Analiza rada algoritama strojnog učenja na zadanom skupu podataka

Krupić, Omar

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

ZAVRŠNI RAD

Omar Krupić

Zagreb, 2023. godina.

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

Analiza rada algoritama strojnog učenja na zadanom skupu podataka

Mentor:

Izv. prof. dr. sc. Tomislav Stipančić, dipl. ing.

Student:

Omar Krupić

Zagreb, 2023. godina.

Izjavljujem da sam ovaj rad izradio samostalno koristeći znanja stečena tijekom studija i navedenu literaturu.

Zahvala pripada uzvišenom Bogu koji mi je bio vodilja u cijelom dosadašnjem studiranju. Posebno bih istaknuo svog mentora izv. prof. dr. sc. Tomislava Stipančića koji mi je bio velika podrška i motivacija prilikom izrade ovog završnog rada, te mu se na tome mnogo zahvaljujem. Također, hvala mojoj porodici i prijateljima koji su prilikom studija bili uz mene.

Omar Krupić

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE
Središnje povjerenstvo za završne i diplomske ispite
Povjerenstvo za završne i diplomske ispite studija strojarstva za smjerove:
proizvodno inženjerstvo, računalno inženjerstvo, industrijsko inženjerstvo i menadžment, inženjerstvo materijala i mehatronika i robotika

Sveučilište u Zagrebu Fakultet strojarstva i brodogradnje	
Datum	Prilog
Klasa: **602 – 04 / 23 – 6 / 1**	
Ur.broj: **15 - 1703 - 23 -**	

ZAVRŠNI ZADATAK

Student: **Omar Krupić** JMBAG: **0035222758**

Naslov rada na hrvatskom jeziku: **Analiza rada algoritama strojnog učenja na zadanom skupu podataka**

Naslov rada na engleskom jeziku: **Machine learning algorithms performance analysis on a given data set**

Opis zadatka:

U svijetu podataka i velike količine informacija algoritmi umjetne inteligencije postaju sve značajniji. U ovisnosti o vrsti i strukturi podataka te metodi strojnog učenja algoritmi se koriste prilikom predviđanja, odlučivanja, klasifikacije, prepoznavanja, i slično. Prije primjene algoritma na zadanom skupu podatke je potrebno obraditi i prilagoditi.

Na zadanom skupu podataka potrebno je izvršiti funkcije prilagodbe podataka uključujući funkcije čitanja i čišćenja podataka, funkcije vizualizacije podataka, te funkcije skaliranja podataka. Nakon toga je potrebno primijeniti tri različita algoritma strojnog učenja (naivni Bayesov klasifikator, klasifikator k-najbližih susjeda te klasifikator logističke regresije) koji će podatke podijeliti u klase. U sljedećem koraku je potrebno usporediti rezultate izvođenja algoritama s gledišta preciznosti te odrediti koji je najuspješniji.

U radu je potrebno navesti korištenu literaturu i eventualno dobivenu pomoć.

Zadatak zadan:

30. 11. 2022.

Zadatak zadao:

Doc. dr. sc. Tomislav Stipančić

Datum predaje rada:

1. rok: 20. 2. 2023.
2. rok (izvanredni): 10. 7. 2023.
3. rok: 18. 9. 2023.

Predviđeni datumi obrane:

1. rok: 27. 2. – 3. 3. 2023.
2. rok (izvanredni): 14. 7. 2023.
3. rok: 25. 9. – 29. 9. 2023.

Predsjednik Povjerenstva:

Prof. dr. sc. Branko Bauer

SADRŽAJ

POPIS SLIKA

POPIS TABLICA

POPIS OZNAKA

Oznaka	Opis
H_j	Hipoteza
$Pr()$	Vjerojatnost
$Pr(\|)$	Uvjetovana vjerojatnost
v_{MAP}	Najvjerojatnija target vrijednost
a_n	Vrijednosni atributi
v_{NB}	Vrijednost Bayesovog klasifikatora
X	Primjer u ulaznom prostoru
μ	Prosječna vrijednost
σ	Varijanca
$p(x)$	Funkcija gustoće
k	Broj klasa
β_i	Težina nezavisnih varijabli
z	Linearna kombinacija nezavisnih varijabli
β_k	Parametri modela logističke regresije
h(x)	Hipoteza modela strojnog učenja
tp	Broj pozitivnih vrijednosti predviđenih kao pozitivne
fp	Broj negativnih vrijednosti predviđenih kao pozitivne
fn	Broj pozitivnih vrijednosti predviđenih kao negativne
tn	Broj negativnih vrijednosti predviđenih kao negativne
x_train	Primjeri za učenje
y_train	Klase primjera za učenje
x_test	Primjeri za testiranje
y_test	Klase primjera za testiranje
y_pred	Predviđena klasa za x_test

SAŽETAK

U svijetu podataka i velike količine informacija algoritmi umjetne inteligencije postaju sve značajniji. U ovisnosti o vrsti i strukturi podataka te metodi strojnog učenja algoritmi se koriste prilikom predviđanja, odlučivanja, klasifikacije, prepoznavanja i slično. U ovom radu bit će prikazana analiza zadanog skupa podataka kroz tri algoritma strojnog učenja. Nakon prilagodbe podataka i njihove pripreme, te nakon primjene algoritama umjetne inteligencije, izvodi se zaključak o valjanosti izvođenja pojedinog algoritma. Svaki algoritam : naivni Bayesov klasifikator, klasifikator k- najbližih susjeda te klasifikator logističke regresije prethodno je opisan u sklopu posebnog poglavlja gdje su izložene sve potrebne pojedinosti za njihovo pravilno razumijevanje.

Ključne riječi: algoritam, strojno učenje, naivni Bayesov klasifikator, klasifikator k-najbližih susjeda, klasifikator logističke regresije, umjetna inteligencija

SUMMARY

In the world of data and large amounts of information, artificial intelligence algorithms are becoming increasingly important. Depending on the type and structure of the data and the machine learning method, algorithms are used for prediction, decision-making, classification, recognition, and the like. In this paper, analysis of a given dataset through three machine learning algorithms is presented and explained. After finishing dataset setup, and after the use of every artificial inteligence algorithm, the conclusion is given about the precision of every algorithm. Each algorithm: naive Bayes classifier, k nearest neighbor, logistic regression is previously described in seperated sections where are displayed all informations and details about their correct use and understanding.

Key words: algorithm, machine learning, naive Bayes classifier, k nearest neighbor, logistic regression, artificial intelligence

1. UVOD

Glavna tematika ovog rada vezana je za strojno učenje te algoritme strojnog učenja kao sastavnog dijela umjetne inteligencije. Povod nastanka ove naučne discipline jeste upravljanje podacima Tako je i sami cilj strojnog učenja izgraditi program koji se dobro prilagođava podacima [1]. Strojno učenje je zapravo područje umjetne inteligencije koje se bavi razvojem algoritama i modela koji omogućavaju računalima učenje i donošenje predviđanja ili odluka bez eksplicitnog programiranja. Postoje različiti algoritmi strojnog učenja, svaki dizajniran za rješavanje različitih vrsta problema. U nastavku, prikazat će se različite vrste algoritama strojnog učenja i njihove primjene. Bit će riječi o linearnoj regresiji koja se koristi za predviđanje kontinuiranih izlaznih varijabli na temelju ulaznih značajki. Zatim o logističkoj regresiji koja se primarno koristi za binarne klasifikacijske zadatke, kao što su detekcija neželjene pošte (spam) ili medicinska dijagnoza. Također spominje se klasifikator k najbližeg susjeda koji se koristi za klasifikaciju i regresiju pronalaženjem najsličnijih podataka u skupu za učenje. Pored pomenutih bit će riječi i o metodi Gaussovog naivnog Bayesa koji je posebno koristan kada su značajke kontinuirane i imaju normalnu distribuciju.

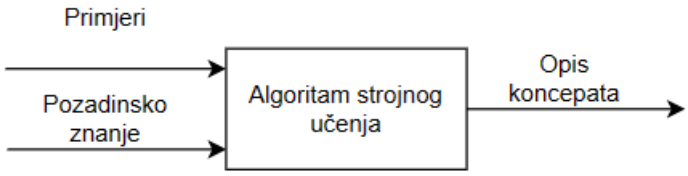
2. OPĆENITO O STROJNOM UČENJU

Naglim razvojem informatičkih nauka, strojno učenje u posljednjih nekoliko desetljeća sve više dobiva na značaju. Kao jedna od najbrže rastućih disciplina današnjice, ovo naučno područje predstavlja vodeće sredstvo za obradu, sortiranje te klasifikaciju podataka. Usljed pojave mnoštva podataka, prvenstveno zbog postojanja društvenih mreža te razvoja sredstava komunikacije, ali i razvoja digitalnih tehnologija u sklopu različitih naučnih polja, napredak ove discipline se nameće sam po sebi, te postaje nužnost. Stoga i nije čudno zašto se kontinuirano povećavanjem količine podataka, konstantno unaprijeđuju metode koje daju rješenje kako takve „hrpe" podataka pretvoriti u smislene cjeline. Samo na taj način obrađeni podatci imaju svoju vrijednost, te kao vid znanja mogu se iskoristiti u različite svrhe. Zbog toga se kaže da u ovisnosti da li su podaci vezani za nauku ili su pak oni privatni, podatci koji su pasivno pohranjeni nisu od velike koristi. Bolji način njihovog iskorištenja jeste pronalaziti nove načine kako iskoristiti takve podatke i pretvoriti ih u koristan produkt ili uslugu. U ovoj transformaciji, strojno učenje igra sve veću i veću ulogu[15]. Ono što zapravo ovu tranformaciju čini mogućom jesu sami **algoritmi**. Algoritmi strojnog učenja kao poseban slijed zadataka i procedura koje dovode do riješenja nekog problema, omogućavaju da se na bazi matematičko-statističkih formula, dođe do egzaktnih i ispravnih rješenja. O kojem god algoritmu da se radi, znanstveni cilj i praktični zadatak je da se teorijski okarakterizira sposobnost pojedinačnog algoritma strojnog učenja i njegova sposobnost rješenja bilo kojeg problema prilikom učenja: Koliko precizno može algoritam naučiti pomoću pojedinog tipa ili opsega podataka? Koliko je algoritam precizan prema greškama modeliranja, pretpostavkama i greškama unutar podataka?[16].

Upravo to jeste i pitanje koje se obrađuje u ovom radu: koji algoritam primjeniti na zadani problem kako bi dobili najbolji rezultat u konačnici?

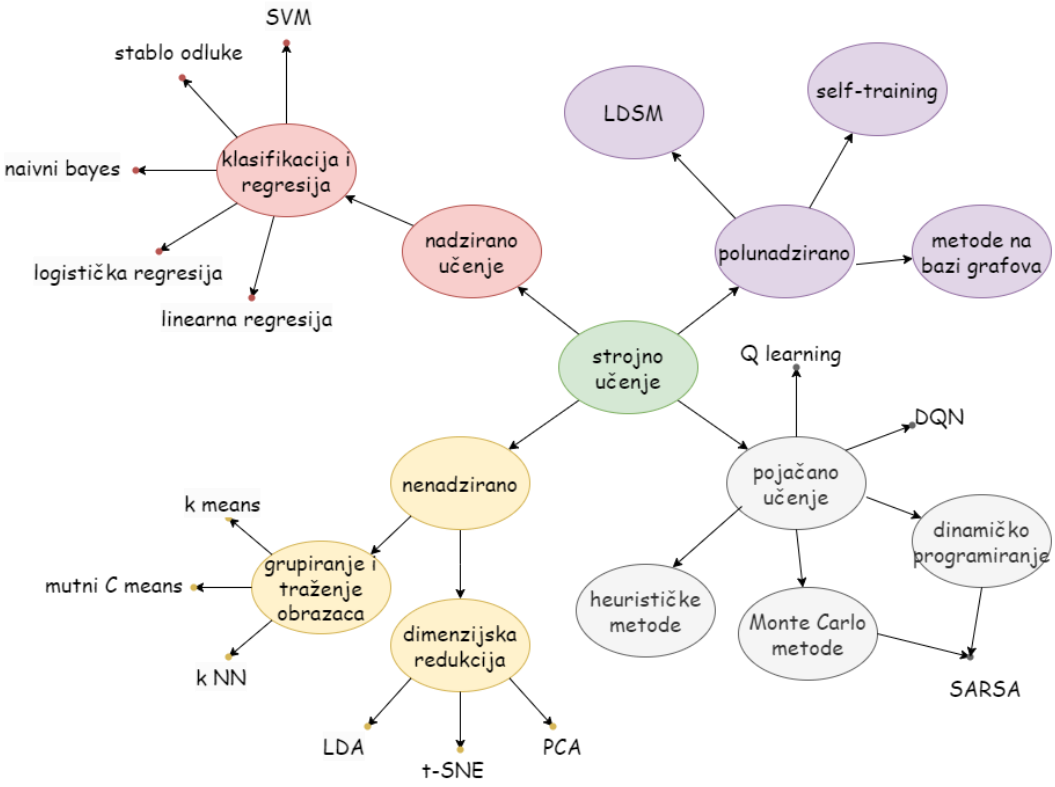
2.1. Kako stroj uči? Podjela strojnog učenja

Metodologija rada koja se primjenjuje pri strojnom učenju izrazito varira. Svaki algoritam radi prema svojevrsnim principima a njegova primjena ovisi o vrsti problema koji je potrebno riješiti. Bez obzira o kojem se algoritmu radi, generalni princip se može izvesti sljedećom formulacijom: sustav učenja cilja određivanje opisa datog koncepta od skupa konceptualnih primjera zadanih od strane učitelja i pozadinskog znanja [1], kako je prikazano prema Slika 1:



