden_layer_sizes=(100,).

6.2.1. Rezultati modela

Nakon upisa svih kodova potrebnih za izgradnju modela logičke regresije, k-najbližih susjeda i modela neuronskih mreža dobivamo sljedeće rezultate prikazane na Tablici 4.:

Tablica 4. Točnost modela prema neskalarnim podacima

Izvor: autor

	kNN	Logistička regresija	Neuronske mreže
Sloj 1	0.725	0.381	0.427
Sloj 2	0.702	0.496	0.396
Sloj 3	0.809	0.450	0.343
Sloj 4	0.807	0.484	0.292
Sloj 5	0.769	0.407	0.392
Sloj 6	0.8	0.446	0.415
Sloj 7	0.692	0.630	0.292
Sloj 8	0.592	0.469	0.392
Sloj 9	0.653	0.507	0.269
Sloj 10	0.746	0.523	0.261
Prosjek	0.729	0.48	0.348

Također je napravljeno testiranje i sa skalarnim vrijednostima. U Tablici 5. prikazani su rezultati samo s neskalarnim vrijednostima.

Tablica 5. Točnosti modela prema skalarnim podacima

Izvor: autor

	kNN	Logistička regresija	Neuronske mreže
Sloj 1	0.687	0.671	0.732
Sloj 2	0.763	0.702	0.809
Sloj 3	0.748	0.694	0.809
Sloj 4	0.738	0.692	0.707
Sloj 5	0.730	0.723	0.761
Sloj 6	0.723	0.676	0.746
Sloj 7	0.776	0.730	0.823
Sloj 8	0.638	0.592	0.646
Sloj 9	0.737	0.638	0.730
Sloj 10	0.738	0.7	0.738
Prosjek	0.728	0.682	0.750

Iz tablica 4. i 5. vidimo rezultate modela. Kada je riječ o neskalarnim vrijednostima, model neuronske mreže ima jako loše rezultate, zato smo koristili podatke sa skalarnim vrijednostima da poboljšamo proces predviđanja. Prosjek predviđanja, koji iznosi 0.348 kod neuronskih mreža, skače na znatno poboljšanje od 0.750. Isto takvo poboljšanje dogodilo se kod logističke regresije, od 0.48 kod neskalarnih vrijednosti naše predviđanje se poboljšava primjenom skalarnih vrijednosti i dobivamo prosjek predikcije od 0.68, dok je kod kNN-a rezultat ostao približno isti. I dalje vidimo da modeli nemaju veliku točnost predviđanja, ali se vidi znatno poboljšanje kada se koriste neskalarni podaci. U konačnici očekivano je da model neuronskih mreža ima najveću prosječnu točnost u ovom modelu. Kada usporedimo rezultate u pojedinačnim slojevima, možemo primijetiti blaga odstupanja u nekim slučajevima. Iz toga zaključujemo da dolazi do problema prilikom predviđanja i treniranja modela i da modeli imaju zapravo neke poteškoće prilikom kategoriziranja podataka i zbog toga dolazi do lošijih rezultata predviđanja. U nastavku rada prikazat će se matrica konfuzije, također će se prikazati krivulja učenja naših modela. Na kraju će se donijeti zaključak i prijedlozi poboljšanja ako se želi koristiti ova baza prilikom daljnjih predviđanja.

6.3. Matrica konfuzije modela

Matrica konfuzije mjeri performanse problema klasifikacije gdje postoje dvije ili više klasa. Matrica konfuzije bitna je za ovu vrstu problema jer možemo vidjeti koliko puta je naš model točno predvidio. Postoje 4 kombinacije prilikom predviđanja rezultata: (towardsdatascience)

- Pravo pozitivan (engl. *True positive*): predviđanje je predvidjelo pozitivnu situaciju i to je točno.
- Pravo negativan (engl. *True negative*): predviđanje je predvidjelo negativnu situaciju i to je točno.
- Lažno pozitivan (engl. *False positive*): predviđanje je predvidjelo pozitivnu situaciju i to nije istina, krivo predviđanje.
- Lažno negativan (engl. *False negative*): Predviđanje je predvidjelo negativnu situaciju i to nije istina, krivo predviđanje.

