

**SVEUČILIŠTE U ZAGREBU
FAKULTET ORGANIZACIJE I INFORMATIKE
VARAŽDIN**

Karlo Gardijan

**Implementacija predviđanja pomoću
strojnog učenja pri razvoju programskih
proizvoda u Microsoft tehnologijama**

DIPLOMSKI RAD

Varaždin, 2023.

SVEUČILIŠTE U ZAGREBU
FAKULTET ORGANIZACIJE I INFORMATIKE
V A R A Ž D I N

Karlo Gardijan

JMBAG: 00161135270

Studij: Organizacija poslovnih sustava

**Implementacija predviđanja pomoću strojnog učenja pri razvoju
programskih proizvoda u Microsoft tehnologijama**

DIPLOMSKI RAD

Mentor:

Izv. prof. dr. sc. Zlatko Stapić

Varaždin, rujan 2023.

ZAHVALA

Posebnu zahvalnost dugujem mom ocu, baki i djedu koji nažalost nije dočekaio ovaj trenutak. Hvala vam na ljubavi, odricanju, potpori, žrtvi i vjeri u mene koju ste mi pružili i tako mi pomogli na mom životnom putu.

Posebno bih se zahvalio i svojoj kumi Karli Ciprijanović na pruženoj podršci kroz ove godine školovanja.

Rad posvećujem svojoj majci koja iako nije bila fizički tu, bila je najveći izvor snage i motivacije da postanem sve ono što jesam danas.

Karlo Gardijan

Izjava o izvornosti

Izjavljujem da je moj diplomski rad izvorni rezultat mojeg rada te da se u izradi istoga nisam koristio drugim izvorima osim onima koji su u njemu navedeni. Za izradu rada su korištene etički prikladne i prihvatljive metode i tehnike rada.

Autor potvrdio prihvatanjem odredbi u sustavu FOI-radovi

Sažetak

Ovaj diplomski rad se bavi temom implementacije predviđanja pomoću strojnog učenja pri razvoju programskih proizvoda u Microsoft tehnologijama. Fokus rada je na razumijevanju strojnog učenja i načina primjene različitih metoda i algoritama kako bismo omogućili precizno predviđanje unutar različitih aplikacija. Strojno učenje predstavlja ključnu komponentu suvremenih informacijskih tehnologija, omogućavajući sustavima da nauče i poboljšaju svoje ponašanje iz iskustava bez izričitog programiranja.

U teorijskom dijelu rada se obrađuju osnovni koncepti strojnog učenja poput nadziranog učenja, nenadziranog učenja i dubokog učenje. Objašnjene su i različite vrste algoritama kao što su neuronske mreže, algoritmi za klasifikaciju i regresiju te je prikazan praktičan primjer obrade podataka.

Nakon teorijskog dijela, rad prelazi u praktičnu fazu. U ovoj fazi razvijen je i implementiran prediktivni model temeljen na strojnom učenju. Prikupljeni podaci su korišteni podatke za treniranje i testiranje modela. Uspješno razvijen prediktivni model moguće je integrirati s drugim programskim proizvodima. Ovo može uključivati različite scenarije, kao što su ugradnja modela u web aplikaciju, mobilnu aplikaciju ili druge Microsoft tehnologije kao što su Azure Cloud Services. Ovim korakom omogućena je praktična primjenu modela u stvarnom svijetu.

Ključne riječi: strojno učenje; umjetna inteligencija; prediktivni modeli; predviđanje; microsoft; algoritmi

Sadržaj

1. Uvod	1
2. Metode i tehnike rada	3
3. Povijest i razvoj umjetne inteligencije	4
3.1. Područja umjetne inteligencije	6
3.1.1. Strojno učenje	7
3.1.2. Neuronske mreže	8
3.1.3. NLP	10
3.1.4. Duboko učenje	11
3.1.5. Kognitivno računarstvo	13
3.1.6. Računalni vid	13
3.2. Prednosti i nedostaci korištenja umjetne inteligencije	14
4. Strojno učenje	16
4.1. Vrste strojnog učenja	16
4.1.1. Nadzirano strojno učenje	17
4.1.2. Strojno učenje bez nadzora	19
4.1.3. Polu-nadzirano strojno učenje	20
4.1.4. Učenje s pojačanjem	21
4.2. Algoritmi strojnog učenja	21
4.2.1. Linearna regresija	22
4.2.2. Algoritmi temeljeni na sličnosti	23
4.2.3. Stablo odlučivanja	24
4.2.4. Naivni Bayesov klasifikator	26
5. Izrada prediktivnih modela i Microsoft tehnologije	28
5.1. Razvoj funkcionalnosti programa	28
5.2. Microsoft tehnologije za razvoj prediktivnih modela	30
6. Praktični dio rada	34
6.1. Prikupljanje, čišćenje i normalizacija podataka	34
6.2. Azure ML i izrada prediktivnog modela	38
6.2.1. Priprema Azure okruženja	38
6.2.2. Učitavanje podataka	41
6.2.3. Izrada modela	43
6.2.4. Evaluacija rezultata	46
6.2.5. Puštanje modela u produkciju	49
6.2.6. Testiranje modela	52
6.2.7. Integracija modela u programski proizvod	54
7. Zaključak	58
Popis literature	60

Popis slika	63
Popis tablica	64
Popis isječaka programskog koda	65
Prilozi	66

1. Uvod

Svijet u kojem se nalazimo mijenja se svakog trenutka. Podcjenjujući je reći da se mijenja svaki dan jer se zaista u svakoj sekundi događa toliko puno stvari. Još na početku mog fakultetskog obrazovanja jedan od profesora je prikazao grafiku što se zapravo zbiva u jednoj internetskoj minuti i kolika količina podataka se zapravo u njoj nalazi. Te brojka svake godine raste, a neki podaci se više ni ne mjere jer su prestali biti relevantni. Svi oni uključeni u odgojno-obrazovni sustav obrazuju naraštaje za neka nova radna mjesta koja još ne postoje ili ih obrazuju za zanimanja koja neće postojati, dok će se neki dio održati i kroz budućnost, ali značajno izmijenjen. Današnje digitalno doba je nezamislivo bez ogromnih količina podataka, a umjetna inteligencija igra jednu od ključnih uloga u eksploataciji, analizi i razumijevanju tih podataka jer ubrzani razvoj tehnologije i sve kompleksniji zahtjevi korisnika postavljaju sve one uključene u razvoj programskih proizvoda pred izazov stvaranja sofisticiranih i prilagodljivih softverskih rješenja.

Sve to oblikuje način na koji radimo, komuniciramo i živimo. Umjetna inteligencija i primjena strojnog učenja, omogućavajući računalima da uče iz podataka i donose samostalne odluke, transformiraju društvo u kojem živimo. Digitalna revolucija koja se zbiva zahvaljujući primjeni modernih koncepata i tehnologija donose nam gomilu novih pametnih rješenja: pametni telefoni, pametni gradovi, pametne kuće, pametni automobili,... Sva ta pametna rješenja svakim danom sve više postaju dio naše svakodnevnice i vrše digitalnu transformaciju mnogih aspekata ljudskog života. Umjetna inteligencija, kao srž te digitalne transformacije, omogućava nam analizu ogromnih količina podataka u stvarnom vremenu. To znači da se odluke donose brže i preciznije nego ikada prije, bilo da je riječ o medicinskoj dijagnozi, praćenju prometa ili personaliziranim preporukama sadržaja na društvenim mrežama. Svaka interakcija s tehnologijom sada je međusobna prilika za učenje – ljudi uče o tehnologiji i ona uči o njima. Važno je razumjeti način na koji funkcioniraju ovi koncepti i shvatiti principe po kojima računala zaključuju kako bi mogli razumjeti benefite koje ova tehnologija donosi te kako korištenjem iste možemo dodati vrijednost programskim proizvodima, ali i razumjeti potencijalne opasnosti zlouporabe ove tehnologije.

Rad je koncipiran u dva dijela: teorijski i praktični dio.

Prvi dio teorijske cjeline se sastoji od uvodnog teorijskog dijela koji sistematizira i objašnjava ulogu umjetne inteligencije prateći njen razvoj i shvaćanje od najranijih početaka do danas. Nakon prvog dijela slijedi analiza i sistematizacija područja strojnog učenja. Rad detaljno objašnjava vrste strojnog učenja i algoritme koji se koriste za njegovu implementaciju. Praktični dio rada se odnosi na pregled i sistematizaciju Microsoft

tehnologija koje podržavaju strojno učenje te se kroz praktični primjer prikazuje ciklus razvoja i implementacije funkcionalnosti predviđanja pomoću strojnog učenja korištenjem Microsoft Azure platforme i Azure Machine Learning alata.

Rad se sastoji od teorijskog i praktičnog dijela rada. Cilj praktičnog dijela rada je pokazati praktičnu primjenu pojmova, metoda i tehnika koji su predstavljeni u teorijskom dijelu. Za primjenu u praktičnom dijelu rada su odabrane Microsoft tehnologije, uvažavajući Microsoft kao jedan od vodećih igrača u tehnološkoj industriji, koji nije samo svjedok ovog trenda, već i ključni doprinositelj razvoju alata i okruženja za primjenu strojnog učenja. Ovaj rad će isto tako naglasiti kako Microsoft pristupa ovim izazovima i kako njihova tehnološka rješenja podržavaju razvoj sofisticiranih i pouzdanih programskih proizvoda kroz analizu implementacije predviđanja pomoću strojnog učenja u Microsoftovim tehnologijama.

Za odabir teme ovog rada prije svega motivirao me veliki potencijal koji vidim u implementaciji ove tehnologije s ciljem unaprjeđenja mogućnosti analiziranja koje su do sada nudile tradicionalne metode i tehnologije. Također, radom sam želio dublje shvatiti i razumjeti način na koji umjetna inteligencija, a posebno strojno učenje funkcioniraju budući da se dosadašnjim obrazovanjem nisam imao prilike dublje susresti s ovom domenom. Nadalje, budući da svoju profesionalnu karijeru planiram graditi u okvirima podatkovnih znanosti smatram kako će mi obrada i razumijevanje ove tematike uvelike pomoći u profesionalnom razvoju.

Također, povezao sam svoju strast prema košarci sa informatikom jer u praktičnom dijelu rada obrađujem temu kreiranja prediktivnih modela koji se koriste za predviđanje košarkaške statistike. Za korištenje Microsoft tehnologija sam se odlučio budući da su upravo oni jedan od lidera ovog područja. Cilj ovog rada je kreiranje prediktivnog modela pomoću strojnog učenja korištenjem Microsoft tehnologija, a provodeći pri tome analizu i prikazujući važnost područja strojnog učenja, ali i umjetne inteligencije i njihovu primjenu u programskim proizvodima koja iz dana u dan postaje sve aktualnija.

2. Metode i tehnike rada

Prije početka pisanja rada proučena je okvirna literatura koja će se koristiti za pisanje rada. Budući da se rad tematski veže na područje strojnog učenja odnosno umjetne inteligencije koje je relativno novo područje mnogi izvori literature su s interneta. Za lakše upravljanje literaturom korišten je alat Zotero. Za pisanje teorijskog dijela rada korištene su razne istraživačke metode kao što je metoda analize koja je služila za rastavljanje složenih pojmova na manje lakše razumljive cjeline. Također, korištena je i metoda klasifikacije koja je služila za sistematiziranje pojmova i definicija. Metoda deskripcije je korištena za opis činjenica bez znanstvenog objašnjenja dok je metoda izviđajnog istraživanja služila za pregled do sada poznatih pojmova i sistematizaciju teorijske podloge koje leži u pozadini praktičnog dijela rada.

Praktični dio rada je izrađen pomoću Microsoft Azure platforme i Azure Machine Learning alata. Za pronalazak početnog skupa podataka je korišteno web mjesto Kaggle, za čišćenje podataka i testiranje izgrađenog modela je korišten Python programski jezik i Visual Studio Code (verzija 1.82) razvojno okruženje. Dok je za izradu programskog proizvoda koji implementira kreirani model korišten Node.js i Python (Flask razvojno okruženje). Za potrebe rada je kreiran i Github repozitorij na kojem su objavljeni materijali potrebni za implementaciju rada kao što je skup podataka, korištene programske skripte i izrađena web stranica.

Za pisanje rada nisu korišteni alati umjetne inteligencije.

3. Povijest i razvoj umjetne inteligencije

U današnjem svijetu, djelovanje umjetne inteligencije (eng. *artificial intelligence*, AI) prožeto je svim sferama našega života. Istovremeno nam je potpuno jasno što taj pojam predstavlja, ali i nejasno kako točno funkcionira ili kako bi se trebao definirati. Iako je teško točno odrediti trenutak kada je sve započelo, povijest razvoja umjetne inteligencije datira od prve polovice 20.stoljeća. Ovisno o autorima, u izvorima je moguće pronaći 1942. i 1950. godinu kao početak razvoja područja umjetne inteligencije. Godine 1942. je američki pisac Isaac Asimov objavio kratku znanstveno fantastičnu priču pod nazivom Runaround (Haenlein i Kaplan, 2019). Radnja Runaround-a je priča o robotu kojeg su razvili inženjeri Gregory Powell i Mike Donavski čija se radnja odvija proučavanjem triju zakona robotike. Robot ne može istovremeno poštovati sva tri zakona te time upadne u beskonačnu petlju u kojoj se ponavlja ponašanje objekta. Radnja priče je nadahnula mnoge autore jer je ovo djelo predstavljalo prve smjernice za razvoj robota kakve danas poznajemo.

Gotovo u isto vrijeme, Alan Turing radi svoje istraživanje danas poznato kao Turingov test (Pinar Saygin et al., 2000). Turing svoj rad objavljuje 1950. godine pod nazivom "Računalni strojevi i inteligencija" (eng. *Computing Machinery and Intelligence*), a započinje ga rečenicom "Predlažem da razmotrimo pitanje: 'Mogu li strojevi razmišljati?'. U Turingovom testu ljudski ispitivač komunicira sa dva igrača tako što s njima razmjenjuje poruke. Ako ispitivač ne može utvrditi koji je igrač računalo, a koji čovjek, smatra se da je računalo prošlo test. Ovaj test je i danas relevantan kada se govori o umjetnoj inteligenciji i testiranju iste iako ga treba uzeti s dozom rezerve budući da ovaj test inteligentnim smatra samo ona računala koja su sposobna komunicirati s ispitivačem na isti način kako to rade ljudi. Također, važno je napomenuti kako računala uvijek pokušavaju ponuditi točan odgovor što se prilikom ispitivanja ovim testom može smatrati kao mana jer ispitivač može lako zaključiti tko od sugovornika nije osoba ako jedan od njih uvijek daje točan odgovor, dok to ona uistinu nije.

Sam pojam umjetne inteligencije je skovan 1956. Godine od strane Marvinia Minskya i Johna McCarthyja prilikom održavanja Dartmouth ljetnog istraživačkog projekta o umjetnoj inteligenciji (eng. *Dartmouth Summer Research Project on Artificial Intelligence*) (Haenlein i Kaplan, 2019). Minsky i McCarthy su umjetnu inteligenciju definirali kao konstrukciju računalnih programa koji rješavaju zadatke koje ljudska bića trenutno obavljaju kvalitetnije jer zahtijevaju mentalne procese visoke razine kao što su perceptivno učenje, organizacija pamćenja i kratkoročno učenje ("History of Artificial Intelligence," bez dat.). Iako je stvorena gotovo prije sedamdeset godina, u vremenu kada je računalna znanost bila u svojim počecima a, računalna snaga nije ni približno dosegala današnje razine, primjećujemo da

njihov koncept umjetne inteligencije i dalje izvanredno odražava suvremeno razumijevanje ovog područja. Današnji računalni sustavi su u stanju obavljati visoko kognitivne zadatke i manipulirati ogromnim količinama podataka, uglavnom zahvaljujući tehnološkom napretku i razvoju hardvera. Ipak, mnogi od temeljnih principa, postulata i algoritama koji stoje u osnovi umjetne inteligencije razvijeni su znatno prije, a o tome će se više raspravljati u nastavku ovog rada.

Danas se pod tim pojmom može pronaći puno toga, od upravljanja logistikom, statistike, poslovnog odlučivanja, ERP sustava, virtualnih asistenata, autonomnih vozila, raznih oblika dijagnostičkih asistenata, alata za kreiranje preciznih predikcija i raznih drugih područja svakodnevnog života. Razlog tome je nepostojanje određene jedinstvene definicije umjetne inteligencije i što ona točno obuhvaća, sa točno određenim pravilima. Svaka definicija je fluidna jer se neka područja koja su nekada smatrana dijelom umjetne inteligencije, zahvaljujući razvoju iste, sada izdvajaju iz tog okvira. Istovremeno se pojavljuju nove teme i područja koja nisu bila zahvaćena tom definicijom, a ubrajaju se u tu domenu. Jedno od objašnjenja umjetne inteligencije jest da je to sposobnost digitalnog računala ili računalno kontroliranog robota da obavlja zadatke koji se obično povezuju s inteligentnim bićima (Copeland, 2023).

Domena umjetne inteligencije uglavnom je usmjerena na sljedeće komponente inteligencije (Copeland, 2023):

- učenje (eng. *learning*)
- rasuđivanje (eng. *reasoning*)
- rješavanje problema (eng. *problem solving*)
- percepciju (eng. *perception*)
- korištenje jezika (eng. *using language*).

Navedene komponente oblikuju način na koji živimo, radimo i komuniciramo. Sveprisutnost umjetne inteligencije prepoznata je u širokom spektru industrija i sektora, donoseći sa sobom ne samo učinkovitost, već i revolucionarne promjene u načinu na koji percipiramo i pristupamo mnogim aspektima svakodnevnog života. U području medicine, umjetna inteligencija se pokazala kao ključan segment. Od dijagnosticiranja rijetkih bolesti do analize ogromnih skupova medicinskih podataka, AI pomaže liječnicima da brže i preciznije postavljaju dijagnoze te planiraju tijek liječenja. Analiza genoma i identifikacija potencijalnih genetskih rizika također su postale ostvarive zahvaljujući dubokom učenju i strojnom učenju. Financijski sektor također ne zaostaje jer umjetna inteligencija omogućuje predviđanje tržišnih kretanja, optimizaciju investicijskih portfelja i automatizaciju trgovinskih procesa. Analiza velikih količina podataka pomaže u otkrivanju uzoraka i trendova koji bi inače mogli

proći nezamijećeno, omogućujući investitorima donošenje informiranih odluka. Transportna industrija doživljava revoluciju zahvaljujući razvoju autonomnih vozila. Umjetna inteligencija upravlja kompleksnim algoritmima koji omogućuju sigurnu navigaciju i interakciju s okolinom.

Osim toga, analitički alati koristeći AI pomažu u optimizaciji ruta, smanjenju gužvi i poboljšanju prometne učinkovitosti. U proizvodnji, umjetna inteligencija podiže razinu efektivnosti i efikasnosti. Automatizacija procesa proizvodnje uz pomoć robota i pametnih strojeva pomaže u povećanju produktivnosti i smanjenju grešaka. Analizom podataka generiranih tijekom proizvodnje, moguće je prepoznati trendove koji ukazuju na potencijalne kvarove, čime se povećava pouzdanost i smanjuju troškovi održavanja. E-trgovina doživljava procvat zahvaljujući personaliziranim preporukama proizvoda. Algoritmi umjetne inteligencije analiziraju povijest kupovina i preferencije korisnika kako bi stvorili personalizirano iskustvo kupovine. Ovo ne samo da povećava vjerojatnost prodaje već i poboljšava zadovoljstvo kupaca. Umjetna inteligencija ima i značajan utjecaj na sigurnost, zdravstvenu skrb, energetiku, obrazovanje pa čak i na umjetnost. Sposobnost AI-a da analizira ogromne količine podataka i prepozna uzorke omogućuje ranu detekciju prijevara, bolje upravljanje resursima u bolnicama, optimizaciju potrošnje energije, personalizirano obrazovanje te čak generiranje umjetničkih djela.

Umjetna inteligencija se zaista proteže kroz gotovo svaki aspekt naših života i poslovanja. Njezina sposobnost analize, obrade podataka i autonomnog djelovanja donosi inovacije koje su nezamislive bile samo nekoliko desetljeća unazad. Ona se kontinuirano razvija pružajući nam pri tome odlične rezultate dobivene njezinom primjenom.

3.1. Područja umjetne inteligencije

Umjetna inteligencije se dijeli na različita subpodručja zbog složenosti i različitosti zadataka i problema koje ona rješava. Ovisno o autorima pojedine literature, moguće je pronaći različite podjele na subpodručja. Područje umjetne inteligencije je moguće podijeliti na ("6 Major Sub-Fields of Artificial Intelligence," 2021):

- strojno učenje (eng. *machine learning*, ML)
- neuronske mreže (eng. *neural network*)
- obrada prirodnog jezika (eng. *natural language processing*, NLP)
- duboko učenje (eng. *deep learning*)
- kognitivno računalstvo (eng. *cognitive computing*)
- računalni vid (eng. *computer vision*).

Kako je prikazano na slici ispod umjetna inteligencija se raščlanjuje na šest područja, ako nema čvrstih granica između njih i često se međusobno preplavljaju. Ova međuovisnost među područjima omogućava programskim proizvodima da učinkovito rješavaju različite izazove, prilagođavajući se dostupnim resursima i kompleksnosti problema s kojima se suočavaju.