Slika 1 Osnovna shema strojnog učenja

Osnova svakog algoritma strojnog učenja jesu podatci. Bez obzira koliko je sofisticiran postojeći algoritam, ukoliko njegov temelj nije pripremljen na pravi način (a temelj su podatci) algoritam neće biti uspješan prilikom testiranja na podatcima iz stvarnih životnih situacija [24]. U ovisnosti o njihovoj vrsti te načinu njihove obrade, strojno učenje se može podijeliti na nekoliko tipova: nadzirano, nenadzirano, polunadzirano, pojačano učenje



Slika 2 Podjela strojnog učenja

2.1.1. Nadzirano učenje

Nadzirano učenje ima za zadatak učenje funkcija koje mapiraju ulazne podatke do izaznih na osnovu ulazno-izlaznih parova. Ovakvom metodom se zaključuje funkcija iz tzv. trening

podataka koji se sastoje iz niza primjera [17]. Specifičnost nadziranog učenja jeste postojanje pozadinskog znanja u vidu **označenih podataka** (eng. *labeled data*). Takve podatke je potrebno pripremiti i kao takve predstaviti zadanom algoritmu. Zato se smatra da su algoritmi nadziranog strojnog učenja oni kojima je potrebna vanjska podrška.[17]. Kao jednostavan primjer označenih podataka može poslužiti fotografija automobila povezanog sa oznakom „automobil“ ili tekstualni opis određenog produkta kome se dodjeljuje njegov naziv. Takvi podatci predstavljaju bazu znanja za daljnji rad algoritma. Recimo da se želi naučiti klasa skupine porodičnih automobila. Postavlja se skup primjeraka automobila. Potrebno je imati skupinu ljudi nad kojima će se provesti ispitivanje. Svaki pojedinac će ocijeniti automobil kao onaj koji je prihvatljiv ili nije prihvatljiv za zadanu klasu (recimo kao prikladan porodični automobil). Učenje klase jeste pronalaženje opisa koji je sačinjen od strane svih pozitivnih i negativnih primjera. Na taj način se čini pretpostavka: za potpuno novi automobil, prema opisu koji je naučen, unaprijed se može svrstati da li spada u klasu porodičnih automobila ili ne bez provođenja upitnika nad ispitanicima [15]. Također ulazni skupovi podataka su podijeljeni na „trening“ i „test“ skupove podataka.[17] Svi algoritmi uče neke vrste obrazaca od trening skupa podataka, i primjenjuju ih na test skupove podataka za predviđanje ili klasifikaciju [17].

Algoritmi koji rade po navedenom principu jesu neuronske mreže, naivni bayes, linearna regresija, logistička regresija, nasumična šuma, i stroj potpornih vektora (SVM). U nastavku rada, neki od pomenutih će se detaljnije obraditi.

2.1.1.1. Klasifikacija

Klasifikacija se može pojasniti na primjeru bankovnog sustava. Banka kao financijska ustanova dodjeljuje određene količine novca putem kreditiranja. Za banku je bitno da je u mogućnosti unaprijed predvidjeti rizik vezan za posudbu, te imati u vidu koja je zapravo vjerojatnost da korisnik neće biti u mogućnosti vratiti cijelokupni iznos posudbe. Ovo je bitno iz razloga da banka osigura svoju zaradu te da ne preoptereti korisnika sa posudbom iznad njegove mogućnosti. Iz tog razloga banka ima zabilježenu povijest posudbi koja sadrži korisničke podatke (poput primanja, ušteđevine, profesije, godina, financijskih aktivnosti u prošlosti i sl.) te da li su korisnici vratili svoje prethodne posudbe ili ne. Cilj je da se od sličnih podataka uspije stvoriti generalno pravilo koje bi povezivalo svojstva korisnika te rizik vezan za njega. To je zapravo model strojnog učenja koji na osnovu prethodno prikupljenih

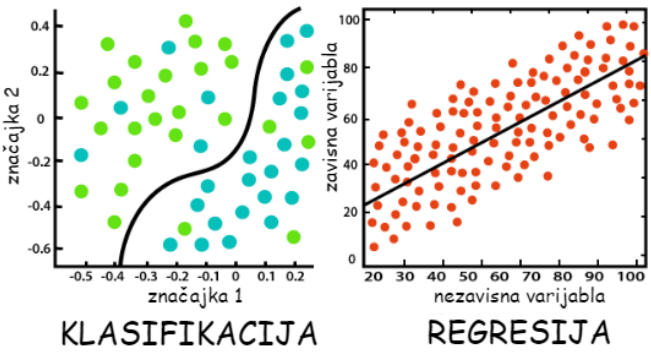
podataka uspjeva izračunati rizik novih aplikanata i odlučiti da li ga prihvatiti ili ne. Upravo ovo je primjer klasifikacijskog problema gdje postoje dvije klase: klasa niskog i klasa visokorizičnog korisnika [15]. Korisnički podatci bi predstavljali početnu točku klasifikacijskog procesa. Na osnovu postojećih podataka o korisnicima, klasifikator obavlja zadatak da odredi kojoj će klasi dodijeliti nove korisnike. Nakon treniranja sa postojećim podatcima, klasifikator određuje pravilo po kojem će se vršiti klasifikacija. To može da bude jednostavno „if – then“ pravilo:

IF prihod > F1 AND ušteđevina > F2 THEN nizak_rizik ELSE visok_rizik

gdje su F1 i F2 dodijeljene vrijednosti. Ovakav primjer zapravo predstavlja diskriminantu, odnosno funkciju koja odvaja primjere različitih klasa.

2.1.1.2. Regresija

Pojam regresija označava statističku metodu kod koje se jedna varijabla predstavlja na osnovu jedne ili više drugih varijabli. Varijabla koja se predstavlja naziva se **zavisna** ili **odzivna varijabla**, dok se ostale varijable koje se koriste za predstavljanje ili predviđanje odziva nazivaju se **nezavisne** varijable. Mnoge nezavisne varijable se tretiraju kao svojevrsni **prediktori** [20].

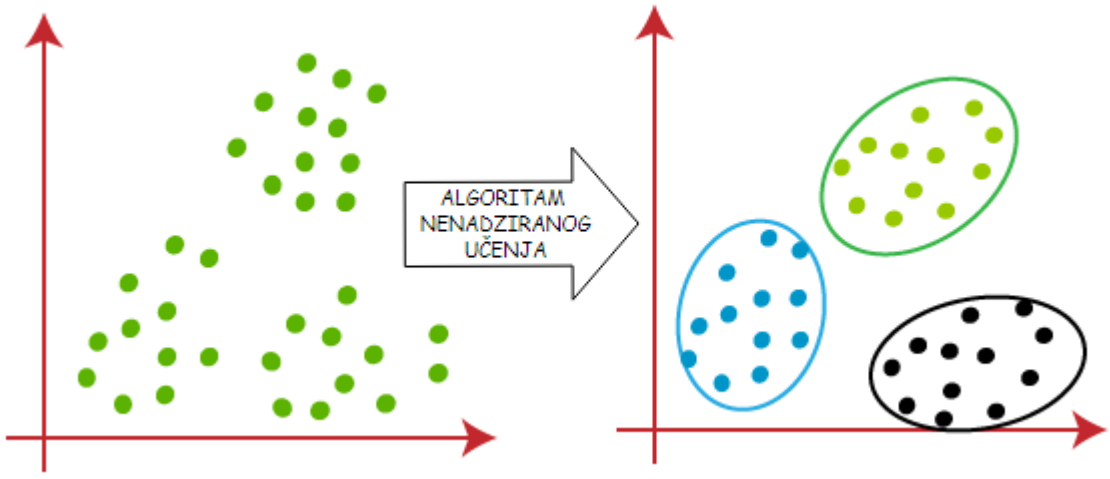


Slika 3 Usporedba grafičkog prikaza podataka nakon postupka klasifikacije i regresije

2.1.2. Nenadzirano učenje

Za razliku od nadziranog učenja, ne postoji točan odgovor i ne postoji učitelj. Algoritmi samostalno osmišljava, otkriva i predstavlja strukture unutar podataka. Algoritmi nenadziranog učenja uče nekoliko značajki od podataka. Kada je novi podatak predstavljen, on koristi prethodno naučene značajke da bi prepoznao klasu podatka [17].

Kod nenadziranog učenja ne postoje klasificirani, niti označeni skupovi podataka, već se radi o „hrpi“ podataka koji nemaju nikakve oznake. Primjer neoznačenog podatka bi bio nekakav nasumični broj, vektor ili matrica koju je iz okoline pokupio senzor i o kojem nemamo više nikakve informacije, niti o njegovom okruženju niti kontekstu. Može se reći da nenadzirano učenje nema početnu bazu prema kojoj bi se referirao. Algoritam sam sastavlja i pronalazi smislene veze između podataka. U nenadziranom učenju, ne postoji nadzornik već postoje samo ulazni podaci. Cilj je da se nađu regularnosti unutar ulaznih podataka [15]. Postoji struktura prema ulaznom prostoru takva da se određeni obrasci ponavljaju češće nego drugi, a cilj je vidjeti što se generalno događa a što ne. U statistici se to naziva **gustoćom procjene**. Jedna od metoda za gustoću procjene jeste **clustering** gdje je cilj da se nađu klasteri ili grupe ulaznih podataka [15].



Slika 4 Grafovi prije i nakon provedbe algoritama nenadziranog učenja

Neke od metoda koje se koriste u sklopu nenadziranog učenja su: analiza glavnih komponenti (eng. *Principal component analysis* ili *PCA*), dekompozicija pojedinačne vrijednosti (eng. *singular value decomposition* ili *SVD*), neuronske mreže, k-means klasteri, probabilističke metode klasteriranja. Jedna od prednosti koja predhodi nadziranom učenju sa nenadziranim klasteriranjem ili dimenzijskom redukcijom jeste da ne postoji kasnija potreba za označenim podacima. Označavanje podataka je skupo. Mi možemo koristiti veće količine neoznačenih podataka za učenje klaster parametara a onda iskoristiti manje skupine nadziranih podataka za učenje druge faze klasifikacije i regresije [15].

2.1.3. Polunadzirano učenje

Predstavlja kombinaciju nadziranog i nenadziranog učenja. Razlika je u tome da za vrijeme treninga, algoritam koristi manje skupine označenih podataka kako bi dobio usmjerenje za klasifikaciju i ekstrakciju značajki od većih neoznačenih skupova podataka. Polunadzirano učenje rješava probleme nedovoljnog broja označenih podataka kod algoritama nadziranog učenja. Nameće se pitanje: koliki je značaj polunadziranog učenja? Najveći značaj jeste upravo u situacijama kada je dobivanje podataka jeftino, dok dobavljanje oznaka košta mnoštvo vremena, truda i novca što je slučaj u mnogo područja primjene. Primjer za to jeste prepoznavanje glasa kod kojeg se gotovo besplatno može snimiti velika količina govora, ali bi njegovo označavanjei iziskivalo osobu koja bi to morala preslušati i napisati transkripte [25].

2.1.4. Pojačano učenje

Pojačano učenje je jedno od 3 osnovne paradigme strojnog učenja, zajedno sa nadziranim i nenadziranim učenjem. To je područje strojnog učenja čiji je zadatak odrediti u kojoj mjeri softverski agenti moraju imati učešća u okruženju da bi se maksimizirala koncepcija kumulativnih nagrada [17]. Može se reći da se na temelju ocjene i evaluacije rezultata određuje povratna informacija koji služi kao korekcija. Kod poticanog učenja povratna veza može biti jednostavni binarni signal (1 - dobro ili 0 - loše), ili kompleksna numerička evaluacija temeljena na odgovarajućoj metrici [26]. Tako agent na osnovu povratne veze prilagođava svoje učenje.

2.2. Naivni Bayesov klasifikator

Naivni Bayesov klasifikator spada u probablističke klasifikatore. Predstavlja klasifikacijsku tehniku zasnovanu na Bayesovom teoremu sa pretpostavkom nezavisnosti između prediktora [17]. U samom imenu se skrije i prethodno pomenuta definicija. Naziv predstavlja kovanicu dvije riječi:

- Naivni: prisustvo određenih značajki je neovisno o prisustvu drugih. Ukoliko se naprimjer uzme da se automobil prepoznaje na osnovu njegovog oblika, dimenzija, te određenih elemenata (npr. kotači, retrovizori, gepek, stakla…) kao prijevozno sredstvo. Na taj način svaka značajka pojedinačno doprinosti identifikaciji bez da ovise jedna o drugoj.
- Bayesov: jer je zasnovan na Bayesovom principu.

Jednostavno rečeno, naivni Bayesov klasifikator podrazumjeva da prisustvo pojedine značajke unutar klase nije povezano sa prisustvom bilo koje druge značajke [17]. Naivni Bayes se pretežito veže za industriju vezanu za klasifikaciju teksta. Najčešće je korišten za klasteriranje i klasifikaciju u ovisnosti o vjerojatnosti uvjeta događanja [17].

2.2.1. Bayesov teorem

Promatramo skup svih mogućih događaja. Pretpostavimo da se on može napisati korištenjem međusobno disjunktnih (onih kojima je presjek prazan skup) događaja $H_j, j = 1 \dots n$. Neka je P bilo koji događaj. Događaj H_j također "razbijaju" P tako da je $P = (P \cap H_1) \cup (P \cap H_2) \cup \dots \cup (P \cap H_n)$.[19]. Tako vrijedi za slučaj da su događaji disjunktni:

$$\begin{aligned} \Pr(\mathrm{P}) &= (P \cap H_1) \cup (P \cap H_2) \cup \dots \cup (P \cap H_n) \\ &= \Pr(H_1)\Pr(P|H_1) + \Pr(H_2)\Pr(P|H_2) + \dots + \Pr(H_n)\Pr(P|H_n) \\ &= \sum_{j=1}^{n} \Pr(H_j)\Pr(P|H_j) \end{aligned} \tag{1}$$

Gdje su:

$\boldsymbol{H_j}$ – hipoteza,

$\mathbf{Pr}(\boldsymbol{H_j})$ – Vjerojatnost H_j (hipoteze),

$\mathbf{Pr}(\boldsymbol{P}|\boldsymbol{H_j})$ – Vjerojatnost događaja P ukoliko je H_j točan.