Tablica matrice konfuzije sastoji se od točnih i predviđenih vrijednosti, na Slici 30. prikazan je kod s kojim će se prikazati matrica konfuzije za podatke dobivene pomoću logističke regresije.

```
print("Matrica konfuzije:\n{}".format(confusion_matrix(y, pred)))
print("Matrica konfucije X_scaled:\n{}".format(confusion_matrix(y, pred_scaled)))
```

Slika 30. Kod za prikazivanje matrice konfuzije logističke regresije

Izvor: autor

Pomoću koda koji je prikazan na Slici 30. program će ispisati tablicu u kojoj će biti prikazani skalarni i neskalarni podaci koji su predviđeni logističkom regresijom.

Tablica 6. Matrica konfuzije logističke regresije

Izvor: autor

Neskalarni podaci	**Predviđene vrijednosti**			
Stvarne vrijednosti		**0**	**1**	**2**
	0	340	62	19
	1	122	184	112
	2	21	94	349
Skalarni podaci	**Predviđene vrijednosti**			
Stvarne vrijednosti		**0**	**1**	**2**
	0	340	62	19
	1	122	184	112
	2	21	94	349

Vidimo da je dijagonala označena plavom bojom jer su sve vrijednosti na dijagonali zapravo najveći brojevi iz pojedinih kategorija. Također vidimo da su u slučaju logističke regresije dobiveni isti rezultati kada se uspoređuju skalarni i neskalarni podaci. Prilikom predviđanja brojeva u tablici korištene su 4 kombinacije koje su navedene ranije u poglavlju: pravo pozitivan, pravo negativan, lažno pozitivan, lažno negativan. Nakon što se dobila matrica konfuzije, sljedeći korak je izračunati preciznost cijelog modela i točnosti po pojedinim kategorijama. Te izračune dobivamo pomoću koda prikazanog na Slici 31.

```
print(classification_report(y, pred))
print("scaled: ")
print(classification_report(y, pred_scaled))
```

Slika 31. Kod za izvještaj predviđanja pomoću logističke regresije

Izvor: autor

Navedeni kod ispisat će dvije tablice, gdje će se prikazati točnost predviđanja pomoću skalarnih i neskalarnih vrijednosti. Tablica 7. prikazat će preciznost, odaziv, F1 i točnost po pojedinim kategorijama koristeći metodu logističke regresije.

Tablica 7. Izvještaj klasifikacije logističke regresije

Izvor: autor

Neskalarni podaci	**Preciznost**	**Odaziv**	**F1**
0	0.70	0.81	0.75
1	0.54	0.44	0.49
2	0.73	0.75	0.74
Točnost			0.67
Makroprosjek	0.66	0.67	0.66
Težinski prosjek	0.66	0.67	0.66
Skalarni podaci	**Preciznost**	**Odaziv**	**F1**
0	0.70	0.81	0.75
1	0.54	0.44	0.49
2	0.73	0.75	0.74
Točnost			0.67
Makroprosjek	0.66	0.67	0.66
Težinski prosjek	0.66	0.67	0.66

F1 pokazatelj je mjera točnosti na skupu podataka, koristi se pri procjeni binarnih klasifikacijskih sustava, tj. pri klasifikaciji podataka u „pozitivne“ i „negativne“. Makroprosjek se računa kao prosjek preciznosti svih klasa, dok se težinski prosjek isto računa prema preciznosti po klasi, ali uzima u obzir broj uzoraka svake klase u podacima. (Khalusova, 2021)

U nastavku rada provedena je i kNN regresija, kojom je dobivena matrica konfuzije prikazana na Slici 32.

```
pred_knn = cross_val_predict(knn, X, y)
pred_knn_scaled = cross_val_predict(knn_scaled, X_scaled, y)
```

```
print(confusion_matrix(y, pred_knn))
print("Scaled:")
print(confusion_matrix(y, pred_knn_scaled))
```

Slika 32. Kreiranje matrice konfuzije pomoću kNN metode

Izvor: autor

Tablica prikazuje izgled matrice konfuzije za kNN model.