Slika 1. Podjela umjetne inteligencije

Izvor: vlastita izrada prema 6 Major Sub-Fields of Artificial Intelligence (2021)

Važno je napomenuti da postoje značajne razlike između ovih područja, kako u složenosti njihove implementacije, tako i u njihovim sposobnostima. Ova raznolikost može značajno utjecati na trajanje razvojnog procesa, konačnu cijenu proizvoda i kvalitetu pružene usluge što je ujedno i najveća motivacija za odabir upravo ove podjele. Rad razmatra svako od ovih područja, istražujući kako se međusobno preplavljaju i kako njihova kombinacija može doprinijeti razvoju inteligentnih rješenja.

3.1.1. Strojno učenje

Među laicima je razumijevanje pojma strojno učenje najčešće istovjetan pojmu umjetna inteligencija budući da na prvi pogled izgleda kako ova dva pojma i jesu istoznačna jer gotovo pa ne postoji nijedan program koji koristi umjetnu inteligenciju, a da u isto vrijeme ne koristi jedan od oblika strojnog učenja. Ipak, ova dva pojma imaju nekoliko ključnih razlika

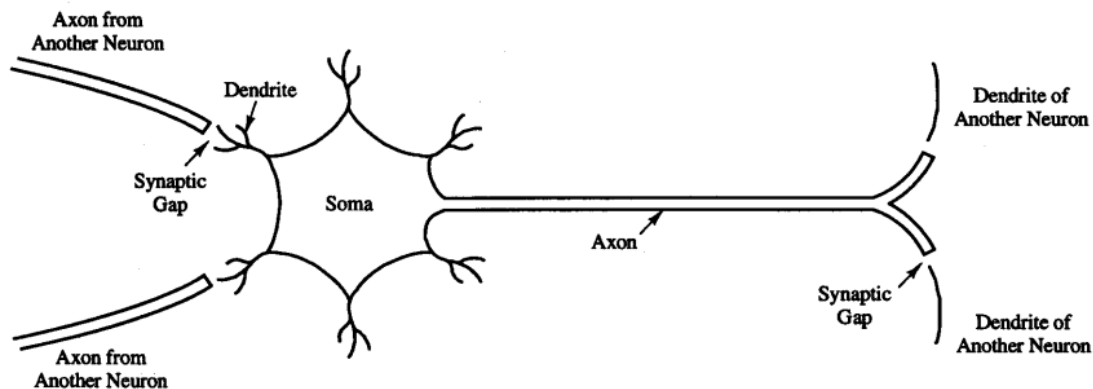
koje valja istaknuti. Najveća razlika je u tome da umjetna inteligencija ipak označava puno šire područje od strojnog učenja, Cilj umjetne inteligencije je stvoriti “inteligentan” računalni sustav odnosno program dok je strojno učenje jedan od alata koji umjetna inteligencija koristi kako bi ostvarila svoju zadaću. Strojno učenje nudi konkretan pristup za ostvarenje cilja umjetne inteligencije.

Strojno učenje je grana umjetne inteligencije koja omogućuje računalima da uče iz podataka i donose odluke gotovo bez ljudskog kontakta te poboljšavaju svoje performanse s iskustvom. Temelji se na razvoju algoritama i modela koji analiziraju podatke kako bi identificirali uzorke i donosili predviđanja ili odluke. Strojima se omogućava pristup informacijama i dopušta im se da samostalno stječu znanje. Ovo jednostavno implicira da se računala potiču da izvrše zadatke bez potrebe da ih posebno programiramo za te svrhe (“6 Major Sub-Fields of Artificial Intelligence,” 2021). Strojno učenje pomaže računalima da postanu autonomnija, prilagodljivija i inteligentnija tako doprinoseći napretku tehnologije i inovacijama. Budući da se javlja sve veća količina podataka, strojno učenje je ključno kako bi se ta hrpa podataka mogla obraditi.

3.1.2. Neuronske mreže

Kao što je spomenuto na početku ovog poglavlja, navedena područja međusobno se isprepliću u mnogim aspektima, a neuronske mreže čvrsto su povezane sa strojnim učenjem. U stvari, neuronske mreže predstavljaju jednu od tehnika koje se koriste unutar strojnog učenja i imaju široku primjenu, posebno u domeni dubokog učenja. Strojno učenje obuhvaća raznolik skup tehnika, a neuronske mreže su samo jedna od tih tehnika. Ipak, zbog svoje izuzetne širine primjene, složenosti i impresivnih rezultata koje postižu, neuronske mreže su izdvojene kao zasebno područje, te se smatraju jednom od najčešće korištenih tehnika unutar strojnog učenja.

Inspiracija neuronskog računalstva je ustvari u ljudskom mozgu. Biološki neuron je zapravo živčana stanica koja se sastoji od dendrita, aksona i tijela stanice. Uz to je još bitno spomenuti sinaptičku pukotinu što je mjesto gdje dva neurona između sebe prenose živčane impulse. Na tom principu funkcioniraju i umjetne neuronske mreže koje se ovdje obrađuju (Fausett, 1994).

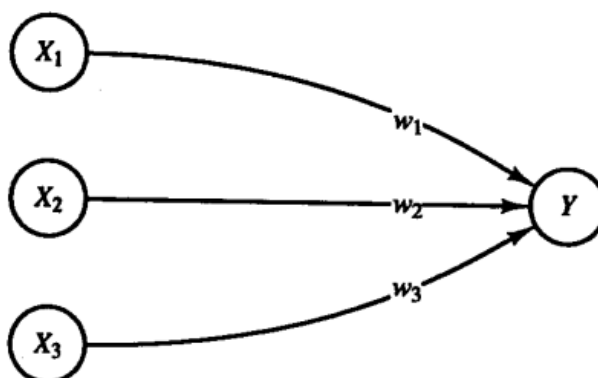


Slika 2. Biološki neuron

Izvor: (Fausett, 1994)

Ukratko, neuroni su specijalizirani za (Eluyode i Akomolafe, 2013) :

- primati informacije iz unutarnjeg i vanjskog okruženja;
- prijenos signala drugim neuronima i efektorskom organu;
- obradu informacija (integracija) i
- odrediti ili modificirati diferencijaciju senzornih receptorskih stanica i efektorskih stanica.

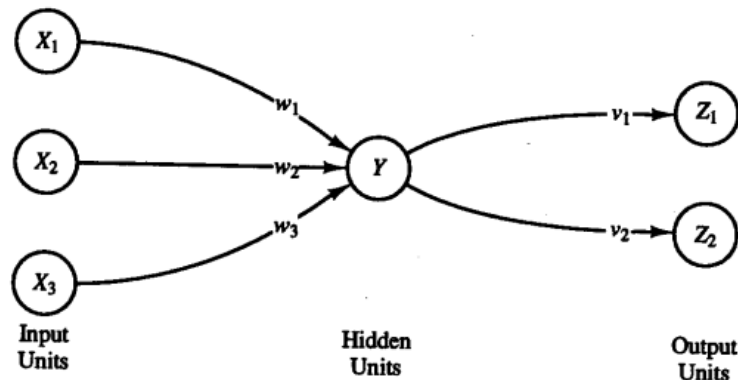


Slika 3. Jednostavni umjetni neuron

Izvor: (Fausett, 1994)

Umjetna neuronska mreža je sustav za obradu informacija koji ima određene karakteristike zajedničke s biološkim neuronskim mrežama. Taj biološki model je poslužio kao inspiracija za matematički prikaz neuronske mreže. Kao što je već rečeno, umjetne neuronske mreže razvijene su kao generalizacije matematičkih modela ljudske kognicije ili neuralne biologije, temeljene na pretpostavkama da (Fausett, 1994):

1. Obrada informacija odvija se na mnogim jednostavnim elementima koji se nazivaju neuroni.
2. Signali se prenose između neurona preko veza.
3. Svaka poveznica ima pridruženu težinu, koja u tipičnoj neuronskoj mreži umnožava preneseni signal.
4. Svaki neuron primjenjuje aktivacijsku funkciju (obično nelinearnu) na svoj neto ulaz (zbroj ponderiranih ulaznih signala) kako bi odredio svoj izlazni signal.



Slika 4. Jednostavna neuronska mreža

Izvor: (Fausett, 1994)

Umjetna neuronska mreža je skup algoritama koji se koriste za identifikaciju elementarnih korelacija među velikim količinama podataka ("6 Major Sub-Fields of Artificial Intelligence," 2021).

3.1.3. NLP

NLP je grana umjetne inteligencije koja omogućava komunikaciju računala i ljudi korištenjem prirodnog jezika. To je metoda računalne analize ljudskih jezika. Oponašanjem ljudskog prirodnog jezika, omogućuje stroju razumijevanje i tumačenje podataka. NLP je metoda za pretraživanje, analizu, razumijevanje i izdvajanje informacija iz tekstualnog unosa ("6 Major Sub-Fields of Artificial Intelligence," 2021). Pomoću NLP-a se mogu izrađivati velike analize sa ogromnom količinom podataka na temelju kojih se kasnije mogu donositi bitne odluke. Neki od primjera korištenja NLP-a koje je moguće susresti u svakodnevnom životu su:

- razgovorni agenti (eng. *chatbots*)
- prevoditelje tekstova
- virtualni asistenti

- aplikacije za provjeru gramatike
- provjera teksta na društvenim mrežama kako bi se ograničio neprimjeren sadržaj
- poboljšanje korisničkog iskustva korisnika
- analiza povratnih informacija
- automatsko pisanje sažetaka teksta.

NLP tehnologija igra ključnu ulogu u današnjem svijetu, omogućavajući računalima da dublje razumiju i komuniciraju s ljudima. Kroz razne primjene poput chatbotova, prevoditelja tekstova i virtualnih asistenata, NLP olakšava svakodnevne aktivnosti i poboljšava interakciju između ljudi i tehnologije. Također, analiza povratnih informacija i automatsko pisanje sažetaka teksta samo su neki od načina na koje NLP doprinosi boljem razumijevanju teksta i informacija koje se generiraju. S obzirom na rastuću količinu podataka, NLP će vjerojatno nastaviti igrati ključnu ulogu u pružanju informacija za donošenje bitnih odluka.

3.1.4. Duboko učenje

Duboko učenje je podskup strojnog učenja koje je u biti neuronska mreža s tri ili više sloja. Ove neuronske mreže pokušavaju simulirati ponašanje ljudskog mozga omogućavajući mu da uči iz velikih količina podataka. Iako neuronska mreža s jednim slojem još uvijek može dati približna predviđanja, dodatni skriveni slojevi mogu pomoći u optimizaciji i poboljšanju točnosti. [8] Kao primjere korištenja dubokog učenja također je moguće izdvojiti primjere navedene iznad u tekstu u dijelu o NLP-u poput chatbotova, virtualnih asistenta, prevoditelja tekstova zato što su sva ova područja u uskoj kooperaciji. Duboko učenje promatra sve moguće ljudske osobine i baze podataka o ponašanju te prolazi učenje pod nadzorom ("6 Major Sub-Fields of Artificial Intelligence," 2021).

Duboko učenje predstavlja izuzetno snažan pristup u području umjetne inteligencije koji je ključan za razumijevanje kompleksnih uzoraka i obrazaca u velikim skupovima podataka. Kroz složene neuronske mreže s više slojeva, duboko učenje omogućava sustavima da izvlače duboka značenja iz informacija, što je često izuzetno teško ili nemoguće za konvencionalne metode analize. Ovaj pristup pokazuje posebno snažne rezultate u područjima kao što su prepoznavanje uzoraka u slikama, analiza prirodnog jezika, autonomna vožnja i mnogi drugi kompleksni zadaci. Kontinuirani razvoj dubokog učenja će biti značajan za nove inovacije i dublje razumijevanje složenih procesa u svijetu oko nas.

Kako je vidljivo na slici iznad duboko učenje je usko povezano s područjima strojnog učenja i umjetne inteligencije, a ovi pojmovi međusobno se dopunjuju. Umjetna inteligencija koristi strojno učenje kao ključnu tehniku kako bi postigla svoje ciljeve, pri čemu se među različitim tehnikama ističe upotreba neuronskih mreža. S druge strane, duboko učenje predstavlja daljnji korak u unaprjeđenju tehnike neuronskih mreža tako da dodaje dodatne slojeve unutar neuronske mreže, što ima za cilj poboljšanje performansi i stvaranje sofisticiranijih i inteligentnih računalnih programa.



Slika 5. Duboko učenje

Izvor: vlastita izrada prema Singapore Computer Society (bez dat.)

Duboko učenje predstavlja naprednu varijaciju tehnike neuronskih mreža koja se koristi za dublje razumijevanje i obradu kompleksnih podataka. Dodavanjem više slojeva, duboko učenje omogućava računalima da prepoznaju dublje i apstraktnije obrasce u podacima, što može rezultirati boljem sposobnošću donošenja odluka i izvođenja inteligentnih zadataka.

3.1.5. Kognitivno računarstvo

Cilj kognitivnog računarstva je pokrenuti i poboljšati interakciju između čovjeka i stroja kako bi se izvršili složeni zadaci i pomoglo u rješavanju problema. Dok rade s ljudima na raznim poslovima, strojevi uče i shvaćaju ljudsko ponašanje i osjećaje u različitim situacijama, a zatim rekreiraju ljudski misaoni proces u računalnom modelu ("6 Major Sub-Fields of Artificial Intelligence," 2021). Kombinacija naučenog razmišljanja u sinergiji sa umjetnom inteligencijom stvara nove proizvode koji imaju najbolje karakteristike od oba svijeta. Pomoću kognitivnog računarstva moguće je donijeti složene odluke u raznim područjima djelovanja. Jedan od primjera kognitivnog računarstva jest Google Assistant.

Kada razmišljamo o kognitivnom računarstvu, važno je naglasiti da računala, bez obzira na njihovu naprednost i inteligenciju, nemaju stvarne emocije ili osjećaje. Umjesto toga, njihovi odgovori temelje se na društvenim normama, pravilima i podacima na kojima su trenirani. Prilikom razvoja takvih programa često se posebno pridaje pažnja tome da računala daju odgovore koji su što je moguće sličniji ljudskim, kako bi korisnicima pružila uvjerljivu iluziju razgovora s stvarnom osobom dok su u interakciji s programom.

Primjerice, sustavi se posebno treniraju da pruže ispriku korisniku u slučaju da ponude netočan odgovor. Računalo u takvim situacijama ne osjeća krivicu, jer nema emocionalnu svijest, ali je programirano da se ispriča korisniku kako bi osiguralo zadovoljstvo korisnika i zadovoljilo društvene norme. Važno je razumjeti da računalni sustavi koji koriste kognitivno računarstvo nemaju svijest, emocije ili namjere. Njihovi odgovori su rezultat algoritama i podataka, a njihova sposobnost za imitiranje ljudskog ponašanja temelji se na programiranju i treniranju, a ne na stvarnim emocijama ili svijesti.

Kognitivno računarstvo predstavlja odlučujući korak prema stvaranju intelligentnih sustava koji ne samo da izvršavaju zadatke, već i razumiju i reagiraju na složene ljudske potrebe. Ova tehnologija omogućava strojevima da ne samo obrade podatke, već i interpretiraju kontekst, emocije i namjere koje stoje iza interakcije s ljudima. Integracija naučenih procesa s umjetnom inteligencijom omogućava stvaranje inovativnih proizvoda koji kombiniraju sposobnosti računala s razumijevanjem ljudskog ponašanja. Kroz primjere poput Google Assistanta, vidimo kako kognitivno računarstvo postiže napredak u ostvarivanju intelligentnih i korisnički orijentiranih rješenja koja obogaćuju način na koji se odvija interakcija sa strojevima u svakodnevnom životu.

3.1.6. Računalni vid

Računalni vid je polje umjetne inteligencije koje omogućuje računalima i sustavima da izvuku značajne informacije iz digitalnih slika, videa i drugih vizualnih inputa i poduzmu

radnje ili daju preporuke na temelju tih informacija. Ako AI omogućuje računalima da razmišljaju, može se reći da im računalni vid omogućava da vide, promatraju i razumiju (*What Is Deep Learning?*, bez dat.). Ovaj dio umjetne inteligencije je široko rasprostranjen u zdravstvenom sektoru za procjenu zdravstvenog stanja pacijenta pomoću MRI skeniranja, X-zraka i drugih tehnika snimanja. U automobilske industrije računalom upravljana vozila i dronovi također koriste računalni vid ("6 Major Sub-Fields of Artificial Intelligence," 2021).

Računalni vid zauzima ključnu ulogu u transformaciji načina na koji računala percipiraju svijet oko sebe. Kroz napredne algoritme dubokog učenja, računalni vid omogućava sustavima da prepoznaju lica, objekte, scene i druge elemente unutar vizualnih sadržaja te da izvuku dublje značenje iz tih informacija, pružajući im sposobnost da donose odluke na temelju onoga što vide. Primjena računalnog vida u različitim sektorima naglašava njegovu svestranost i sposobnost prilagodbe kako bi riješio različite izazove u modernom svijetu.

Kako bi računala uistinu mogla vidjeti potrebno je jako puno podataka na kojima ćemo trenirati algoritam odnosno program. Osim podataka računalni vid koristi duboko učenje pomoću kojeg obrađuje dobivene podatke, a praksa je pokazala da su konvolucijske neuronske mreže tehnika koja pruža najbolje rezultate kada je u pitanju računalni vid.

Konvolucijske neuronske mreže omogućuju računalima da na primjer sliku koja ulazi kao input u sustav rastave na piksele te svakom od piksela pridruže odgovarajuću oznaku. Računalo oznake koristi kako bi nad njima proveo konvolucije pomoću kojih računalo može predvidjeti što se nalazi na slici. Konvolucija je matematički operacija u kojoj se dvije funkcije spajaju u novu funkciju. Neuronska mreža pokreće konvolucije koje kreiraju predviđanje te se provjerava točnost predikcije, a postupak se ponavlja sve dok predviđanja ne postanu sve točnija (*What Is Computer Vision?*, bez dat.).

Jednostavan primjer korištenja računalnog vida je razvoj programa koji će na temelju fotografije predvidjeti nalazi li se na slici pas ili vuk. Model mora biti u stanju na temelju fizičkih karakteristika kao što je na primjer veličina i oblik ušiju, dužina očnjaka, veličina glave, veličina i oblik šapa i slično točno prepoznati koja životinja se nalazi na slici. Ovo je moguće upravo zahvaljujući računalnom vidu, ali je za kreiranje uspješnog modela potrebno pripremiti kvalitetan set podataka nad kojima ćemo trenirati algoritam.

3.2. Prednosti i nedostaci korištenja umjetne inteligencije

Umjetna inteligencija predstavlja jedno od najznačajnijih dostignuća suvremene tehnologije, s potencijalom da značajno transformira način na koji živimo i radimo. Unatoč

svojim brojnim prednostima, ona donosi i određene nedostatke, a njihova složena dinamika oblikuje naš odnos prema tehnološkim inovacijama. Prednost je svakako značajan iskorak u automatizaciji i efikasnosti. Sposobnost strojeva da izvršavaju rutinske zadatke brže i preciznije od čovjeka značajno povećava produktivnost u mnogim sektorima.

Osim toga, korištenjem AI-a mogu se obraditi i analizirati velike količine podataka u kratkom vremenskom roku, omogućavajući bolje informacije i donošenje odluka temeljenih na činjenicama. Još jedan ključni aspekt je personalizacija usluga jer umjetna inteligencija može analizirati obrasce ponašanja korisnika te pružiti personalizirane preporuke i usluge. Ovo je posebno važno u marketingu i trgovini, gdje se personalizacijom može povećati angažman korisnika i poboljšati korisničko iskustvo. Kao što je već spomenuto, AI također ima značajne primjene u medicini. Sposobnost brze i precizne analize medicinskih podataka pomaže zdravstvenim stručnjacima u dijagnosticiranju bolesti. Kreiranje točnih prediktivnih modela u raznim životnim sferama koje se kreću od primjene u sportskim kolektivima preko najprometnijih svjetskih burzi do vremenskim prognoza samo su neke od mogućnosti umjetne inteligencije.

Unatoč brojnim prednostima, umjetna inteligencija ima i svoje nedostatke i ograničenja. Jedan od najpoznatijih nedostataka jest gubitak radnih mjesta zahvaljujući automatizaciji koja rezultira smanjenjem potrebe za radnom snagom na repetitivnim zadacima u sektorima poput proizvodnje, trgovine i uslužnih djelatnosti. Jedan od ključnih nedostataka je također i nedostatak kreativnosti i empatije računala. Dok strojevi mogu izvršavati zadatke temeljene na logici, njima nedostaje kreativno razmišljanje i sposobnost empatije tako da ipak određena zanimanja nije moguće sasvim zamijeniti upotrebom modernih tehnologija i koncepata. Osim toga, postoji zabrinutost da AI može donositi odluke koje su u suprotnosti s etičkim i moralnim načelima, jer mu nedostaje moralna svijest kakvu imaju samo ljudi.

Naposljetku, pretjerana ovisnost o tehnologiji je još jedan važan nedostatak. Kako AI preuzima više zadataka koje su nekad obavljali ljudi, postoji rizik da ćemo izgubiti vještine i sposobnosti koje su nam nekada omogućavale neovisnost. Ovisnost o tehnologiji može dugoročno utjecati na našu sposobnost da rješavamo probleme i donosimo odluke bez pomoći strojeva.

