Ocjena kvalitete biblioteka programa otvorenog koda za strojno učenje u jeziku Python

Zovko, Zoran

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Zoran Zovko

OCJENA KVALITETE BIBLIOTEKA PROGRAMAOTVORENOG KODA ZA STROJNO UČENJE U JEZIKU PYTHON

ZAVRŠNI RAD

Varaždin, 2019.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Zoran Zovko

Matični broj: 0016126953

Studij: Informacijski sustavi

OCJENA KVALITETE BIBLIOTEKA PROGRAMA OTVORENOG KODA ZA STROJNO UČENJE U JEZIKU PYTHON

ZAVRŠNI RAD

Mentor:

Prof. dr. sc. Božidar Kliček

Varaždin, kolovoz 2019.

Zoran Zovko

Izjava o izvornosti

Izjavljujem da je moj završni 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 prihvaćanjem odredbi u sustavu FOI-radovi

_____

Sažetak

Strojno učenje je doživjelo jako brz razvoj posebno zadnjih nekoliko godina. Od raznih implementacija u društvenim mrežama i svakodnevnim aplikacija do govornih asistenata i autonomnih vozila. Često same realizacije izgledaju dosta impresivno, ali u pozadini svega se nalaze podaci i algoritmi. Podaci se prikupljaju raznim mjerenjima i anketiranjima te s obzirom na vrstu podataka i željeni rezultat odabire se najpogodniji algoritam za izradu modela.

Postoje četiri vrste strojnog učenja: nadgledano učenje, nenadgledano učenje, polu-nadgledano učenje i učenje podrškom. Svaka vrsta strojnog učenja sadrži svoju skupinu algoritama te se oni koriste s obzirom na vrstu podataka i željeni rezultat.

Izrada modela obavlja se u nekom od alata ili programskih jezika. Među njima je i najpopularniji jezik današnjice Python. On sadrži na tisuće biblioteka namijenjenih strojnom učenju, a najpopularnije među njima su Scikit – learn i TensorFlow koje će se koristiti za usporedbu s alatima. Također uz biblioteke postoji i jako puno alata koji se koriste za strojno učenje, među njima spadaju i Weka i RapidMiner koji su također korišteni u usporedbi.

Ključne riječi: Strojno učenje, Python, klasteriranje, logistička regresija, neuronske mreže

Sadržaj

1. Uvod

U poglavlju „Strojno učenje“ definirati će se pojmovi poput strojnog učenja, nadgledanog i nenadgledanog učenja, algoritama, modela… Navesti će se podjela strojnog učenja te će biti posebno objašnjena svaka od podjela. Nakon podijele biti će govora o modelima strojnog učenja i njihovoj podjeli. Na kraju poglavlja govorit će se o programskom jeziku Python, njegovoj primjeni i popularnosti, njegovim bibliotekama koje se koriste za strojno učenje poput Scikit – learna i TensorFlowa… Nakon Pythona dolaze na red alati koji će biti korišteni u usporedbi s bibliotekama.

U poglavlju nakon „Usporedba biblioteka i komercijalnih alata“ biti će opisan rad te prikazani dijelovi programskih kodova što je zapravo opisani dio praktičnog rada. Također biti će prikazane i slike rada u alatima i programskom jeziku Python te prezentirani rezultati i izvješća. Nakon što se opiše rad biblioteka i alata biti će napravljena usporedba između istih te za kraj iznijeti ocjena.

2. Strojno učenje

Iako je strojno učenje područje koje postoji već neko vrijeme, ono se počelo ozbiljnije pratiti tek zadnjih dvadesetak godina. Njegova velika ekspanzija i stavljanje sve veći naglasak na njega i njegovu primjenu u stvarnom svijetu dogodila se zadnjih nekoliko godina. Glavni „krivac“ za takvu brzu ekspanziju strojnog učenja je zasigurno brzi razvoj računala i tehnologije. Kako više ne postoje toliki problemi pohrane i također problemi brzine postaju manji ostvaruju se sve idealniji uvjeti da strojno učenje razvije svoj puni potencijal.

Mnogi ljudi kada čuju „umjetna inteligencija“ ili „strojno učenje“ odmah zamišljaju futurističke slike robota nalik na ljude koji su ih zamijenili u obavljanju njihovog posla, ali umjetna inteligencija i strojno učenje su nešto drugo od samih robota i što je najvažnije ono je već niz godina tu. Najbolji pokazatelj koliko se strojno učenje „ukorijenilo“ u živote ljudi pokazuje postojanje mnogih implementacije strojnog učenja. Neke od aplikacija koje svakodnevno koristi stotine milijuna ljudi kao što su Facebook, Instagram ili Twitter najčešće strojno učenje koriste za prikazivanje najrelevantnijih novosti korisniku na temelju njegove aktivnosti na aplikaciji. Strojno učenje je rasprostranjeno i šire od samih aplikacije. To zasigurno vidimo u sve većoj popularnosti raznih kućnih asistenata koji koriste strojno učenje za prepoznavanje govora i autonomnih vozila koji koriste strojno učenje za izbjegavanje prepreka na cesti te za snalaženje u prostoru.

Jedno od prvih primjena strojnog učenja je zasigurno filtriranje spam E – mail poruka. Ono funkcionira na taj način da uočava riječi i fraze koje se neuobičajeno često koriste u primjerima spam poruka kako bi mogao detektirati nove spam poruke. U tradicionalnom algoritmu za filtriranje svaki put kada se promijene riječi u nekoj spam poruci bilo bi nužno mijenjati i sam algoritam kako bi onemogućio slanje poruka, dok bi neki od algoritama strojnog učenja namijenjen za otkrivanje spam poruka uočio neuobičajenu veliku upotrebu određenih riječi ili fraza te tako ih označio kao riječi ili fraze koje se koriste u potencijalnim spam porukama. Na temelju samo ovog primjera moguće je uočiti ogroman potencijal koje nam strojno učenje pruža te opravdava sve veću zainteresiranost publike za ovo područje.

Globussoft na svojim stranicama navodi četiri područja u kojima se umjetna inteligencija već dobrano ukorijenila. Prvo od njih su banke i financijski sustavi. Banke koriste umjetnu inteligenciju u mnogim aktivnostima poput financijskih operacija, ulaganja novca u dionice, upravljanja različitim svojstvima i još mnogo toga. Ispostavilo se kako je umjetna inteligencija puno uspješnija u obavljanju ovih aktivnosti od čovjeka i vrlo je vjerojatno da će u budućnosti u potpunosti zamijeniti čovjeka na ovim područjima. Drugo područje je medicina

gdje se uz virtualnog asistenta za osobnu zdravstvenu zaštitu umjetna inteligencija koristi i za istraživanje i analitiku. Kako bi se čovjeka zamijenilo u obavljanju opasnog i monotonog posla umjetna inteligencija se primjenjuje u području teške industrije. Roboti su zamijenili čovjeka u prijevozu opasnog tereta u skladištima te sastavljanju automobilskih motora što su samo dva ogledna primjera gdje se koristi umjetna inteligencije kako bi se zamijenio čovjek. Ljudske greške u svakodnevnom poslu su česte i očekivane te da bi se minimizirao broj grešaka ili čak u potpunosti uklonio u područjima poput zračnog transporta koristi se umjetna inteligencija. Važnost umjetne inteligencije u ovom trenutku je postala nezamisliva te je vidljivo u kojem smjeru ide primjena umjetne inteligencije u svakodnevnom životu [1].

U ovom poglavlju prvo će se definirati samo strojno učenje te glavni pojmovi tog područja poput podataka, algoritama, šuma, označenih i neoznačenih podataka itd. Nakon definiranja pojmova obavit će se podjela strojnog učenja na četiri vrste: nadgledano strojno učenje, nenadgledano strojno učenje, učenje podrškom i polu – nadgledano učenje. Poslije obavljenog definiranja i podjele biti će govora o prediktivnom modelu, tri vrste modela te pojmovima prenaučenosti i podnaučenosti. Na kraju poglavlja biti će izneseni osnovni podaci programskog jezika Python, njegovih biblioteka i alata koji će se koristiti u usporedbi te će se navesti glavne prednosti i mane biblioteka u usporedbi s alatima.

2.1. Definiranje i podjela

Idealna definicija strojnog učenja ne postoji, ali barem možemo približno dobro opisati suštinu strojnog učenja. Strojno učenje je znanstvena studija koja ima za cilj omogućiti računalu razumijevanje stvarnih problema okoline u kojoj se nalazi, sudjelovanju u njihovom rješavanju te učenju na prethodnom iskustvu. Slično zapažanje iznio je i jedan od pionira strojnog učenja Arthur L. Samuel u svojoj knjizi „Neka studija o strojnom učenju u igri dame“ gdje navodi sljedeću definiciju „Strojno učenje je područje proučavanja koje daje mogućnost računalu da uči bez da je izričito programirano“ [2]. Iako ove definicije ne obuhvaćaju posao koji strojno učenje obavlja, one dovoljno dobro govore koja je sama suština strojnog učenja.

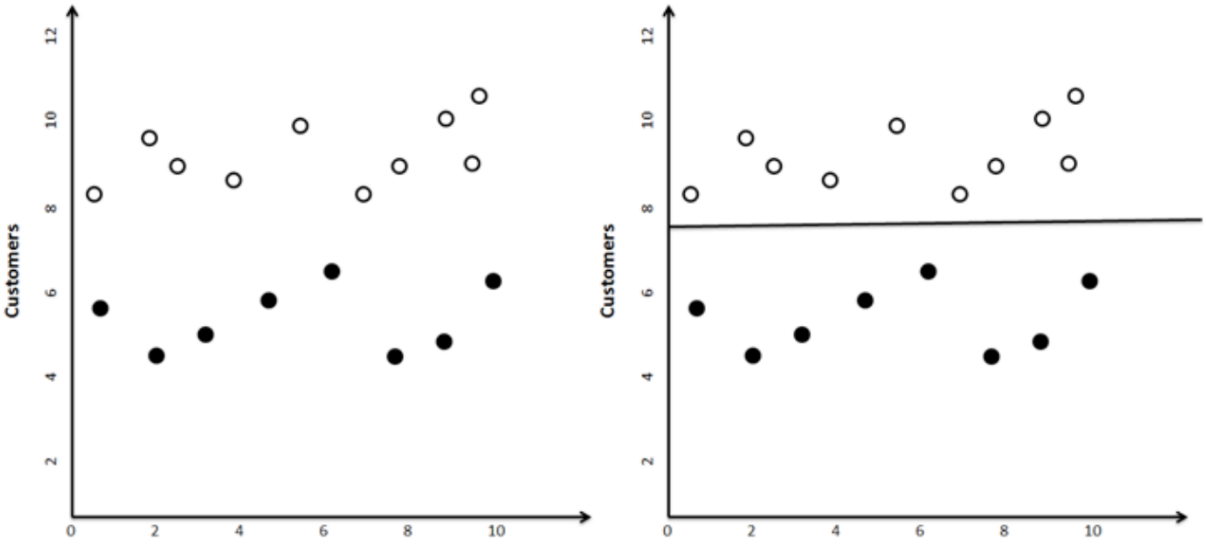
Posao strojnog učenja je izgradnja modela kojeg ćemo kasnije koristiti za dobivanje potrebnih rezultata. Izgradnja modela temelji se na podacima i algoritmu. Podaci koji se koriste pri izradi modela obično se dobivaju prethodnim anketiranjima ili mjerenjima. Kako sama anektiranja i mjerenja su dosta puta provedena od strane ljudi neki od podataka znaju biti pogrešni. To je često veliki problem kod strojnog učenja koji može dovesti do izrade lošeg modela. Također još jedan problem kod podatak se javlja kada su podaci oštećeni ili dio podatka nedostaje taj događaj u strojnom učenju se nazivaju šum. Za razliku od netočnih