Tablica 8. Matrica konfuzije kNN regresije

Izvor: autor

Neskalarni podaci	**Predviđene vrijednosti**			
Stvarne vrijednosti		**0**	**1**	**2**
	0	330	73	18
	1	111	233	74
	2	35	96	333
Skalarni podaci	**Predviđene vrijednosti**			
Stvarne vrijednosti		**0**	**1**	**2**
	0	342	65	14
	1	83	252	83
	2	15	92	357

Iz tablice vidimo da su skalarne vrijednosti dale bolji rezultat. Također vidimo da je ovaj model dao bolje rezultate od logističke regresije, što nije uobičajeno. Nadalje je prikazan izvještaj za podatke korišten metodom k – najbližih susjeda.

Tablica 9. Izvještaj klasifikacije kNN regresije

Izvor: autor

Neskalarni podaci	**Preciznost**	**Odaziv**	**F1**
0	0.69	0.78	0.74
1	0.58	0.56	0.57
2	0.78	0.72	0.75
Točnost			0.69
Makroprosjek	0.69	0.69	0.68
Težinski prosjek	0.69	0.69	0.69
Skalarni podaci	**Preciznost**	**Odaziv**	**F1**
0	0.78	0.81	0.79
1	0.62	0.60	0.61
2	0.79	0.77	0.78
Točnost			0.73
Makroprosjek	0.73	0.73	0.73
Težinski prosjek	0.73	0.73	0.73

Vidimo da model prilikom određivanja visokih cijena ima najveću preciznost. Također, skalirani podaci pokazali su bolje rezultate od neskaliranih. Zadnji model koji još nije odrađen su neuronske mreže. Neuronske mreže trebale bi pokazati najveću točnost, na sljedećim slikama prikazani su kod i tablica za implementaciju konfuzije neuronskih mreža.

```
pred_nn = cross_val_predict(nn, X, y)
pred_nn_scaled = cross_val_predict(nn_scaled, X_scaled, y)

print(confusion_matrix(y, pred_nn))
print("Scaled:")
print(confusion_matrix(y, pred_nn_scaled))
```

Slika 33. Kod za tablicu konfuzije neuronskih mreža

Izvor: autor

Tablica 10. Matrica konfuzije neuronskih mreža

Izvor: autor

Neskalarni podaci	**Predviđene vrijednosti**			
Stvarne vrijednosti		**0**	**1**	**2**
	0	353	63	5
	1	93	250	75
	2	10	87	367
Skalarni podaci	**Predviđene vrijednosti**			
Stvarne vrijednosti		**0**	**1**	**2**
	0	353	63	5
	1	93	250	75
	2	10	87	367

U tablici vidimo da su najtočnija predviđanja u najskupljoj klasi. Kao i u prethodnim modelima i ovdje je prikazan izvještaj za klasifikaciju. Tablica prikazuje klasifikacijski izvještaj neuronskih mreža.

Tablica 11. Izvještaj klasifikacije neuronskih mreža

Izvor: autor

Skalarni podaci	**Preciznost**	**Odaziv**	**F1**
0	0.77	0.84	0.81
1	0.62	0.60	0.61
2	0.82	0.79	0.81
Točnost			0.74
Makroprosjek	0.74	0.74	0.74
Težinski prosjek	0.74	0.74	0.74

Kao što je očekivano, ovaj model dao je najveću preciznost i točnost, što se vidi iz prikazane tablice. U nastavku rada još će biti prikazana krivulja učenja.

6.4. Krivulja učenja

Krivulja učenja nam pruža bolji uvid u performanse našeg modela. Na Slici 34. prikazan je kod crtanja krivulje učenja.