4. Strojno učenje

Strojno učenje je izraz koji je skovao Arthur Samuel, informatičar iz IBM-a i pionir u AI-u i računalnim igrama. Samuel je dizajnirao računalni program za igranje dame (eng. checkers). Što je program više igrao, to je više učio iz iskustva, koristeći algoritme za predviđanje (*What Is Machine Learning?*, bez dat.). Strojno učenje istražuje analizu i konstrukciju algoritama koji mogu učiti iz podataka i predviđati ih.

Ovi uvidi potom pokreću donošenje odluka unutar aplikacija i poslovanja, idealno utječući na ključne pokazatelje rasta. Kako se veliki podaci nastavljaju širiti i rasti, potražnja na tržištu za podatkovnim znanstvenicima će se povećavati. Od njih će se tražiti da pomognu identificirati najrelevantnija poslovna pitanja i podatke za odgovore na njih (Burns, bez dat.). Strojno učenje omogućava računalima da prepoznaju obrasce i donose zaključke iz podataka bez da su eksplicitno programirana za svaku pojedinu situaciju. To ima potencijal transformirati način na koji radimo i živimo, dopuštajući nam da razvijamo personalizirane usluge, bolje razumijemo kompleksne probleme te brže donosimo informirane odluke. Uz napredak tehnologija kao što su duboko učenje i velike količine dostupnih podataka, strojno učenje postaje sve moćnije i korisnije.

Proces strojnog učenja započinje prikupljanjem podataka. Kvaliteta i brojnost prikupljenih podataka su najvažniji za stvaranje uspješnog modela. Sljedeća faza je obrada sirovih (eng. *raw*) podataka s ciljem pročišćavanja, verifikacije i validacije prikupljenih podataka kako bi se odabrao set podataka koji će se pustiti u algoritam. Odabrani set podataka je potrebno podijeliti na 3 podseta koji se koriste za treniranje, validaciju i testiranje odabranog seta. Treniranje odabranog seta započinje puštanjem podataka u algoritam. Nakon čega slijedi provjera rezultat i evaluacija ispravnosti modela te kreiranje korekcija ako je moguće (*Machine Learning Workflow*, bez dat.).

4.1. Vrste strojnog učenja

Područje strojnog učenja dijeli se u potkategorije ovisno o vrsti problema koje rješavaju. Na temelju metoda i načina učenja, strojno učenje se uglavnom dijeli na četiri vrste, a to su (*Types of Machine Learning - Javatpoint*, bez dat.):

- Nadzirano strojno učenje (eng. *supervised machine learning*)
- Strojno učenje bez nadzora (eng. *unsupervised machine learning*)
- Polu-nadzirano strojno učenje (eng. *semi-supervised machine learning*)
- Učenje s podrškom (eng. *reinforcement learning*).

Odabir određene vrste strojnog učenja ovisi o specifičnom problemu, podacima s kojima se u tom trenutku raspolaže i podacima koji se žele dobiti.

Kako je na slici ispod navedeno strojno učenje možemo podijeliti u 4 najčešće i najpoznatije kategorije ovisno o stupnju nadzora.



Slika 6. Podjela strojnog učenja

Izvor: vlastita izrada prema Javatpoint (bez dat.)

Daljnji rad pojašnjava svaku kategoriju posebno. Postoji još kategorija koje možemo dodati u ovu skupinu kao što su detekcija anomalija (eng. *anomaly detection*), prenošenje znanja (eng. *transfer learning*), samonadziruće učenje (eng. *self-supervised learning*), ali one neće biti detaljnije objašnjene u sklopu ovog rada.

4.1.1. Nadzirano strojno učenje

Nadzirano strojno učenje, potkategorija je strojnog učenja i umjetne inteligencije te je primjer dobre prakse. Kao što mu samo ime govori, nadzirano strojno učenje temelji se na nadzoru. To znači da u tehnici nadziranog učenja mi treniramo strojeve pomoću "označenog" skupa podataka, a na temelju treninga stroj predviđa izlaz. Ovdje označeni podaci određuju da su neki od ulaza već preslikani na izlaz. Što je još dragocjenije, možemo reći; prvo treniramo stroj s ulazom i odgovarajućim izlazom, a zatim tražimo od stroja da predvidi izlaz pomoću skupa testnih podataka (Ali, 2022). Nadzirano strojno učenje je proces koji uključuje korištenje označenih podataka za treniranje algoritama kako bi klasificirali podatke ili precizno predvidjeli rezultate. Tijekom iteracija modela s ulaznim podacima, model prilagođava svoje težine dok ne postigne odgovarajuće rezultate, a to se postiže kroz proces unakrsne provjere.

Glavni cilj tehnike nadziranog učenja je mapiranje ulazne varijable x s izlaznom varijablom y .



Slika 7. Nadzirano strojno učenje

Izvor: vlastita izrada prema Ali (2022)

Nadzirano učenje može se podijeliti u dvije vrste problema koje se javljaju prilikom rudarenja podataka:

- **klasifikacija** - vrsta nadziranog strojnog učenja gdje algoritmi uče iz dostupnih podataka kako bi mogli predvidjeti ishod ili neki događaj u budućnosti. Klasifikacijski algoritmi koriste se za predviđanje ishoda, ako ishod može uzeti dvije moguće vrijednosti kao što su točno ili netočno, zadano ili nema zadane postavke, da ili ne - ovo je poznato kao binarna klasifikacija. Ako ishod sadrži više od dvije moguće vrijednosti, to je poznato kao višeklasna klasifikacija (Ali, 2022). Postoji više klasifikacijskih algoritama poput: nasumična šuma, stabla odlučivanja, logistička regresija, potporni vektorski strojevi (eng. *support vector machines*) (*Supervised Machine Learning* - Javatpoint, bez dat.).
- **regresija** - vrsta nadziranog strojnog učenja koji se koriste ako postoji odnos između ulazne varijable i izlazne varijable (Ali, 2022). Regresija se koristi za razumijevanje odnosa između zavisnih i nezavisnih varijabli. Postoji više regresijskih algoritama poput: Linearna regresija, ne-linearna regresija, regresijska stabla, Bayesova linearna regresija i polinomna regresija (*Supervised Machine Learning* - Javatpoint, bez dat.).

Nadzirano učenje danas omogućava da se rješavaju različiti svakodnevni problemi zahvaljujući tome što modeli uče kroz vrijeme na dostupnim setovima podataka (eng. *dataset*).

4.1.2. Strojno učenje bez nadzora

Strojno učenje bez nadzora, potkategorija je strojnog učenja i umjetne inteligencije. Kako samo ime govori riječ je o učenju bez nadzora odnosno bez “označenih” podataka kakav je slučaj kod nadziranog strojnog učenja. Budući da u ovom slučaju nemamo ulaze koji su preslikani na izlaz cilj je istrenirati algoritam da može pronaći sličnosti, obrasce i strukturu unutar ulaznih podataka i na temelju njih donijeti predikciju.

Strojno učenje bez nadzora, često poznato kao otkrivanje razreda (eng. *class discovery*), predstavlja granu strojnog učenja u kojoj algoritam nastoji identificirati obrasce ili strukturu u podacima bez uputa putem označenim izlaza, što je suprotno nadziranom strojnom učenju. Ključna razlika između ova dva oblika učenja je da strojno učenje bez nadzora ne može koristiti unakrsnu validaciju kao tehniku za povezivanje ulaza i izlaza budući da nema označene podatke (Gentleman i Carey, 2008).



Slika 8. Strojno učenje bez nadzora

Izvor: vlastita izrada prema Gentleman i Carey (2008)

Ne nadzirano strojno učenje razlikuje dvije vrste pristupa za kreiranje modela, a to su modeli bazirani na asocijacijama i modeli bazirani na klasterima. Analiza pomoću klastera omogućuje grupiranje podataka koji imaju najviše zajedničkih osobina u iste grupe dok

podaci koji nemaju toliko sličnosti ostaju razdvojeni. Model asocijacija se koristi za pronalaženje veza između varijabli u velikim skupinama podataka te na taj način povezuje podatke ovisno o njihovim međuovisnostima (*Unsupervised Machine Learning - Javatpoint*, bez dat.) .

Model asocijacija se pokazao posebno uspješnim u poboljšanju prodaje i korisničkog iskustva. Naime, model temeljen na tehnici asocijacija za osobu koja kupuje proizvod X (recimo da je taj proizvod košarkaška lopta) će pretpostaviti da osoba ima tendenciju kupiti proizvod Y (recimo košarkaške tenisice ili dres). Ovo je banalan primjer koji za cilj ima objasniti način na koji ova metoda funkcionira, a ako ju koristimo na velikim skupovima podataka rezultati mogu biti prilično iznenađujući, ali točni.

Neki od najzastupljenijih algoritama koji se koriste kod strojnog učenja bez nadzora su (*Unsupervised Machine Learning - Javatpoint*, bez dat.):

- K najbližih susjeda (eng. *K-nearest neighbours*)
- K sredina (eng. *K-Means*)
- Hijerarhijsko klasteriranje (eng. *Hierarchical clustering*)
- Detekcija anomalija (eng. *Anomaly detection*)
- Apriori algoritam (eng. *Apriori algorithm*)
- Neuronske mreže (eng. *Neural networks*).

Prednost nenadziranog strojnog učenja leži u njegovoj jednostavnosti pri pronalasku odgovarajućeg skupa podataka, otkrivanju skrivenih obrazaca među podacima i širokom spektru primjene. Međutim, ovaj pristup također prate i nedostaci, uključujući ograničenu kontrolu nad rezultatima koju uzrokuje osjetljivost na ulazne parametre te ovakvi algoritmi mogu dati različite rezultate sa svakom izvedbom. Također, teškoće u interpretaciji, jer nemamo potpunu sigurnost kako su algoritmi kreirali obrasce budući da ne postoje referentne oznake između ulaza i izlaza.

4.1.3. Polu-nadzirano strojno učenje

Polu-nadzirano strojno učenje potkategorija je strojnog učenja i umjetne inteligencije. Kako samo ime govori riječ je o učenju koje je mješavina prethodne dvije metode. Glavna ideja prilikom stvaranja polu-nadziranog učenja je pokušati spojiti najbolje od oba pristupa i tako ostvariti bolje performanse modela.

Glavna ideja ove kategorije je koristiti mali dio označenih podataka kakav se koristi prilikom rada s nadziranim strojnim učenjem i veliki dio ne označenih podataka koje koristi strojno učenje bez nadzora (A. Gupta, 2019).

Iako polu-nadzirano strojno učenje ime svoje izazove jer je osjetljivo na omjere označenih i neoznačenih podataka nad kojima je treniran. Prednosti ove kategorije su ušteda vremena i resursa jer nije potrebno označiti sve primjere. Također, ovaj pristup može poboljšati performanse modela strojnog učenja bez nadzora.

4.1.4. Učenje s pojačanjem

Učenje s pojačanjem je posebna kategorija strojnog učenja. Ona se razlikuje od svih do sada navedenih kategorija iako ima neka zajednička svojstva kao i ostale tri kategorije. Učenje s pojačanjem kako i samo ime govori je vrsta učenja u kojoj se pojačavaju “osjećaji” onoga koji uči.

Ideja ovakvog pristupa je napraviti model u kojem algoritam koji se trenira s podacima dobiva samo ulazne podatke te ne zna koji su izlazni podaci (za razliku od nadziranog učenja), ali ipak ova kategorija se razlikuje i od nenadziranog učenja budući da je ovaj pristup baziran na nagradama i kaznama. Ideja je da se za svaku točnu predikciju algoritam nagradi, a za svaku negativnu kazni s ciljem stvaranja pouzdanog modela (Sutton i Barto, 2018).

4.2. Algoritmi strojnog učenja

Za potrebe strojnog učenja podatkovni stručnjaci moraju imati duboko i opširno razumijevanje matematičkih algoritama koji pokreću strojno učenje. Osim potrebnih matematičkih i statističkih znanja važno je i razumijevanje početnog seta podataka na visokoj razini kako bi odabrali upravo onaj algoritam koji će pružiti najkvalitetnije rezultate budući da nije svaki algoritam jednako uspješan u kreiranju modela za svaki tip problema. Rad je do sada obradio više algoritama koje strojno učenje koristi, a u ovom poglavlju rad skreće pažnju na nekoliko algoritama o kojima u dosadašnjem radu nije bilo toliko govora, a spadaju u jedne od najzastupljenijih algoritama strojnog učenja.

Algoritmi koji će se detaljnije obraditi u nastavku rada su:

- Linearna regresija (eng. *Linear regression*)
- Algoritmi temeljeni na sličnosti (eng. *clustering algorithms*)
- Stablo odlučivanja (eng. *Decision tree*)
- Naivni Bayesov klasifikator (eng. *Naive Bayes classifier*).

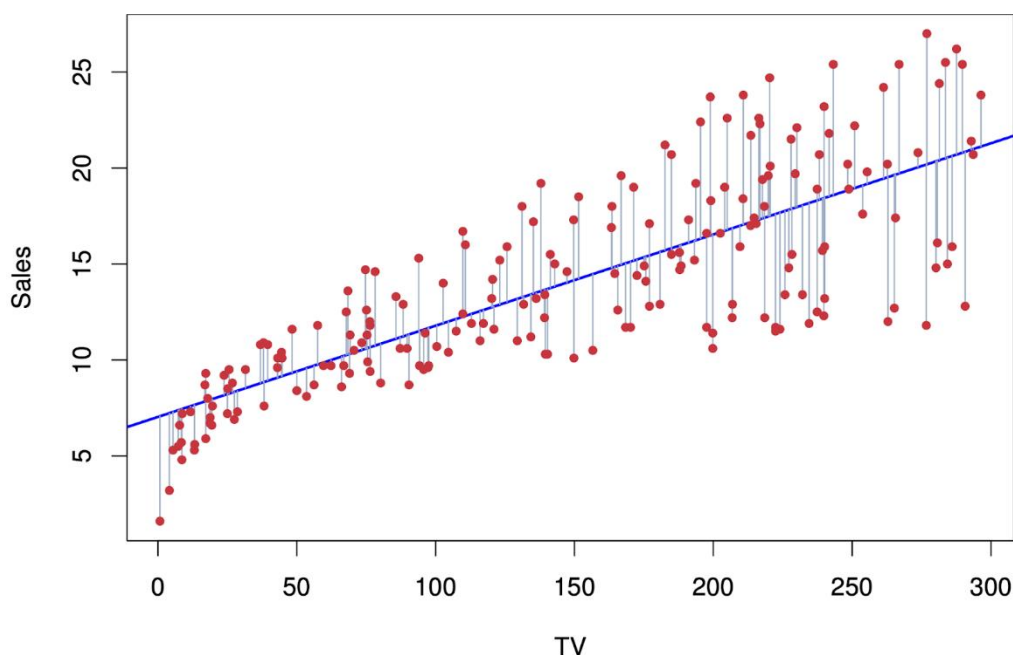
Navedeni algoritmi su odabrani jer svojim karakteristikama, sličnostima, ali i razlikama omogućuju rješavanje širokog spektra problema i zadataka koji se rješavaju pomoću umjetne inteligencije i strojnog učenja. Linearna regresija je jednostavan algoritam kojeg se najčešće

koristi za modeliranje veza i relacija između ulaznih i izlaznih podataka. Algoritmi temeljeni na sličnosti grupiraju podatke u skupine ovisno o njihovim sličnostima te na temelju pripadnosti pojedinoj grupi stvaraju predikcije. Stablo odlučivanja s druge strane je algoritam koje je temeljen na pravilima te se pokazao odličnim u rješavanju problema klasifikacije. Naivni Bayesov algoritam određivanjem vjerojatnostima nad kontinuiranim varijablama uspješno rješava klasifikacijske probleme. Umjetne neuronske mreže su algoritam koji je obrađen u prijašnjim poglavljima. Inspiriran ljudskim neuronima omogućuje rješavanje složenih problema zahvaljujući međusobno povezanim umjetnim neuronima organiziranim u slojeve, pri čemu svaki neuron ima određenu težinu i pristranost (Kumar Talaviya, 2023).

4.2.1. Linearna regresija

Linearna regresija predstavlja tip nenadziranog strojnog učenja koji procjenjuje linearnu povezanost između zavisne varijable i jedne ili više nezavisnih varijabli. Kada imamo samo jednu nezavisnu varijablu nazivamo ju univarijatna linearna regresija (eng. *univariate linear regression*), dok se u slučaju više od jedne nezavisne varijable koristi naziv multivarijatna linearna regresija (eng. *multivariate linear regression*). Osnovni cilj ovog algoritma jest pronaći optimalnu linearnu jednadžbu koja može predvidjeti vrijednost zavisne varijable na temelju nezavisnih varijabli (M. Gupta, 2018).

Na slici ispod se nalazi predviđanje koje je algoritam linearne regresije ponudio temeljem ulaznog skupa podataka. Predviđanje je označeno plavom bojom dok crveni krugovi označavaju podatke. Regresija je ostvarena promatranjem dviju varijabli. Varijable "TV" na x osi koja u ovom slučaju označava zavisnu varijablu i varijable "Sales" koja označava prodaju televizora. Regresija je ostvarena pronalaženjem funkcije (na grafu označena plavom bojom) koja povezuje varijable što omogućuje da se u slučaju jedne nepoznate varijable njena vrijednost može predvidjeti promatranjem vrijednosti poznate varijable i regresijske funkcije.



Slika 9. Linearna regresija

Izvor: (Bacallado i Taylor, 2022)

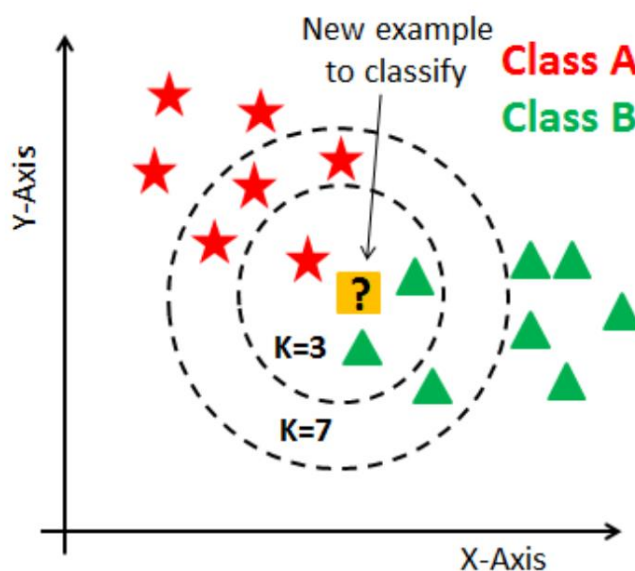
Regresijske funkcije nisu uvijek linearne, ali je linearna funkcija uzeta kao primjer zbog svoje jednostavnosti. Mogu poprimati vrijednosti raznih matematičkih funkcija kao što su logistička, eksponencijalna, polinomijalna, sigmoidalna ili na primjer racionalna funkcija. Također, linearna funkcija je u ovom slučaju posebno važna budući da se ista koristi i u praktičnom dijelu rada, ali o tome više u nastavku rada.

4.2.2. Algoritmi temeljeni na sličnosti

Zajednička komponenta ovih algoritama je grupiranje podataka temeljem sličnosti. Učenje temeljeno na sličnosti se još naziva i lijeno učenje (eng. *lazy learning*). Ova skupina algoritama također spada u algoritme koji se koriste prilikom nadziranog strojnog učenja. Algoritmi koji se ističu u ovoj kategoriji svojom obuhvatnom primjenom u industriji je algoritam K-najbližih susjeda (eng. *K-nearest neighbours*) i algoritam K-sredina (eng. *K-means*).

Algoritam K-najbližih susjeda je jednostavan algoritam za čije provođenje je prvo potrebno odrediti vrijednost K koja predstavlja broj susjeda koji tražimo. Nakon što smo odredili vrijednost K potrebno je izračunati udaljenosti, sličnost između varijabli. Udaljenost među podacima se računa Euklidskom formulom za udaljenost točaka u prostoru. Nakon čega se uspoređuju dobivene udaljenosti te se kao predviđanje određuje ona vrijednost koja prevladava među K najbližih susjeda (Oreški, 2022).

Na slici ispod je prikazan primjer predviđanja temeljen na algoritmu K-najbližih susjeda. Podaci klase A su označeni crvenom bojom dok su podaci klase B označeni zelenom bojom, a nova ulazna varijabla za koju treba napraviti predviđanje je označena s žutim kvadratom i znakom upitnika. U prvom slučaju je vrijednost $K=3$ te će sustav predvidjeti kako novi podatak ima vrijednost klase B budući da se u prva tri susjeda nalaze dva podatka klase B i jedan podatak klase A. Koristeći analogno razmišljanje ako vrijednosti K pridružimo broj 7 algoritam će predvidjeti da je vrijednost novog podatka klasa A. Iz primjera je vidljivo kako je ovaj algoritam osjetljiv na K vrijednost koju odaberemo stoga moramo biti posebno oprezni prilikom određivanja vrijednosti K .