podataka većina algoritama nije osjetljiva na šumove, ali kod nekih algoritama zna zadavati probleme. Zbog prethodno navedenih problema prije same izrade modela potrebno je napraviti obradu i pripremu podataka u vidu eliminacije netočnih podataka ili onih podataka koji uvelike odskakuju od ostatka ukoliko je takva eliminacija moguća s obzirom na broj podataka. Skupove ovakvih podataka možemo zamišljati kao tablicu u kojoj reci tablice predstavljaju atribute, a stupci tablice predstavljaju vrijednosti atributa. U takvim skupovima podataka podacima nije dodijeljeno nikakvo značenje pa se nazivaju neoznačeni podaci. Oni su najčešće dobiveni mjerenjem, nije ih teško pribaviti je ih danas ima jako puno i nije ih teško prikupiti. Ako takvim podacima dodijelimo značenje oni postaju označeni podaci i danas je to najčešća radnja kod pripreme podataka [3], [4].

Algoritmi strojnog učenja nemaju unaprijed isprogramirane funkcionalnosti nego je njegova funkcionalnost zadana primjerima za učenje. Oni obuhvaćaju postupke obrade podataka te postupke optimiranja slobodnih parametara postupka obrade na podatcima za učenje [5].

Postoje razne podjele strojnog učenja, ali ovdje će biti podijeljeno u četiri glavne skupine: nadgledano učenje, nenadgledano učenje, polu-nadgledano učenje i učenje podrškom [4].

U **nadgledanom učenju** podaci za trening trebaju biti označeni. Karakteristični zadatak strojnog učenja za nadgledano učenje je klasifikacija. Klasifikacija je način identificiranja tehnike grupiranja za određeni skup podataka na takav način da se ovisno o vrijednosti ciljanog ili izlaznog atributa, cijeli skup podataka može kvalificirati da pripada klasi. Na slici ispod grafički je prikazana klasifikacija gdje su uzorci podijeljeni u dvije skupine gdje je jedna označena praznim kružićima, a druga skupina prikazana ispunjenim kružićima [3].



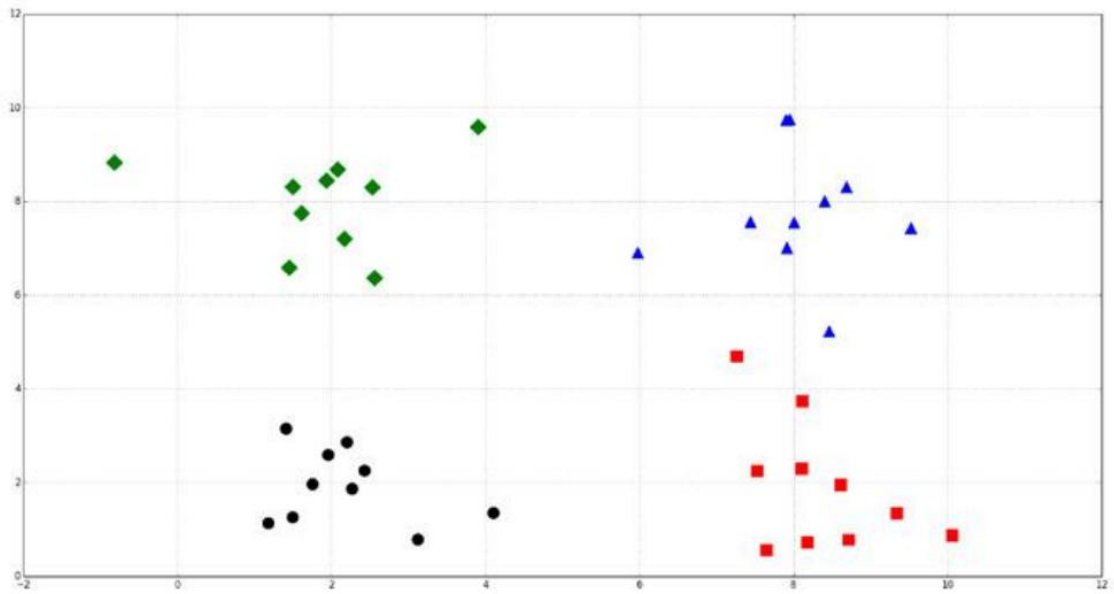
Slika 1. Grafički prikaz klasifikacije [3]

Još jedan karakteristični zadatak strojnog učenja je regresija. Regresija predviđa numeričku vrijednost s obzirom na unesenu vrijednost temeljem naučenih pravila. Postoji nekoliko vrsta regresija, neke od njih su: linearna regresija, logistička regresija, regresija podržavajući vektora… Na slici ispod prikazan je grafički primjer linearne regresije [6], [7].

Slika 2. Grafički prikaz linearne regresije [3]

Slika 2 prikazuje cijenu pizze u dolarima s obzirom na promjer u inčima. Linearna funkcija prikazana na slici najbolje prikazuje kontinuitet rasta cijene s obzirom na rast promjera pizze te na temelju nje može se odrediti cijena pizze s obzirom na promjer.

Za razliku od nadgledanog učenja u **nenadgledano učenje** nalazi prirodno grupiranje slučajeva na temelju postojećih neoznačenih podataka. Nenadgledano učenje podrazumijeva zadatke klasteriranja, asocijacije te vizualizacije i smanjenja dimenzija. Klasteriranje je proces grupiranja sličnih objekata u zajedničke klastere. Neki od najvažnijih algoritama klasteriranja su: algoritam klasterizacije metodom K-srednjih vrednosti, kraće k – means, hijerarhijska analiza klaster (eng. *Hierarchical Cluster Analysis* – HCA) i maksimizacija izuzetaka. Na slici ispod prikazano je klasteriranje k – means algoritmom [4], [8], [9].

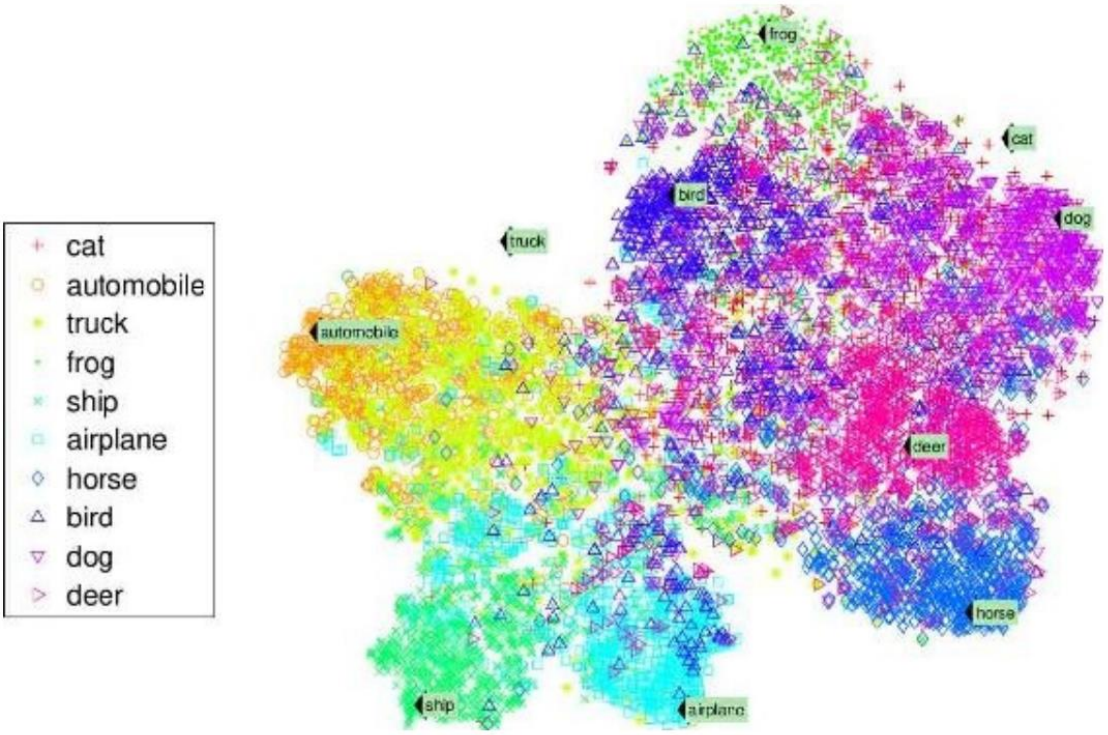


Slika 3. Grafički prikaz K-Means algoritma [10]

Na slici 3 vidljivo je kako su uzorci prema vrijednosnim svojstvima podijeljeni u četiri skupine gdje je jedna skupina označena zelenim rombom, druga skupina je prikazana crnim točkicama, treća skupina je prikazana plavim trokutima i zadnja skupina je prikazana crvenim kvadratićima.

Algoritmi za učenje asocijacije s obzirom na varijable, izdvajaju i definiraju pravila koja se mogu primijeniti na skupu podataka i demonstriraju učenje temeljeno na iskustvu, a samim tim i predviđanje. Predstavnici ove vrste algoritama su apriori algoritam te klasteriranje klasa ekvivalentnosti i rešetkasti presjek odozdo prema gore (eng. *Equivalence Class Clustering and bottom-up Lattice Traversal – ECLAT*) [3]. Algoritmi vizualizacije su dobar pokazatelj nenadgledanog učenja jer ih „hranimo“ s puno kompleksnih i neoznačenih podataka. Oni

pokušaju očuvati strukturu koliko mogu kako bi se moglo razumjeti kako su podaci organizirani i identificirali neočekivane uzorke. Nakon vizualizacije slijedi smanjenje dimenzije u kojem je cilj pojednostavit podatke bez da se izgubi previše informacija. Na slici 4 prikazan je algoritam stohastičkog ugrađivanja susjeda s t-distribucijom (eng. *T-distributed Stochastic Neighbor Embedding* – t-SNE) vizualizacije [4].