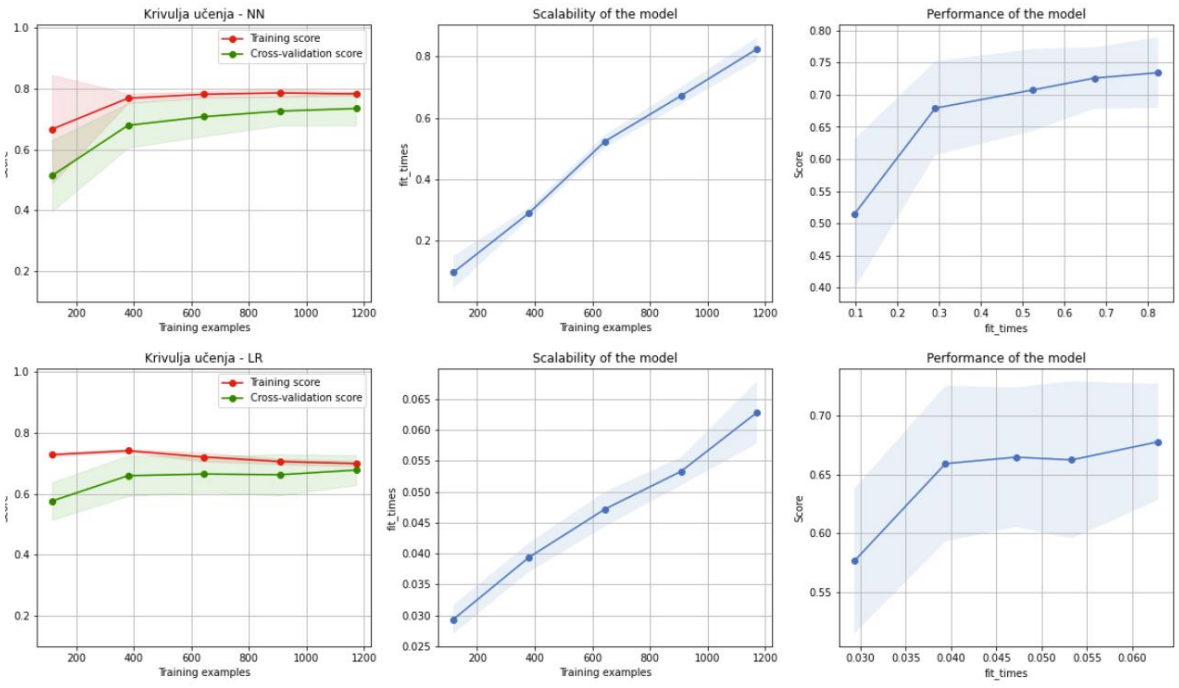
```
title1 = "Krivulja učenja - NN"
cv1 = kfold
estimator1 = MLPClassifier()
plot_learning_curve(estimator1, title1, X_scaled, y, ylim=(0.1, 1.01), cv=cv1, n_jobs=4)

title2 = "Krivulja učenja - LR"
cv2 = kfold
estimator2 = LogisticRegression()
plot_learning_curve(estimator2, title2, X_scaled, y, ylim=(0.1, 1.01), cv=cv2, n_jobs=4)
```

Slika 34.. Izvještaj klasifikacije neuronskih mreža

Izvor: autor

Grafički prikaz krivulje učenja prikazan je na Slici 35.



Slika 35. Grafički prikaz krivulje učenja

Izvor: autor

Krivulja učenja neuronskih mreža pokazuje bolje rezultati od one kod logističke regresije, ali i dalje točnost modela neuronskih mreža nije ona koja se priželjkuje. Kao što se iz grafa vidi, krivulja točnosti na skupu za treniranje nije na razini da bi se moglo sa sigurnošću reći da će donijeti veliku točnost. Kreće se od 0.65 do 0.8, dok je kod uspješnijih predviđanja između 0.9

i 1.0. Dobar znak u ovom slučaju je da CV postepeno raste. Također, ako pogledamo ostale krivulje koje se tiču neuronskih mreža, možemo vidjeti da točnost raste s vremenom. Kod vremena kalkulacije vidimo da je potrebno više vremena za izračun prilikom porasta skupa za treniranje.