Slika 10. Primjer primjene K-najbližih susjeda

Izvor: (Shah, 2021)

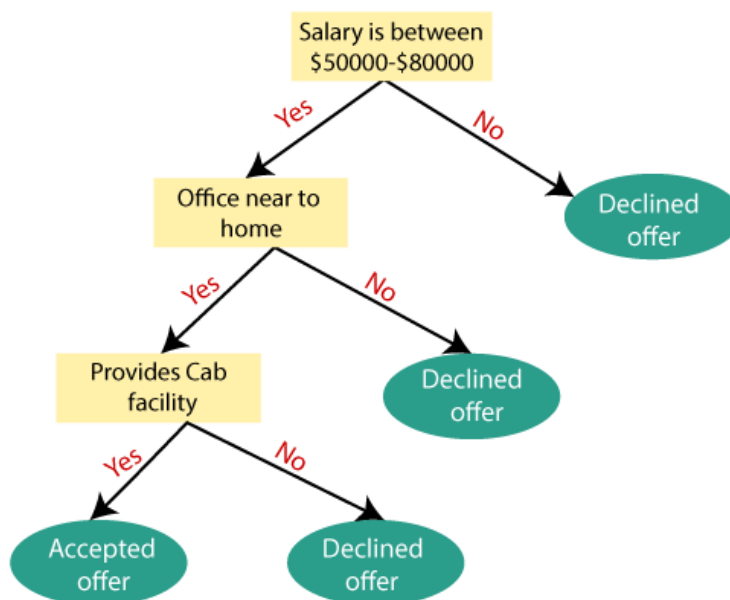
Ovaj algoritam je posebno osjetljiv na skalu na kojoj se nalazi ovisno o početnim podacima nad kojima je algoritam treniran jer varijanca za jedan podatak može biti nekoliko stotina dok već za drugi može biti nekoliko tisuća. Stoga je potrebno prvo normalizirati podatke, a neke od normalizacija koje se koriste su z-score normalizacija, MiniMax normalizacija i normalizacija decimalnog skaliranja (Oreški, 2022).

4.2.3. Stablo odlučivanja

Stablo odlučivanja algoritam je strojnog učenja koji se temelji na zaključivanju na temelju pravila. Algoritam također spada u algoritme nadziranog strojnog učenja što znači da je model potrebno trenirati nad podacima koji imaju označene podatke. Drugim riječima ulazni podaci na kojima treniramo algoritam moraju imati odgovarajuće izlazne podatke.

Stablo odlučivanja započinje korijenskim čvorom nakon kojeg slijede čvorovi odluke (eng. *decision node*) koji granaju stablo. Kada govorimo o binarnim stablima odlučivanja svaki čvor ima dva čvora koja se granaju iz njega. Na kraju stabla kada više nema čvorova u kojima je potrebno donijeti odluku slijede dva lista (eng. *leaf node*) stabla. Upravo listovi predstavljaju predikciju odnosno odluku koju algoritam donosi na temelju ulaznih podataka. U svakom čvoru odluke unutar stabla ovisno o tome koji je uvjet zadovoljen dolazi do prelaska na idući čvor (*Decision Tree Algorithm in Machine Learning - Javatpoint*, bez dat.).

Na slici iznad je prikazan primjer binarnog stabla odlučivanja. Zelenom bojom su označeni listovi stabla dok su žutim pravokutnicima prikazani čvorovi odluke. Svaki čvor odluke ima pridružen uvjet te se ovisno o njegovom ispunjenju prelazi na idući čvor odnosno kreće lijevo ili desno kroz stablo. Na prikazanom primjeru kandidat će prihvatiti ponudu tvrtke o zaposlenju ako mu tvrtka ponudi godišnju plaću u vrijednosti između 50.000,00 i 80.000,00 dolara, ured mu bude blizu doma te ako mu poslodavac osigura prijevoz. U svim ostalim slučajevima ishod je odbijanje ponude.



Slika 11. Primjer stabla odlučivanja

Izvor: (*Javatpoint*, bez dat.)

Navedeni primjer je pojednostavljen prikaz stabla odlučivanja koje se koristi prilikom strojnog učenja. Stabla odlučivanja koja koriste modeli strojnog učenja su mnogo složenija s puno više grana te su u pravilu vrlo rijetko binarna.

4.2.4. Naivni Bayesov klasifikator

Naivni Bayesov klasifikator je jedan od jednostavnih algoritama klasifikacije. Široka primjena ovog algoritma leži u njegovoj jednostavnosti te mogućnosti kreiranja brzih predikcija. Za razliku od ostalih algoritama u ovom poglavlju, naivni Bayesov algoritam spada u skupini algoritama koji učenje temelje na vjerojatnosti.

Algoritam se zove Bayesov zato što implementira Bayesov teorem. Matematički teorem koji je dobio ime po Britanskom matematičaru iz 18. stoljeća Thomasu Bayesu. Teorem daje vjerojatnost nekog događaja na temelju informacija koje su ili bi mogle biti povezane s tim događajem. Formula za izračunavanje Bayesovog teorema je (Hayes, 2023):

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

gdje je

- $P(A)$ vjerojatnost događaja A
- $P(B)$ vjerojatnost događaja B
- $P(B | A)$ vjerojatnost da će se dogoditi događaj B ako se dogodio događaj A
- $P(A | B)$ vjerojatnost da će se dogoditi događaj A ako se dogodio događaj B
- $P(A \cap B)$ vjerojatnost da se dogodi događaj A i događaj B

Kada se radi s malim brojem ulaznih podataka, ovaj algoritam pokazuje izvanredne rezultate. No, kako se broj ulaznih podataka povećava i broj uvjetnih ovisnosti između tih ulaznih atributa raste. S obzirom na to da, čisti Bayesovom teorem, zahtijeva izračunavanje svih međusobnih ovisnosti između atributa, to dovodi do potrebe za izračunom velikog broja vjerojatnosti. Ovo značajno otežava rad s tim algoritmom i produžuje vrijeme potrebno za njegovu implementaciju. Stoga se odlučilo pojednostaviti algoritam za potrebe strojnog učenja. Definirano je da ulazni atributi nemaju zajedničke ovisnosti, čime se znatno smanjuje broj vjerojatnosti koje treba izračunati i značajno skraćuje vrijeme potrebno za dobivanje predikcija. Ova modifikacija originalnog Bayesovog teorema poznata je kao Naivni Bayesov algoritam (*Naive Bayes Classifier in Machine Learning - Javatpoint*, bez. dat.).

Naivni Bayesov klasifikator je pronašao široku primjenu u raznim sferama kao što su filtriranje neželjene pošte ili detekciji neželjenih komentara (eng. *spam filtering*), bankarskim

sustavima za određivanje kreditne sposobnosti, u zdravstvenom sustavu kao pomoć pri detekciji bolesti, financijskom i javnom sektoru za razvrstavanje vrijednosnih papira i dokumenata, ali i u sustavima za autentikaciju korisnika budući da jako dobro prepoznaje rukopis, ali i govor.

5. Izrada prediktivnih modela i Microsoft tehnologije

Prilikom izrade programskih proizvoda u novije vrijeme sve češće susrećemo potrebu za implementacijom umjetne inteligencije odnosno strojnog učenja. Korisnicima raznih programskih proizvoda funkcionalnosti koje nudi strojno učenje, a pogotovo prediktivni modeli donose dodatnu vrijednost softverskim proizvodima što su prepoznali developeri i arhitekti programskih proizvoda te sve češće unutar svojih aplikacija nude i funkcionalnosti strojnog učenja kako bi poboljšali korisničko iskustvo i kreirali uslugu koja će korisniku donijeti dodanu vrijednost, ali i napravili iskorak u odnosu na konkurenciju.

5.1. Razvoj funkcionalnosti programa

Funkcionalnost predviđanja temeljenog na strojnom učenju se proširila među programskim proizvodima iz raznih sektora svakodnevnog života. Aplikacije za analizu sportskih događaja uvelike primjenjuju mogućnosti strojnog učenja kako bi kreirali kvalitetne prognoze za buduća događanja. Sustavi unutar financijskog sektora također uvode prediktivne modele kako bi prognozirali kretanja na burzi, promjene u vrijednostima valuta, kretanje cijena nekretnina i slično. Aplikacije za upravljanje prometom također koriste prediktivne modele kako bi predvidjeli potencijalne kritične dionice i tako na vrijeme intervenirali i preusmjerili promet kako bi se izbjegli prometni kolapsi i gužve. Autonomna vozila također koriste strojno učenje kako bi predvidjeli potencijalne opasnosti na cesti i na vrijeme reagirali. Predviđanje pomoću strojnog učenja se uvodi i u aplikacije za prognoziranje dugoročnih vremenskih prognoza. Sustavi za upravljanje pametnim gradovima također implementiraju razne mogućnosti predviđanja pogonjenog strojnim učenjem i obradom velikih količina podataka.

Iz svega navedenog možemo zaključiti kako predviđanje pogonjeno strojnim učenjem pronalazi svoju primjenu u velikom broju programskih proizvoda raznih namjena. Implementacija ove tehnologije može biti posebno izazovna budući da zahtijeva koordinaciju stručnjaka različitih područja te podatkovna znanost i podaci dobivaju na značaju. Implementacija ovakve funkcionalnosti može biti izuzetno skupa te je potrebno s velikim oprezom ući u razvoj iste. Kako bi se osigurala njena uspješna primjena i ostvarivanje dodane vrijednosti potrebno je pratiti smjernice dobre prakse i postaviti realna očekivanja.

Smjernice dobre prakse za uspješnu integraciju funkcionalnosti predviđanja pomoću strojnog učenja unutar programskog proizvoda su (Cottman, 2020):

- Jasno definirati problem koji se želi riješiti i što se očekuje od rješenja
- Postaviti metrike za evaluaciju dobivenog rješenja
- Prilagoditi arhitekturu postojećeg sustava ovakvoj integraciji
- Prikupiti što je moguće veći skup relevantnih podataka
- Pripremiti set podataka koji će se koristiti za treniranje modela (obraditi i transformirati podatke)
- Odabrati tehniku strojnog učenja koja će se koristiti (poželjno je odabrati nekoliko tehnika i algoritama te usporediti dobivene rezultate)
- Trenirati model nad pripremljenim setom podataka
- Testirati model na testnim skupom podataka
- Implementirati model ili rezultate unutar programskog proizvoda
- Testirati programski proizvod i uspješnost integracije
- Održavati i ažurirati cjelokupan sustav
- Dokumentirati rad i integraciju
- Pripremiti upute za korištenje i educirati korisnike
- Pratiti zadovoljstvo korisnika

Smjernice predstavljaju okvir uspješnog vođenja projekta integracije funkcionalnosti predviđanja pomoću strojnog učenja unutar razvoja programskog proizvoda te ne garantiraju uspješnu implementaciju, ali joj povećavaju vjerojatnost. Smjernice nisu strogo definirane i podložne su promjenama budući da svaki projekt nosi svoje izazove i ograničenja gotovo je nemoguće definirati egzaktne korake kojih se razvojni tim mora držati kako bi uspješno implementirao ovakav projekt.

Najčešći izazovi s kojim se razvojni timovi susreću prilikom implementacije ovog tipa projekata su osiguranje kvalitete podataka. Algoritmi strojnog učenja su osjetljivi na podatke na kojima su trenirani te je potrebno osigurati velike količine kvalitetnih podataka. Nadalje, integracija u postojeće sustave također može biti izazovna budući da strojno učenje može zahtijevati posebnu infrastrukturu kako hardvera tako i softvera kako bi efikasno radila. Također, prediktivni modeli zahtijevaju konstantno održavanje što može biti prilično skupo i trošiti vrijeme koje bi se moglo uložiti u razvoj drugih proizvoda ili novih značajki (PlanetTogether, 2023).

5.2. Microsoft tehnologije za razvoj prediktivnih modela

Microsoft kao jedna od vodećih svjetskih kompanija za izradu softvera u svojoj ponudi usluga ima i širok asortiman alata, platformi i servisa koji omogućuju razvoj i implementaciju strojnog učenja pri razvoju programskih proizvoda.

Neki od najznačajnijih alata koje Microsoft nudi su sljedeći:

Microsoft Azure platforma¹ je platforma u oblaku (eng. *cloud*) koja korisnicima nudi veliku skupinu Microsoftovih usluga zajedno s izgrađenom infrastrukturom Microsoftom pogonjenih sustava te servisima dostupnima po potrebi. Azure je jedan od svjetskih lidera u ovom području stoga korisnicima nudi nekoliko usluga pomoću kojih mogu implementirati i razvijati modele strojnog učenja. Azure Machine Learning server je servis u oblaku koji korisnicima nudi mogućnost razvoja prediktivnih modela te njihovo objavljivanje unutar Azure programskog okruženja. Azure auto ML je podsustav unutar Azure Machine Learning servera koji korisnicima nudi mogućnosti automatiziranog razvoja algoritama strojnog učenja. Azure auto ML funkcionira na principu crne kutije budući da korisnici moraju definirati ulazne parametre i podatke dok sam program automatski razvija prediktivni model i generira predviđanje. Osim razvoja vlastitih modela strojnog učenja Azure platforma korisnicima omogućava i korištenje nekih od unaprijed izrađenih algoritama za obavljanje raznih zadataka pomoću Azure Cognitive Services alata.



Slika 12. Logo Microsoft Azure platforme

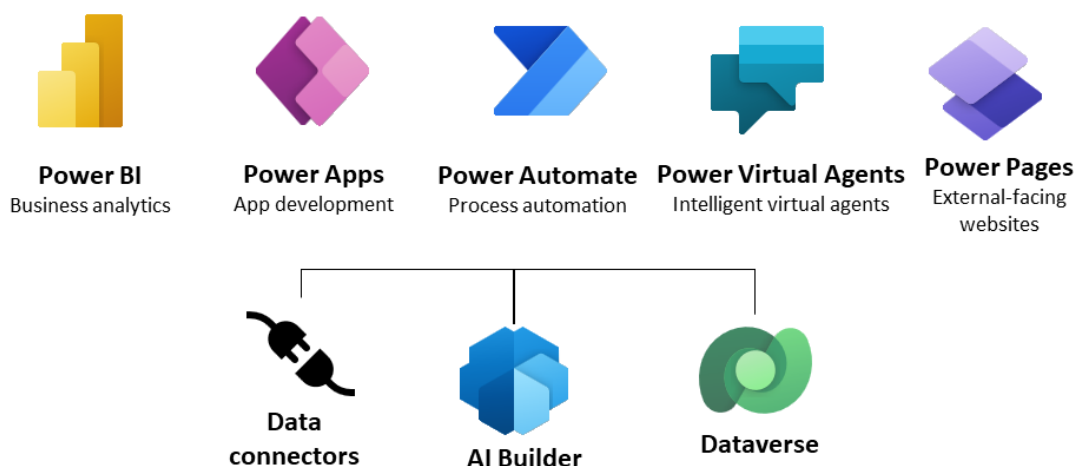
Izvor: (Cloud Computing Services | Microsoft Azure)

Snaga Microsoft Azure platforme leži u mogućnosti integracije i kolaboracije između usluga i servisa koje platforma nudi. Tako na primjer poslovni korisnici unutar Microsoft Synapse Analytics servisa mogu integrirati Machine Learning server za razvoj modela strojnog učenja i unaprjeđenje analitičkih mogućnosti.

Microsoft Power Platform² je moćna platforma za izradu poslovnih aplikacija niskog koda koja između ostalih mogućnosti korisnicima nudi i implementaciju i razvoj modela strojnog učenja unutar aplikacija.

¹ <https://azure.microsoft.com/en-us>

² <https://powerplatform.microsoft.com/en-us/>



Slika 13. Microsoft Power platforma

Izvor: (Vivek, 2023)

Unutar Microsoft Power Platform ukomponirani su i sljedeći Microsoft produkti:

- Power BI
- Power Apps
- Power Automate
- Power Virtual Agents
- Power Pages.

Microsoft Power BI ³moćan je alat za vizualizaciju podataka i kreiranje izvještaja u svrhu analitike, a svojim korisnicima nudi i implementaciju modela strojnog učenja kako bi poboljšali svoje izvještaje uvođenjem prediktivne analitike. U praksi je često korišten zbog vizualizacije podataka, automatizirane izrade izvještaja, lake integracije s drugim platformama, napredne analitike, široke dostupnosti (usluga u oblaku) i mogućih proširenja

Microsoft SQL Server Machine Learning Services ⁴alat je koji omogućuje korisnicima da unutar SQL baze podataka pokreću prediktivne modele izrađene u R ili Python programskim jezicima. On omogućava široku lepezu algoritama, brzu obradu podataka, laganu upotrebu, optimiziranost i skalabilnost. Na taj način korisnici ne moraju koristiti neko od rješenja ili platformi u oblaku kako bi dobili pristup alatima i mogućnostima koje Microsoft nudi u polju strojnog učenja.

³ <https://powerbi.microsoft.com/en-us/>

⁴ <https://microsoft.github.io/sql-ml-tutorials/>

Microsoft ERP (eng. *Enterprise Resource Planning*) rješenja također nude razne mogućnosti korištenja strojnog učenja. Microsoft među svojim rješenjima nudi uslugu koja se posebno ističe u sposobnostima implementacije modela strojnog učenja, a to je **Microsoft Dynamics 365**⁵ platforma.



Slika 14. Logo Microsoft Dynamics 365

Izvor: (Microsoft)

Microsoft Dynamics je platforma za implementaciju ERP rješenja koja se sastoji od tri glavna proizvoda koja obuhvaćaju cjelokupno poslovanje organizacije, Dynamics 365 Finance and Operations, Dynamics 365 Business Central i Dynamics 365 Sales. Odabir jednog od ovih rješenja ili više njih ovisi o potrebama i veličini organizacije, ali je važno za napomenuti da svi imaju veliki broj ekstenzija koje mogu implementirati, a uključuju razne metode umjetne inteligencije i strojnog učenja s ciljem poboljšanja poslovanja.

Ostale tehnologije koje Microsoft nudi, a da podržavaju implementaciju i razvoj prediktivnih modela i strojnog učenja su na primjer **Microsoft Bot Framework**⁶ alat koji služi za razvoj chatbotova, ali također omogućuje korištenje strojnog učenja budući da je u ovom području ono posebno važno. Nadalje, **Microsoft Azure Notebooks**⁷ je alat u oblaku baziran na sustavu Jupyter Notebook servisa koji omogućuje razvoj, testiranje i dijeljenje modela strojnog učenja.

Microsoft svojim korisnicima nudi pregršt alata za implementaciju strojnog učenja i prediktivnih modela, ali sam izbor alata može biti prilično izazovan. Kako bi se korisnik odlučio za tehnologiju koja najviše odgovara njegovim potrebama potrebno je napraviti analizu dostupnih tehnologija, jasno definirati ciljeve i problem koje će model rješavati te sukladno budžetu i vremenu odabrati tehnologiju koja mu najviše odgovara. Najveća snaga Microsoft tehnologija leži upravo u njihovom ekosustavu i infrastrukturi. Microsoft svoje proizvode prilagođava korisnicima pri tome cijelo vrijeme osluškujući potrebe tržišta. Brza ažuriranja i popravci, personalizacija, korisnička podrška, interoperabilnost, mogućnost

⁵ <https://dynamics.microsoft.com/en-us/>

⁶ <https://dev.botframework.com/>

⁷ <https://visualstudio.microsoft.com/vs/features/notebooks-at-microsoft/>

integracije servisa i usluga prema potrebama korisnika, a pogotovo nakon pojave Azure platforme objašnjava zašto se sve veći broj korisnika odlučuje za implementaciju Microsoft rješenja.

6. Praktični dio rada

Praktični dio rada prikazuje cjelokupan proces izrade modela strojnog učenja koji započinje prikupljanjem sirovih podataka, a završava puštanjem razvijenog modela na korištenje. U radu je kreiran prediktivni model za predviđanje broja poena koje će neki igrač ostvariti na košarkaškoj utakmici. Model je kreiran pomoću Azure Machine Learning (Azure ML) alata unutar Microsoft Azure platforme, a za čišćenje podataka (eng. *data cleansing*) je korišten Python programski jezik. U nastavku je prikazan proces izrade modela strojnog učenja kroz sljedeće faze:

- Prikupljanje, čišćenje i normalizacija podataka
- Pripremanje Azure ML razvojnog okruženja
- Učitavanje podataka
- Razvoj modela
- Evaluacija rezultata
- Puštanje modela u produkciju
- Testiranje

6.1. Prikupljanje, čišćenje i normalizacija podataka

Za potrebe rada skup podataka je preuzet sa stranice Kaggle⁸, a u prilogu radu se nalazi link na kojem je isti dostupan. Budući da se radi o modelu koji predviđa broj poena koje će igrač ostvariti skup podataka je veza za igračevu statistiku te sadrži relevantne podatke o igračevoj učinkovitosti.

Na slici ispod su prikazani sirovi podaci koji su pronađeni u preuzetom skupu podataka. Kao što je vidljivo na slici postoje prazni redci, polja koja nisu relevantna za model kao što je na primjer ime igrača, a postoji mogućnost da postoje dupli zapisi. Stoga je potrebno sirove podatke očistiti i transformirati u ulazni set podataka za treniranje odabranog algoritma strojnog učenja.