Općenito, vjerojatnost presjeka dvaju događaja računa se formulom [19]:

$$\Pr(\mathrm{A} \cap \mathrm{B}) = \Pr(A)\Pr(B|A) \tag{2}$$

Ukoliko se na formulu (2) primjeni jednakost $B = H_j$ i $A = P.$, dobije se:

$$\Pr(H_j|P) = \frac{\Pr(H_j)\Pr(P|H_j)}{\sum_{j=1}^{n}\Pr(H_j)\Pr(P|H_j)}, \tag{3}$$

gdje $\Pr(H_j|P)$ predstavlja aposteriornu (naknadnu) vrijednost hipoteze nakon što smo saznali događaj [19].

2.2.2. Bayesov klasifikator

Bayes-ov pristup klasifikaciji novih instanci jeste dodjeljivanje najvjerojatnije vrijednosti v_{MAP} zadanih vrijednosnih atributa $\langle a_1, a_2, \dots a_n \rangle$ koji opisuju instancu.

$$v_{MAP} = argmax_{v_j \in V} \mathrm{P}(v_j | a_1, a_2, \dots a_n) \quad (4)$$

Izraz (4) se može preoblikovati pomoću izraza (3) [18]:

$$v_{MAP} = argmax_{v_j \in V} \frac{P(a_1, a_2, \dots a_n | v_j) P(v_j)}{P(a_1, a_2, \dots a_n)}$$

$$= argmax_{v_j \in V} P(a_1, a_2, \dots a_n | v_j) P(v_j) \quad (5)$$

Već je spomenuto da su atributi koji opisuju instancu međusobno nezavisni i ne utječu na međusobnu vjerojatnost. Stoga se može reći da je vjerojatnost svih atributa $a_1, a_2, \dots a_n$ jednaka produktu vjerojatnosti za pojedinačne atribute: $P(a_1, a_2, \dots a_n | v_j) = \prod P(a_i | v_j)$. Ukoliko se ovaj izraz zamjeni sa jednadžbom (5) dobije se izraz naivnog Bayes-ovog klasifikatora:

$$v_{NB} = argmax_{v_j \in V} P(v_j) \prod \prod P(a_i | v_j) \quad (6)$$

Gdje je v_{NB} označava vrijednost dobivenu pomoću naivnog Bayesovog klasifikatora.

Dakle metoda naivnog Bayesovog učenja određuje vjerojatnosti različitih instanci $P(v_j)$ i $P(a_i | v_j)$, na osnovu njihovih učestalosti u sklopu trening podataka. Skup procjena korespondira naučenoj hipotezi. Hipoteza potom klasificira svaku novu instancu primjenjujući pravilo dato jednadžbom (6) [18].

Za primjer se može uzeti skup podataka koji se koriste kao trening podatci, te opisuju da li pacijent boluje od upale pluća. Podatci su prikazani prema Tablica 1.

Pacijent	Temperatura	Kašalj	Bol u prsima	Upala pluća
P1	Visoka	Suh	Bez boli	DA
P2	Visoka	Bez kašlja	Jaka	NE
P3	Blago povišena	Bez kašlja	Blaga	DA
P4	Blago povišena	Suh	Jaka	DA
P5	Normalna	Sa sekretom	Bez boli	NE
P6	Normalna	Suh	Jaka	DA
P7	Normalna	Suh	Jaka	NE
P8	Visoka	Bez kašlja	Bez boli	DA
P9	Visoka	Suh	Blaga	NE
P10	Blago povišena	Suh	Blaga	DA

Tablica 1 Prisustvo bolesti u ovisnosti o simptomima

Skup podataka je opisan atributnim vrijednostima (temperatura, kašalj, bol u prsima). Pomoću naivnog Bayesovog klasifikatora i trening podataka vrši se predikcija vrijednosti: da li pacijent boluje od upale pluća ili ne. Prema jednadžbi (5) određuje se v_{NB}:

$$argmax_{v_j \in [da,ne]} P(v_j) P(Temperatura = visoka | v_j)\, P(Kašalj = suh, nadražen | v_j)$$
$$P(Bol\ u\ prsima = jaka | v_j) \tag{7}$$

Za izračun v_{NB} potreban je određeni broj vjerojatnosti koje mogu biti procijenjene iz trening podataka. Tako vrijedi:

$$P(Upala\ pluća = da) = \frac{6}{10} = 0{,}6$$

$$P(Upala\ pluća = ne) = \frac{4}{10} = 0{,}4$$

Slično se može odraditi i za uvjetovane vjerojatnosti:

$$P(Temperatura = Visoka | Upala\ pluća = DA) = \frac{3}{10} = 0{,}3$$

$$P(Temperatura = Visoka | Upala\ pluća = NE) = \frac{1}{10} = 0{,}1$$

Koristeći vjerojatnosti procjena za ostale atributne vrijednosti, prema jednadžbi (7) se određuje:

$$P(DA)P(Visoka|DA)P(Suh|DA)P(Bez\ boli|DA) = 0.6 \cdot 0{,}2 \cdot 0{,}4 \cdot 0{,}2 = 0{,}0096$$

$$P(NE)P(Visoka|NE)P(Suh|NE)P(Bez\ boli|NE) = 0.4 \cdot 0{,}2 \cdot 0{,}2 \cdot 0{,}1 = 0{,}0016$$

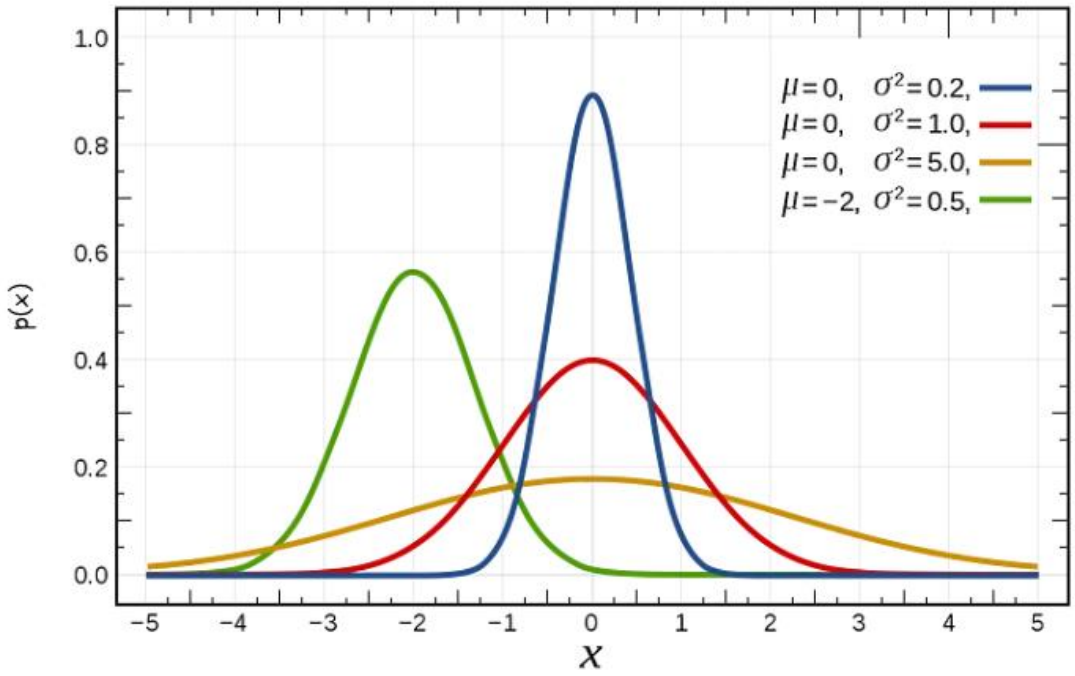
Naivni Bayesov klasifikator dodjeljuje target vrijednost Upala pluća = da novoj instanci na osnovu vjerojatnosti naučenih iz trening podataka. Nadalje, može se izvršiti normalizacija navedenih vrijednosti. Stoga se može izvršiti uvjetovana vjerojatnost: target vrijednost za promatrane atributne vrijednosti. Za ovaj slučaj to iznosi: $\frac{0{,}0096}{0{,}0096+0{,}0016} = 0{,}00857$

2.2.3. Gaussova distribucija

X je normalno distribuiran sa prosjekom μ i varijancom σ^2, označeno kao $N(\mu, \sigma^2)$, ukoliko je funkcija gustoće jednaka:

$$p(x) = \frac{1}{\sqrt{2\pi\sigma}} exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right], -\infty < x < +\infty \tag{8}$$

Mnogi problemi u stvarnom životu barem približno prate pravilo normalne distribucije u obliku zvona. U ovom slučaju $\boldsymbol{\mu}$ predstavlja tipičnu vrijednost i $\boldsymbol{\sigma}$ definira koliko instanci varira oko zadane vrijednosti [15].

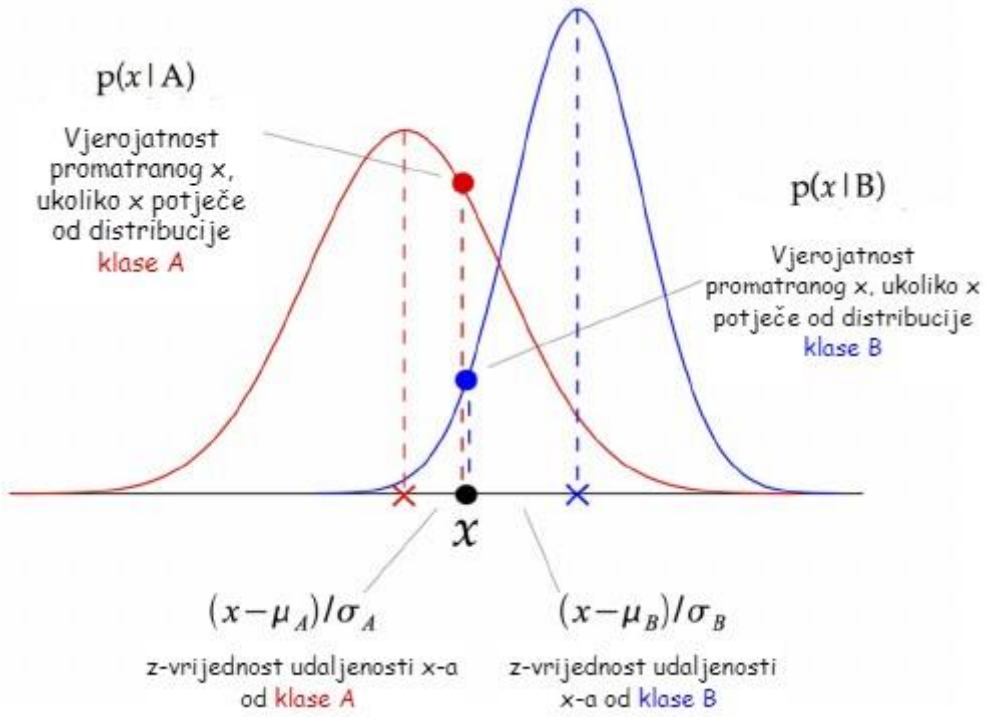


Slika 5 Normalna distribucija u obliku zvona u ovisnosi o μ i σ

U prethodnom poglavlju prikazan je Gaussov pristup klasifikaciji. Korištena varijabla je u tom slučaju bila određena diskretna vrijednost. Međutim što u slučaju da je X kontinuirana varijabla? U tom slučaju može se pretpostaviti da će njena distribucija biti normalna, te da će ona moći slijediti funkciju gustoće prikazanu formulom (8). Na osnovu nje se tada mogu izračunati vjerojatnosti. Ako pretpostavimo da X zaista prati Gauss-ovu distribuciju, tada se mora zamjeniti vjerojatnost gustoće normalne distribucije i nazvati je Gaussov naivni Bayes. Da bi izračunali ovu formulu potreban je prosjek i varijanca od X [29]:

$$p(x|y=c) = \frac{1}{\sqrt{2\pi\sigma}} exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right] \tag{9}$$

Na taj način pomoću Gauss-ove funkcije gustoće možemo načiniti predviđanje mijenjanjem postojećih parametara sa novim unosima vrijednosti i kao rezultat Gaussova funkcija daje procjenu vjerojatnosti novounesene vrijednosti.



Slika 6 Prikaz Gaussovog naivnog klasifikatora

2.3. Klasifikator k najbližih susjeda

Klasifikator k najbližih susjeda predstavlja jednu od osnovnih klasifikacijskih metoda. Algoritam k najbližih susjeda (skraćeno KNN) je jednostavan nadzirani algoritam strojnog učenja koji se koristi za rješavanje klasifikacijskih i regresijskih problema. Jednostavan je za primjenu i razumjevanje, ali mu je velika mana to što postaje značajno sporiji ukoliko se poveća veličina upotrebljenih podataka [17]. Razlog za to jeste upravo princip rada koji podrazumjeva računanje udaljenosti između točaka. Povećanim brojem podataka dolazi do povećanja broja operacija prilikom izvođenja algoritma. Kako se navodi u primjeru, strategija učenja u sklopu klasifikatora najbližih susjeda najsličnija je pamćenju. Kao da se želi zapamtiti kakav bi odgovor trebao biti kada pitanje ima određene karakteristike (zasnovano na okolnostima ili prijašnjim primjerima) nego da se zapravo zna pravi odgovor, te da se razumije pitanje na osnovu značenja i određenih klasifikacijskih pravila [24]. Zbog toga su za klasifikator k najbližih susjeda veoma bitni trening podaci.