Slika 4. t-SNE algoritam [4]

Slika 4 prikazuje 2D prikaz podataka kako bi se lakše razumjeli podaci. Na slici su prikazano deset vrsta podataka kao što su: mačke, automobili, kamioni, žabe, brodovi… Svaka vrsta podataka je označena pripadajućom oznakom.

Kod **polu-nadgledanog učenja** algoritmi se mogu bavit djelomično označenim podacima, obično puno neoznačenih podataka i malo označenih podataka. Oni su kombinacije algoritama za nadgledano učenje i algoritama za nenadgledano učenje [4].

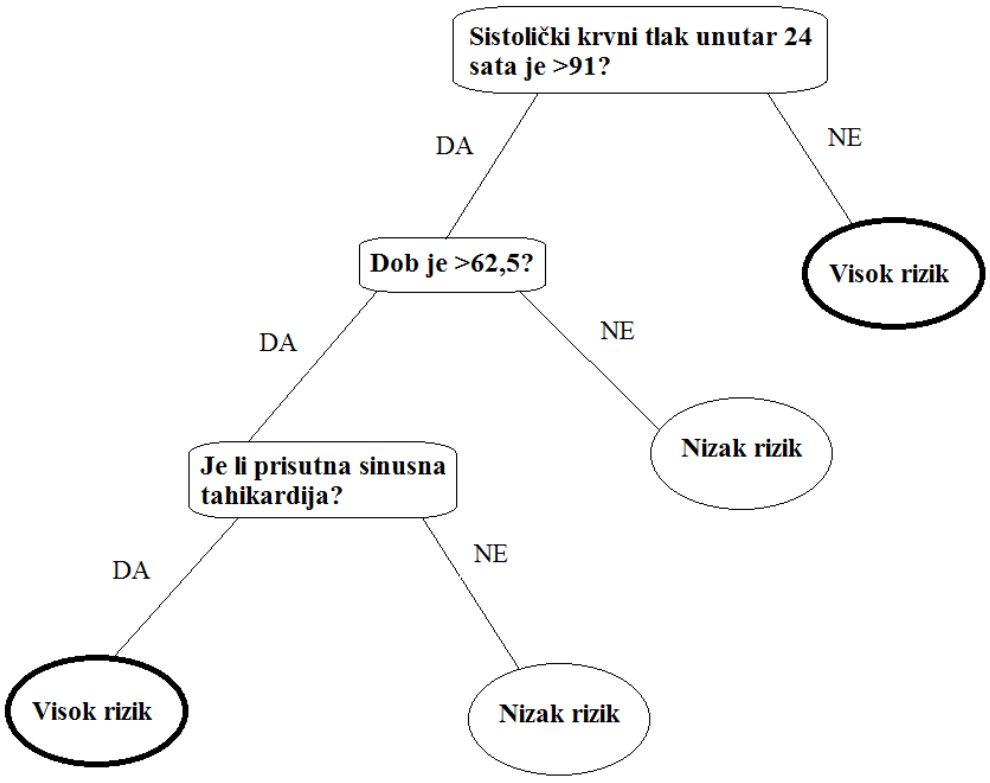
Učenje podrškom je potpuno drukčije s obzirom na prethodne vrste strojnog učenja. Ono je posebna je vrsta strojnog učenja gdje učenje polazi od povratnih informacija iz okoline, a tehnika učenja je iterativna i prilagodljiva [3].

2.2. Prediktivni model

Modeli su centar svake implementacije strojnog učenja. On opisuje podatke koji su promatrani u sustavu. Modeli su rješenja algoritma primijenjenog na skupu podataka. U mnogim slučajevima modeli se primjenjuju na nove skupove podataka koji pomažu modelima da nauče novo okruženje i također ih prognoziraju. Modele možemo kategorizirati u tri kategorije: logički modeli, geometrijski modeli i probabilistički modeli [3].

2.2.1. Logički modeli

Logički modeli su više algoritamski i pomažu da se izvuče skup pravila izvodeći algoritme iterativno. Primjer logičkog modela je stablo odlučivanja [3]. Slika ispod prikazuje stablo odlučivanja radi li se o visokorizičnom ili niskorizičnom pacijentu.



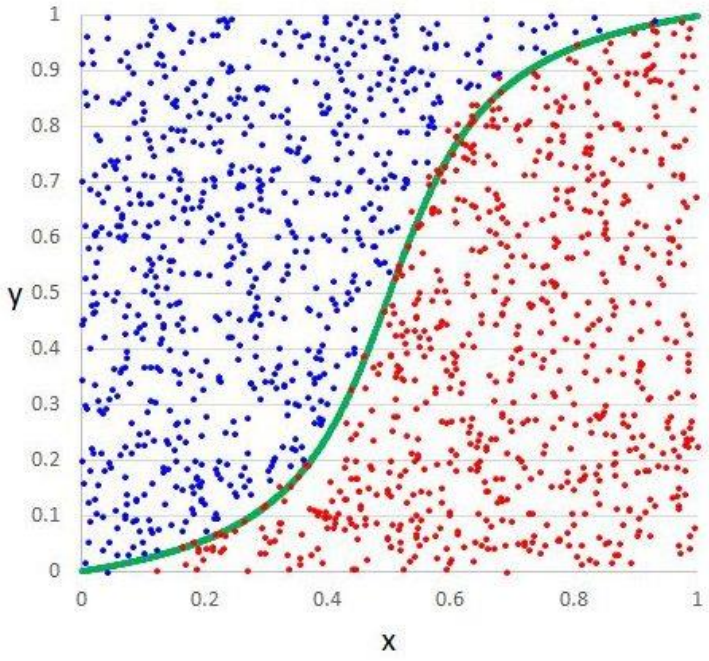
Slika 5. Stablo odlučivanja [11]

Stablo odlučivanja na slici iznad je vrlo jednostavan prikaz određivanja radi li se o visokorizičnom ili niskorizičnom pacijentu. Ukoliko pacijent unutar 24 sata ima sistolički krvni tlak manji ili jednak 91 tada se radi o visokorizičnom pacijentu, ukoliko je on viši od 91 postavlja se potanje je li dob pacijenta viša od 62,5 godina? Ukoliko je dob pacijenta niža ili jednaka

62,5 godina tada se radi o niskorizičnom pacijentu, u suprotnom se postavlja novo pitanja, a to je je li prisutna sinusna tahikardija? Ukoliko kod ispitanog pacijenta nije prisutna sinusna tahikardija tada se radi o niskorizičnom pacijentu, u suprotnom se radi o visokorizičnom pacijentu.

2.2.2. Geometrijski model

Geometrijski model koristi geometrijske koncepte kao što su linije, ravnine i udaljenosti. Ovakvi modeli često se izrađuju za jako velike skupove podataka. Često linearna transformacija pomaže usporedbi različitih metoda strojnog učenja [3]. Na slici 6 prikazan je geometrijski model izrađen algoritmom logističke regresije.



Slika 6. Logistička regresija [12]

Logistička regresija se koristi kada je potrebno predvidjeti rezultate koji se mogu prikazati u binarnom brojevnom sustavu tj. da postoje dvije vrste izlaza. Na slici 6 vidljiva je podjela uzoraka sa vrijednostima svojstava X i Y gdje je jedna vrsta uzoraka označena plavom bojom, a druga vrsta uzoraka označena crvenom bojom. Sigmoidna funkcija vidljiva na slici dijeli uzorke na dva dijela te tako određuje pripada li određeni uzorak jednoj odnosno drugoj skupini.

2.2.3. Probabilistički model

Probabilistički model je statistički model koji uključuje statističke tehnike. Ovakvi modeli su bazirani na strategiji da definiraju vezu između dvije varijable. Taj se odnos može izvesti sa sigurnošću jer uključuje korištenje slučajnog pozadinskog postupka. U mnogo slučajeva podskup skupa podataka se može razmotriti za obradu.

2.2.4. Prenaučenost i podnaučenost

Prenaučenos je problem koji se javlja kada su podaci previše kompleksni ili je podataka premalo. Prenaučenost se lako može uočiti tako što kod skupa podataka koji nam služi za trening imamo izrazito malu stopu pogrešaka, dok se kod skupa podataka za testiranje događaju velike stope pogrešaka. To znači da je model zapravo počeo pamtiti podatke i radi dobro sve dok skup podataka nema podatke koji odskaču od uobičajenih. Podnaučenost je suprotno od prenaučenosti i ona se javlja kada je model previše jednostavan kako bi naučio osnovnu strukturu podataka.

2.3. Python i alati za strojno učenje

Python je interpreterski programski jezik visoke razine i opće namjene. Najčešće se koristi za pisanje manjih skripti. Pythonova sintaksa je dosta slična engleskom jeziku stoga je izuzetno jednostavan za učenje. Zbog svoje jednostavnosti Python je postao jako popularan za biblioteke različite namjene. Popularnost Pythona dokazuje i istraživanje IEEE Spectruma iz 2018. godine koji svrstava Python kao najpopularniji programski jezik [13]. Širok broj biblioteka kojim raspolaže dovelo je Python do statusa jednog od najpopularnijih programskih jezika za strojno učenje. Za strojno učenje u Pythonu koriste se razne biblioteke koje su specificirane za rad na posebnim područjima kao na primjer NumPy biblioteka koja je odlična za rad s višedimenzionalnim poljima te posjeduje jako veliku kolekciju matematičkih funkcija, biblioteka SciPy koja je pogodna za optimizaciju, linearnu algebru te statistiku i biblioteka Theano koja je također izuzetna na poljima matematike i statistike. Iako se i same koriste za strojno učenje navedene biblioteke se često koriste u implementaciji drugih opširnijih biblioteka koje su više specificirane za strojno učenje poput Scikit – learn ili TensorFlow koje će se koristiti u radu i koje će biti objašnjene nešto kasnije.