⁸ <https://www.kaggle.com/>

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
13	Kyle Anderson	PF	29	MIN	69	46	28.4	3.7	7.2	0.509	0.6	1.5	0.41	3	5.7	0.536	0.553
14	Giannis Antetokounmpo	PF	28	MIL	63	63	32.1	11.2	20.3	0.553	0.7	2.7	0.275	10.5	17.6	0.596	0.572
15	Thanasis Antetokounmpo	PF	30	MIL	37	0	5.6	0.5	1.2	0.435	0	0.2	0	0.5	1	0.526	0.435
16	Cole Anthony	PG	22	ORL	60	4	25.9	4.6	10.2	0.454	1.3	3.4	0.364	3.4	6.7	0.5	0.516
17	OG Anunoby	SF	25	TOR	67	67	35.6	6.3	13.2	0.476	2.1	5.5	0.387	4.2	7.7	0.539	0.556
18	Ryan Arcidiacono	PG	28	TOT	20	4	8.6	0.5	1.9	0.243	0.4	1.2	0.348	0.1	0.7	0.071	0.351
19	Ryan Arcidiacono	PG	28	NYK	11	0	2.4	0.1	0.5	0.2	0.1	0.3	0.333	0	0.2	0	0.3
20	Ryan Arcidiacono	PG	28	POR	9	4	16.2	0.9	3.6	0.25	0.8	2.2	0.35	0.1	1.3	0.083	0.359
21	Deni Avdija	SF	22	WAS	76	40	26.6	3.3	7.6	0.437	0.9	3.1	0.297	2.4	4.6	0.53	0.497
22	Deandre Ayton	C	24	PHO	67	67	30.4	7.8	13.2	0.589	0.1	0.4	0.292	7.7	12.9	0.597	0.592
23	Udoka Azubuike	C	23	UTA	36	4	10	1.6	2	0.819	0	0	0	1.6	2	0.819	0.819
24	Marvin Bagley III	C	23	DET	42	25	23.6	4.8	9.1	0.529	0.5	1.6	0.288	4.4	7.5	0.579	0.554
25	Patrick Baldwin Jr.	SF	20	GSW	31	0	7.3	1.4	3.5	0.394	1	2.7	0.381	0.4	0.8	0.44	0.541
26																	
27	LaMelo Ball	PG	21	CHO	36	36	35.2	8.2	20	0.411	4	10.6	0.376	4.2	9.4	0.45	0.51
28	Mo Bamba	C	24	TOT	49	7	15.7	2.4	4.9	0.485	1	2.5	0.387	1.4	2.4	0.59	0.585
29	Mo Bamba	C	24	ORL	40	6	17	2.7	5.4	0.495	1.1	2.7	0.398	1.6	2.7	0.594	0.596
30	Mo Bamba	C	24	LAL	9	1	9.8	1.2	3	0.407	0.6	1.8	0.313	0.7	1.2	0.545	0.5

Slika 15. Prikaz sirovih podataka

Čišćenje podataka je inicijalni korak u procesu analize podataka. Ova ključna operacija uključuje pripremu i provjeru podataka, obično se odvija prije osnovne analize u procesu. Čišćenje podataka se odnosi na uklanjanje pogrešnih i otkrivanje lažnih podataka te ispravak istih.

Odabrani set podataka sadrži sljedeće podatke:

Tablica 1. Popis atributa unutar početnog seta podataka

Atribut	Tip atributa	Opis
Igrač	string	Ime i prezime igrača (atribut koji će se u fazi čišćenja ukloniti)
Pos	string	Pozicija na kojoj igrač igra (kategorička varijabla)
Age	number	Starost igrača zaključeno s 1. Veljače 2023.
Tm	string	Ekipa u kojoj igrač igra (kategorička varijabla)
G	number	Broj utakmica koje je igrač odigrao
GS	number	Broj utakmica koje je igrač započeo
MP	number	Ukupan broj minuta koje je igrač proveo u igri
FG	number	Ukupan broj poena iz polja igrača
FGA	number	Ukupan broj šuteva iz polja igrača

FG%	number	Učinkovitost igrača iz polja (FG/FGA) igrača
3P	number	Broj zabijenih trica igrača
3PA	number	Broj šutiranih trica igrača
3P%	number	Učinkovitost igrača u šutu za tri poena (3P/3PA) igrača
2P	number	Broj zabijenih šuteva za dva poena igrača
2PA	number	Ukupan broj šuteva za dva poena igrača
2P%	number	Učinkovitost igrača u šutu za dva poena (2P/2PA) igrača
eFG%	number	Ukupna učinkovitost igrača $(FG+0.5*3)/FGA$ igrača
FT	number	Broj zabijenih slobodnih bacanja igrača
FTA	number	Ukupan broj šutiranih slobodnih bacanja igrača
FT%	number	Učinkovitost s linije slobodnih bacanja (FT/FTA) igrača
ORB	number	Prosječan broj napadačkih skokova igrača
DRB	number	Prosječan broj obrambenih skokova igrača
TRB	number	Ukupan prosječan broj skokova igrača
AST	number	Prosječan broj asistencija igrača
STL	number	Prosječan broj ukradenih lopti igrača
BLK	number	Prosječan broj šuteva koje je igrač blokirao
TOV	number	Prosječan broj lopti koje igrač gubi
PF	number	Prosječan broj osobnih pogrešaka koje igrač radi
PTS	number	Prosječan broj poena koje igrač zabija (traženi izlaz iz modela)

Svi podaci koji se nalaze u početnom skupu podataka se odnose na statistiku igrača u NBA ligi za sezonu 2022./2023. Odabrani skup podataka je posebno pogodan za daljnju implementaciju budući da sadrži gotovo sve bitne statističke elemente koji se prate na košarkaškim utakmicama te ulaze u službenu statistiku igrača. Što odabrani skup podataka čini lako dostupnima, preciznim, točnim i otvorenim za proširenja, ali i učitavanje novih podataka.

Podaci su očišćeni korištenjem mogućnosti Python programskog jezika te pandas i os biblioteka dok su prazni redovi ručno izbrisani iz datoteke. Na ispod prikazanom odsječku programskog koda prikazana je skripta za obradu sirovih podataka u skup podataka koji će se koristiti za razvoj modela. Program prvo čita .csv datoteku u kojoj se nalaze podaci nakon čega se popunjavaju vrijednosti koje nedostaju s nulom. Zatim se brišu redovi koji su duplikati te dvije kategoričke varijable u ovom slučaju "Pos" i "Tm" se transformiraju u kontinuirane varijable pomoću funkcije `get_dummies`.

```
import pandas as pd
import os

file_path= 'nba_data_processed.csv'
if os.path.exists(file_path):
    data= pd.read_csv(file_path)
    print("File found")
else:
    print(f"File not found.")

try:
    data = pd.read_csv('nba_data_processed.csv')
    data.fillna(value=0, inplace=True)
    print("Rows without values modified")

    data.drop_duplicates(inplace=True)
    print("Duplicates dropped")

    data=pd.get_dummies(data, columns=['Pos'])
    data=pd.get_dummies(data, columns=['Tm'])
    print("Categorical data modified")

    from sklearn.preprocessing import MinMaxScaler
    scaler = MinMaxScaler()
    data[data.columns] = scaler.fit_transform(data[data.columns])

    data.to_csv('processed_nba_data.csv', index=False)
    print("Data modified and saved")
except Exception as e:
    print(f"An error occurred: {str(e)}")
```

Programski kod 1. Čišćenje podataka

Nadalje, kako bi se osigurala što veća moguća preciznost i izbjeglo da veliki ulazni parametri dominiraju nad algoritmom provedena je i normalizacija skupa podataka MiniMax metodom. Postupak MiniMax normalizacije je objašnjen u prethodnim poglavljima.

processed_nba_data - Excel

Pretraži

Karlo Gardijan

DatotekaPolaznoUmetanjeCrtanjeRaspored straniceFormulePodaciPregledPrikazAutomatizirajPomoć

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje

Umetanje</

Slika 16. Snimka zaslona, pripremljen ulazni skup podataka

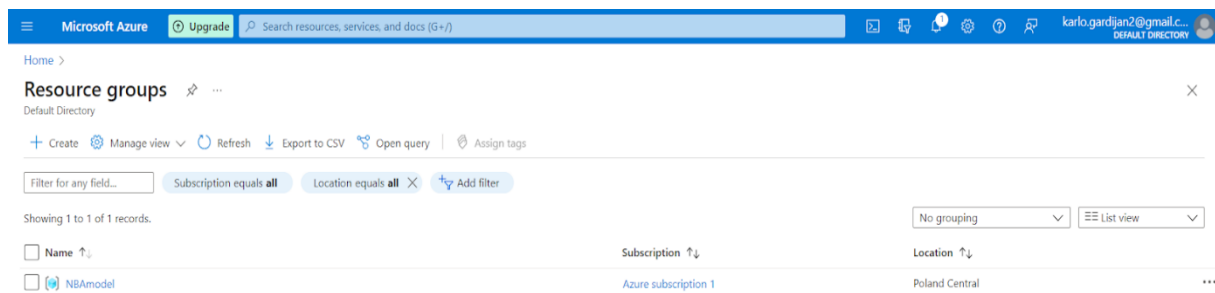
Očišćen i normaliziran skup podataka koji će se koristiti kao ulaz u sustav. Kao što je vidljivo na slici svi podaci su u rasponu između 0 i 1.

6.2. Azure ML i izrada prediktivnog modela

Azure ML je alat u oblaku (eng. *cloud*) unutar Microsoft Azure platforme koji služi za implementaciju strojnog učenja i njenu integraciju s ostalim Microsoft uslugama i proizvodima unutar Azure platforme te je pomoću njega izrađen model za potrebe ovog rada. Također, Azure ML omogućuje integraciju i s ostalim programskim proizvodima koji nisu nužno dio Azure platforme kao što su na primjer razni web servisi.

6.2.1. Priprema Azure okruženja

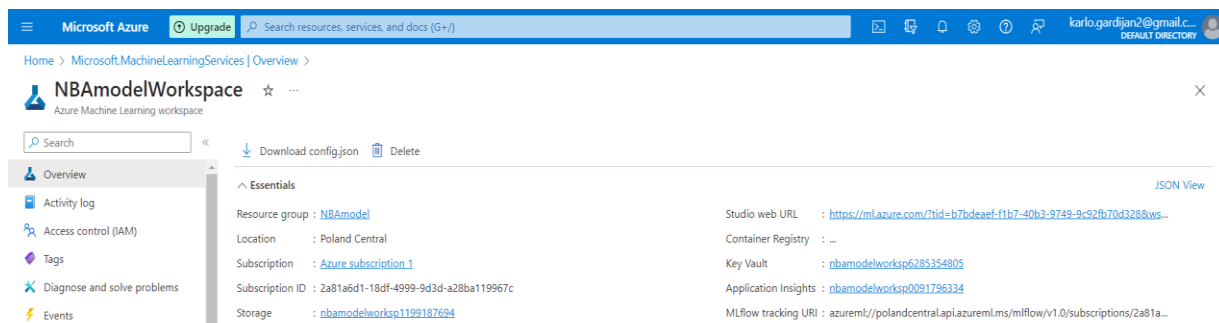
Prije nego započnemo s radom unutar Azure ML-a potrebno je pripremiti okruženje za rad i razvoj. Priprema Azure okruženja započinje izradom Azure korisničkog računa gdje korisnik odabire tip pretplate i namjenu za koju će se račun koristiti. Idući korak je pregledavanje same Azure platforme i upoznavanje s njenim servisima. Kako bi se mogli upoznati sa servisima prvo moramo kreirati grupu resursa.



Slika 17. Kreiranje grupe resursa

Grupa resursa je u stvari kontejner (eng. *container*) koji grupira sve resurse koje koristimo prilikom rada s Azure servisima u jednu logičku cjelinu te sačinjava osnovni dio rada s Azure platformom.

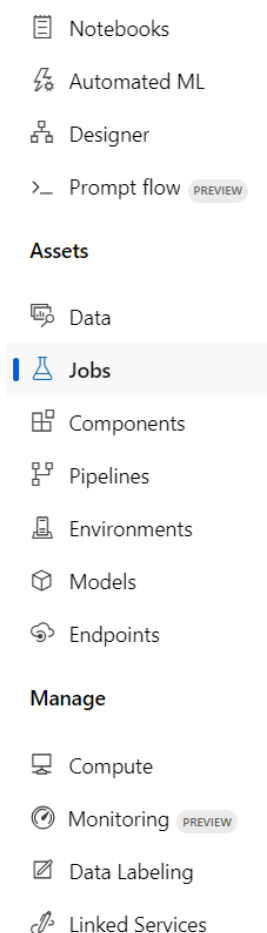
Nakon kreiranje grupe resursa još je potrebno stvoriti radni prostor (eng. *workspace*) nakon čega je moguće započeti s radom unutar Azure Machine Learning Studija. Radni prostor je potrebno kreirati kako bi sustav mogao upravljati našim projektom. Sustav automatski alocira potrebne resurse za rad na projektu, upravlja pravima pristupa, postavlja razvojno okruženje, upravlja korisničkom pretplatom s ciljem praćenja potrošnje resursa koji se naplaćuju i slično.



Slika 18. Kreiranje radnog prostora

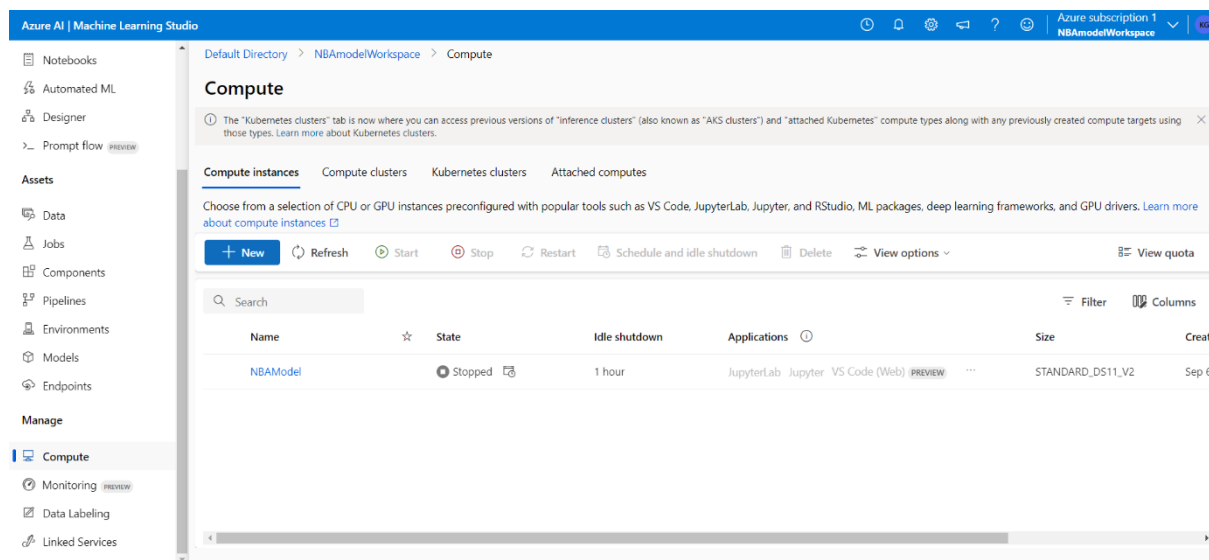
Kreiran je radni prostor unutar prethodno kreirane grupe resursa s lokacijom „*Poland Central*“ budući da je to najbliža lokacija Hrvatskoj koju ovaj alat nudi. Nakon što je na ovaj način kreiran radni prostor moguće je pristupiti Azure Machine Learning Studiju iz kojeg će se dalje razvijati prediktivni model. Nakon čega započinjemo s radom u Azure ML Studio alatu.

Na slici ispod prikazana je alatna traka Azure ML Studio alata u kojem je izrađen prediktivni model za potrebe rada. Alatna traka se sastoji od više funkcionalnosti koje su potrebne za implementaciju strojnog učenja. Funkcionalnosti koje su korištene u nastavku rada su redom prema pojavljivanju u alatnoj traci. „*Data*“ pomoću koje je učitani početni skup podataka. „*Jobs*“ funkcionalnost unutar koje su se zakazivali poslovi i izvođenje kreiranih modela. Također unutar ove funkcionalnosti je moguće pratiti uspješnost izvođenja svakog pojedinog modela kao i evaluirati njegove rezultate. „*Pipelines*“ je funkcionalnost unutar koje su izgrađeni modeli. „*Endpoints*“ se koristi za praćenje modela koji su objavljeni i prikupljanje potrebnih informacija za povezivanje na model. „*Compute*“ funkcionalnost unutar koje se priprema virtualno okruženje za razvoj modela.



Slika 19. Alatna traka Azure ML Studio alata

Sljedeći korak u postavljanju okruženja je kreiranje virtualne instance koja simulira računalo i radnu okolinu unutar koje se model razvija. Kako je vidljivo na slici ispod instance je trenutno zaustavljena te je definirano da se pokreće isključivo ručno te da se nakon pokretanja automatski isključuje nakon jednog sata rada. Ovo je napravljeno kako bi se sačuvali računalni resursi od nepotrebnog zauzimanja i optimizirale performanse sustava.

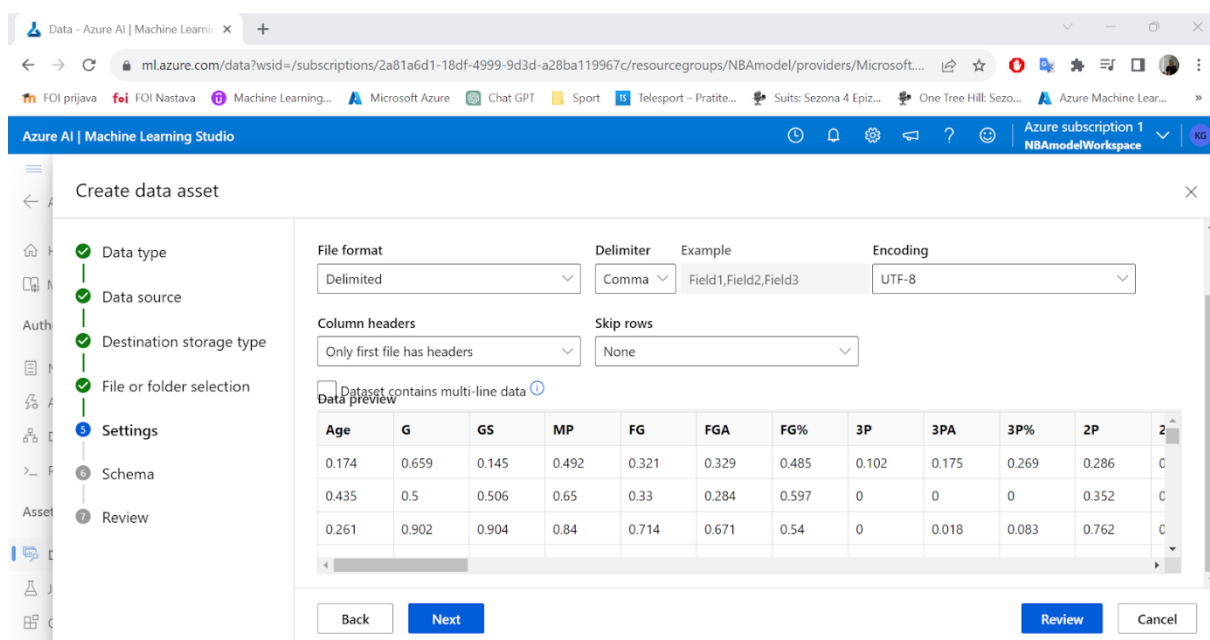


Slika 20. Kreirana virtualna instance razvojnog okruženja

6.2.2. Učitavanje podataka

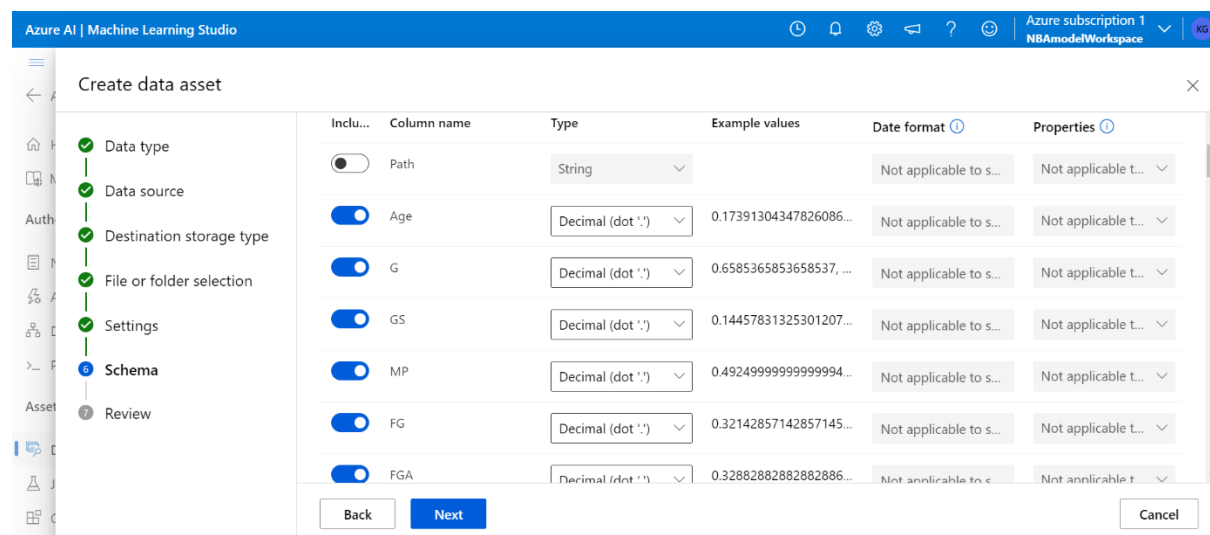
Izrada prediktivnog modela započinjem učitavanjem podatkovnog skupa u Azure ML. U ovom slučaju se učitavaju dva različita skupa podataka. Prvo se učitava normalizirani skup podataka, a nakon toga očišćen skup podataka koji nije normaliziran. Učitavaju se oba skupa kako bi se u procesu evaluacije rješenja odabrao najbolji model za predviđanje.