Kod logističke regresije nešto je lošija situaciju nego kod neuronskih mreža, što je očekivano, ali rezultati nisu na razini očekivanja. Krivulja točnosti na skupu za treniranje kod logističke regresije ne doseže ni 0.8, zapravo samo po toj krivulji vidimo da će točnost ovog modela biti znatno manja od onoga kod neuronskih mreža. Kada usporedimo ostale krivulje logističke regresije, vidimo da ne pokazuju uspješnost kao kod neuronskih mreža, ali i dalje pokazuju poboljšanje rezultata tijekom određenog vremena, odnosno točnost modela raste nakon više vremena. Kada usporedimo razinu najveće točnosti kod neuronskih mreža, to se događa kod 900 opservacija, dok kod logističke regresije to dolazi već kod 390 opservacija. No, ukoliko gledamo cross-validation metodu, vidimo da je točnost na nižoj razini. Ukoliko treba odlučiti koji će se model koristiti u praksi, uvijek se treba odlučiti za neuronske mreže.

7. Rasprava

Izrađene su tri metode na temelju strojnog učenja. Pomoću metoda k-najbližih susjeda, logističke regresije i neuronske mreže nastojao se izgraditi model strojnog učenja koji će predviđati cijene računala koje su formirane u Python programskom okruženju. Metode funkcioniraju na temelju klasifikacije cijena računala. Metode nisu pokazale očekivanu točnost, ali istraživanje ipak ispunjava očekivanja da će neuronske mreže ostvariti najveću točnost od tri metode strojnog učenja. Ipak, graf krivulje učenja pokazuje da naša neuronska mreža s vremenom sve točnije predviđa.

Ono što je pokazalo jako loše rezultate je logistička regresija. Da bi se pokušali poboljšati rezultati provelo se istraživanje na skalarnim i neskalarnim podacima, ali nije se ostvarilo željeno poboljšanje. Kada pogledamo na krivulju učenja, vidimo da je preciznost relativno loša, ne kreće se ni blizu željenih rezultata. Kod logističke regresije ukoliko ciljne oznake nemaju linearnu korelaciju sa svojstvima, dolazi do pogrešaka. Zbog toga često dolazi do loše točnosti modela, iako je model imao relativno jednostavan skup podataka. Kada vidimo da dolazi do takvih loših rezultata, moramo se odlučiti za novi model ili jednostavno moramo poboljšati podatke za bolje treniranje.

Dok se kNN metoda inače smatra najlošijom, u ovom primjeru pružila je očekivane rezultate. Ono što iznenađuje kod kNN metode je njezina točnost i praćenje točnosti s logističkom regresijom.

7.1. Prednosti i nedostaci programskog rješenja

Prednosti ovog programskog rješenja polaze iz veoma jednostavnog načina implementacije i primjene. Python okruženju možemo pristupiti pomoću besplatnog programa Anaconda, koji je poznat kao program za osobnu primjenu. Program može dosegnuti veliku točnost, ali moramo paziti na podatke i na treniranje tih podataka jer želimo da naš model ima točnost veću od 90 %. Prednost ovog modela polazi i od brzine dobivanja rezultata.

Glavni nedostaci su moguće greške prilikom predviđanja. Model ne uspijeva postići željenu točnost podataka, kada dolazi do toga moramo se vratiti na korak treniranja podataka i proučiti bazu podataka da vidimo jesu li nam podaci dobri. Moramo znati da postoji mnogo različitih modela koji bi dali bolje rezultate, zato ako dođe do nezadovoljavajućih rezultata, moramo pogledati naše mogućnosti i promijeniti metodu predviđanja.

7.2. Poboljšanja modela

Program je moguće poboljšati tako da se proširi baza podataka koja bi omogućila bolju klasifikaciju. Ako bi se model trebao implementirati u praksi, najbolji izbor bile bi neuronske mreže, koje su pružile najbolje rezultate. Jedan od problema ovog modela je potreba za stalnim ažuriranjem podataka. Zbog brze promjene cijena, brze proizvodnje i napretka komponenti računala program bi bio najbolji kada bi se implementirao za svaku komponentu posebno. Stvorio bi se jedan velik sustav koji bi se sastojao od potprograma koji bi predviđali cijene za određene komponente, kako bi se garantirala velika točnost predviđanja.