Ovaj algoritam pretpostavlja da su sve instance u određenom odnosu na točku u prostoru unutar n-dimenzijskog prostora. Najbliži susjed instance je definiran uvjetima standardne Euklidske udaljenosti. Drugim riječima, odabrana instanca X može biti opisana kao vektor:

$$\langle a_1(x), a_2(x), \dots\ a_n(x)\rangle \tag{10}$$

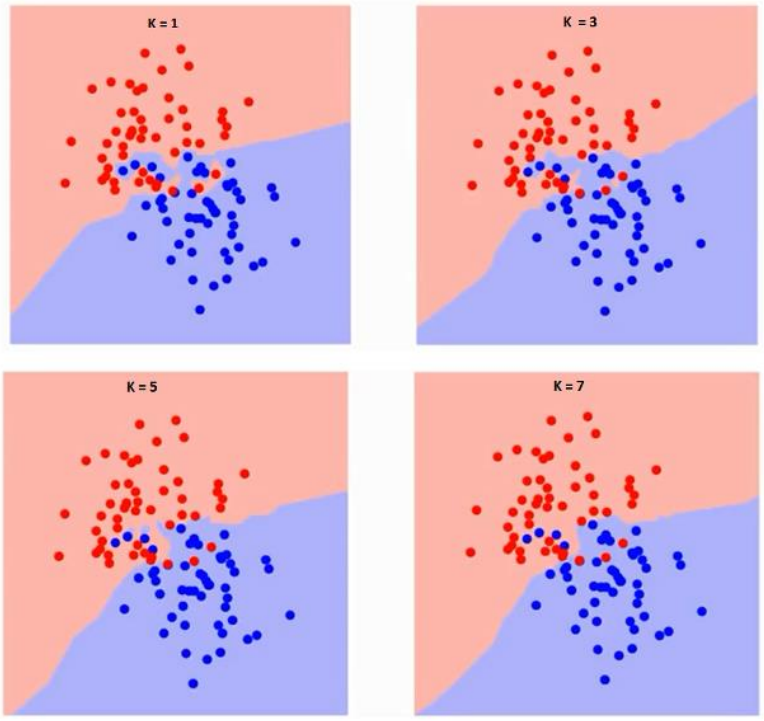
Gdje $a_r(x)$ označava vrijednost r-tog atributa instance X. Tada je udaljenost između dvije instance x_i i x_j definirana kao $d(x_i, x_j)$ gdje je [18]:

$$d(x_i, x_j) = \sqrt{\sum_{r=1}^{n} (a_r(x_i) - a_r(x_j))^2} \tag{11}$$

Euklidska udaljenost je najčešće korištena mjera udaljenosti, i oni su ograničeni kao vektori stvarnih vrijednosti. Formula (11) zapravo mjeri pravocrtnu udaljenost između odabrane točke u prostoru i drugih točaka.

Neizostavan korak pri provođenju klasifikatora k najbližih vrijednosti jeste određivanje faktora K.

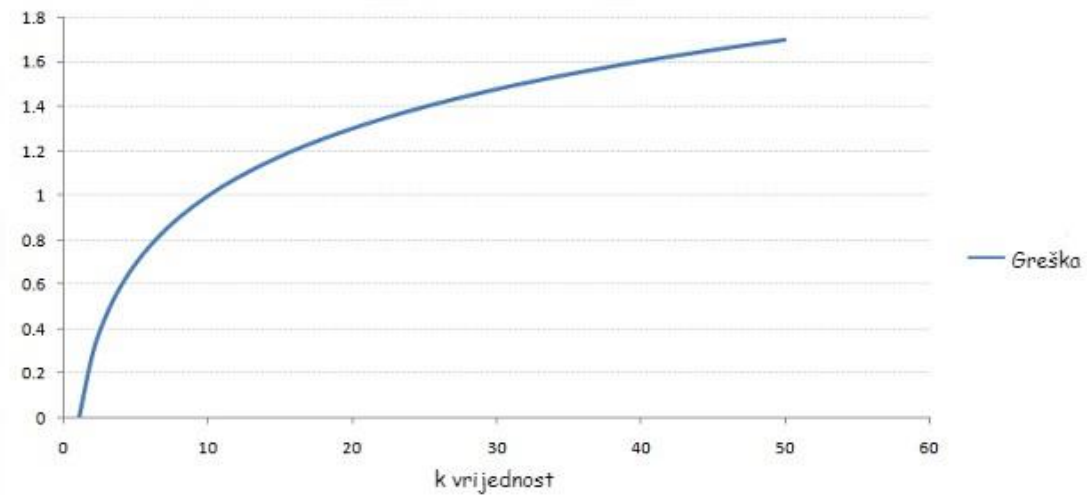
K vrijednost, cijeli broj, je broj susjeda koje algoritam mora uzeti u razmatranje da bi se dobio odgovor [24]. U ovisnosti o veličini i vrsti podataka potrebno je prilagoditi njegovu vrijednost. Što je manji K parametar, to će se algoritam lakše prilagoditi podatcima koji se prezentiraju, što može dovesti do prekomjerne prilagodbe već do optimalnog određivanja granice razdvajanja između klasa. Što je veći K parametar, to se više udaljava od skokova i padova realnih podataka, koji deriviraju blagim krivuljama između klasa unutar podataka [24]. Primjer slikovito prikazuje [Slika 7.].



Slika 7 Izgled granice razdvajanja između klasa u ovisnosti o vrijednosti K faktora

Na slici iznad [Slika 7.] se vidi kako se mijenja granica između dvije različite klase podataka prilikom povećanja vrijednosti K faktora. Kod iznosa faktora K=1 vidi se da granica obuhvata sve plave vrijednosti. Što je vrijednost K veća, tako i granica između klasa postaje „glađa“.

U obzir treba uzeti i grešku koja se također javlja kao posljedica povećanja faktora K. Za slučaj gdje je K=1, stopa greške koja se javlja jednaka je 0 [Slika 8].



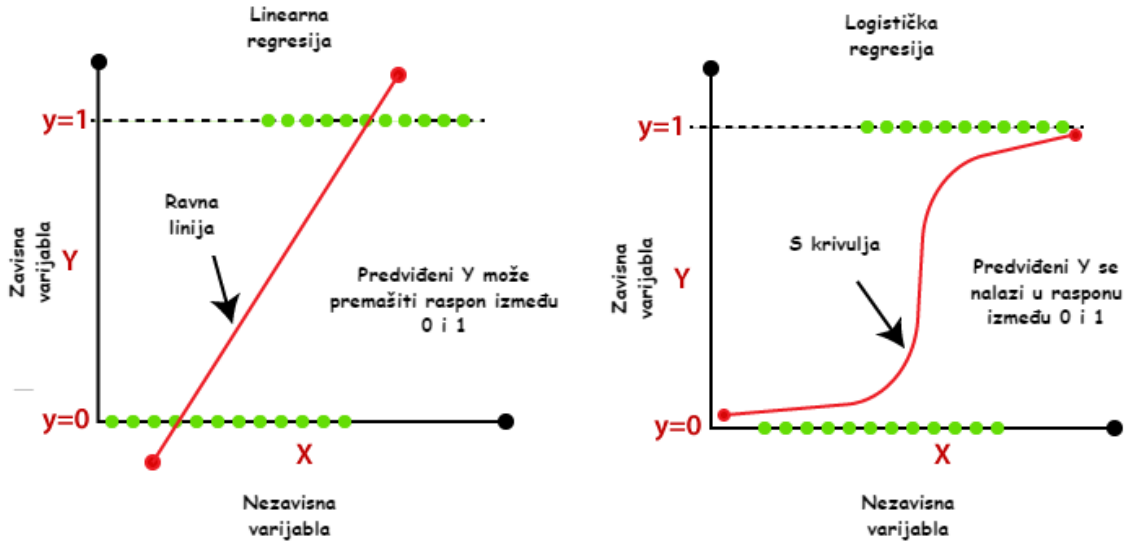
Slika 8 Porast greške u odnosu na vrijednost K

Kao odabir vrijednosti k može se primijeniti jednostavno pravilo: prvo se proba najbliži cijeli broj kvadratnog korijena broja primjera slobodnih kao parametar K unutar KNN. Naprimjer ukoliko imamo 1000 primjera, počne se sa K=31 a onda se smanjuje vrijednost u mrežnom pretraživanju poduprijeto sa unakrsnim vrednovanjem.[24].

2.4. Klasifikator logističke regresije

Logistička regresija se koristi za predviđanje kategorički zavisnih varijabli koristeći zadani skup podataka kao nezavisnih varijabli. Osnovni princip rada algoritma logističke regresije jeste baziran na konceptu predikcijskog modela čija je osnovna svrha klasifikacija uzoraka. Stoga se može reći da logistička regresija spada pod klasifikacijski algoritam.

Jedna od osnovnih primjera logističke regresije jeste linearna regresija.



Slika 9 Prikaz razlika između linearne i logističke regresije

Linearna regresija je standrardni ili osnovni regresijski model u kojem prosjek odziva se procjenjuje na osnovu pojedinačnog prediktora. Osnovni model se lako proširuje i postaje multivarijabilan linearni model, tako da linearna regresija ima više od jednog prediktora. Pošto linearna regresija za predviđanje koristi prosječnu vrijednost odziva, mora postojati određena greška. U slučaju linearne regresije, greška je normalno raspoređena (krivulja u obliku zvona). Odziv je teoretski normalan za zadanu vrijednost prediktora. Zbog toga se linearni model često referira kao **Gaussova regresija** [20]. Drugim riječima pod Gaussovom regresijom se podrazumjeva da odziv kao i postojeće greške imaju normalnu raspodjelu, što

će se podrazumjevati i u ovom slučaju. Tako je varijanca σ^2 konstantna tokom promatranja, pa su i promatranja unutar modela nezavisna [21].

Analogno modelu normalne regresije koji se zasniva na Gaussovoj funkciji distribucije vjerojatnosti (u nastavku *gfdv*), binarni odziv modela je izveden iz Bernoullijeve distribucije, koja je podskup binomne *gfdv* sa binominalnim nazivnikom uzimajući vrijednost 1. Bernoullijev *gfdv* se može izraziti kao:

$$f(y_i; \pi_i) = \pi_i^{y_i}(1 - \pi_i)^{1-y_i} \tag{12}$$

Binarna logistička regresija je izvedena iz kanonskog oblika Bernoullijeve raspodjele. Bernoullijev *gfdv* je dio eksponencijalne porodice raspodjele vjerojatnosti, čije svojstvo omogućava mnogo lakšu procjenu njegovih parametara od tradicionalnih metoda [21].

Postoje 3 vrste logističke regresije: dvočlana, višečlana, ordinalna. Ona koja će se proučavati i čiji se primjer provodi u nastavku pomoću programskog jezika Python jeste dvočlana. Ukoliko je odziv binarnog tipa, odnosno ukoliko on poprima vrijednosti 1 i 0 tj. istina ili laž, tada je riječ o takozvanoj dvočlanoj regresiji. Vrijednosti odziva mogu uveliko varirati u ovisnosti o vrsti istraživanja koje se provodi. Za primjer se može uzeti prolaznost studenata na određenom ispitu. Prediktori bi u tom slučaju bili prolaz: 1 odnosno istina, te pad ; 0 ili laž. Kao odziv dat kao prosječna vrijednost može se uzeti postotak riješenosti ispita vrijednosti od 0 do 100%. Klasifikacija se u tom slučaju vrši po principu da se student s obzirom na osvojeni postotak dodijeli prema prediktorima kao prolaz ili pad. Ukoliko je procentualna vrijednost bliža prediktoru 1, automatski mu se dodjeljuje, te student je zadovoljio prolaznost. Isto vrijedi i za vrijednosti bliže prediktoru 0, u tom slučaju student je pao ispit.

Logistička regresija uključuje niz formula za izračunavanje vjerojatnosti i parametara modela. Neke od njih su:

Logistička funkcija (Sigmoidna funkcija)

$$P(Y = 1) = \frac{1}{1 + e^{-z}} \tag{13}$$

Linearna kombinacija:

$$z = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k \tag{14}$$

Gdje je $P(Y = 1)$ vjerojatnost da je ovisna varijabla Y jednaka 1, a z je linearna kombinacija nezavisnih varijabli X_i s njihovim težinama β_i.

Parametri modela logističke regresije $(\beta_0, \beta_1, \dots, \beta_k)$ procjenjuju se pomoću metoda poput maksimalne izglednosti ili gradijentnog spusta. Ovi parametri određuju oblik krivulje sigmoidne funkcije a stoga i kako se model ponaša [30].

Log-Odds

Log-Odds ili logit transformacija izračunava se kako bi se dobila linearna kombinacija varijabli [31]:

$$\ln\left(\frac{P(Y=1)}{1-P(Y=1)}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k \tag{15}$$

3. ANALIZA ALGORITAMA UNUTAR PYTHON-A

3.1. O Python-u

Python je jedan od najpoznatijih svjetskih *open-source* programskih jezika. Obuhvaća višestruke paradigme kao što su: objektno orjentirano, proceduralno, funkcionalno strukturirano te reflektivno programiranje. Njegov razvoj kasnih 80-ih započinje **Guido van Rossum**, kao nastavak na ABC programski jezik, a prvi put je objavljen 1991 kao verzija Python 0.9.0.[2].

Python posjeduje mnoštvo modula koji omogućavaju njegovu široku primjenu. Upravo ta modularnost je od velike koristi pri dodavanju programabilnog interfejsa već postojećim aplikacijama. Najbolji opis ovog programskog jezika jeste jezgrovitost sa mnoštvom library-a koji omogućavaju široku primjenu. Zaključno sa 14. studenim 2022, Python Package Index (PyPI), službeni repozitorij Python software-a stvoren od strane trećih lica, uključuje preko 415 000 paketa sa širokim rasponom funkcionalnosti uključujući:

- Automatizaciju
- Analizu podataka
- Baze podataka
- Dokumentaciju
- Grafički korisnički interfejs
- Mobilne aplikacije
- **Strojno učenje** itd.