Na slici ispod je prikazana snimka zaslona Azure ML alata koja prikazuje proces učitavanja skupa podataka u Azure ML. Na prikazanoj snimci zaslona je vidljiv ekran s odabirom delimitera koji se koristi kod zapisa podataka. Zatim vrsta enkodiranja što je u ovom slučaju UTF-8 te odabir datoteka u kojima se nalaze nazivi atributa. Budući da za potrebe rada dodajemo dvije .csv datoteke, ali odvojeno ovdje bismo opciju u kojoj nazive atributa dobijemo iz prve datoteke. Nazivi atributa u obje datoteke su identični stoga budući da se radi o istim atributima, a razlikuju se samo po tome jesu li normalizirani ili ne.



Slika 21. Snimka zaslona Azure ML, učitavanje podatkovnog skupa

Sljedeći korak u postupku učitavanja podataka je odabrati podatke koji se učitavaju u podatkovni skup. Azure ML ovdje nudi razne mogućnosti kao što je mogućnost odabira samo onih atributa koji su potrebni. Nadalje, važno je provjeriti i točno odrediti tip podatka svakog atributa koji učitavamo budući da pogrešno definiran tip podataka može kasnije uzrokovati greške u modelu.



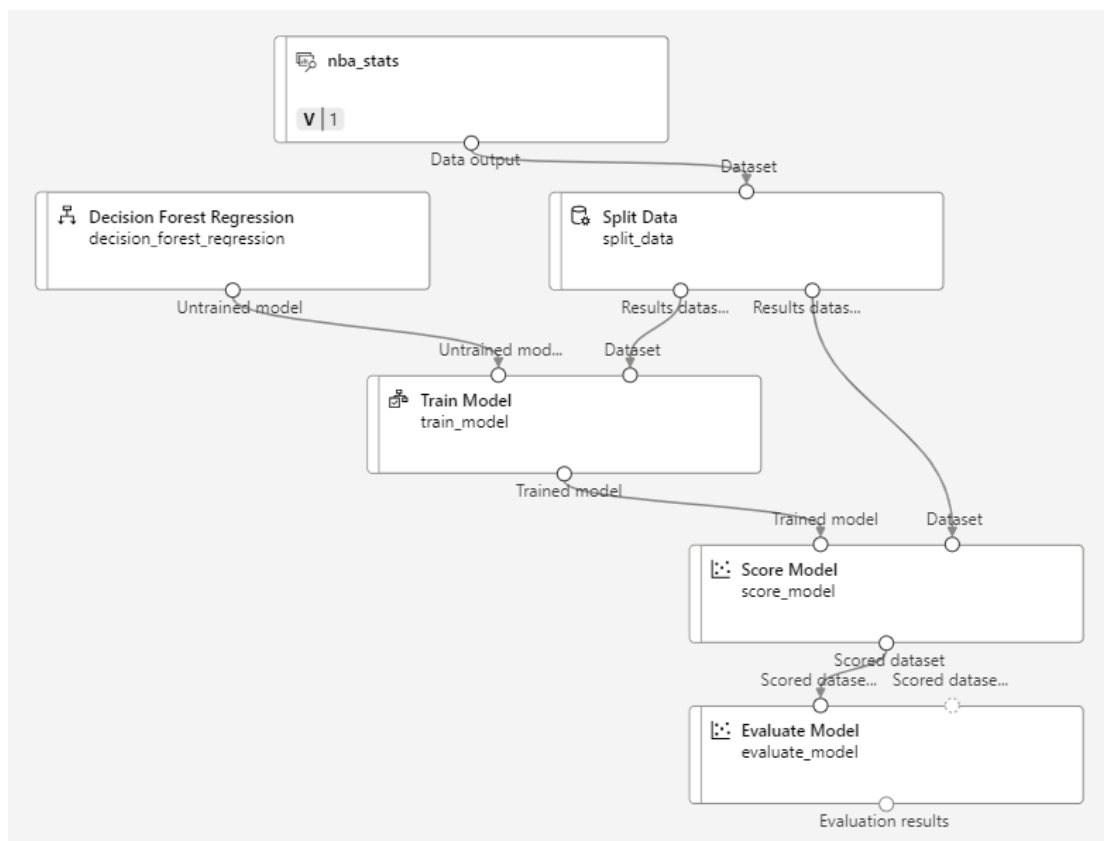
Slika 22. Odabir podataka koji se učitavaju u skup podataka

Azure ML omogućuje učitavanje raznih izvora kao što su web mjesta, baze podataka, skladišta podataka, lokalne mape, ali i lokalne datoteke. Također omogućuje i integraciju s ostalim servisima dostupnim u Azure platformi kao što je Azure Synapse Analytics. Za potrebe rada učitane su dvije lokalne .csv datoteke.

6.2.3. Izrada modela

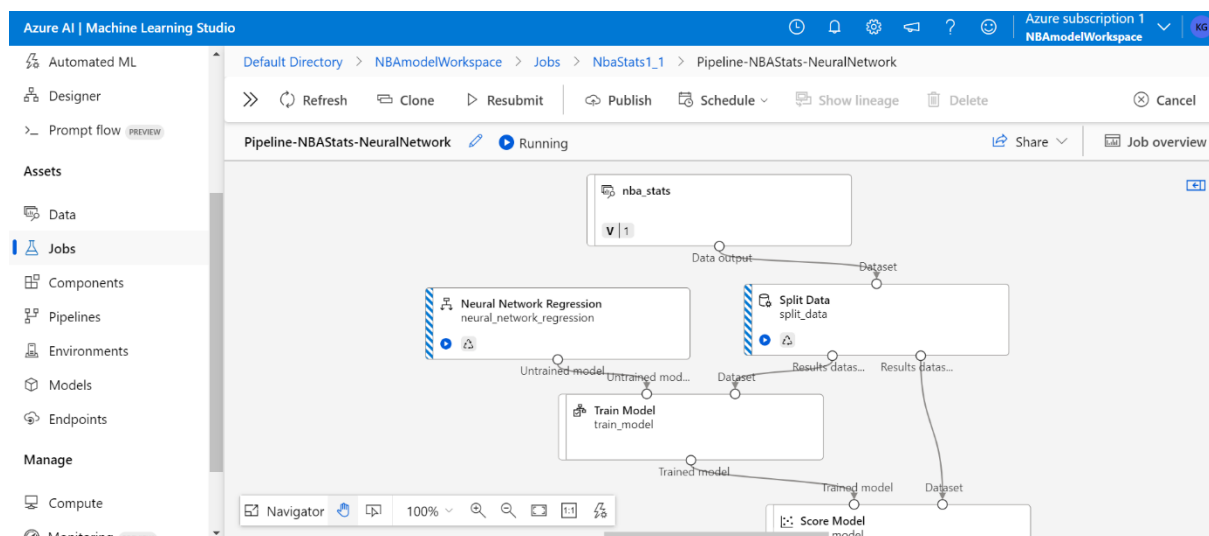
Sljedeći korak u procesu kreiranja modela je razvoj puta (eng. *pipeline*) kojim ćemo trenirati model. Alat nudi širok spektar mogućnosti i transformacija koje je moguće napraviti nad podacima kako bi se kreirao precizan model. Samo sučelje alata je poprilično intuitivno i lako za korištenje. Svi elementi su u grafičkom obliku te za kreiranje modela nije potrebno pisanje programskog koda. Transformacije je potrebno slijedno posložiti i pravilno povezati.

Na slici ispod vidljiv je grafički prikaz transformacija koje su provedene nad ulaznim podacima kako bi se dobio krajnji model. Kao što je vidljivo prvo je učitani skup podataka, zatim su podaci podijeljeni u skup za treniranje i skup za testiranje modela. Odabran je algoritam strojnog učenja pomoću kojeg su podaci trenirani, a nakon uspješnog treniranja modela isti je testiran na testnom skupu podataka te su evaluirani rezultati testiranja.



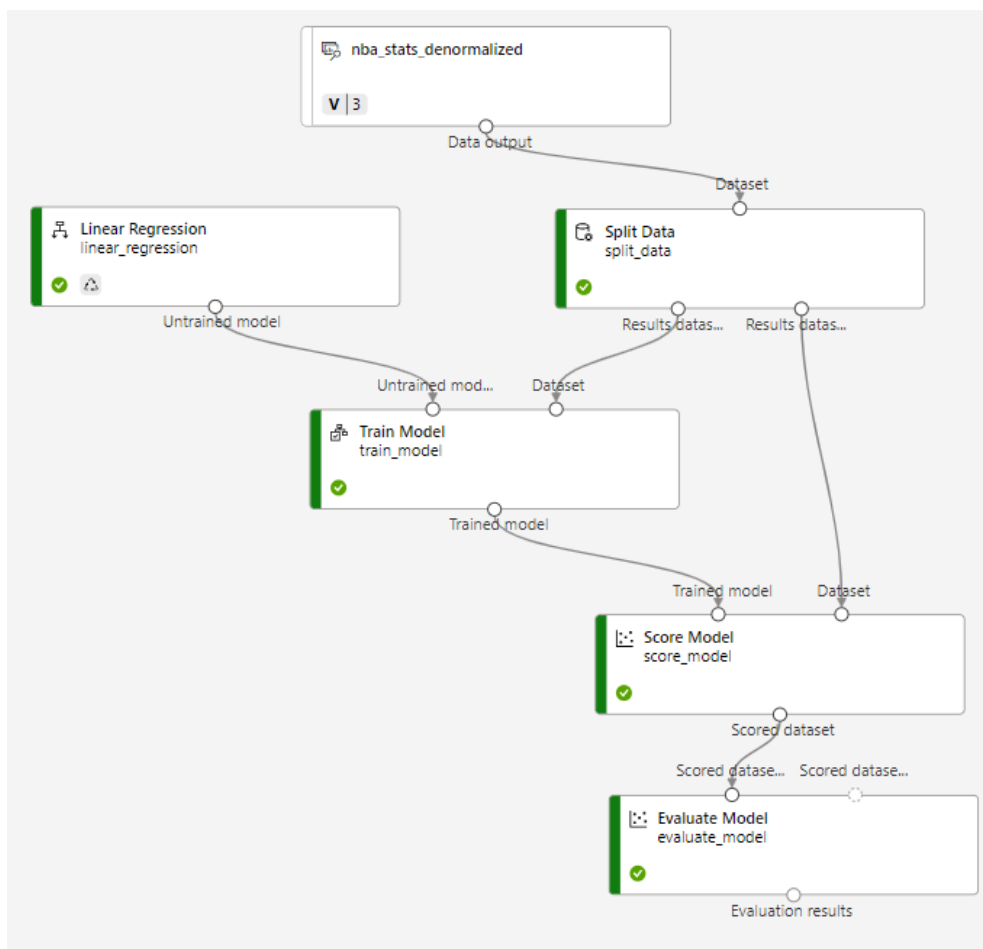
Slika 23. Snimka zaslona iz Azure ML, prikaz procesa izrade modela

Na slici ispod je prikazan proces treniranja modela strojnog učenja. Nakon što je model generiran potrebno je provjeriti njegovu ispravnost te ako sustav ne pronade greške moguće je započeti testiranje modela. Testiranje može potrajati ovisno o ukupnoj veličini podatka na kojima se model trenira.



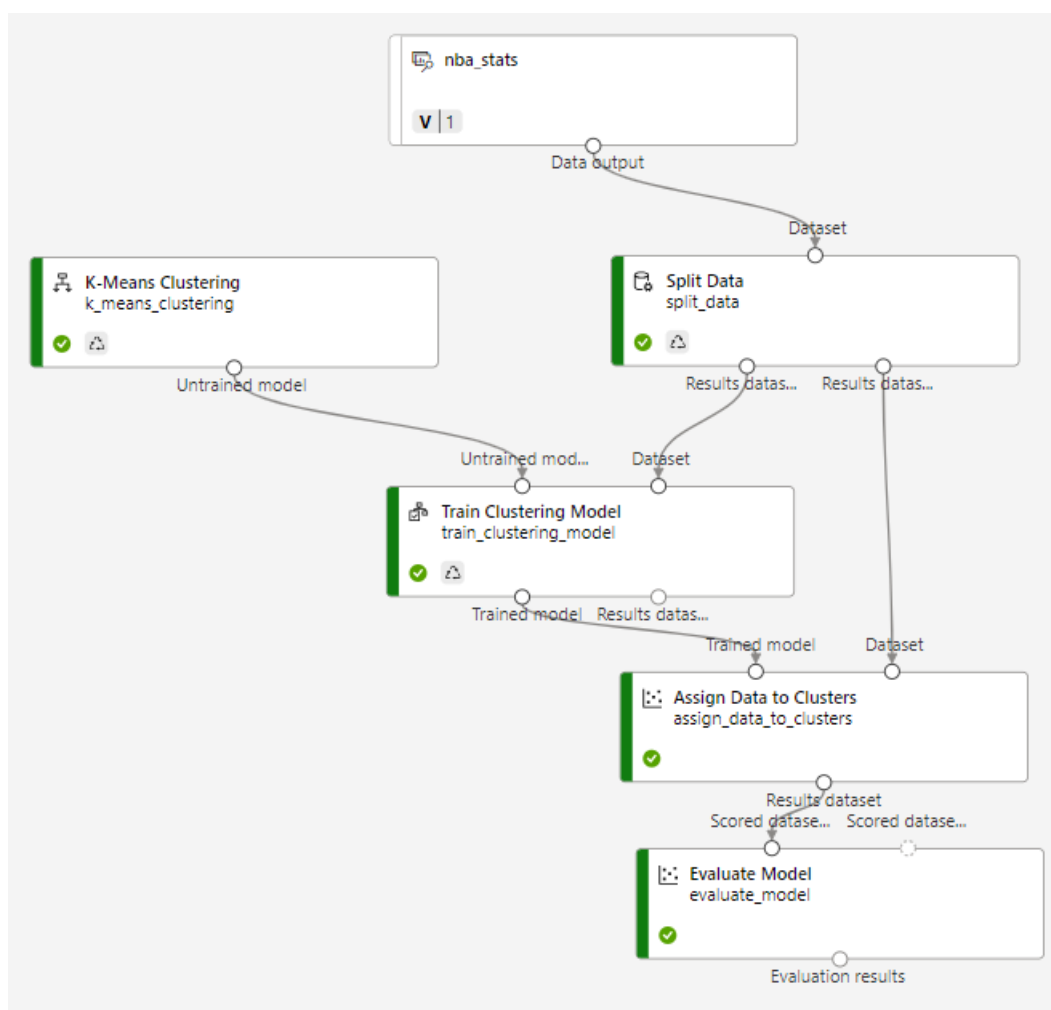
Slika 24. Treniranje modela strojnog učenja

Ako je testiranje uspješno provedeno moguće je provjeriti rezultate testiranja te evaluirati snagu modela. Nakon uspješno završenog testiranja potrebno je osvježiti prozor kako bi se prikazao uspješno treniran model kako je prikazano na slici ispod.



Slika 25. Uspješno treniran model

Prikazani model se odnosi na algoritam linearne regresije, a za potrebe rada su još izrađeni modeli koji su trenirani algoritmima neuronskih mreža, slučajne šume i K-sredina koji se nešto razlikuje budući da je riječ o drugoj vrsti algoritma stoga će on biti prikazan u nastavku rada. Dijagrami s ostalim algoritmima nisu prikazani u radu budući da su kreirani po istom principu, a razlikuju se samo po odabranom algoritmu. Iznimka je jedino algoritam K-sredina budući da se radi o algoritmu klasteriranja dok su ostali algoritmi regresijski stoga se njegov model ipak razlikuje od ostalih pa je on prikazan u nastavku.



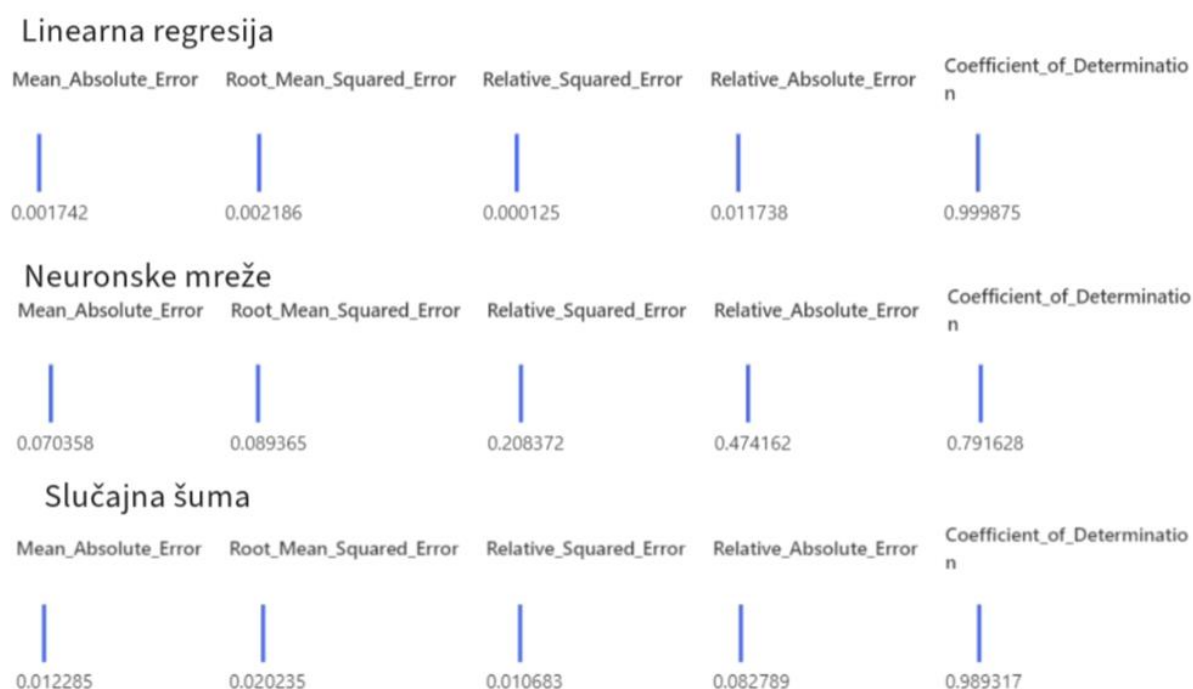
Slika 26. Trenirani model algoritmom K-sredina

Prikazan je trenirani model pomoću algoritma K-sredina. Kako je vidljivo na dijagramu ovaj model koristi drukčije elemente u odnosu na prethodno kreirane modele. K-sredina je klasifikacijski algoritam dok su ostali algoritmi koji su korišteni za izgradnju modela regresijski. Stoga ne čudi kako je potrebno koristiti drukčije elemente kako bi se kreirao model. Konkretno u prikazanom primjeru najvažniji element kojeg je potrebno dodati je „*Assign Data to Clusters*“, a osim njega potrebno je koristiti „*Train Clustering Model*“ umjesto običnog modela za treniranje.

6.2.4. Evaluacija rezultata

Kako je u prethodnom poglavlju navedeno kreirana su četiri modela s ciljem dokazivanja važnosti odabira pogodnog algoritma za svaki pojedinačni problem. Važno je za napomenuti kako prilikom strojnog učenja ne postoji nijedan algoritam koji je uvijek bolji od ostalih već rezultati ovise o problemu koji se rješava te ulaznim podacima što je i potvrđeno evaluacijom algoritama.