Strojno učenje nije rezervirano samo za programske jezike, to pokazuje izrazito velik broj alata koji su namijenjeni upravo strojnom učenju. Dosta alata poput Weke, RapidMinera i

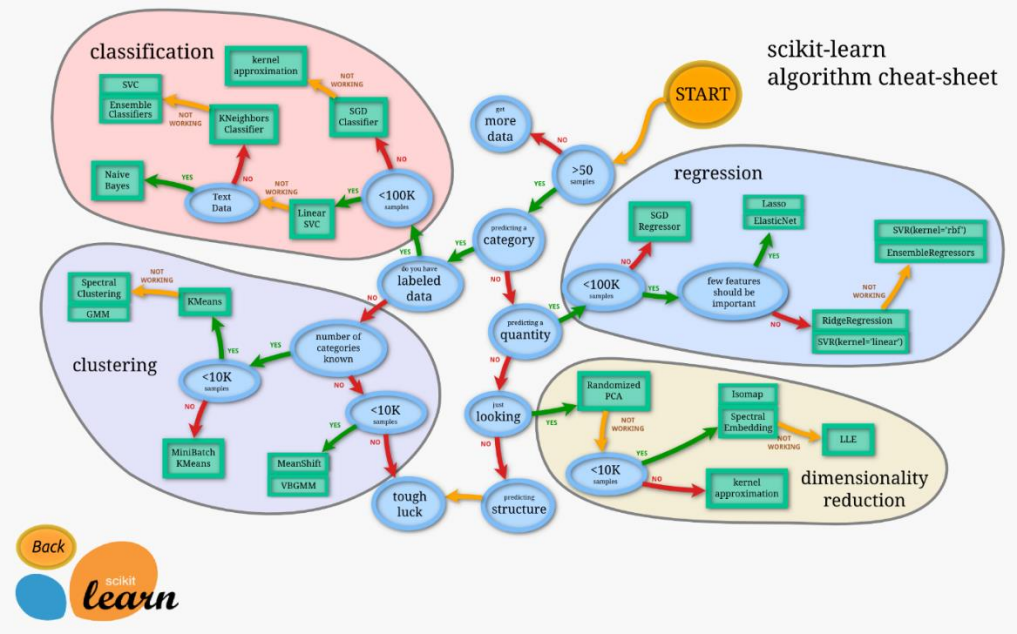
Accord.NETa pruža sve potrebno za izradu modela od same pripreme podataka, preko odabira algoritma i samog učenja do predviđanja. Svi navedeni alati koriste sučelje te nisu potrebne vještine programiranja kako bi se služili njima. Vizualizacija modela je vrlo elegantno riješena i postavlja se pitanje zašto su nam potrebne biblioteke i programiranje kad imamo alate?

Iz navedenog vidimo kako korištenje alata u nekim slučajima naveliko olakšava posao. Iako u dosta slučajeva model koji dobijemo korištenjem alata je dovoljan za naše potrebe, veliki broj tih alata se naplaćuje i ponekad ta cijena i nije toliko mala. Često modeli dobiveni alatima se teže integriraju te ti modeli budu dosta sporiji u usporedbi s modelima dobivenih iz pisanog programskog koda pomoću neke od biblioteka. Pogodnosti koje nosi otvoreni kod biblioteka poput mogućnosti da bilo tko može unaprijediti samu biblioteku su doprinijele da biblioteke za razliku od alata brzo dobivaju moderne algoritme. S druge strane za rad s bibliotekama je potrebno znati vještine programiranja što katkad može predstavljati problem.

U ostatku pod poglavlja će se detaljnije opisati biblioteke za rad u Pythonu poput Scikit – learna i TensorFlowa te alati s kojim će biblioteke biti uspoređene poput Weke i RapidMinera.

2.3.1. Scikit – learn

Scikit – learn je zasigurno najpopularnija Pythonova biblioteka za strojno učenje. Zašto je to tako govori nam širok spektar primjenjivih algoritama klasifikacije, regresije i klasteriranja kao što su metoda potpornih vektora, slučajna šuma, k – means, DBSCAN…[14] Na slici ispod možemo vidjeti i mapu strojnog učenja biblioteke Scikit – learn.



Slika 7. Scikit - learn mapa [15]

Na slici je prikazan vodič za korisnike biblioteke u obliku dijagrama toka koji prikazuje kako bi se trebalo postupati problemu s obzirom na situaciju u kojoj se nalazite. Također na slici su prikazani i algoritmi strojnog učenja koji su dostupni unutar ove biblioteke .

Glavne prednosti Scikit – learna su te što sadrži mnogobrojne algoritme strojnog učenja što smo dosad već nekoliko puta napomenuli. Također velika prednost ove biblioteke je ta što je brza, laka za korištenje i ima jako dobru podršku. S toga i ne čudi zašto je najpopularnija biblioteka za strojno učenje [16].

S druge strane glavne mane ove biblioteke je ta što se manje fokusira na statistiku za razliku od programskog jezika R [16].

2.3.2. TensorFlow

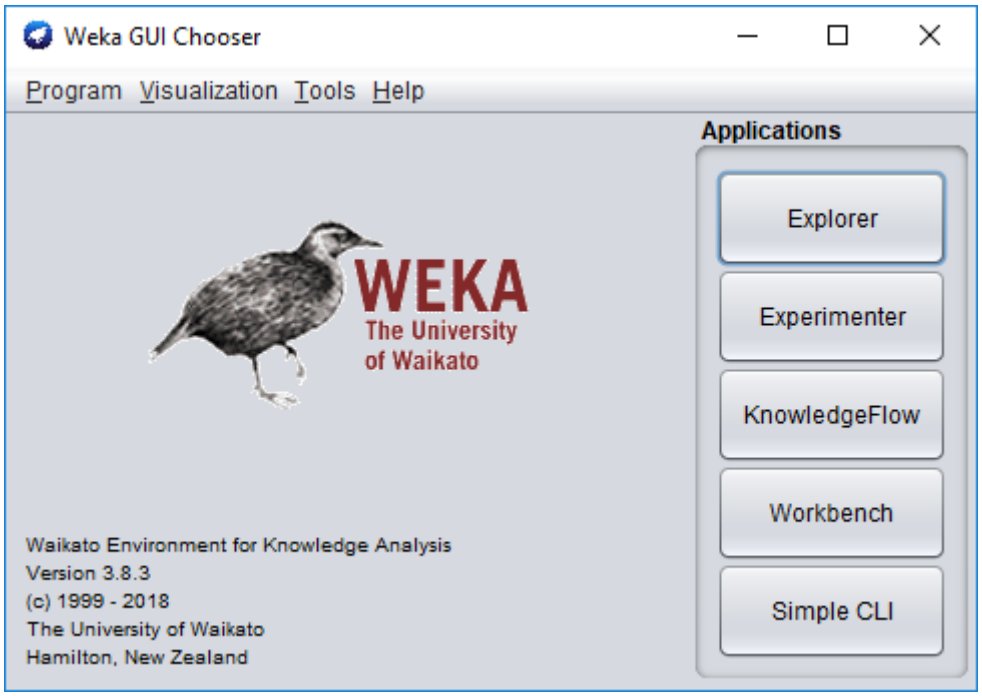
TensorFlow je besplatna biblioteka otvorenog koda za strojno učenje specijalizirana za neuronske mreže. Najpopularnija je biblioteka za strojno učenje kad su u pitanju neuronske mreže. Razvijen je od strane Googlea. Za razliku od Scikit – learna nije razvijen samo za Python nego i za jako puno drugih jezika. Također postoji i manja verzija TensorFlowa za slabije uređaje, tako da i ne čudi zašto je TensorFlow toliko popularan [17].

Prednosti korištenja TensorFlowa su zasigurno njegov prikaz grafova i podržanost od strane Googlea što mu omogućava brza ažuriranja i česta nova izdanja s novim značajkama [18].

Nedostatci Tensorflowa su ti da je sporiji u usporedbi s drugim bibliotekama i nema podršku za Windows [18].

2.3.3. Weka

Weka je besplatan alat za strojno učenje razvijen od strane Sveučilišta u Waikatu. Pisan je u Javi. Sadrži alate za pripremu podataka, klasifikaciju, regresiju, klasteriranje i vizualizaciju [19]. U ovom alatu moguće su radnje od samog učitavanja podataka u obliku baze podataka ili CSV datoteke, pripreme podataka, odabira nekog do algoritama strojnog učenja, izrade modela predviđanja… Na slici ispod prikazano je početno sučelje alata Weka.



Slika 8. Sučelje Weke

Prednosti Weka alata su te što ima širok spektar mogućnosti za pripremu podataka što mu je i najveća značajka i uključeni su algoritmi za rudarenje podataka. Dok bi najveći nedostatci ovog alata bili zasigurno nedostatak novih tehnika, dokumentacija je dosta ograničena te u grafičkom sučelju nisu implementirane sve mogućnosti [20].

2.3.4. Rapidminer

Rapidminer je platforma koja uključuje okruženja za pripremu podataka, strojno učenje, duboko učenje, rudarenje teksta i predviđenje. Ova platforma koristi se u poslovnim i komercijalnim svrhama, a najviše za istraživanja i edukaciju. Za razliku od Weke Rapidminer nije u potpunosti besplatan, već je njegova besplatna verzija ograničena jednim logičkim procesom i s 10 000 podataka [21]. Rapidminer sadrži mnoge algoritme s Weke [22].

Prednosti ovog alata bi bile te da sadrži preko tisuću petsto metoda za integraciju i transformaciju podataka, analizu te modeliranje. Također ovaj alat ima jako veliku fleksibilnost i pruža integraciju velikog broja algoritama [22].

Najveće mane Rapidminera bi bile te ograničene mogućnosti podjele skupa podataka na skup za trening i skup za test te više je namijenjen za ljude koji se bave bazama podataka [22].

3. Usporedba biblioteka i komercijalnih alata

Za potrebe testiranja korištena su tri skupa podataka: mnist.csv, iris.csv i mtcars.csv. Nad navedenim skupovima izvršena su tri algoritma strojnog učenja, a to su: K – means klasteriranje, logistička regresija i neuronske mreže.

Skup podataka **mnist.csv** je skup podataka koji sadrži slike 28X28 piksela na kojima su prikazani ručno pisani brojevi. Skup podataka je podijeljen u dva dijela za trening i za testiranje. Skup podataka za trening sadrži 60 000 slika, a skup za testiranje sadrži 10 000 slika. Izrađen je u Nacionalnom institutu za standarde i tehnologiju i najveći je skup podataka koji sadrži ručno pisane znamenke. Nad ovim skupom podataka je izvršen algoritam dubokih neuronskih mreža.

Skup podataka **iris.csv** je skup podataka koji sadrži podatke o tri vrste perunika: iris setosa, iris versicolor i iris virginica. Skup podataka sadrži 50 elemenata entiteta svake vrsta perunike. Svaki element entiteta ima četiri atributa: dužina čašičnog listića, širina čašičnog listića, dužina latice i širina latice. Nad ovim skupom podatak je primijenjen algoritam K – Means klasteriranje. Skup podataka sadrži 150 elemenata entiteta i često se koristi kao primjer za klasteriranje.