3.2. Metodologija rješavanja

U ovom radu korišena je verzija Pythona 3.11.Prije provedbe i analize algoritama strojnog učenja, zadani skup podataka će se provesti kroz funkcije **čitanja**, **čišćenja**, **vizualizacije**, **skaliranja** te na koncu **implementacije** samih algoritama.

3.2.1. Biblioteke, paketi i moduli

Za uspješnu provedbu zadatka, pomoću naredbe **import as**, uvode se potrebni moduli i biblioteke kako je prikazano na Slika 10. Pregled biblioteka i paketa Postoji više načina uvođenja biblioteka, a u nastavku će se koristiti uvođenje modula dodjeljivanjem skraćenica. To nam pruža korištenje svih funkcija modula, te omogućava da se upisom odgovarajuće skraćenice pozovemo na odgovarajući modul.

```
In [2]: import numpy as np
        import pandas as pd
        import seaborn as sns
        import matplotlib.pyplot as plt
        %matplotlib inline
```

Slika 10. Pregled biblioteka i paketa

3.2.1.1. Pandas

Biblioteka Pandas stavlja fokus na rad sa podacima, i obuhvaća njihovu analizu i obradu. U inžinjerskoj praksi, za uspješno rješavanje problema potrebno je provesti valjanu analizu podataka koji su pohranjeni u bazama. Baze podataka u sklopu različitih modula mogu se pozivati pomoću različitih programa, a jedan od često korištenih je Microsoft Excel. Informacije unutar ovih modula se pozivaju pomoću Microsoft Excel file-a koji se nalazi u istom folderu kao i zadani modul. Python programski jezik koristi biblioteku Pandas za obradu *spread-sheet* formata datoteke kao što su Excel-ovi stoga i da čita podatke unutar Excel datoteke [3].

Pandas je biblioteka za rad s podatkovnim okvirima (*pandas dataframe*), strukturama za tablično skladištenje svih vrsta podataka, ne samo numeričkih.[4]

3.2.1.2. NumPy

Numpy je biblioteka za rad s matricama i algebarskim operacijama [4]. Predstavlja osnovni paket korišten za naučno računanje unutar Pythona. Podrazumjeva operacije sa objektima višedimenzijskih nizova, nizom rutina za brze operacije nad nizovima, gdje se podrazumjevaju matematičke, logičke operacije, manipulaciju oblicima, sortiranja, selektiranja, I/O, diskretnu Fourierovu transformaciju, osnove linearne algebre, osnovne statističke operacije, nasumične simulacije i sl.

3.2.1.3. Seaborn

Uvodi se kao vrsta nadogradnje na biblioteku **matplotlib,** koju koristi za crtanje plot grafova i usko se veže sa **pandas** strukturama podataka.

3.2.1.4. Matplotlib.pyplot

Matplotlib.pyplot je popularna biblioteka u Pythonu koja se koristi za stvaranje grafova i vizualizaciju podataka. Ova bibilioteka pruža bogat skup alatki za crtanje različitih vrsta grafova, dijagrama i sličnih vizualizacija. Pomoću **matplotlib.pyplot** mogu se jednostavno prikazati podatci, istražiti obrasci i prenijeti svoje rezultate drugima na jasan i privlačan način. Služi kao svojevrsni interfejs funkciji matplotlib. Podsjeća na MATLAB-ov način prikazivanja grafova. Dodatak %matplotlib inline služi za prikaz pokrenutog koda u sklopu Jupyter Notebook-a. Ova komanda čini da se u sklopu Jupyter Notebooka usporedno pojavi više grafičkih prikaza u sklopu bilješke.

3.2.2. Učitavanje i analiza zadanog skupa podataka

Sljedeći korak nakon implementacije svih potrebnih modula i biblioteka jeste učitanje zadanog skupa podataka. Podatci potrebni za analizu su zadani u formi **csv** datoteke. Csv datoteka jeste vrsta datoteke koja omogućava da podatci budu spremljeni u tabularnom formatu. Csv predstavlja akronim koji znači „*comma seperated values*“ i predstavlja pojedinačne podatke odvojene zarezima. Mogu se učitavati u sklopu bilo kojeg programa pogodnog za tabelarni prikaz a najčešće se koristi Google spreadsheets ili Microsoft excel [5].

Csv datoteku učitavamo pomoću funkcije **pd.read_csv** i to prema slici 12 [Slika 12]. Prilikom učitavanja moguće je da će se pojaviti error prema slici 11 [Slika 11], pa je u tom slučaju potrebno provjeriti ispravnost unesene putanje te dodati slovo r ispred znaka navoda [Slika 12]

```
SyntaxError: (unicode error) 'unicodeescape' codec can't decode bytes in position 2-3: truncated \UXXXXXXXX escape
```

Slika 11 Greška u sintaksi

```
In [3]: df = pd.read_csv(r'C:\Users\Omar\Desktop\Classified Data.csv')
        df.head()
```

Slika 12 Učitavanje zadanog skupa podataka

Funckija **df.head()** služi za prikaz 5 prvih redova unutar skupa podataka. Analogno df.head() funkciji, **df.tail()** vraća vrijednost posljednjih 5 redova skupa podataka Unosom brojčane vrijednosti unutar zagrade prikazuje zadani broj redova.

Učitavanjem funkcije df.head() povratno se dobiva:

Out[4]:

	Unnamed: 0	WTT	PTI	EQW	SBI	LQE	QWG	FDJ	PJF	HQE	NXJ	TARGET CLASS
0	0	0.913917	1.162073	0.567946	0.755464	0.780862	0.352608	0.759697	0.643798	0.879422	1.231409	1
1	1	0.635632	1.003722	0.535342	0.825645	0.924109	0.648450	0.675334	1.013546	0.621552	1.492702	0
2	2	0.721360	1.201493	0.921990	0.855595	1.526629	0.720781	1.626351	1.154483	0.957877	1.285597	0
3	3	1.234204	1.386726	0.653046	0.825624	1.142504	0.875128	1.409708	1.380003	1.522692	1.153093	1
4	4	1.279491	0.949750	0.627280	0.668976	1.232537	0.703727	1.115596	0.646691	1.463812	1.419167	1

Slika 13 Prikaz prvih 5 redova zadanog skupa podataka

3.2.3. Čitanje podataka

Potrebno je prikazati kolika je veličina promatranog skupa podataka, i to se čini pomoću funkcije **df.shape**:

```
In [167…  df.shape
```

Slika 14 Funkcija prikaza broja redaka i stupaca

Kao izlaznu informaciju dobiva se sljedeće:

```
Out[137]: (1000, 9)
```

Slika 15 Prikaz broja redaka i stupaca

Na osnovu slike 15 [Slika 15.] vidimo da zadani skup podataka predstavlja matricu dimenzija 1000 x 9. Prikaz svih podataka nije od velike koristi te se prikazuje samo jedan njegov dio (kako je opsiano u poglavlju **3.2.2.**), dok se za analizu koristi cijeli skup podataka.

Sljedeći korak jeste funkcija **df.info()** koja vraća sljedeće podatke [Slika 16.]:

- Range index – predstavlja broj unosa
- Data columns – ukupan broj redova
- Nazive kolona
- Non-Null Count – Oznaka za polja koja ne prihvataju nula vrijednosti
- Dtype – Govori o vrsti podataka
- Vrijednosti iskorištene memorije

```
In [5]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 12 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Unnamed: 0     1000 non-null   int64
 1   WTT            1000 non-null   float64
 2   PTI            1000 non-null   float64
 3   EQW            1000 non-null   float64
 4   SBI            1000 non-null   float64
 5   LQE            1000 non-null   float64
 6   QWG            1000 non-null   float64
 7   FDJ            1000 non-null   float64
 8   PJF            1000 non-null   float64
 9   HQE            1000 non-null   float64
 10  NXJ            1000 non-null   float64
 11  TARGET CLASS   1000 non-null   int64
dtypes: float64(10), int64(2)
memory usage: 93.9 KB
```

Slika 16 Informacije o zadanom skupu podataka

Funkcija **df.describe()** prikazuje opis podataka kao što su ukupan broj redaka, prosječnu vrijednost podataka, njihovu standardnu devijaciju, najvišu i najnižu vrijednost, kao i vrijednost za zadani postotak ispod kojeg se nalaze vrijednosti manje od zadane [6].

```
In [6]: df.describe()
```

Out[6]:

	Unnamed: 0	WTT	PTI	EQW	SBI	LQE	QWG	FDJ	PJF	HQE	NXJ	TARGET CLASS
count	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.00000
mean	499.500000	0.949682	1.114303	0.834127	0.682099	1.032336	0.943534	0.963422	1.071960	1.158251	1.362725	0.50000
std	288.819436	0.289635	0.257085	0.291554	0.229645	0.243413	0.256121	0.255118	0.288982	0.293738	0.204225	0.50025
min	0.000000	0.174412	0.441398	0.170924	0.045027	0.315307	0.262389	0.295228	0.299476	0.365157	0.639693	0.00000
25%	249.750000	0.742358	0.942071	0.615451	0.515010	0.870855	0.761064	0.784407	0.866306	0.934340	1.222623	0.00000
50%	499.500000	0.940475	1.118486	0.813264	0.676835	1.035824	0.941502	0.945333	1.065500	1.165556	1.375368	0.50000
75%	749.250000	1.163295	1.307904	1.028340	0.834317	1.198270	1.123060	1.134852	1.283156	1.383173	1.504832	1.00000
max	999.000000	1.721779	1.833757	1.722725	1.634884	1.650050	1.666902	1.713342	1.785420	1.885690	1.893950	1.00000

Slika 17 Funkcija df.describe()

Za provjeru može se iskoristiti **df.isnull()** funkcija koja povratno vraća broj polja sa nul-vrijednostima. Funkcionira po logičkom principu *boolean*, pa 0 predstavljaju upravo ona polja sa vrijednostima 0.

```
In [7]: df.isnull().sum()
```

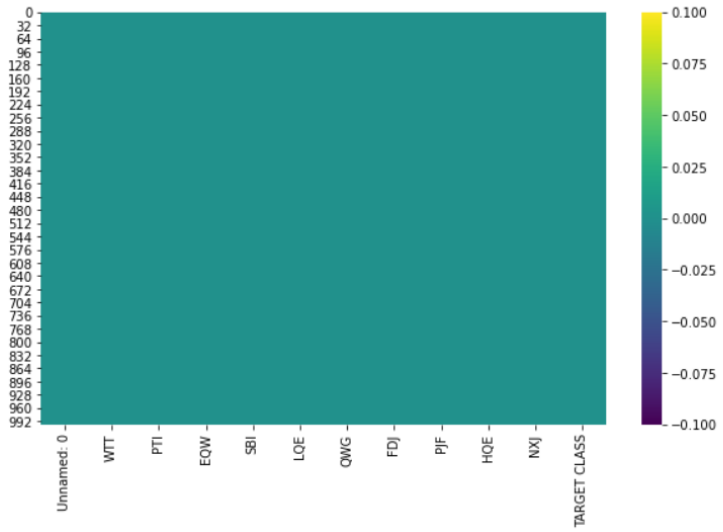
```
Out[7]: Unnamed: 0     0
        WTT            0
        PTI            0
        EQW            0
        SBI            0
        LQE            0
        QWG            0
        FDJ            0
        PJF            0
        HQE            0
        NXJ            0
        TARGET CLASS   0
        dtype: int64
```

Slika 18 Tablica nul-vrijednosti

Dobiveni output potvrđuje da su svi postojeći podatci ispravni.

Grafički prikaz zadanih podataka vrši se preko funkcije **plt.figure()**, u sklopu koje se definira i veličina „heat map“ grafičkog prikaza. U nastavku, **sns.heatmap()** kreira grafikon za vrijednosti dobivene **df.isnull()** funkcijom, gdje se definira i boju i izgled grafikona. Tako s pomoću cmap preciziramo željenu boju prikaza [Slika 19.]

```
In [38]: plt.figure(figsize = (10,6))
         sns.heatmap(df.isnull(),cmap='viridis')
Out[38]: <AxesSubplot:>
```



Slika 19 Grafički prikaz – „heat map“

Sada se i grafički može vidjeti da su vrijednosti dobivene funkcijom df.isnull() jednake boje i da ona iznosi 0 [Slika 19.], što je još jedan dokaz o ispravnosti zadanog skupa podataka.

Prema slici 16 [Slika 16.], koja prikazuje vrstu podataka po pojedinom stupcu, vidi se da vrijednosti integer nalaze u stupcu pod nazivom „TARGET CLASS“. Upravo taj stupac je onaj čije se vrijednosti predviđaju, a one kao konačni ishod mogu dati 0 ili 1 (istina ili laž) [7]. Korištenjem funkcije **df.iloc()** ispisuje se upravo taj stupac na način da se u uglatu zagradu upisuje sljedeće [:-1]. Dobiveni rezultati se vide na slici 20 [Slika 20.].

```
In [29]: # target variable
         df.iloc[:,-1].values
```

```
Out[29]: array([1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0,
                0, 1, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0,
                0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1,
                1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1,
                1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0,
                0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0,
                0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1,
                0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1,
                1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0,
                0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0,
                0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0,
                1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0,
                0, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1,
                0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1,
                1, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0,
                1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1,
                0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1,
                1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0,
                0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0,
                0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 1,
                0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1,
                0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1,
                0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0,
                1, 0, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0,
                1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0,
                0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
                0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1,
                1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1,
                1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0,
                0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1,
                1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0,
                1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1,
                0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 0, 0, 1,
                0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1,
                0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1,
                0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1,
                0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1,
                0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0,
                1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1,
                0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1,
                1, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0,
                0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1,
                1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1,
                0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1,
                1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1,
                0, 1, 0, 1, 1, 1, 0, 1, 1, 1], dtype=int64)
```

Slika 20 Ispis stupca „TARGET CLASS“

3.2.4. Korelacije

Za nastavak analize podataka potrebno je definirati korelacije između varijabli. **Korelacija** predstavlja statističku analizu veza i ovisnosti između varijabli. Korelacije omogućavaju proučavanje kako intenziteta povezanosti tako i usmjerenosti relacija između dva skupa varijabli. Nadalje, da bi algoritmi strojnog učenja radili ispravno, nužno je provjeriti da li su dvije ili više varijabli usko povezane jer zbog toga može doći do grešaka unutar algoritma (pojam poznat kao **multikolinearnost**).