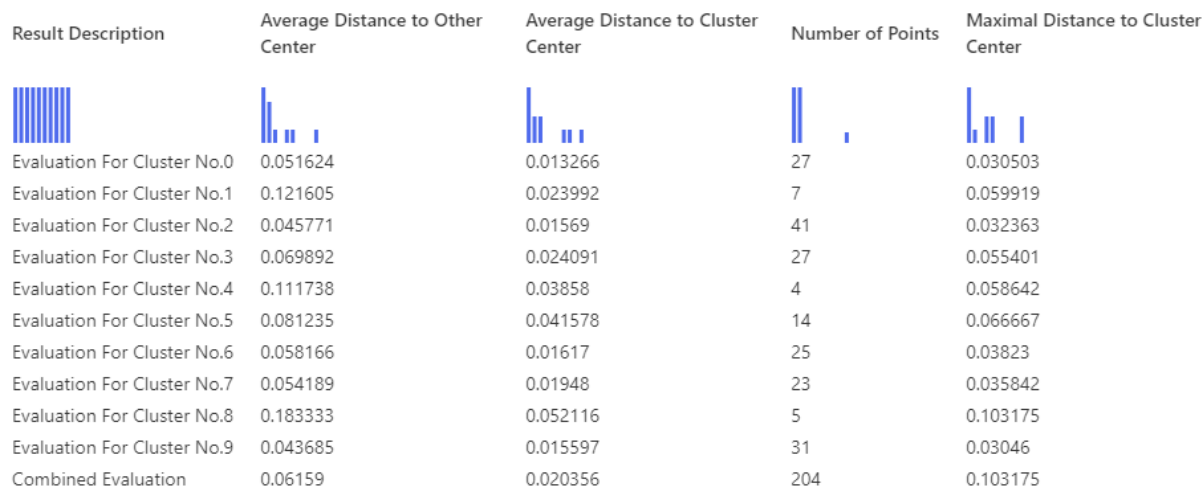
Za konkretan problem koji ovaj rad rješava možemo pretpostaviti kako će linearna regresija biti najbolji odabir budući da možemo pretpostaviti da postoji linearna ovisnost između ulaznih parametar i izlazne varijable. Na primjer ako košarkaš ima veću minutažu za pretpostaviti je da će postići veći broj poena od igrača koji ne igra mnogo ili na primjer ako igrač uzima veliki broj šuteva možemo pretpostaviti da će zabiti više poena od igrača koji ne šutira mnogo.



Slika 27. Evaluacija rezultata, regresijski algoritmi

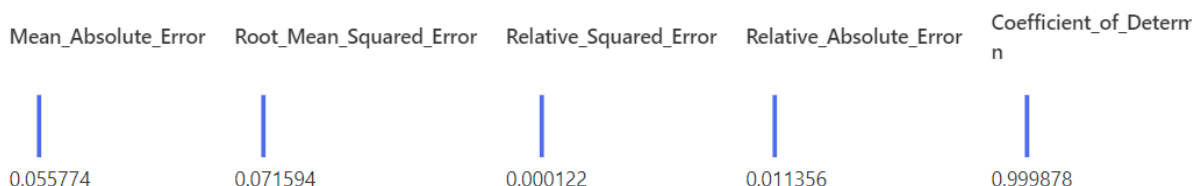
Kako je vidljivo iz rezultata evaluacije algoritama prikazanim na slici iznad algoritam koji dominira u ovom slučaju je linearna regresija što potvrđuje pretpostavku koja je ranije u radu definirana. Najslabije rezultate u ovom slučaju postiže algoritam neuronskih mreža što je bilo i za očekivati budući da se je skup podataka za treniranje poprilično malen, a neuronske mreže su algoritam koji daje točnije rezultate kada se trenira na velikim skupovima podataka. Neuronske mreže bi pružile bolji rezultat kada bi se koristio prošireniji i veći skup podataka. Na primjer ako bi pratili i defenzivne karakteristike igrača i timova, aktualnu formu igrača, proširili skup sa starijim podacima, ali možda i dodali još nekoliko liga.

Algoritam K - sredina moramo analizirati odvojeno od regresijskih algoritama budući da se radi o algoritmu klasifikacije stoga i rezultati evaluacije izgledaju nešto drukčije. Točna razlika između ova dva tipa algoritama je objašnjena u prethodnim poglavljima.



Slika 28. Evaluacija rezultata, metodom K-sredina

Iz rezultata algoritam K-sredina koji je u ovom slučaju testiran na 10 centara vrijednosti od kojih su neki pružili bolje, a neki lošije rezultate. Na primjer klaster broj 2 pruža najbolje rezultate dok klaster 4 ima jako slab rezultat. Rezultat ovog algoritma bi se popravio kada bi se korigirao broj klastera na manji broj, ali se može pretpostaviti kako bi teško postigao rezultate algoritma linearne regresije.



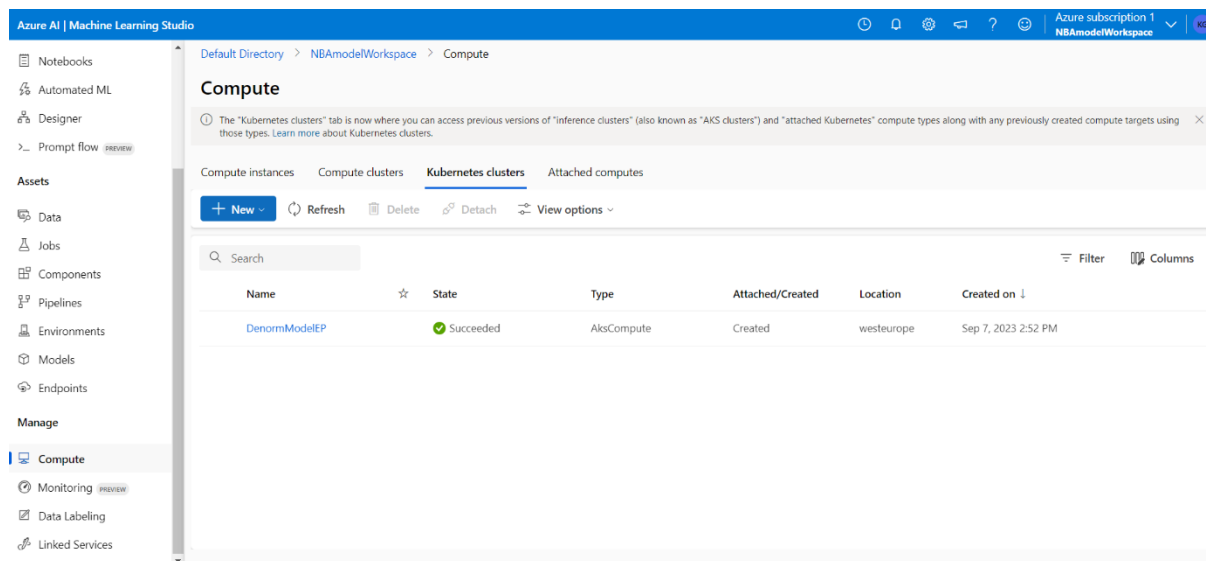
Slika 29. Evaluacija rezultata nenormaliziranih podataka metodom linearne regresije

Zbog svega navedenog u daljnjem radu će se koristiti algoritam linearne regresije i njegov model budući da je pružio najbolje rezultate. Dosada testirani podaci su se odnosili na normaliziran skup podataka, a sada će se isti proces napraviti i s podacima koji nisu normalizirani. Budući da se radi o ulaznom skupu podataka koji se odnosi na košarkašku statistiku pojedinog igrača podaci nemaju širok raspon vrijednosti te možemo pretpostaviti da bi algoritam mogao dati dobre rezultate i bez prethodne normalizacije što bi kasnije moglo olakšati korištenje algoritma.

Kao što je vidljivo na slici iznad algoritam je vratio prilično dobre rezultate koji su ipak nešto lošiji od normaliziranog modela, ali razliku u ovom slučaju možemo pa i zanemariti.

6.2.5. Puštanje modela u produkciju

Sljedeći korak u procesu dizajniranja modela je puštanje istog u produkciju. Odnosno omogućiti korištenje kreiranog modela korisnicima i integraciju s ostalim programskim proizvodima. Prvi korak u ovom postupku je kreiranje nove instance Kubernetes klastera koja je zadužena za upravljanje kontejnerom za orkestraciju servisa koji omogućuju puštanje modela u produkciju.

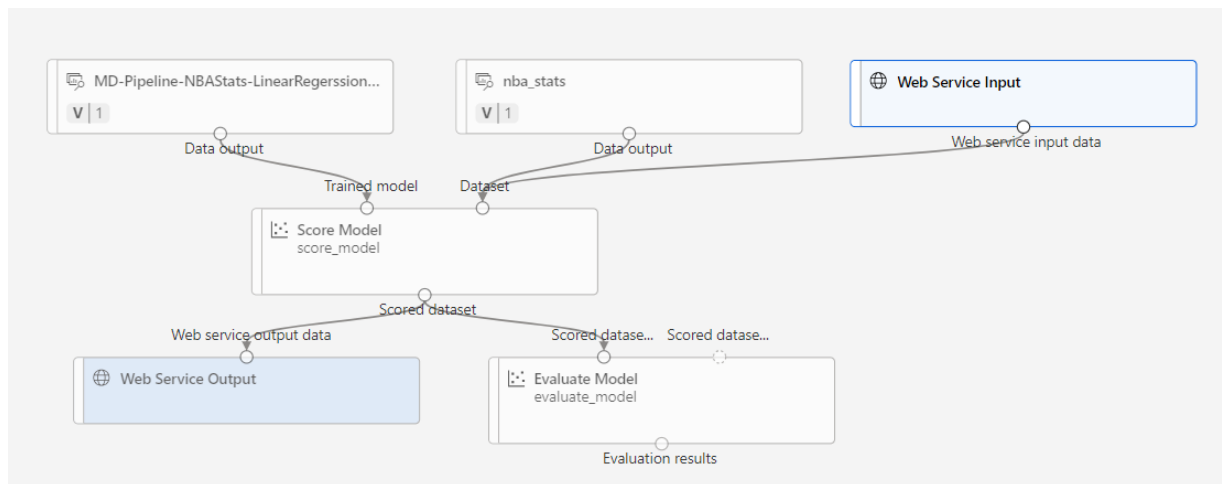


Slika 30. Kreiran AKS klaster

Kako je prikazano na slici ispod uspješno je kreiran Kubernetes klaster. U konkretnom primjeru je kreiran Azure Kubernetes servis (eng. *Azure Kubernetes Service*, AKS) pomoću koji je objavljen odabrani model.

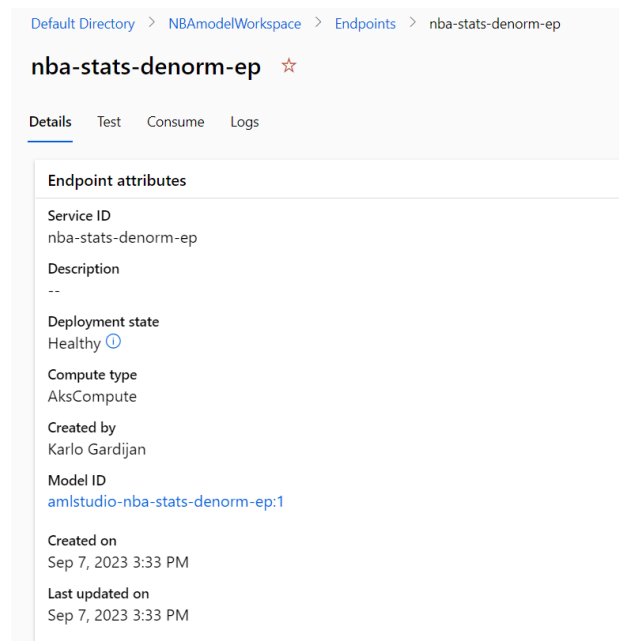
Sljedeći korak u ovom postupku je kreiranje novog dijagrama puta (eng. *inference pipeline*) koji omogućuje dodavanje ulaznih podataka u model, ali i ispis novih predviđanja.

Na slici ispod je prikazana novokreirana struktura modela koja omogućuje primanje ulaznih podataka s raznih web mjesta, ali i ispis predviđanja. Ova arhitektura omogućuje povezivanje na sustav s raznih poslužitelja i integraciju s drugim programskim proizvodima te ispisivanje rezultata predviđanja. Novokreirani model prima dvije vrste podataka, a to su podaci koji se nalaze u našem početnom skupu podataka i podatke koje korisnik unosi.



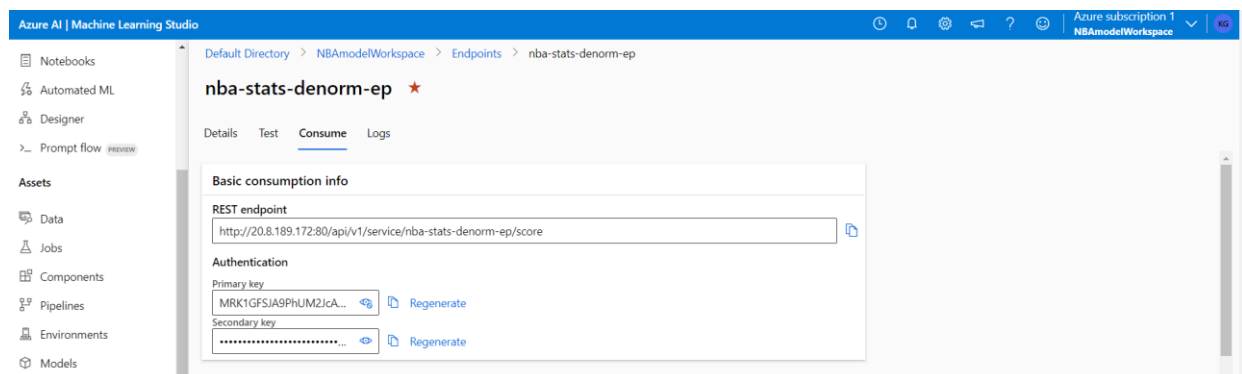
Slika 31. Kreiran *inference pipeline*

Sljedeći korak je objaviti model odnosno kreirati kranju točku (eng. *endpoint*) koja će ostalim programskim proizvodima omogućiti spajanje s kreiranim modelom i njegovo nesmetano korištenje. Kreiranje krajnje točke ima za cilj pripremiti okruženje za objavljivanje modela na način da se za model kreira jedinstveni identifikator i token za autentifikaciju pomoću kojih će ostali servisi i usluge po potrebi komunicirati s odabranim modelom.



Slika 32. Uspješno objavljen model spreman za korištenje

Kreirani model je uspješno obavljen i spreman za korištenje i integraciju s drugim programskim proizvodima. Potrebno je obratiti pažnju na opciju „*Deployment status*“ budući da njena vrijednost neposredno nakon objavljivanja modela još neko vrijeme zadržava vrijednost „*In transition*“ jer za to vrijeme nije moguće pristupiti modelu. Modelu se može pristupiti tek nakon što navedena opcija poprimi vrijednost „*Healthy*“.



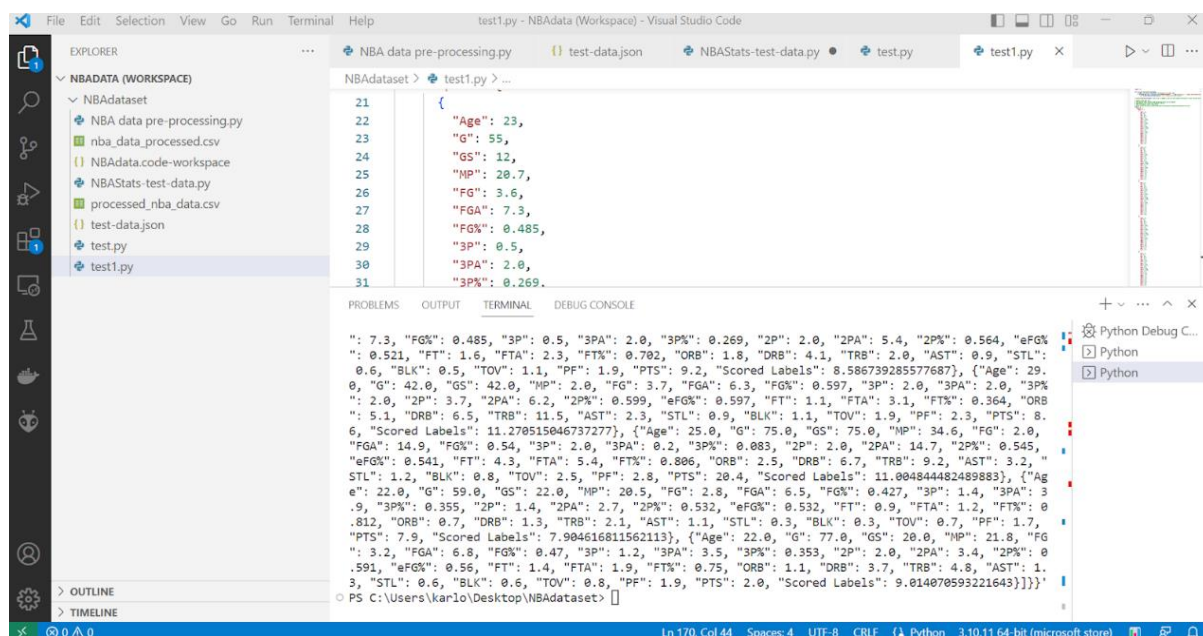
Slika 33. Podaci za integraciju modela

Nakon što je model uspješno objavljen sustav je generirao jedinstveni URL preko kojega je moguće pristupiti modelu te je generiran token za autentifikaciju korisnika. Ovo je posebno važno budući da se pomoću ovih elemenata omogućuje korištenje modela. Odnosno spajanje na model i integraciju s drugim programskim proizvodima. U nastavku rada je prikazano kako se ova dva podatka koriste za pristup modelu pomoću skripte izrađene u Python programskom jeziku.

6.2.6. Testiranje modela

Zadnji korak u procesu razvoja funkcionalnosti predviđanja pomoću strojnog učenja je testiranje konekcije na objavljeni model i provjera rezultata. Konekcija je testirana pomoću skripte napisane Python programskim jezikom u kojoj je kao ulazni podataka u sustav kreiran novi, do sada nepoznat igrač te se od algoritma očekuje predviđanje prosječnog broja poena koje će novi igrač ostvariti u idućoj sezoni.

Na slici iznad je prikazana snimka zaslona iz razvojnog okruženja Visual Studio Code u kojem je pomoću Python programskog jezika testirana skripta za konekciju i testiranje kreiranog prediktivnog modela.



Slika 34. Testiranje objavljenog modela i stvaranje predikcija

Model je kreirano predviđanje za pet novih igrača. Rezultat koji je model vratio se odnosi na prosječan broj poena koje će svaki od igrača ostvariti, a oni su redom: 8.58, 11.27, 11, 7.9 i 9.01 poena po utakmici. Za svakog igrača su postavljeni različiti ulazni parametri koji do sada nisu poznati modeli odnosno nisu se nalazili u ulaznom skupu podataka.

Na ispod prikazanom odsječku programskog koda vidljivi su ulazni podaci za jednog novog igrača kojem je model kreirao predviđanje da će postići 9.02 poena u prosjeku po utakmici te `url` i `api_key` za povezivanje na model. Cijela skripta je dostupna na Github repozitoriju koji se nalazi u prilogu rada.

```

{
  "Age": 22,
  "G": 77,
  "GS": 20,
  "MP": 21.8,
  "FG": 5.2,
  "FGA": 7.8,
  "FG%": 0.47,
  "3P": 1.2,
  "3PA": 4.5,
  "3P%": 0.353,
  "2P": 3.0,
  "2PA": 4.4,
  "2P%": 0.591,
  "eFG%": 0.56,
  "FT": 1.4,
  "FTA": 1.9,
  "FT%": 0.75,
  "ORB": 1.1,
  "DRB": 3.7,
  "TRB": 4.8,
  "AST": 2.3,
  "STL": 1.6,
  "BLK": 0.6,
  "TOV": 2.8,
  "PF": 1.9,
  "PTS": 3.0
}
]
},
"GlobalParameters": {}
}
...
body = str.encode(json.dumps(data))
url = 'http://20.8.189.172:80/api/v1/service/nba-stats-denorm-ep/score'
api_key = 'MRK1GFSJA9PhUM2JcAyxRgGgZieIyJLD'

```

Programski kod 2. Testiranje objavljenog modela

Testiranjem objavljenog modela završava proces razvoja funkcionalnosti predviđanja pomoću algoritama strojnog učenja te kreirani modeli postaju dostupni za integraciju u programske proizvode. To nikako ne znači da u ovom trenutku prestaje razvoj modela te da se nakon njegove implementacije u programski proizvod završava rad na modelu. U sklopu životnog ciklusa modela nakon njegove integracije u razne programske proizvode spada i njegovo održavanje te dodatna optimizacija modela.

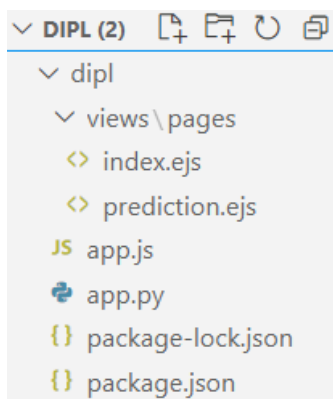
S vremenom može doći i do pretreniranosti modela na što treba posebno paziti, a upravo s ciljem izbjegavanja takvih situacija je potrebno održavati model, ali ovaj rad se time neće detaljnije baviti. Potrebno je naglasiti kako je rad i razvoj prediktivnih modela kontinuiran proces koji započinje odlukom za implementaciju funkcionalnosti predviđanja

unutar programskog proizvoda, a završava u trenutku kada završava životni ciklus programskog proizvoda koji koristi model.

Nadalje, još jedna važna stavka koja je možda do sada bila i zapostavljena u radu je pisanje dokumentacije. Za vrijeme izrade cjelokupnog modela potrebno je uredno i pravovremeno dokumentirati proces razvoja modela kako bi se njime što lakše i jednostavnije upravljalo kroz trajanje cjelokupnog životnog ciklusa programskog proizvoda.