Skup podataka **mtcars.csv** je skup podataka koji sadržava podatke o automobilima. Podaci su prikupljeni iz američkog magazina Motor Trend. O 32 modela automobila dodano je 12 atributa:

- Naziv marke i model automobila
- Mpg – koliko je moguće prijeći milja po galonu goriva
- Cyl – broj cilindra
- Disp –
- Hp – konjske snage
- Drat – omjer stražnje osovine
- Wt – težina u funtama
- Qsec – potrebno vrijeme za prelazak četvrtine milje
- Vs – motor (0 = V tip motora, 1 = normalan tip motora)
- Am – transmisija (0 = automatski mjenjač, 1 = ručni mjenjač)
- Gear – broj prednjih zupčanika
- Carb – broj rasplinjača

Nad ovim skupom podataka obavljen je algoritam logističke regresije za predviđanje omjera stražnje osovine i broja rasplinjača. Na slici ispod vidljiv je prikaz prvih 5 podataka u skupu podataka mtcars.csv.

	car_names	mpg	cy1	disp	hp	drat	wt	qsec	vs	am	gear	carb
0	Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
1	Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
2	Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
3	Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
4	Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2

Slika 9. Primjer podataka skupa podataka mtcars.csv

U pod poglavljima opisat će se sam rad u alatima i rad s bibliotekama te napraviti usporedba rada u bibliotekama u odnosu na alate.

3.1. Scikit – learn u odnosu na Weku i RapidMiner

Za usporedbu biblioteke Scikit – learn sa alatom Weka korišten je algoritam logističke regresije i skup podataka mtcars.csv te algoritam klasteriranja i skup podataka iris.csv, dok je za usporedbu navedene biblioteke s alatom RapidMiner korišten također algoritam klasteriranja i skup podataka iris.csv

3.1.1. Izrada modela logističke regresije u biblioteci Scikit - learn

Za rad sa logističkom regresijom u Pythonu u biblioteku Scikit – learn bilo je potrebno i uključiti neke module iz biblioteka kao što su: Numpy, Pandas, Scipy, Seaborn i Matplotlib. To možemo vidjeti u programskom kodu ispod.

```
import numpy as np
import pandas as pd

from pandas import Series, DataFrame

import scipy
from scipy.stats import spearmanr
```

```
from pylab import rcParams
import seaborn as sb
import matplotlib.pyplot as plt

import sklearn
from sklearn.preprocessing import scale
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn import preprocessing
```

Kao što možemo vidjeti iz programskog koda dodatne biblioteke korištene su za mogućnost prikaza i obrade podataka. Iza toga su postavljeni parametri za vizualizaciju.

```
get_ipython().magic('matplotlib inline')
rcParams['figure.figsize'] = 5, 4
sb.set_style('whitegrid')
```

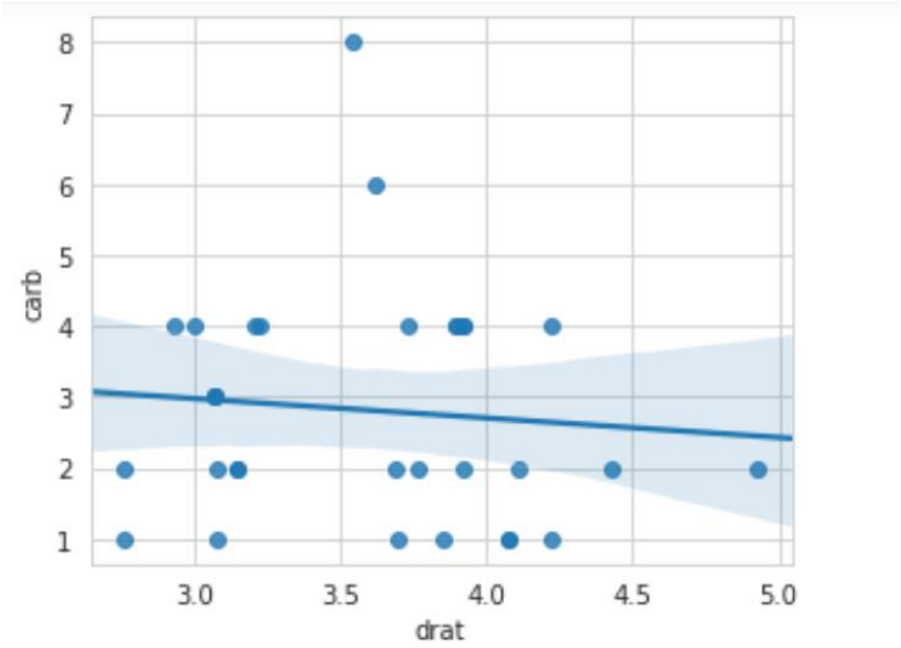
Nakon što su postavljeni parametri za vizualizaciju bilo je potrebno učitati skup podataka te dodijeliti nazive stupcima.

```
adress = '/home/zoran/Preuzimanja/Python-data-science-master/mtcars.csv'
cars = pd.read_csv(adress)
cars.columns = ['car_names', 'mpg', 'cy1', 'disp', 'hp', 'drat', 'wt',
'qsec', 'vs', 'am', 'gear', 'carb']
cars.head()
```

Nakon učitanog skupa podataka potrebno je odrediti podskup koji ćemo predviđati, a to su kako smo naveli omjer stražnje osovine i broj rasplinjača.

```
cars_data = cars.ix[:, (5,11)].values
cars_data_names = ['dart','carb']
y = cars.ix[:,9].values
```

Varijable koje se predviđaju su postavljene i sad se može pogledati graf. Na slici ispod prikazan je graf logističke regresije.



Slika 10. Grafički prikaz logističke regresije za varijable drat i carb

Nakon što je prikazan graf potrebno je provjeriti jesu li ove dvije varijable nezavisne jedna o drugoj. To se provjerava sljedećim programskim kodom.

```
drat = cars['drat']
carb = cars['carb']
spearmanr_coefficient, p_value = spearmanr(drat, carb)
print ("Spearmanr Rank Correlation Coefficient " '%0.3f' %
(spearmanr_coefficient))
```

Dobiveni rezultat iznosi da je korelacije između ove dvije varijable -0,125 što je dovoljno dobro. Za kraj je potrebo pokrenuti i ocijeniti model što se može vidjeti u slijedećem programskom kodu:

```
X = scale(cars_data)
LogReg = LogisticRegression()
LogReg.fit(X,y)
print (LogReg.score(X, y))
y_pred = LogReg.predict(X)
from sklearn.metrics import classification_report
print(classification_report(y, y_pred)).
```

Traženi rezultat vidljiv je na slici ispod.

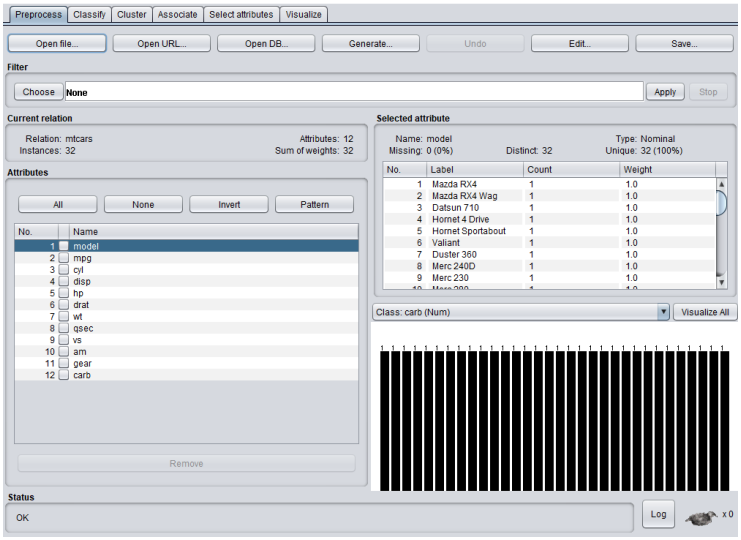
	precision	recall	f1-score	support
0	0.88	0.79	0.83	19
1	0.73	0.85	0.79	13
micro avg	0.81	0.81	0.81	32
macro avg	0.81	0.82	0.81	32
weighted avg	0.82	0.81	0.81	32

Slika 11. Ocjena modela logističke regresije

Iz ocjene je vidljivo da je preciznost modela 0.82, osjetljivost modela iznosi 0.81, vrijednost f1 iznosi također 0.81 i na kraju je vidljivo da su obrađeni svi podaci.

3.1.2. Izrada modela logističke regresije u alatu Weka

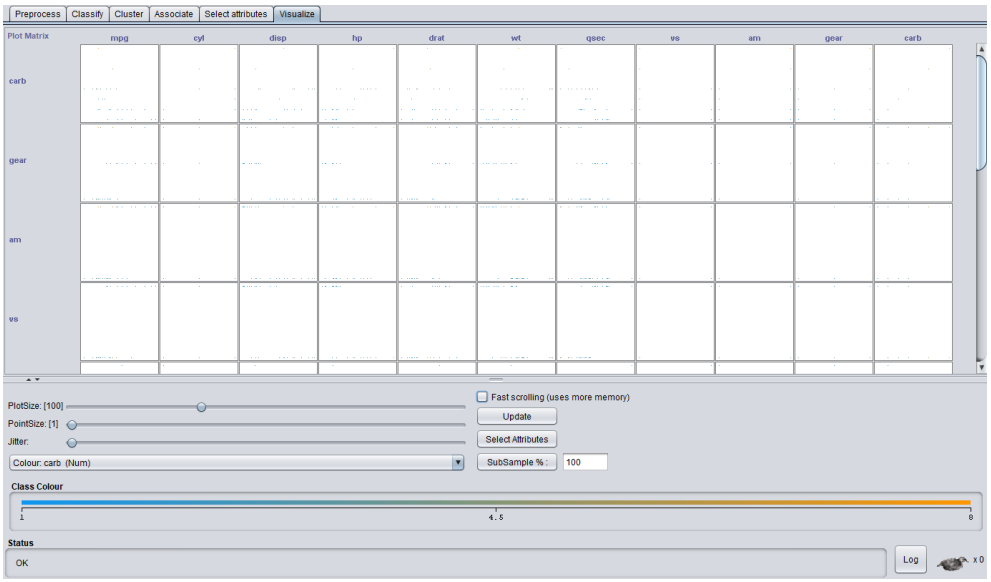
Za razliku od biblioteke Scikit – learn u alatu Weka nema potrebe za kodiranjem već se sav posao odvija u sučelju. Pritiskom na gumb Open file… otvara se prozorčić u kojem se odabire skup podataka.