Kao primjer korelacije može se uzeti primjer ocjena završnog ispita te ukupnog prosjeka ocjena. Tablica 2 sadrži ocjene završnog ispita te ukupne prosjeke ocjena 15 učenika. Postavlja se pitanje: kakve obrasce možemo pronaći među ovim podacima? Promatrajući Tablicu 2 da se primjetiti da uglavnom oni studenti koji imaju visok prosjek ocjena često

imaju i visoke ocjene na završnom ispitu. Slično vrijedi i za one učenike sa srednjim i nižim prosjecima. Može se zaključiti da postoji tendencija da studenti postižu slične vrijednosti (više, srednje ili niže vrijednosti) za obe varijable. Učinak studenata u slučaju obje varijable je povezan, drugim riječima, postoji **korelacija** između dvije varijable [10].

Quantitative SAT Scores and Final Examination Scores for 15 Introductory Psychology Students*

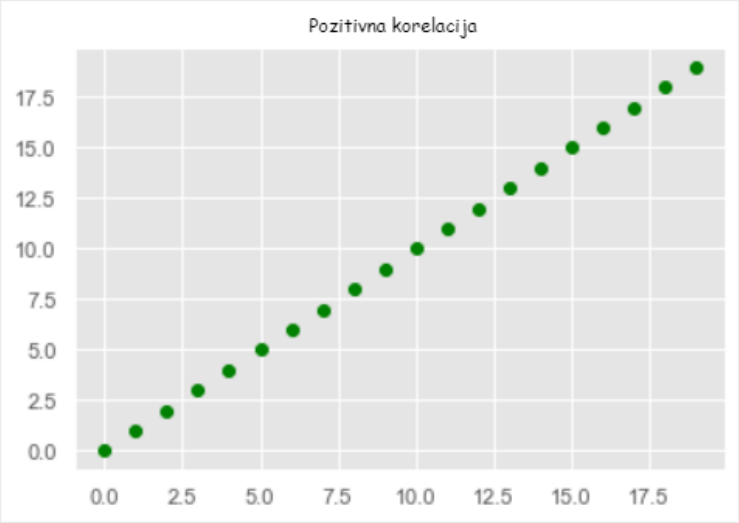
Student	*Quantitative SAT Score (X)*	*Final Examination Score (Y)*
1	595	68
2	520	55
3	715	65
4	405	42
5	680	64
6	490	45
7	565	56
8	580	59
9	615	56
10	435	42
11	440	38
12	515	50
13	380	37
14	510	42
15	565	53
Σ	8,010	772
	$\overline{X} = 534.00$	$\overline{Y} = 51.47$
	$s_x = 96.53$	$s_y = 10.11$

*Note: s_x and s_y are sample standard deviations.

Tablica 2 Prosjek ocjena i ocjene završnog ispita

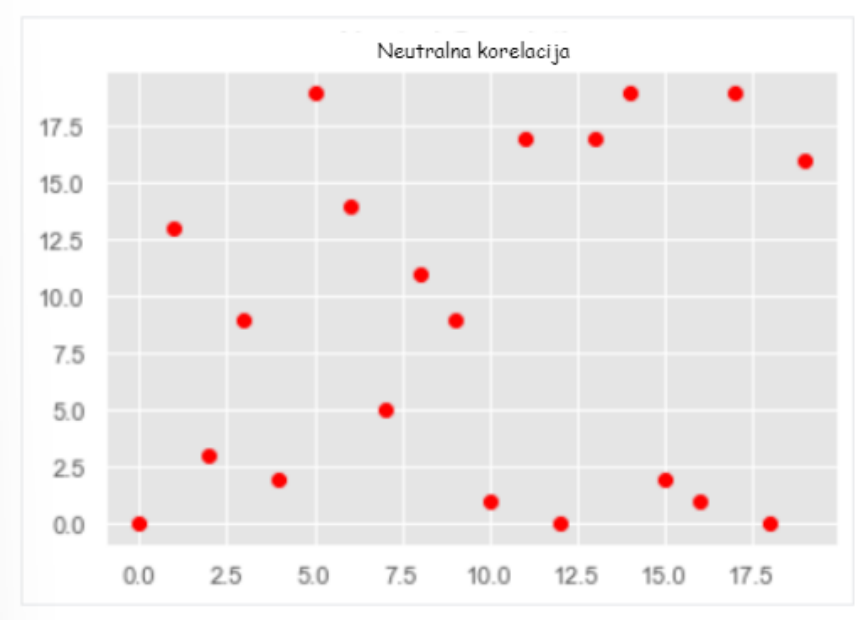
Tri su vrste korelacija:

- Pozitivne korelacije: Dvije varijable su u pozitivnoj korelaciji ukoliko se njihove vrijednosti kreću u istom smjeru. Naprimjer, na slici ispod, kako se vrijednost X povećava, tako se povećava i vrijednost Y konstantnim rastom:



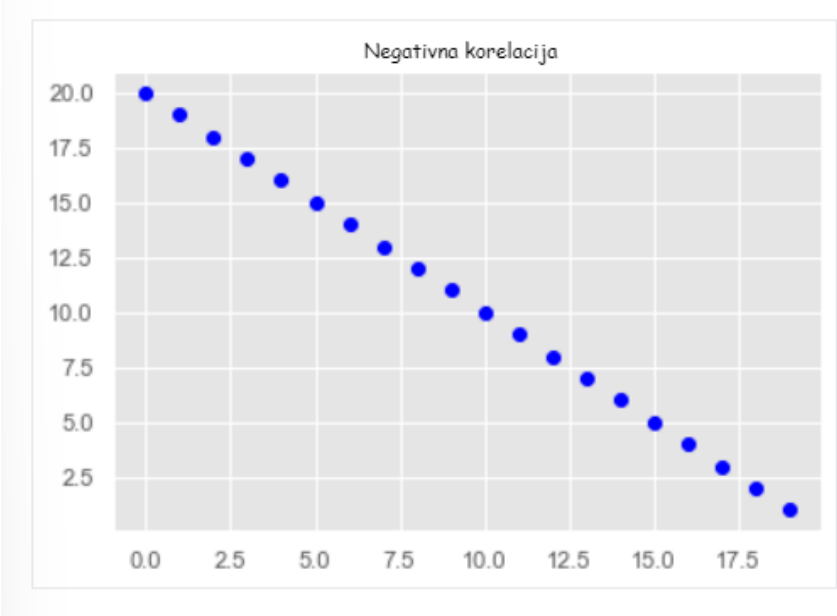
Slika 21 Primjer pozitivne korelacije

- Neutralne korelacije: Ne postoji veza između promjene vrijednosti X i Y. U ovom slučaju vrijednosti su potpuno nasumične i ne prikazuju nikakvu naznaku korelacija, kako je prikazano na slici 22 [Slika 22.]:



Slika 22 Primjer neutralne korelacije

- Negativne korelacije: Varijable X i Y će biti u negativnoj korelaciji kada se njihove vrijednosti mijenjaju u suprotnim smjerovima, pa se tako prema slici 23. [Slika 23.] vrijednost X povećava, dok se vrijednost Y smanjuje konstantnim padom:[8].



Slika 23 Primjer negativne korelacije

3.2.5. Korelacijski faktori

Korelacijski koeficjenti predstavljaju statističku sumaciju koja mjeri intenzitet i smjer u kojem su dvije varijable povezane jedna sa drugom. Jedna od prednosti korelacijskih koeficijenata je ta da oni procjenjuju korelaciju između dvije varijable na standardiziran način, što znači da će vrijednost koeficijenta uvijek biti na skali između 1 i -1. Postoji više korelacijskih koeficijenata, međutim najpoznatiji jeste **Pearsonov** korelacijski koeficijent [8].

Za određivanje vrijednosti korelacijskih koeficijenata zadanog data frame-a, u Pythonu se koristi funkcija **df.corr()**. Unutar zagrade moguće je naznačiti željenu metodu određivanja korelacija (npr. method='pearson', method='kendall'), a ukoliko zagrada ostane prazna, tada se kao zadano koristi Pearsonova metoda [9]. Dobiveni rezultati funkcije prikazani su na slici 24 [Slika 24.].

```
In [30]: df.corr()
```

Out[30]:

	Unnamed: 0	WTT	PTI	EQW	SBI	LQE	QWG	FDJ	PJF	HQE	NXJ	TARGET CLASS
Unnamed: 0	1.000000	0.012981	-0.038424	0.005651	0.058392	0.011102	-0.000945	-0.036872	-0.014241	0.020570	0.007118	0.022593
WTT	0.012981	1.000000	-0.235255	-0.301018	-0.101517	-0.083401	-0.183628	-0.148100	-0.330977	0.324981	-0.073889	0.551394
PTI	-0.038424	-0.235255	1.000000	0.271908	0.001580	0.097322	0.173701	0.095060	0.246387	-0.266242	0.035834	-0.468748
EQW	0.005651	-0.301018	0.271908	1.000000	-0.040291	0.144539	0.182021	0.059533	0.323857	-0.324656	0.126143	-0.598120
SBI	0.058392	-0.101517	0.001580	-0.040291	1.000000	-0.007983	0.022026	-0.007666	0.037767	0.078032	0.145045	0.028874
LQE	0.011102	-0.083401	0.097322	0.144539	-0.007983	1.000000	0.049944	0.035750	0.083734	-0.143929	0.044496	-0.205043
QWG	-0.000945	-0.183628	0.173701	0.182021	0.022026	0.049944	1.000000	0.098062	0.199189	-0.181809	0.036078	-0.327664
FDJ	-0.036872	-0.148100	0.095060	0.059533	-0.007666	0.035750	0.098062	1.000000	0.122888	-0.116969	-0.012923	-0.214885
PJF	-0.014241	-0.330977	0.246387	0.323857	0.037767	0.083734	0.199189	0.122888	1.000000	-0.363736	0.028012	-0.571483
HQE	0.020570	0.324981	-0.266242	-0.324656	0.078032	-0.143929	-0.181809	-0.116969	-0.363736	1.000000	-0.129283	0.643989
NXJ	0.007118	-0.073889	0.035834	0.126143	0.145045	0.044496	0.036078	-0.012923	0.028012	-0.129283	1.000000	-0.111730
TARGET CLASS	0.022593	0.551394	-0.468748	-0.598120	0.028874	-0.205043	-0.327664	-0.214885	-0.571483	0.643989	-0.111730	1.000000

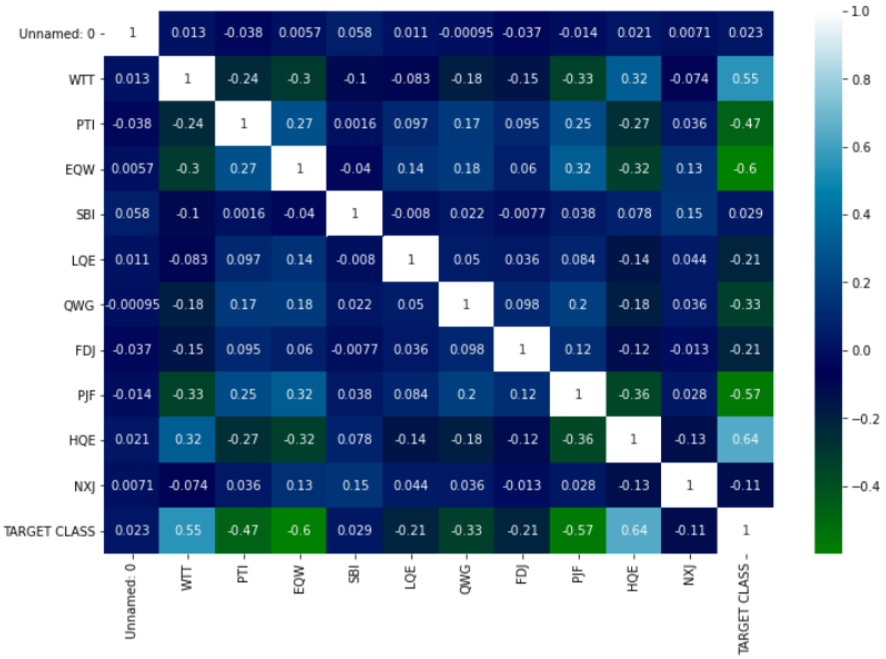
Slika 24 Prikaz korelacijskih faktora

Analizirajući sliku 24 [Slika 24.] da se primjetiti da se na dijagonali matrice nalaze vrijednosti 1. To je upravo zbog toga što se na tim pozicijama podudaraju jednake zadane vrijednosti te kao takve tvore pozitivni korelacijski koeficijent iznosa 1. Vrijednosti mimo glavne dijagonale nam govore koliko je veliko odstupanje skupa podataka od linearne ovisnosti.