8. Zaključak

Umjetna inteligencija i strojno učenje svakim danom sve više postaju popularniji te se sve više razvijaju. Strojno se učenje razvilo do te mjere da se danas primjenjuje u svim aspektima naših života, i zapravo ne možemo funkcionirati bez sustava umjetne inteligencije.

U prvom dijelu rada je uvod u umjetnu inteligenciju i objašnjena je njezina povijest i podjela, također su spomenute neke prednosti i mane koje danas muče brojne ljude. Mnoge muči problem hoće li nas strojevi zamijeniti u našim poslovima, također mnogi znanstvenici raspravljaju jesu li strojevi opasnost za ljude ili su oni tu da nam samo pruže korist i unaprijede naše živote. Nadalje je objašnjeno strojno učenje i njegovi modeli koji obuhvaćaju nadzirano, nenadzirano i polunadzirano učenje.

U drugom dijelu rada, koji je obuhvaćao metodologiju, objašnjene su metode i algoritmi koji će biti primijenjeni u istraživanju. Algoritmi koji su objašnjeni obuhvaćali su linearnu i logičku regresiju, kNN algoritam i neuronske mreže. Objašnjen je svaki model koji je predstavljao dobru podlogu za treći dio rada – uvod u okruženje Python. Na Python, jedno od najpoznatijih programskih okruženja za primjenu strojnog učenja, nadovezao se i program Anaconda koji je služio za implementaciju Python okruženja i pomoću kojega se provelo cijelo istraživanje.

U zadnjem dijelu je provedeno istraživanje na temelju višerazredne klasifikacije cijena računala. Prikazan je postupak izgradnje modela u kojemu su se koristile već spomenute metode logističke regresije, k-najbližih susjeda i neuronske mreže. Detaljno je objašnjen svaki korak, od pisanja koda do dobivenih rezultata pomoću slika, tablica i grafikona. Na kraju rezultati istraživanja nisu pokazali veliku točnost, odnosno očekivanu točnost, ali je i dalje donesen zaključak da su neuronske mreže najučinkovitija metoda. Također su doneseni prijedlozi poboljšanja gdje je potrebno doraditi bazu podataka da bi proces treniranja bio kvalitetniji.

Literatura

Anon., 2016. *Future of life.* [Mrežno]
Available at: https://futureoflife.org/background/benefits-risks-of-artificial-intelligence/

Anon., 2019. *Data Flair.* [Mrežno]
Available at: https://data-flair.training/blogs/artificial-intelligence-advantages-disadvantages/

Anon., 2019. *Machine Learning Interviews.* [Mrežno]
Available at: https://machinelearninginterview.com/topics/machine-learning/how-does-knn-algorithm-work-what-are-the-advantages-and-disadvantages-of-knn/

Anon., 2020. *DataRobot.* [Mrežno]
Available at: https://www.datarobot.com/blog/semi-supervised-learning/

Anon., 2020. *Statology.* [Mrežno]
Available at: https://www.statology.org/linear-regression-assumptions/

Anon., 2021. *Built in.* [Mrežno]
Available at: https://builtin.com/artificial-intelligence

Anon., 2021. *Geek For Geeks.* [Mrežno]
Available at: https://www.geeksforgeeks.org/mathematical-explanation-of-k-nearest-neighbour/

Anon., 2022. *Altexsoft.* [Mrežno]
Available at: https://www.altexsoft.com/blog/semi-supervised-learning/

Anon., 2022. *Coursera.* [Mrežno]
Available at: https://www.coursera.org/articles/what-is-python-used-for-a-beginners-guide-to-using-python

Anon., n.d. *AWS Amazon.* [Mrežno]
Available at: https://aws.amazon.com/what-is/neural-network/

Balodi, T., 2019. *Analytic Steps.* [Mrežno]
Available at: https://www.analyticssteps.com/blogs/introduction-machine-learning-supervised-and-unsupervised-learning