Slika 12. Učitavanje skupa podataka u Weku

Nakon učitanih podataka u kartici Classify nalazi se gumb za odabir algoritama. Odabirom algoritma logističke regresije moguće je dodatno postaviti atribute te pritiskom na

gumb start izraditi model. Svi grafički prikazi vidljivi su u kartici Visualize što možemo vidjeti na slici ispod.



Slika 13. Prikaz kartice Visualize

3.1.3. Izrada modela klasteriranja u biblioteci Scikit – learn

Za izradu modela klasteriranje u programskom jeziku Python koristeći biblioteku Scikit – learn kao u slučaju modela logističke regresije potrebno je uključiti dodatne module određenih biblioteke poput Numpya, Pnadas biblioteke i Matplotliba. Uključivanje ovih biblioteka prikazano je u programskom kodu ispod. U izradi modela korišten je skup podataka iris.csv.

```
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt

import sklearn
from sklearn.cluster import KMeans
from mpl_toolkits.mplot3d import Axes3D
from sklearn.preprocessing import scale
```

```
import sklearn.metrics as sm
from sklearn import datasets
from sklearn.metrics import confusion_matrix,classification_report
```

Kako bi se mogla obaviti vizualizacija podataka pomoću biblioteke Matplotlib potrebno je podesiti parametre.

```
get_ipython().magic('matplotlib inline')
plt.rcParams['figure.figsize'] = 7, 4
```

Skup podataka iris.csv je unaprijed implementiran u skupove podataka koji dolaze s bibliotekom Sckit – learn, stoga ju nije potrebno učitavati iz nekog vanjskog direktorija. Učitavanje skupa podataka te postavljanje okvira podataka obavlja se sljedećim programskim kodom.

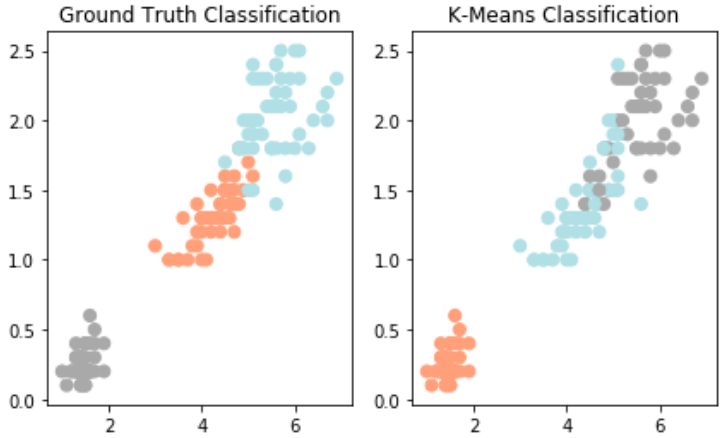
```
iris = datasets.load_iris()
X = scale(iris.data)
Y = pd.DataFrame(iris.target)
variable_names = iris.feature_names
X[0:10,]
```

Kako bi se izradio model potrebno je odrediti broj klastera i parametar random_state. Nakon postavljenih parametara postavlja se nazivi stupaca te slijedi izrada i grafički prikaz modela što je vidljivo u programskom kodu ispod. Za prikaz modela upotrjebljene su varijable dužine i širine latice.

```
clustering = KMeans(n_clusters=3, random_state = 5)
clustering.fit(X)
iris_df = pd.DataFrame(iris.data)
iris_df.columns =
['Sepal_Length','Sepal_Width','Petal_Length','Petal_Width']
Y.columns = ['Mete']
color_theme = np.array(['darkgray','lightsalmon','powderblue'])
plt.subplot(1,2,1)
plt.scatter(x=iris_df.Petal_Length,y=iris_df.Petal_Width,c=color_theme[iris
.target], s=50)
plt.title('Ground Truth Classification')
plt.subplot(1,2,2)
```

```
plt.scatter(x=iris_df.Petal_Length,y=iris_df.Petal_Width,c=color_theme[clus
tering.labels_], s=50)
plt.title('K-Means Classification')
```

Nakon prikazanog koda prikazuje se graf vidljiv na sljedećoj slici koji jasno prikazuje klastere.



Slika 14. Grafički prikaz klasteriranja cvjetova perunike

Nakon prikazanog grafa slijedi prikaz ocjene modela koja je prikazana na slici ispod.

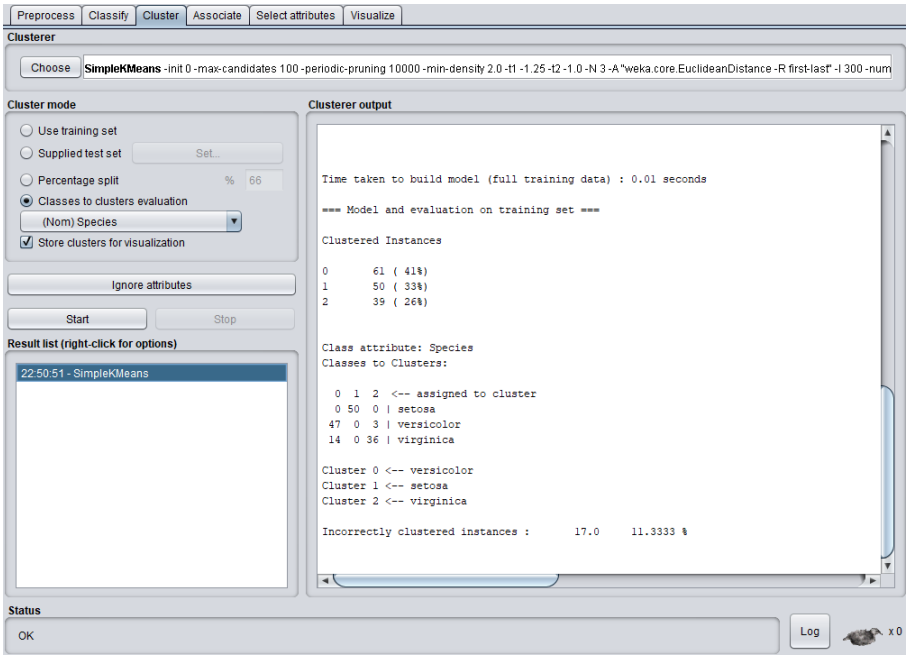
	precision	recall	f1-score	support
0	1.00	1.00	1.00	50
1	0.74	0.78	0.76	50
2	0.77	0.72	0.74	50
micro avg	0.83	0.83	0.83	150
macro avg	0.83	0.83	0.83	150
weighted avg	0.83	0.83	0.83	150

Slika 15. Ocjena modela klasteriranja

Na slici je jasno vidljivo da preciznost modela iznosi 0.83, osjetljivost je također 0.83 kao i mjera f1 te je na kraju vidljivo da su obrađeni svi članovi entiteta.

3.1.4. Izrada modela klasteriranja u alatu Weka

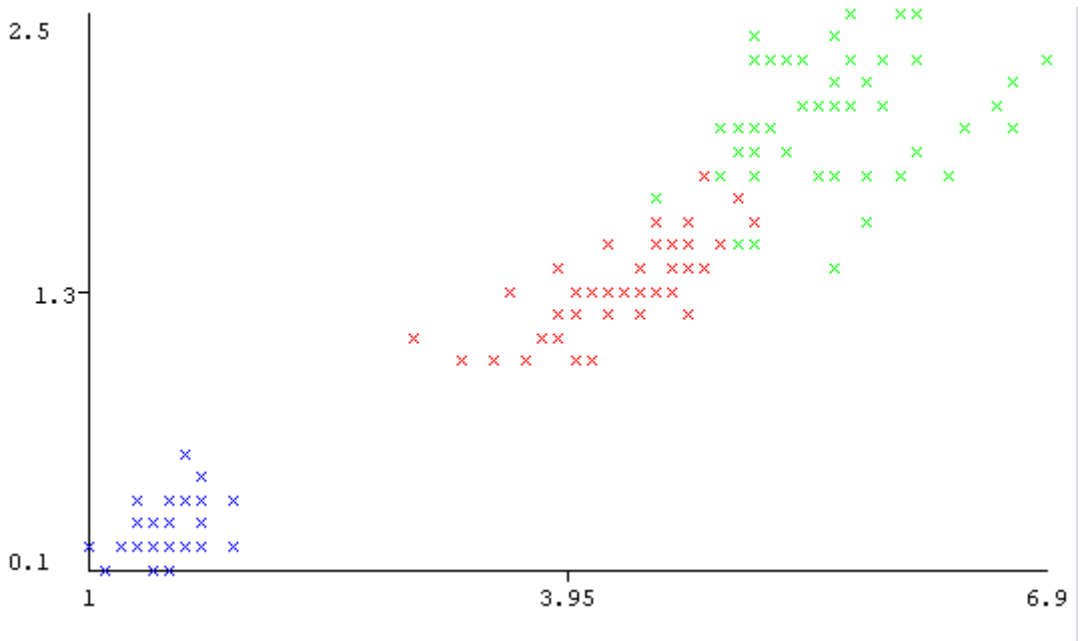
Postupak učitavanja skupa podataka u alat Weka je identičan kao za logističku regresiju pa nije potrebno ponovno objašnjavati postupak. Za razliku od Logističke regresije postupak izrade modela klasteriranja odvija se u kartici Cluster Vidljivoj na slici ispod.



Slika 16. Kartica Cluster u alatu Weka

Kako je vidljivo na slici odabran je algoritam klasteriranja K – Means te su postavljeni parametri kao što je broj klastera koji iznosi 3. Također na slici se vidi i izvješće klastera koje se izradi automatski s izradom modela.

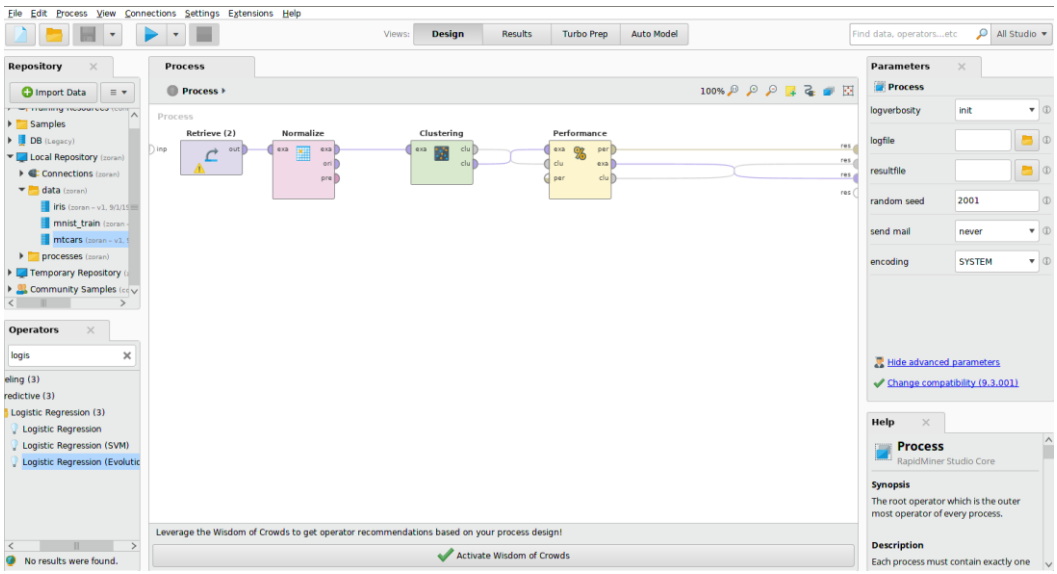
Grafički prikaz je vidljiv u kartici Visualize te grafički prikaz varijabli duljine i širine latice je vidljiv na slici ispod.