Grafički prikaz vrijednosti prikazanih na slici 24 [Slika 24.] dobiva se uz pomoć već korištene funkcije plt.figure() i sns.heatmap():

```
In [35]: plt.figure(figsize=(12,8))
         sns.heatmap(df.corr(),annot = True,cmap='ocean')
```

```
Out[35]: <AxesSubplot:>
```



Slika 25 Grafički prikaz korelacijskih faktora

Interpretacija koeficijenata čija je vrijednost blizu 0 može biti zahtjevna. Postoji više metoda kako interpretirati takve vrijednosti koeficijenata. Vrijednost blizu 0 upućuje na to da su takve varijable statistički nezavisne, što znači da vrijednosti jedne varijable ne pružaju nikakve informacije ili ne utječu na vrijednosti druge varijable. Za ispis minimalnih vrijednosti korelacijskih varijabli koristimo funkciju df.corr().min(), gdje nam dodatak „:min()“ određuje najniže vrijednosti u retku:

```
In [40]: df.corr().min()
```

```
Out[40]: Unnamed: 0     -0.038424
         WTT            -0.330977
         PTI            -0.468748
         EQW            -0.598120
         SBI            -0.101517
         LQE            -0.205043
         QWG            -0.327664
         FDJ            -0.214885
         PJF            -0.571483
         HQE            -0.363736
         NXJ            -0.129283
         TARGET CLASS   -0.598120
         dtype: float64
```

Slika 26 Najniže vrijednosti korelacijskih koeficijenata

Za ispis potpuno podudarnih korelacija, onih dijagonalnih vrijednosti, analogno koristi se funkcija **df.corr().max()**:

```
In [12]: df.corr().max()
Out[12]: WTT            1.0
         PTI            1.0
         EQW            1.0
         SBI            1.0
         LQE            1.0
         QWG            1.0
         FDJ            1.0
         PJF            1.0
         HQE            1.0
         NXJ            1.0
         TARGET CLASS   1.0
         dtype: float64
```

Slika 27 Potpuno podudarne korelacije

3.2.6. Čišćenje podataka

Na osnovu analize dobivenih podataka, potrebno je odrediti kolone za brisanje te provesti takozvano čišćenje podataka. Odabiru se one kolone kod kojih se primjeti odstupanje na način da ne uzimamo niti najniže niti najviše vrijednosti jer one mogu utjecati na preciznost procjene.

```
In [41]: del df['Unnamed: 0']
         del df['NXJ']
         del df['SBI']
         df.head()
```

Out[41]:

	WTT	PTI	EQW	LQE	QWG	FDJ	PJF	HQE	TARGET CLASS
0	0.913917	1.162073	0.567946	0.780862	0.352608	0.759697	0.643798	0.879422	1
1	0.635632	1.003722	0.535342	0.924109	0.648450	0.675334	1.013546	0.621552	0
2	0.721360	1.201493	0.921990	1.526629	0.720781	1.626351	1.154483	0.957877	0
3	1.234204	1.386726	0.653046	1.142504	0.875128	1.409708	1.380003	1.522692	1
4	1.279491	0.949750	0.627280	1.232537	0.703727	1.115596	0.646691	1.463812	1

Slika 28 Brisanje korelacija

Nakon brisanja, ispisuje se trenutna matrica korelacija:

```
In [42]: df.corr()
```

Out[42]:

	WTT	PTI	EQW	LQE	QWG	FDJ	PJF	HQE	TARGET CLASS
WTT	1.000000	-0.235255	-0.301018	-0.083401	-0.183628	-0.148100	-0.330977	0.324981	0.551394
PTI	-0.235255	1.000000	0.271908	0.097322	0.173701	0.095060	0.246387	-0.266242	-0.468748
EQW	-0.301018	0.271908	1.000000	0.144539	0.182021	0.059533	0.323857	-0.324656	-0.598120
LQE	-0.083401	0.097322	0.144539	1.000000	0.049944	0.035750	0.083734	-0.143929	-0.205043
QWG	-0.183628	0.173701	0.182021	0.049944	1.000000	0.098062	0.199189	-0.181809	-0.327664
FDJ	-0.148100	0.095060	0.059533	0.035750	0.098062	1.000000	0.122888	-0.116969	-0.214885
PJF	-0.330977	0.246387	0.323857	0.083734	0.199189	0.122888	1.000000	-0.363736	-0.571483
HQE	0.324981	-0.266242	-0.324656	-0.143929	-0.181809	-0.116969	-0.363736	1.000000	0.643989
TARGET CLASS	0.551394	-0.468748	-0.598120	-0.205043	-0.327664	-0.214885	-0.571483	0.643989	1.000000

Slika 29 Nova matrica korelacija

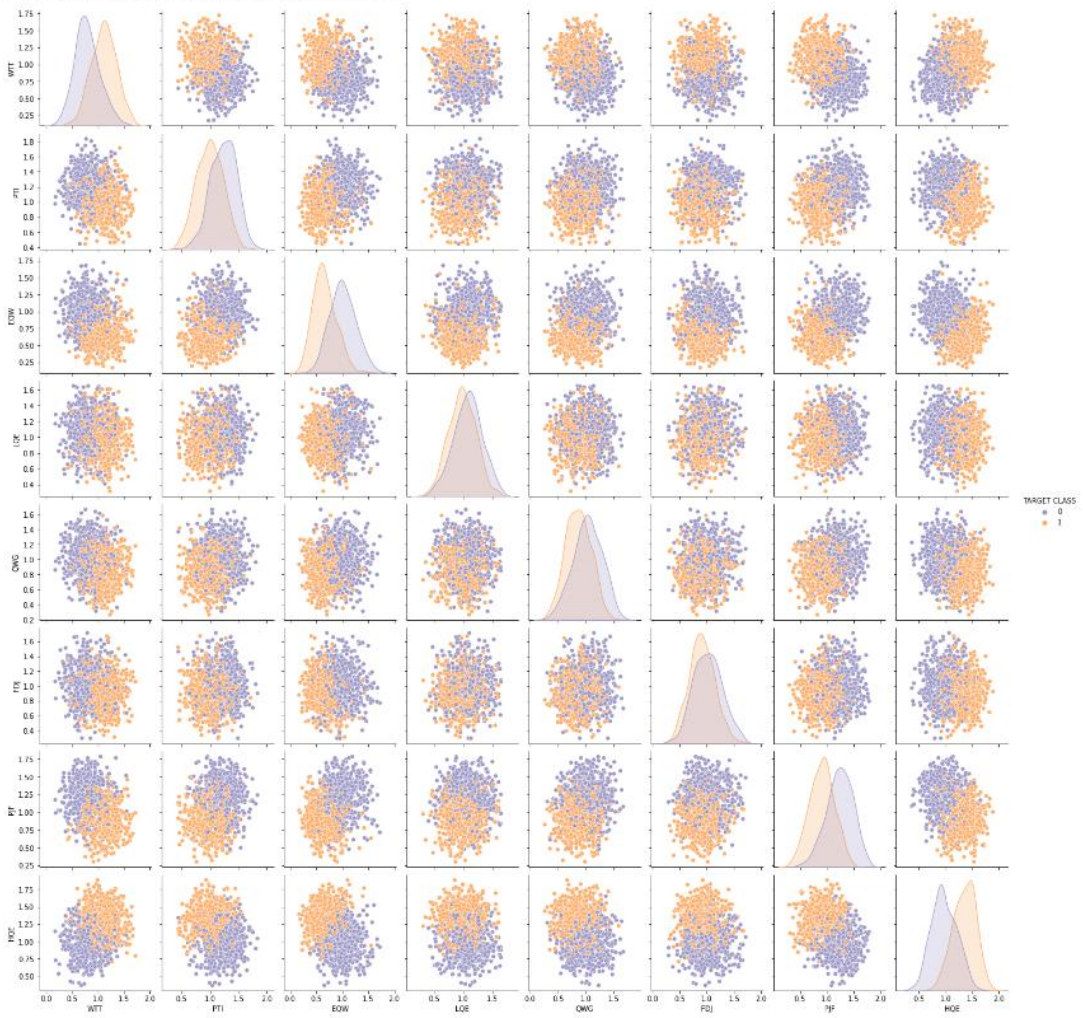
3.2.7. Vizualizacija

Vizualni prikaz podataka postižemo funkcijom **sns.pairplot()**, u sklopu koje se definira koja klasa se želi istaknuti, koji skup podataka želi prikazati, te kakvu vrstu i izgled grafičkog prikaza dobiti.

Na slici 30 [Slika 30.]. nalazi se vizualni prikaz korelacija koje su predhodno dobivene nakon čisćenja kako je prikazano na slici 29. [Slika 29.]. Svaki redak i stupac predstavljaju jednu kolonu zadanog skupa podataka. Na dijagonali se nalaze histogrami koji zajedno sačinjavaju presjek i prikazuju korelacije. U vidu točkica prikazane su ulazne vrijednosti, dok se bojom ističe da li one poprimaju vrijednosti 1 ili 0 u sklopu klase „TARGET CLASS“ (plava predstavlja $h(x)$=0, narančasta $h(x)$=1).

```
In [88]: sns.pairplot(hue='TARGET CLASS',data=df,palette='tab20c_r')
```

```
Out[88]: <seaborn.axisgrid.PairGrid at 0x1b529d68ee0>
```

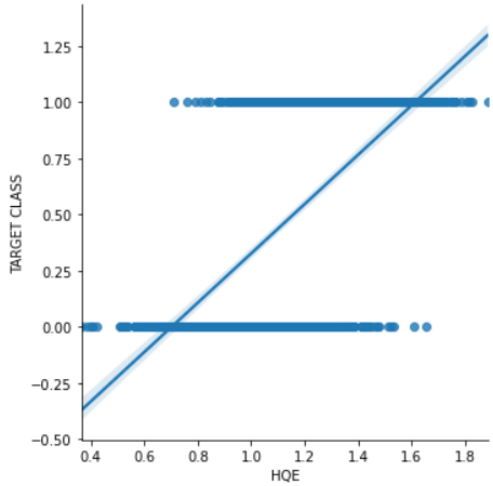


Slika 30 Vizualni prikaz korelacija dobivenih nakon čišćenja

Dodatni prikaz vrši se pomoću funkcije **lmplot()** gdje „lm“ predstavlja skraćenicu od „linear model“. Zadanom funkcijom prikazujemo grafikon prema slici 31 [Slika 31.]. Prema dijagramu se vidi da je korelacija „TARGET CLASS“ i „HQE“ klase linearno pozitivna.

```
In [92]: sns.lmplot(x = 'HQE',y = 'TARGET CLASS',data = df,markers='o')
```

```
Out[92]: <seaborn.axisgrid.FacetGrid at 0x1b52ea23af0>
```

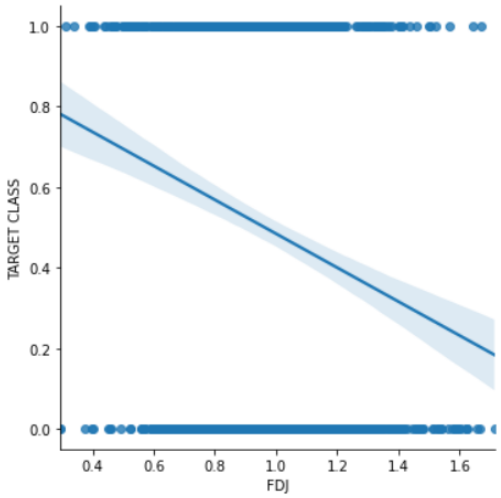


Slika 31 Linearna ovisnost „TARGET CLASS“ i „HQE“ kolona

Za razliku od prethodnog primjera, korealcija „TARGET CLASS“ i „FDJ“ klase poprima linearan pad:

```
In [101…: sns.lmplot(x = 'FDJ',y = 'TARGET CLASS',data = df,markers='o')
```

```
Out[101…: <seaborn.axisgrid.FacetGrid at 0x1b52ef3ee80>
```



Slika 32 Linearna ovisnost „TARGET CLASS“ i „FDJ“ kolona

3.2.8. Skaliranje

Da bi daljnja analiza zadani podataka bila uspješna potrebno je izvršiti proces skaliranja.