Barla, N., 2022. *v7Labs.* [Mrežno]
Available at: https://www.v7labs.com/blog/deep-learning-guide#h1

Boog, J., 2022. *10 BEST MACHINE LEARNING SOFTWARE [2022 LIST].* [Mrežno]
Available at: https://theqalead.com/tools/machine-learning-software/

Christopher, A., 2021. *Medium.* [Mrežno]
Available at: https://medium.com/swlh/k-nearest-neighbor-ca2593d7a3c4

Das, S., 2020. *Analytics Diamag.* [Mrežno]
Available at: https://analyticsindiamag.com/why-jupyter-notebooks-are-so-popular-among-data-scientists/

Gatto, L., 2021. *Bioinformatics.* [Mrežno].

Hintze, A., 2016. *The Conversation.* [Mrežno]
Available at: https://theconversation.com/understanding-the-four-types-of-ai-from-reactive-robots-to-self-aware-beings-67616

Keith, 2021. *A Brief History of Machine Learnig.* [Mrežno]
Available at: https://www.dataversity.net/a-brief-history-of-machine-learning/#

Khalusova, M., 2021. *Maria Khalusova.* [Mrežno]
Available at: https://www.mariakhalusova.com/posts/2019-04-17-ml-model-evaluation-metrics-p2/#micro--macro--weighted-averaged-precision

Kumar, A., 2022. *Vitaflux.* [Mrežno]
Available at: https://vitalflux.com/linear-vs-logistic-regression-differences-examples/

Lončarić, S., n.d. *FER.* [Mrežno]
Available at: https://www.fer.unizg.hr/_download/repository/01-Uvod-1s.pdf

Maillo, R. T. i. H., 2017. kNN-IS: An Iterative Spark-based design of the k-Nearest Neighbors classifier for big data. U: s.l.:an., p. 3.

Mali, K., 2021. *Analytics Vidhya.* [Mrežno]
Available at: https://www.analyticsvidhya.com/blog/2021/10/everything-you-need-to-know-about-linear-regression/

Marco, M. D., 2018. *Diplomski rad.* [Mrežno]
Available at: http://repozitorij.fsb.hr/8941/3/De_Marco_2018_diplomski.pdf

Martinez, J. C., 2020. *Live Code Stream.* [Mrežno]
Available at: https://livecodestream.dev/post/7-steps-of-machine-learning/

Mijwil, M. M., 2017. *Linkedin.* [Mrežno]
Available at: https://www.linkedin.com/pulse/artificial-neural-networks-advantages-disadvantages-maad-m-mijwel/

Milošević, D., 2018. *Analitika i vizualizacija podataka.* [Mrežno]
Available at: https://dusanmilosevic.com/12-python-biblioteka-za-data-science/

Mishra, S., 2017. *Towards Data Science.* [Mrežno]
Available at: https://towardsdatascience.com/unsupervised-learning-and-data-clustering-eeecb78b422a

Nau, R., 2020. *Statistical forecasting.* [Mrežno]
Available at: https://people.duke.edu/~rnau/411home.htm

Nduati, J., 2020. *Section.* [Mrežno]
Available at: https://www.section.io/engineering-education/introduction-to-neural-networks/

Rout, A. R., 2020. *Geek For Geeks.* [Mrežno]
Available at: https://www.geeksforgeeks.org/ml-advantages-and-disadvantages-of-linear-regression/

Selig, J., 2022. *Expert.ai.* [Mrežno]
Available at: https://www.expert.ai/blog/machine-learning-definition/

Stuart Russell, P. N., 1995. U: *Artificial Inteligence: A Modern Approach.* s.l.:an., p. 3.

Vijay Raghavan, V. G. V. G. C. R., 2016. *Cognitive Computing: Theory and Applications.* s.l.:an.

Yufeng, 2017. *Towards Data Science.* [Mrežno]
Available at: https://towardsdatascience.com/the-7-steps-of-machine-learning-2877d7e5548e

Popis slika

Popis tablica