Slika 17. grafički prikaz klasteriranja cvijeta perunike u alatu Weka

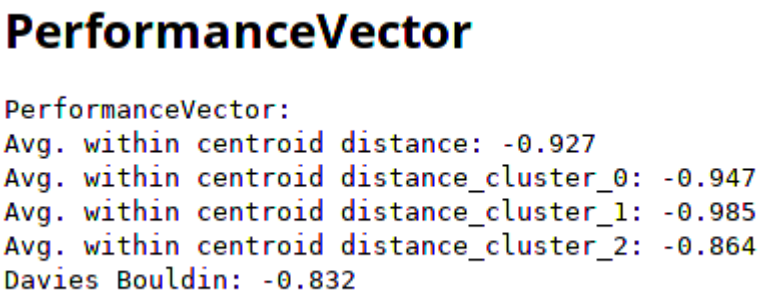
3.1.5. Izrada modela klasteriranja u alatu RapidMiner

Za razliku od Pythona koji koristi kodiranja i alata Weka, alat RapidMiner koristi modeliranje modela. Kako se može vidjeti na slici 18 model se sastavlja u središnjem dijelu alata. Nakon što je učitan model podatka te po potrebi se obradili podaci kreće se s modeliranje. Na početku se postavlja prvi kvadratić koji predstavlja skup podataka. Nakon što je učitan skup za potrebe ovog modela napravljena je normalizacija te se nakon toga pristupilo postupku klasteriranja. Pritiskom na kvadratić klasteriranje moguće je postaviti potrebne paramtre za izradu modela. Na kraju modela kako bi se izradilo izvješće potrebno je dodati kvadratić performanse.

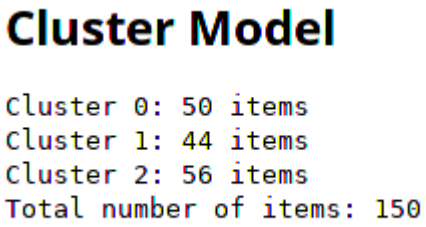


Slika 18. Izrada modela u alatu RapidMiner

Generirane performanse modela te broj članova entiteta podijeljenih po određenom klasteru su prikazani na slikama 19 i 20. Grafički prikaz varijabli dužine i širine latice nije potreban jer se ne razlikuje od grafova generiranih u alatu Weka i bibliotekom Matplotlib.



Slika 19. Performanse modela klasteriranja cvijeta perunike u alatu RapidMiner



Slika 20. Podjela članova entiteta po klasterima

Na slici 20 prikazana je podjela članova entiteta po klasterima. Klaster 0 označava skupinu cvjetova iris setosa, klaster 1 označava skupinu cijetova iris versicolor te klaster 2 označava cvijet virginica.

3.1.6. Usporedba biblioteke Scikit – learn s alatima Weka i RapidMiner

Rad s bibliotekom Scikit – learn zahtijevao je instalaciju dosta dodatnih biblioteka za razliku od alata kod kojih je sama instalacija dosta jednostavna. Također izrada modela je dosta složenija jer zahtjeva vještine kodiranja i dobro poznavanje područja strojnog učenja. Kako kod ove usporedbe nisu korišteni veliki skupovi podataka sama izrada modela kako u alatima tako i u jeziku Python je tekla izrazito brzo.

Kod izrade ovih modela daleko jednostavnije je bilo korištenje alata kod kojih se uz par klikova vrlo jednostavno mogao izraditi model te nakon same izrade odmah su bili dostupni izvještaji, ocjena modela te vizualizacija. Samim time izrada modela je bila jako pojednostavljena.

S druge strane kako je već navedeno za korištenje biblioteke Scikit – leran bilo je potrebno uključivanje dodatnih modula drugih biblioteka kako bi se mogla izvršiti vizaulizacija te obrada podataka. Brzina samih modela u sva tri slučaja je bila na najvišoj razini.

Sljedeća tablica prikazuje usporedbu alata Weka i RapidMiner s bibliotekom Scikit – learn.

Tablica 1: Usporedba biblioteke Scikit – learn s alatima Weka i RapidMiner temeljem modela klasteriranja

	Preciznost	**Brzina učenja**
Scikit – learn	0,83	<1 sec
Weka	0,88	<1 sec
RapidMiner	0,92	<1 sec

Brzina učenja biblioteke kako Scikit – leran tako i alata Weka i RapidMiner iznosila je daleko ispod jedne sekunde najvećim dijelom zbog malog skupa podataka koji se koristio. Alat RapidMiner neočekivano je postigao najbolju ocjenu preciznosti od čak 0,92, zatim slijedi alat Weka s 0,88 te na kraju je biblioteka Scikit – learn s 0,83.

Usporedba biblioteke Scikit – learn s alatom Weka temeljena modelom izrađenim algoritmom logističke regresije prikazana je sljedećom tablicom.

Tablica 2: Usporedba biblioteke Scikit – learn s alatom Weka temeljem modela logističke regresije

	Preciznost	**Brzina učenja**
Scikit – learn	0,82	<1 sec
Weka	0,72	<1 sec
RapidMiner	-	-

Poput modela koji je izrađen algoritmom klasteriranja brzina učenja ovih modela je također iznosila daleko ispod jedne sekunde najvećim djelom zbog malog skupa podataka nad kojim se provodilo učenje. Mjera preciznosti modela izrađenog bibliotekom Scikit – learn je oznosila 0,82, dok je mjera preciznosti modela izrađenog alatom Weka iznosila 0,72.

Istraživanje Škole računalnih znanosti i statistike o brzini i performansama biblioteka za strojno učenje i alata gdje je uzeto u obzir petnaest biblioteka i alata među kojima su biblioteka Scikit – learn i alat Weka navodi biblioteku Scikit – learn kao najbržu od petnaest ispitanih, a alat Weka zauzima četvrto mjesto iza biblioteka Tensorflow i MLib. Alat RapidMiner nije obuhvaćen ovim istraživanjem [23].

3.2. Biblioteka TensorFlow u odnosu na alate Weka i RapidMiner

Za usporedbu biblioteke TensorFlow s alatima Weka i RapidMiner korišten je jako velik skup podataka te algoritam dubokih neuronskih mreža kako bi se ispitale krajnje granice alata.

Skup podataka mnist.csv je korišten za izradu prediktivnog modela koji će na temelju ručno napisanih znamenki točno detektirati o kojoj se znamenki radi.

3.2.1. Izrada modela dubokih neuronskih mreža u biblioteci TensorFlow

Poput biblioteke Scikit – learn za prikaz i interpretaciju podataka potrebno je koristiti pojedine module iz biblioteka kao što su Numpy i biblioteka Matplotlib. Na početku pisanog programskog koda potrebno je uključiti biblioteku TensorFlow. Nakon što je biblioteka uključena potrebno je učitati skup podataka te postaviti potrebne parametre kao što su broj slojeva, broj neurona po sloju… Sve to možemo vidjeti u programskom kodu ispod.

```
import tensorflow as tf
mnist = tf.keras.datasets.mnist
(x_train, y_train), (x_test, y_test) = mnist.load_data()
x_train = tf.keras.utils.normalize(x_train, axis=1)
x_test = tf.keras.utils.normalize(x_test, axis=1)

model = tf.keras.models.Sequential()
model.add(tf.keras.layers.Flatten())
model.add(tf.keras.layers.Dense(128, activation=tf.nn.relu))
model.add(tf.keras.layers.Dense(128, activation=tf.nn.relu))
model.add(tf.keras.layers.Dense(10, activation=tf.nn.softmax))

model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])
model.fit(x_train, y_train, epochs=3)
```

Nakon što je model izrađen potrebno je izračunati pogotke i gubitke čije je računanje vidljivo u programskom kodu ispod.

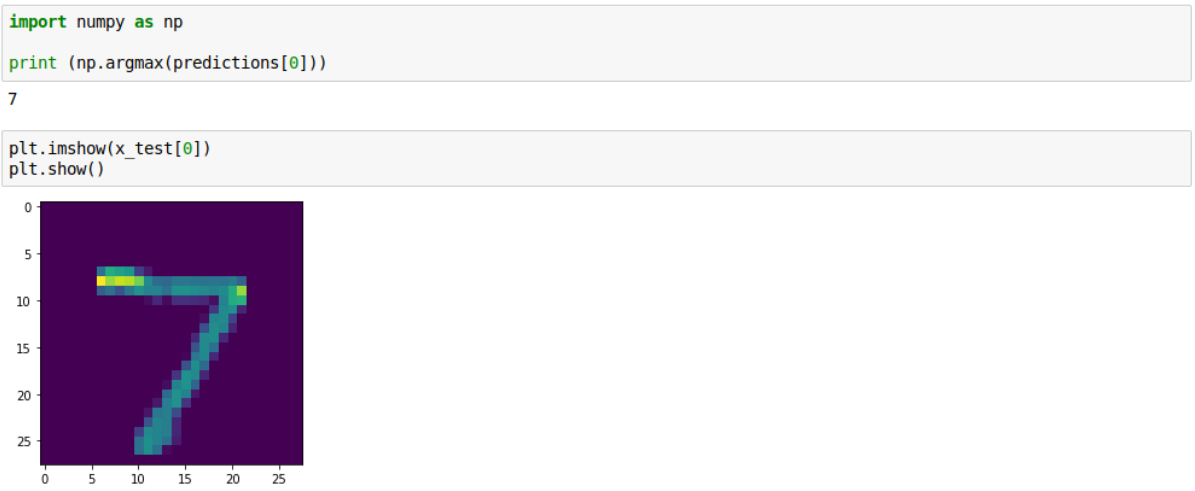
```
val_loss, val_acc = model.evaluate(x_test, y_test)
print(val_loss, val_acc)
```

Stopa gubitaka iznosi 0.11 dok je stopa pogodaka 0.97. Kako je stopa gubitaka i pogodaka zadovoljavajuća može se prijeći na predviđanje. U programskom kodu ispod vidljiva je izrada prediktora te interpretacija predikcije pomoću biblioteke Numpy.

```
predictions = model.predict(x_test)
print(predictions)
import numpy as np
print (np.argmax(predictions[0]))

plt.imshow(x_test[0])
plt.show()
```

Primjer pogotka vidljiv je na slici ispod gdje je vidljiv pogodak broja 7.