6.2.7. Integracija modela u programski proizvod

Nakon što je prediktivni model uspješno obavljen i testiran slijedi integracija modela u izrađeni programski proizvod. Programski proizvod je jednostavna web stranica izrađena pomoću Node.js platforme i Python programskog jezika, Flask razvojnog okruženja za izradu web aplikacija. Izrađena web stranica koristi Node.js za primanje korisnikovog unosa, ali i ispis rezultata na zaslonu dok je pomoću Pythona implementirana komunikacija s kreiranim modelom. Kod za izradu web stanice, ali i Python programskog rješenja za povezivanje na prediktivni model je dostupan unutar Github repozitorija koji se nalazi u prilogima radu.



Slika 35. Organizacija repozitorija

Prilikom spajanja na web stranicu korisnik unosi podatke za kreiranje predviđanja odnosno igračevu statistiku. Izgled web forme za unos podataka je prikazan na slici ispod.

The screenshot shows a web browser window with the title "NBA Stats Prediction". The address bar shows "localhost:3000". The form is titled "NBA Stats Prediction" and contains the following input fields with their respective values:

Field	Value
Age:	23
G:	55
GS:	23
MP:	20.7
FG:	3.6
FGA:	7.3
FG%:	0.485

Slika 36. Forma za upis podataka, prvi dio

Korisnik unosi igračevu statistiku jedan po jedan element dok se ne unesu svi potrebni podaci kako bi se predviđanje moglo uspješno kreirati. Budući da aplikacija od korisnika očekuje unos igračeve statistike svi podaci moraju biti numerički. Nakon što je korisnik unio sve potrebne podatke klikom na gumb započinje slanje podataka prema modelu te se kreira predviđanje.

The screenshot shows the second part of the data entry form. The input fields and their values are:

Field	Value
TRB:	2.8
AST:	0.9
STL:	0.6
BLK:	0.5
TOV:	1.1
PF:	1.9
PTS:	9.2

At the bottom of the form is a blue button labeled "OK".

Slika 37. Forma za upis podataka, drugi dio

Nakon klika web forma aplikaciji prosljeđuje korisnikov unos te se aplikacija spaja na ranije definiranu krajnju točku modela. Spajanje je omogućeno preko funkcije `invoke_api` koja prima podatke koje je korisnik unio u JSON obliku i ključ za povezivanje na model koji je generiran u fazi testiranja modela. Funkcija se zatim povezuje na model te mu šalje zahtjev, a nakon što dobije odgovor odnosno predikciju funkcija istu prosljeđuje do forme koja se otvara u novom prozoru i korisniku prikazuje predikciju modela.

```
def invoke_api(data, api_key):

    body = str.encode(json.dumps(data))
    url = 'http://20.8.189.172:80/api/v1/service/nba-stats-denorm-ep/score'

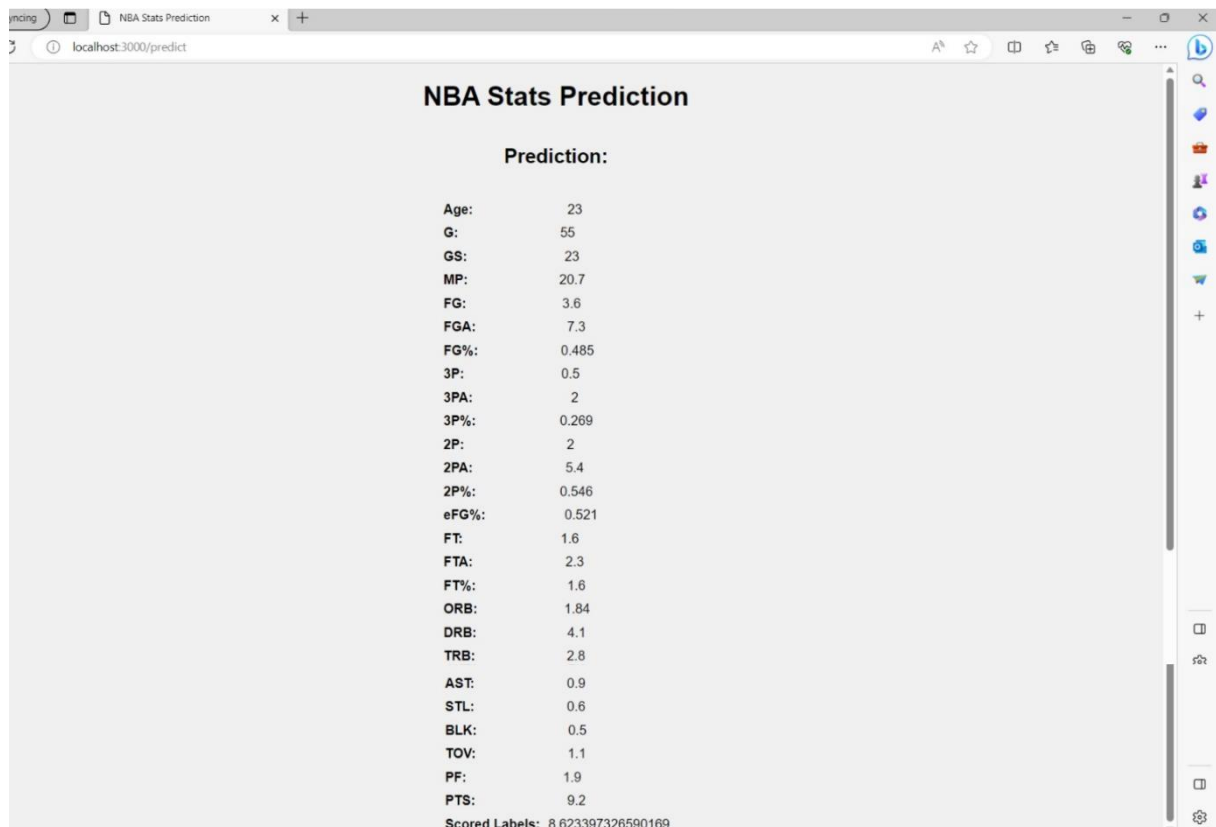
    if not api_key:
        raise Exception("A key should be provided to invoke the endpoint")
    headers = {'Content-Type': 'application/json', 'Authorization': ('Bearer ' + api_key)}

    req = urllib.request.Request(url, body, headers)

    try:
        # Send the request and get the response
        response = urllib.request.urlopen(req)
        # Read and return the result
        result = response.read()
        return result.decode("utf8", 'ignore')
```

Programski kod 3. Funkcija za povezivanje na model

Model vraća predikciju u JSON formatu stoga kako bi se podaci korisniku prikazali u što čitljivijem i preglednijem obliku prije nego se vrate do forme za prikaz podaci se enkodiraju u UTF-8 format. Forma s predikcijom izgleda kao na slici ispod.



NBA Stats Prediction	
Prediction:	
Age:	23
G:	55
GS:	23
MP:	20.7
FG:	3.6
FGA:	7.3
FG%:	0.485
3P:	0.5
3PA:	2
3P%:	0.269
2P:	2
2PA:	5.4
2P%:	0.546
eFG%:	0.521
FT:	1.6
FTA:	2.3
FT%:	1.6
ORB:	1.84
DRB:	4.1
TRB:	2.8
AST:	0.9
STL:	0.6
BLK:	0.5
TOV:	1.1
PF:	1.9
PTS:	9.2
Scored Labels: 8.623397326590169	

Slika 38. Korisniku prikazana predikcija modela

Ovisno o korisnikovim unosima model kreira predikciju, a podaci se ispisuju na način da se korisniku prvo ispišu parametri koje je unio nakon čega se ispisuje predikcija koju model generira. U prikazanom slučaju model je predvidio da će prosječan broj poena koje će igrač postizati u tekućoj sezoni biti 8.62.

7. Zaključak

Ovaj diplomski rad pokušava analizirati i sistematizirati područje strojnog učenja kao dijela umjetne inteligencije. Rad se sastoji od dva glavna dijela, a to su teorijski i praktični dio. Teorijski dio započinje analizom pojma umjetne inteligencije i njegovog razvoja kroz prošlost za koju možemo zaključiti kako se njeno poimanje i nije previše mijenjala još od onoga kako su ga definirali Minsky i McCarthy u kasnih pedesetih.

Uvodni dio rada teoretski obrađuje područja umjetne inteligencije koja su definirana u nekoliko kategorija, a to su strojno učenje, neuronske mreže, NLP, duboko učenje, kognitivno računarstvo i računalni vid za koje možemo ustvrditi kako se isprepliću te je teško govoriti o tvrdim granicama između područja. Za kraj uvodnog dijela rada su predstavljene prednosti i nedostaci korištenja umjetne inteligencije od kojih bi istaknu kao najveću prednost mogućnost obrade velike količine podataka u kratkom vremenu te kreiranja izuzetno preciznih predikcija. Dok je najveća mana gubitak radnih mjesta na repetitivnim poslovima gdje roboti pogonjeni nekim oblikom umjetne inteligencije često s lakoćom zamjenjuju ljude.

Glavni dio teorijskog rada sistematizira područje strojnog učenja u četiri vrste: nadzirano strojno učenje, strojno učenje bez nadzora, polu-nadzirano strojno učenje i učenje s pojačanjem. Zatim su predstavljeni i objašnjeni algoritmi strojnog učenja za koje je pretpostavljeno, a kasnije u praktičnom dijelu rada i dokazano da ne postoji određeni algoritam koji uvijek daje bolje rezultate od ostalih već izbor algoritma koji će pružiti najbolje rezultate ovisi o problemu koji se pokušava riješiti, ali i dostupnom podatkovnom skupu na kojem se algoritam trenira. Teorijski dio rada završava pregledom Microsoft tehnologija koje omogućuju implementaciju predviđanja pomoću strojnog učenja.

U praktičnom dijelu rada je prikazan cjelokupan proces razvoja prediktivnog modela korištenjem Microsoft Azure Machine Learning alata. Razvoj je započeo prikupljanjem početnog skupa podataka koji je zatim očišćen i transformiran kako bi se mogao koristiti za treniranje modela. Izrađeno je nekoliko modela korištenjem različitih algoritama strojnog učenja čiji su podaci evaluirani te je izabran model koji je donio najbolje rezultate, a to je u ovom slučaju bio model izrađen pomoću algoritma linearne regresije. Budući da su svi modeli trenirani i evaluirani na istom skupu podataka rezultate možemo smatrati relevantnima za usporedbu jer su svi modeli imali jednake uvjete za razvoj. Odabrani model je objavljen pomoću Azure ML-a te je za njega kreirana krajnja točka pomoću koje je model testiran. Nakon što je model uspješno objavljen integriran je u programski proizvod čime je postao dostupan korisnicima na testiranje i konzumiranje. Izrađeni programski proizvod za

potrebe rada je jednostavna web aplikacija izgrađena pomoću Node.js dok se za komunikaciju s modelom koristi Python odnosno Flask unutar Python programskog jezika.

Najizazovniji dio rada bila je izrada praktičnog dijela rada jer isti obuhvaća gotovo pa cjelokupan proces razvoja (nedostaje dodatna optimizacija modela) i integracije prediktivnog modela u programski proizvod. Budući da mi je ovo bio prvi takav projekt bilo je poprilično izazovno ukomponirati sve korištene tehnologije, ali moram priznati kako mi je u cijelom procesu uvelike pomogla Azure platforma koja ima opsežnu dokumentaciju i zbilja veliku zajednicu korisnika.

Popis literature

- [1] 6 Major Sub-Fields of Artificial Intelligence. (2021.). *Medium*. Preuzeto 20. 4. 2023 s <https://rancholabs.medium.com/6-major-sub-fields-of-artificial-intelligence-77f6a5b28109>
- [2] Ali, M. (2022). *Supervised Machine Learning*. Preuzeto 3. 6. 2023 s <https://www.datacamp.com/blog/supervised-machine-learning>
- [3] Bacallado, S., i Taylor, J. (2022). *Simple linear regression*. Preuzeto 26.6.2023. s <https://web.stanford.edu/class/stats202/notes/Linear-regression/Simple-linear-regression.html>
- [4] Burns, E. (bez dat.). *What Is Machine Learning and Why Is It Important?* Enterprise AI. Preuzeto 20. 4. 2023 s <https://www.techtarget.com/searchenterpriseai/definition/machine-learning-ML>
- [5] Copeland, B. J. (2023). Artificial intelligence (AI) | Definition, Examples, Types, Applications, Companies, & Facts | Britannica. U *Britannica*. Preuzeto 16. 4. 2023 s <https://www.britannica.com/technology/artificial-intelligence>
- [6] Council of Europe. (bez dat.). History of Artificial Intelligence. *Council of Europe*., Preuzeto 14. 4. 2023 s <https://www.coe.int/en/web/artificial-intelligence/history-of-ai#:~:text=The%20term%20%22AI%22%20could%20be,because%20they%20require%20high%2Dlevel>
- [7] Eluyode, O. S., i Akomolafe, D. T. (2013). Scholars Research Library Comparative study of biological and artificial neural networks. *European Journal of Applied Engineering and Scientific Research*, 2, 36–46.
- [8] Fausett, L. V. (1994). *Fundamentals of neural networks: Architectures, algorithms, and applications*. Prentice-Hall.
- [9] Gentleman, R., i Carey, V. J. (2008). Unsupervised Machine Learning. U F. Hahne, W. Huber, R. Gentleman, i S. Falcon, *Bioconductor Case Studies* (str. 137–157). Springer New York. https://doi.org/10.1007/978-0-387-77240-0_10

- [10] Gupta, A. (2019). Semi-Supervised Learning in ML. *GeeksforGeeks*. Preuzeto 20. 6. 2023 s <https://www.geeksforgeeks.org/ml-semi-supervised-learning/>
- [11] Gupta, M. (2018). Linear Regression in Machine learning. *GeeksforGeeks*. Preuzeto 20. 8. 2023 s <https://www.geeksforgeeks.org/ml-linear-regression/>
- [12] Haenlein, M., i Kaplan, A. (2019). A Brief History of Artificial Intelligence: On the Past, Present, and Future of Artificial Intelligence. *California Management Review*, 61(4), 5–14. <https://doi.org/10.1177/0008125619864925>
- [13] Hayes, A. (2023). *Bayes' Theorem: What It Is, Formula, and Examples*. Investopedia. Preuzeto 25. 8. 2023 s <https://www.investopedia.com/terms/b/bayes-theorem.asp>
- [14] IBM. (bez dat.-a). *What is Computer Vision? | IBM*. Preuzeto 20. 4. 2023. S <https://www.ibm.com/topics/computer-vision>
- [15] IBM. (bez dat.-b). *What is Deep Learning? | IBM*. Preuzeto 20. 4. 2023., s <https://www.ibm.com/topics/deep-learning>
- [16] IBM. (bez dat.-c). *What is Machine Learning? | IBM*. Preuzeto 20. 4. 2023. s <https://www.ibm.com/topics/machine-learning>
- [17] Javatpoint. (bez dat.-a). *Decision Tree Algorithm in Machine Learning—Javatpoint*. Preuzeto 29. 5. 2023. s <https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm>
- [18] Javatpoint. (bez dat.-b). *Naive Bayes Classifier in Machine Learning—Javatpoint*. Preuzeto 25.8.2023. s <https://www.javatpoint.com/machine-learning-naive-bayes-classifier>
- [19] Javatpoint. (bez dat.-c). *Supervised Machine learning—Javatpoint*. Preuzeto 3.6.2023. s <https://www.javatpoint.com/supervised-machine-learning>
- [20] Javatpoint. (bez dat.-d). *Types of Machine Learning—Javatpoint*. *Www.Javatpoint.Com*. Preuzeto 29.5.2023. s <https://www.javatpoint.com/types-of-machine-learning>
- [21] Javatpoint. (bez dat.-e). *Unsupervised Machine learning—Javatpoint*. Preuzeto 15.6.2023. s <https://www.javatpoint.com/unsupervised-machine-learning>

- [22] Kumar Talaviya, A. (2023). *Top 10 machine learning algorithms with their use-cases—DataKwery*. DataKwery. Preuzeto 9.9.2023. s <https://www.datakwery.com/post/top-10-ml-algorithms-with-use-cases/>
- [23] *Machine Learning Workflow: Streamlining Your ML Pipeline*. (bez dat.). Preuzeto 17.7.2023. s <https://www.run.ai/guides/machine-learning-engineering/machine-learning-workflow>
- [24] Microsoft. (bez dat.). *Business Applications | Microsoft Dynamics 365*. Preuzeto 9. 9. 2023. s <https://dynamics.microsoft.com/en-us/>
- [25] Microsoft Azure. (bez dat.). *Cloud Computing Services | Microsoft Azure*. Preuzeto 9. 9. 2023. s <https://azure.microsoft.com/en-us>
- [26] Oreški, D. (2022). *Strojno učenje temeljeno na sličnosti*. Inteligentni sustavi [Moodle]. Sveučilište u Zagrebu, Fakultet organizacije i informatike, Varaždin
- [27] Pinar Saygin, A., Cicekli, I., i Akman, V. (2000). Turing Test: 50 Years Later. *Minds and Machines*, 10(4), 463–518. <https://doi.org/10.1023/A:1011288000451>
- [28] PlanetTogether. (2023). *Incorporating Machine Learning into Product Development*. Preuzeto 28.8.2023. s <https://www.planettogether.com/blog/incorporating-machine-learning-into-product-development>
- [29] Singapore Computer Society. (bez dat.). *Simplifying the Difference: Machine Learning vs Deep Learning—Singapore Computer Society*. Preuzeto 25.4.2023. s <https://www.scs.org.sg/articles/machine-learning-vs-deep-learning>
- [30] Sutton, R. S., i Barto, A. G. (2018). *Reinforcement learning: An introduction* (Second edition). The MIT Press.
- [31] Vivek, K. (2023). *What is Microsoft Dataverse? - Power Apps*. Preuzeto 11. 9. 2023. s <https://learn.microsoft.com/en-us/power-apps/maker/data-platform/data-platform-intro>

Popis slika

Slika 1. Podjela umjetne inteligencije.....	7
Slika 2. Biološki neuron	9
Slika 3. Jednostavni umjetni neuron	9
Slika 4. Jednostavna neuronska mreža	10
Slika 5. Duboko učenje.....	12
Slika 6. Podjela strojnog učenja.....	17
Slika 7. Nadzirano strojno učenje	18
Slika 8. Strojno učenje bez nadzora	19
Slika 9. Linearna regresija	23
Slika 10. Primjer primjene K-najbližih susjeda	24
Slika 11. Primjer stabla odlučivanja	25
Slika 12. Logo Microsoft Azure platforme	30
Slika 13. Microsoft Power platforma	31
Slika 14. Logo Microsoft Dynamics 365.....	32
Slika 15. Prikaz sirovih podataka.....	35
Slika 16. Snimka zaslona, pripremljen ulazni skup podataka	38
Slika 17. Kreiranje grupe resursa.....	39
Slika 18. Kreiranje radnog prostora	39
Slika 19. Alatna traka Azure ML Studio alata.....	40
Slika 20. Kreirana virtualna instanca razvojnog okruženja	41
Slika 21. Snimka zaslona Azure ML, učitavanje podatkovnog skupa	42
Slika 22. Odabir podataka koji se učitavaju u skup podataka.....	42
Slika 23. Snimka zaslona iz Azure ML, prikaz procesa izrade modela.....	43
Slika 24. Treniranje modela strojnog učenja	44
Slika 25. Uspješno treniran model	45
Slika 26. Trenirani model algoritmom K-sredina	46
Slika 27. Evaluacija rezultata, regresijski algoritmi.....	47
Slika 28. Evaluacija rezultata, metodom K-sredina	48
Slika 29. Evaluacija rezultata nenormaliziranih podataka metodom linearne regresije.....	48
Slika 30. Kreiran AKS klaster.....	49
Slika 31. Kreiran inference pipeline	50
Slika 32. Uspješno objavljen model spreman za korištenje.....	51
Slika 33. Podaci za integraciju modela	51
Slika 34. Testiranje objavljenog modela i stvaranje predikcija.....	52
Slika 35. Organizacija repozitorija	54
Slika 36. Forma za upis podataka, prvi dio	55
Slika 37. Forma za upis podataka, drugi dio	55
Slika 38. Korisniku prikazana predikcija modela	57

Popis tablica

Tablica 1. Popis atributa unutar početnog seta podataka	35
---	----

Popis isječaka programskog koda

Programski kod 1. Čišćenje podataka	37
Programski kod 2. Testiranje objavljenog modela.....	53
Programski kod 3. Funkcija za povezivanje na model.....	56

Prilozi

[1] Početni set podataka, Kaggle, <https://www.kaggle.com/datasets/joebeachcapital/nba-player-statistics?resource=download>

[2] Github repozitorij, <https://github.com/kgardijan/funkc-predvidanja-diplomski-rad>

[3] Kreirani model je dostupan za spajanje preko REST *endpointa* na linku: <http://20.8.189.172:80/api/v1/service/nba-stats-denorm-ep/score>. Za spajanje je još potreban primary key (token): MRK1GFSJA9PhUM2JcAyxRgGgZielyJLD

Prilikom spajanja na *endpoint* modelu je potrebno proslijediti ulazne podatke u JSON formatu na način kako ih model i očekuje primjer skripte s ulaznim podacima se nalazi na priloženom Github repozitoriju u mapi resursi pod nazivom „Skripta za testiranje objavljenog modela“.