Slika 21. Pogodak broja 7 pomoću modela izrađenog s bibliotekom TensorFlow

3.2.2. Usporedba biblioteke TensorFlow s alatima Weka i RapidMiner

U tablici ispod prikazana je usporedba biblioteke TensorFlow s alatima Weka i RapidMiner.

Tablica 3: Usporedba biblioteke TensorFlow s alatima Weka i RapidMiner

	Točnost	**Gubitak**	**Brzina učenja**
TensorFlow	96,67%	0,1145	15 sec
Weka	96,32%	0,1337	873 sec
RapidMiner	-	-	-

Izrada modela dubokih neuronskim mreža s ovako velikim skupom podataka u jeziku Python je protekla gotovo jednakom brzinom kao i izrada modela klasteriranja i logističke regresije s malim brojem podataka. Potrebno vrijeme za izradu modela je bilo 15 sekundi. Iako je potrebno kodirati sama izrada modela je bila izrazito jednostavna te brzina modela za predviđanje je jako brza. Sama izrada modela je protekla bez većih hardverskih napora. Točnost modela izrađenog pomoću biblioteke TensorFlow iznosila je 96,67% s gubitkom odnosno odstupanjem dobivenog rezultata od očekivanog rezultata od 0,1145.

S druge strane izrada modela u alatima Weka i Rapidminer je uzrokovala određene probleme. Alat RapidMiner nakon što je učitavanje skupa podataka trajalo nekoliko minuta nije uspio izradit model jer je potrošio cijelu radnu memoriju računala te daljnje korištenje računalom moglo se nastavit jedino uz resetiranje istog.

Brzina učenja obavljenog pomoću alata Weka iznosila je 873 sekunde odnosno 14 minuta i 33 sekunde što je znatno povećanje s obzirom na biblioteku TensorFlow. Iako je brzina učenja bila znatno lošija s obzirom na biblioteku TensorFlow točnost koja iznosi 96,32% nije previše odstupala od rezultata postignutih s bibliotekom TensorFlow. Gubitak modela iznosi 0,1337 što također nije preveliko odstupanje od biblioteke TensorFlow.

Već spomenuto istraživanje Škole računalnih znanosti i statistike također navodi biblioteku Tensorflow kao drugu u poretku mjerenja brzine i performansi biblioteka i alata namijenjenih strojnom učenju što je dva mjesta iznad alata Weka. Navedeni podatak potvrđuje tezu da je biblioteka TensorFlow brža u usporedbi s alatima. Alat RapidMiner nije obuhvaćen ovim istraživanje, ali s obzirom da se radio o alatu koji za izradu modela koristi modeliranje očito je da je lošije pozicioniran s obzirom na brzinu i performanse kako od alata Weka tako i od biblioteke TensorFlow [23].

4. Zaključak

Nakon odrađene izrade modela te napravljene usporedbe očito je koje su prednosti, a koje mane biblioteka namijenjenih strojnom učenju. Samim time što su biblioteke otvorenog koda omogućava im jednostavno praćenje standarda te brzo ažuriranje samih biblioteka. Velika prednost ovih biblioteka kako je vidljivo iz primjera s dubokim učenjem je brzina koja ne dolazi do tolikog značaja ukoliko se koristi skup s manjim brojem podataka. Također sama implementacija modela u aplikacije ili neku od hardverskih opcija je znatno jednostavnija za razliku od alata.

Što se tiče alata, oni svoje prednosti imaju u jednostavnoj instalaciji te korištenju. Pozivajući se na usporedbu manjih skupova podataka vidljivo je da je brzina izrade modela jednako brza kao u jeziku Python. Samo korištenje je višestruko lakše posebno za ljude koji se ne snalaze baš u kodiranje ili im jezik Python nije toliko poznat. Također vizualizacija rezultata te generiranje izvještaja dolazi odmah uz izradu modela što skraćuje dosta vremena. Velika mana ovih alata je susretanje s velik skupom podataka gdje učenje ukoliko se može izvršiti traje jako dugo vremena.

Dobiveni rezultati nakon testiranja modela klasifikacije algoritmom K – means dobiveni su pomalo neočekivani rezultati gdje je najveću preciznost imao model izrađen alatom RapidMiner, nakon toga model izrađen alatom Weka i na kraju model izrađen bibliotekom Scikit – learn. Za razliku od modela izrađenih algoritmom K – means testiranjem modela izrađenih algoritmom logističke regresije dali su očekivane rezultate gdje je najbolji rezultat preciznosti ostvario model izrađen bibliotekom Scikit – learn. Rezultati testiranja modela biblioteke TensorFlow s alatima Weka i RapidMiner nisu pokazali velika odstupanja u točnosti i gubitku modela izrađenog pomoću biblioteke TensorFlow i modela izrađenog alatom Weka. Najveća razlika kod ovih modela pokaza se kod vremena potrebnog za učenje koje je bilo drastično kraće kod biblioteke TensorFlow.

Za izradu manjih modela korištenje alata je jako dobar izbor ukoliko to uvjeti dozvoljavaju te korištenje alata poput RapidMinera je izrazito dobro za edukaciju jer vizualiziraju svaku radnju potrebnu u izradi modela. Ukoliko nam je model potreban za neke ozbiljnije stvari bolji bi odabir bio korištenje Pythona i njegovih biblioteka jer pružaju jako dobre brzine.

Popis literature

[1] R. Bharadwaj, „What is the Importance of Artificial Intelligence in Everyday Life?“, *Globussoft*, 2017. [Na internetu]. Dostupno na: https://globussoft.com/importance-of-artificial-intelligence/. [Pristupljeno: 12-ruj-2019].

[2] A. L. Samuel, *Some Studies in Machine Learning Using the Game of Checkers*. IBM Journal of Research and Development, 1959.

[3] S. Gollapudi, *Practical Machine Learning*. Birmingham: Packt Publishing, 2016.

[4] A. Geron, *Hands - On Machine Learning with Scikit - Learn & TensorFlow*. Sebastopol: O'Reilly Media, 2017.

[5] S. Šegvić, „‚Duboko učenje' Uvodno predavanje“, *Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva*. [Na internetu]. Dostupno na: http://www.zemris.fer.hr/~ssegvic/du/du0intro.pdf.

[6] I. Goodfellow, Y. Bengio, i A. Courville, *Deep Learning*. Cambridge, Massachusetts: The MIT Press, 2015.

[7] M. Mohri, A. Rostamizadeh, i A. Talwalker, *Fundations of Machine Learning*. Cambridge, Massachusetts: The MIT Press, 2012.

[8] B. Kliček, „Uvod i stabla odlučivanja“, *Sveučilište u Zagrebu, Fakultet organizacije i informatike*, 2018. [Na internetu]. Dostupno na: https://elfarchive1819.foi.hr/mod/resource/view.php?id=1910.

[9] K. P. Murphy, *Machine Learning: A Probabilistic Perspectiv*. Cambridge, Massachusetts: The MIT Press, 2012.

[10] G. Hackling, *Mastering Machine Learning with scikit-learn*. Birmingham: Packt Publishing, 2014.

[11] N. Mehić, „Homo sapiens - razumno biće?“, *Istraži me*, 2014. [Na internetu]. Dostupno na: http://www.istrazime.com/wp-content/uploads/2014/05/vgjvzhuztgrt.png.

[12] „Logistic Regression with Python“, *Medium*, 2019. [Na internetu]. Dostupno na:

https://miro.medium.com/max/1200/0*gKOV65tvGfY8SMem.png.

[13] S. Cass, „The 2018 Top Programming Languages“, *IEEE Spectrum*, 2018. [Na internetu]. Dostupno na: https://spectrum.ieee.org/at-work/innovation/the-2018-top-programming-languages.

[14] F. i zur. Pedregosa, „Scikit-learn: Machine Learning in Python“, *Journal of Machine Learning Research*, 2011. [Na internetu]. Dostupno na: http://jmlr.org/papers/v12/pedregosa11a.html.

[15] „Choosing the right estimator [slika]“. [Na internetu]. Dostupno na: https://scikit-learn.org/stable/_static/ml_map.png.

[16] Indiana University, „Scikit-learn“, *Indiana University*, 2019. [Na internetu]. Dostupno na: http://dskb.soic.indiana.edu/catalog/b0dcd397-de2f-4dc3-904f-e7925939e1e8.

[17] Google, „TensorFlow: Open source machine learning“, 2015. [Na internetu]. Dostupno na: https://www.youtube.com/watch?v=oZikw5k_2FM.

[18] H. Patel, „TensorFlow Pros and Cons — The Bright and the Dark Sides“, *Medium*, 2018. [Na internetu]. Dostupno na: https://medium.com/@patelharshali136/tensorflow-pros-and-cons-the-bright-and-the-dark-sides-5fefbb11fb96.

[19] „Weka 3: Machine Learning Software in Java“. [Na internetu]. Dostupno na: https://www.cs.waikato.ac.nz/ml/weka/.

[20] F. Ducatelle, „Software for the Data Mining Course“, *The University od Edinburgh*. [Na internetu]. Dostupno na: http://www.inf.ed.ac.uk/teaching/courses/dme/html/software2.html.

[21] M. Hofmann i R. Klinkenberg, *RapidMiner: Data Mining Use Cases and Business Analytics Applications*, 1. izdanje. Boca Raton, Florida: Taylor & Francis, CRC Press, 2013.

[22] K. Rangra i K. L. Bansal, „Comparative Study of Data Mining Tools“, *Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, sv. 4, izd. 6, str. 222, 2014.

[23] J. Beel, „Experience with and Preference of Machine-Learning Libraries: scikit-learn vs. Tensorflow vs. Weka … [What Machine-Learning Students

Think/Like/Know/Are …]“, *School of Computer Science and Statistics*, 2018. [Na internetu]. Dostupno na: https://www.scss.tcd.ie/joeran.beel/blog/2018/01/28/what-machine-learning-students-think-like-know-are-experience-with-and-preference-for-machine-learning-libraries-scikit-learn-vs-tensorflow-vs-weka/. [Pristupljeno: 12-ruj-2019].

Popis slika

Popis tablica