Shahriyari je u 2019. pokazao da izvedba normalizacije ima značajan efekt na različite pristupe strojnog učenja [11]. Također proučavanja od strane Ambarwari-ja iz 2020. pokazuju da su tehnike skaliranja kao što su MinMax normalizacija i standardizacija imale značajne efekte na analizu podataka [12]. Normalizacija podataka je reorganizacija podataka unutar baze podataka na način da korisnici mogu iskoristiti za daljnje upite i analize [13]. Skaliranje ima za svrhu ujednačavanje podataka kako bi svi podaci posjedovali jediničnu varijancu [Slika 33.]. Na taj način niti jedna varijanca neće imati veći red veličina od druge te neće dovesti do nesposobnosti modela da uči iz drugih značajki [14].

```
In [46]: from sklearn.preprocessing import StandardScaler

In [47]: sc = StandardScaler()

In [54]: scale = sc.fit(df.drop(['TARGET CLASS'],axis = 1))

In [55]: scaled_features = scale.transform(df.drop('TARGET CLASS',axis = 1))

In [59]: scaled_features

Out[59]: array([[-0.12354188,  0.18590747, -0.91343069, ..., -0.79895135,
                 -1.48236813, -0.9497194 ],
                [-1.08483602, -0.43034845, -1.02531333, ..., -1.12979749,
                 -0.20224031, -1.82805088],
                [-0.78870217,  0.33931821,  0.30151137, ...,  2.59981844,
                  0.28570652, -0.68249379],
                ...,
                [ 0.64177714, -0.51308341, -0.17920486, ..., -2.26133896,
                 -2.36249443, -0.81426092],
                [ 0.46707241, -0.98278576, -1.46519359, ..., -0.42204066,
                 -0.03677699,  0.40602453],
                [-0.38765353, -0.59589427, -1.4313981 , ..., -0.7262528 ,
                 -0.56778932,  0.3369971 ]])
```

Slika 33 Vrijednosti nakon postupka skaliranja

3.3. Modeli predviđanja – algoritmi strojnog učenja

Prema slici 34 [Slika 34] za ulazne podatke se dodjeljuju skalirane značajke i sve se sprema pod X. Pod Y se spremaju izlazni podaci iz data framea iz klase TARGET CLASS. Također uvodi se funkcija train_test_split koja podatke dijeli na dvije skupine: za treniranje i za testiranje. Podjela je nasumična a omjer u kojem će podatci biti podijeljeni jeste 30% za testiranje i 70% za treniranje, što se definira u sklopu funkcije train_test_split kao test_size = 0.30.

```
In [108…
X = scaled_features
y = df['TARGET CLASS']

In [122…
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size = 0.30,random_state = 1)
```

Slika 34 Definiranje skupa podataka za treniranje i testiranje

3.4. Algoritam Gaussov naivni Bayesov klasifikator

Iz **sklearn_naive_bayes** biblioteke uvodio GaussianNB modul prema slici 35 [Slika 35]. „nb“ predstavlja klasifikator naivnog Bayesa, „fit“ predstavlja metodu za trening stojnog učenja za zadani skup podataka.

```
In [123…
from sklearn.naive_bayes import GaussianNB
nb = GaussianNB()
nb.fit(X_train,y_train)
y_pred = nb.predict(X_test)
```

Slika 35 Uvođenje GaussianNB modula

S pomoću y_pred ispisuje se predikcija na novom skupu podataka „X_test“, te se rezultati predikcija pohranjuju u varijablu „y_pred“.

```
In [124…
y_pred

Out[124…
array([1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1,
       1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0,
       1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1,
       0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0,
       0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1,
       1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0,
       0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1,
       0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0,
       0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0,
       1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0,
       1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0,
       1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1], dtype=int64)
```

Slika 36 Ispis rezultata predikcija

Na slici ispod [Slika 37.] prikazuje se tzv. matrica zabune. Matrica zabune je reprezentativna matrica gdje su točni odgovori u redovima a oni predviđeni u stupcima [24].

```
In [125…
from sklearn.metrics import confusion_matrix,classification_report
print(confusion_matrix(y_test,y_pred))

[[152   9]
 [  5 134]]
```

Slika 37 Ispis matrice zabune

Prikaz klasifikacijskog izvještaja prikazan je na slici ispod [Slika 38.].

```
In [126…
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.97      0.94      0.96       161
           1       0.94      0.96      0.95       139

    accuracy                           0.95       300
   macro avg       0.95      0.95      0.95       300
weighted avg       0.95      0.95      0.95       300
```

Slika 38 Klasifikacijski izvještaj

3.5. Algoritam klasifikator k najbližih susjeda

Na isti način kao i u prethodnom poglavlju, uvodi se modul koji izvršava funkcije klasifikatora k najbližih susjeda:

```
In [132…
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier()
knn.fit(X_train,y_train)
y_pred = knn.predict(X_test)
```

Slika 39 Modul klasifikatora k najbližeg susjeda

U nastavku se nastoji pokupiti valjana K vrijednost. Provjera se vrši u rasponu greške od 1 do 40. Za svaku vrijednost K se poziva klasifikator k najbližeg susjeda a zatim se odabire vrijednost K koja ima najmanju stopu greške:

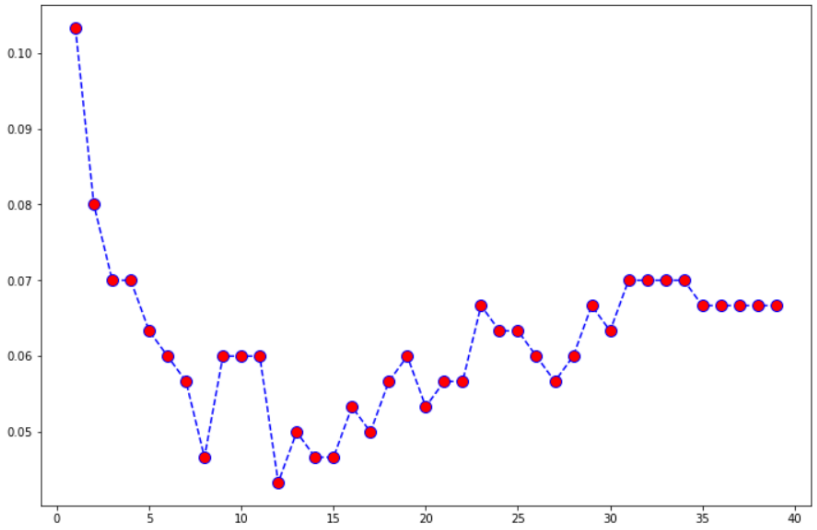
```
In [136…
# Best K value
Error_Rate = []
for k in range(1,40):
    knn = KNeighborsClassifier(n_neighbors=k)
    knn.fit(X_train,y_train)
    y_pred = knn.predict(X_test)
    Error_Rate.append(np.mean(y_pred!=y_test))
```

Slika 40 Odabir K vrijednosti

Već poznatim funkcijama plt.figure() i plt.plot() postiže se grafički prikaz stope greške (ordinata), kao i K vrijednosti (apscisa):

```
In [141…
plt.figure(figsize=(12,8))
plt.plot(range(1,40),Error_Rate,color='blue',linestyle='dashed',marker="o",markerfacecolor='red',markersize=10)

Out[141… [<matplotlib.lines.Line2D at 0x1b53153c610>]
```



Slika 41 Grafički prikaz K vrijednosti

Za vrijednost k = 12 očitava se greška u iznosu oko 0,02 koja je zapravo najniža srednja vrijednost greške. Drugim riječima 12 najbližih susjeda je potrebno za procjenu klase novog ulaza uz prisustvo najmanje greške.

U nastavku za odabranu vrijednost K=12, vrši se procjena preciznosti ovog algoritma:

```
In [183…
knn = KNeighborsClassifier(n_neighbors=12)
knn.fit(X_train,y_train)
y_pred = knn.predict(X_test)
print(classification_report(y_test,y_pred))
```

	precision	recall	f1-score	support
0	0.95	0.95	0.95	129
1	0.94	0.95	0.95	121
accuracy			0.95	250
macro avg	0.95	0.95	0.95	250
weighted avg	0.95	0.95	0.95	250

Slika 42 Rezultati algoritma najbližih susjeda sa odabranom vrijednosti K = 12

3.6. Algoritam klasifikator logističke regresije

Slično kao i što je objašnjeno prethodno (vidi **poglavlje 3.3**), podatke dijelimo na one za treniranje i testiranje. U ovom slučaju dijelit će se u omjeru 1:4 (25% ukupnih podataka koristit će se za testiranje):

```
In [159…
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size = 0.25,random_state = 1)
```

Slika 43 Podjela skupa podataka za treniranje i testiranje

Kolona „**precision**“ predstavlja omjer zadan prema formuli:

$$\frac{tp}{tp + fp} \tag{16}$$

gdje je:

tp-broj pozitivnih vrijednosti koje su predviđene kao pozitivne (ispravno predviđanje)

fp broj negativnih vrijednosti koje su predviđene kao pozitivne (pogrešno predviđanje). Preciznost zapravo govori o sposobnosti klasifikatora da ne označi uzorke kao pozitivne ukoliko su oni negativni.

Kolona „**recall**“ predstavlja omjer zadan prema formuli:

$$\frac{tp}{tp + fn} \tag{17}$$

tp-broj pozitivnih vrijednosti koje su predviđene kao pozitivne (ispravno predviđanje)

fn broj pozitivnih vrijednosti koje su predviđene kao negativne (pogrešno predviđanje)

Ova kolona govori o sposobnosti klasifikatora da pronađe pozitivne vrijednosti.

Kolona „**f1-score**“ predstavlja svojevrsni prosjek preciznosti i opoziva, gdje je najviša vrijednost koju poprima jeste 1 a najniža 0. Može se predstaviti kao umnožak odziva i preciznosti pomnožen sa 2 te podijeljen za njihovim zbrojem:

$$\frac{2 * recall * precision}{recall + precision} \tag{18}$$

Kolona „**support**“ označava broj pojavljivanja u svakoj klasi unutar y_test klase. Uneravnotežen support unutar trening podataka mogu ukazivati na slabu strukturu unutar zabilježenih rezultata u sklopu klasifikatora, i može ukazivati na potrebu novog raspoređivanja uzoraka te njihov rebalans [22].

Pod „macro avg“ podrazumjeva se postupak određivanja makro prosječne preciznosti i to prosjeka svih vrijednosti preciznosti za različite klase. Slično vrijedi i za makro prosječne vrijednosti za sve recall rezultate koji su prikazani na slici 42 [Slika 42].

Također makro F1 izračunava vrijednosti F1 rezultate odvojene po klasama bez korištenja težinskih vrijednosti, dok se prosječne težinske vrijednosti računa zaseban rezultat za svaku klasu, a kada se zbroje zajedno koriste težinsku vrijednost koja ovisi o broju točnih oznaka za svaku klasu:

U konačnici dobivena preciznost algoritma iznosi 96%. Izračunava se prema formuli [23]:

$$\frac{tp + tn}{tp + tn + fp + fn} \tag{19}$$

gdje je:

tp-broj pozitivnih vrijednosti koje su predviđene kao pozitivne (ispravno predviđanje)

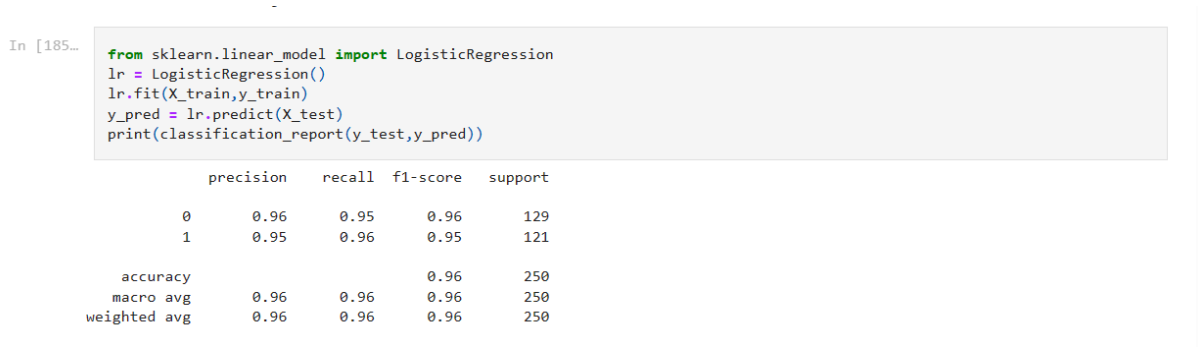
fp broj negativnih vrijednosti koje su predviđene kao pozitivne (pogrešno predviđanje).

tn broj negativnih vrijednosti koje su predviđene kao negativne (ispravno predviđanje)

fn broj pozitivnih vrijednosti koje su predviđene kao negativne (pogrešno predviđanje)

Suma ukupno točnih pozitivnih i točno negativnih vrijednosti podijeljeno sa cijelim brojem primjeraka.

Na slici ispod [Slika 44.] prikazan je klasifikacijski izvještaj zadanog algoritma.



Slika 44 Prikaz klasifikacijskog izvještaja

4. ZAKLJUČAK

U ovom uspoređivanju tri popularna algoritma strojnog učenja - K-najbližih susjeda, Logističke regresije i Naivnog Bayesa – uzete su u obzir njihove karakteristike, prednosti i ograničenja kako bi se bolje razumjela njihova primjena i potencijalni scenariji upotrebe.

Algoritam K-najbližih susjeda je jednostavan algoritam za klasifikaciju i regresiju koji se temelji na sličnosti između podataka. Prednosti koje ga odlikuju su jednostavnost implementacije i prilagodljivost različitim vrstama podataka. KNN ima ograničenja učinkovitosti za velike skupove podataka i zahtijeva odgovarajući izbor parametra K, broja susjeda. Ovaj algoritam je koristan kada se podaci grupiraju u lokalne skupine ili kada nema jasnih linearnih granica između klasa.

Logistička regresija je linearni model za klasifikaciju i može se koristiti za binarnu ili višeklasnu klasifikaciju. Prednost mu je interpretabilnost modela i brza konvergencija prilikom učenja. Logistička regresija pretpostavlja linearnu vezu između značajki i ciljne varijable, što može biti ograničavajuće za složene probleme. Vrlo je pogodan kada želimo razumjeti utjecaj pojedinih značajki na izlaznu klasifikaciju i kada je interpretacija važna.

Naivni Bayes je probabilistički algoritam za klasifikaciju temeljen na Bayesovoj teoremi. Prednosti uključuju brzu konvergenciju, dobre performanse na tekstualnim podacima i jednostavnu implementaciju. Naivni Bayes pretpostavlja nezavisnost između značajki, što može biti nerealno za stvarne podatke.

Svaki od ovih algoritama ima svoje mjesto u strojnom učenju, i odabir između njih ovisi o specifičnim karakteristikama podataka i ciljevima analize. KNN je prikladan za lokalno grupiranje podataka, logistička regresija za interpretaciju i linearno modeliranje, dok je Naivni Bayes često koristan za obradu tekstualnih podataka. Važno je pažljivo razmotriti zahtjeve problema i osobine podataka pri odabiru algoritma kako bismo postigli najbolje rezultate u zadatku strojnog učenja.

LITERATURA

[1] Ethem Alpaydin, Strojno učenje, Mate marketing tehnologija, The MIT Press 2021. pristupljeno 13.07.2023.

[2] https://en.wikipedia.org/wiki/Python_(programming_language), pristupljeno 14.07.2023.

[3] https://www.nsk.hr/ispis-rezultata/ (Hrcak-kljucne rijeci pandas python-ugur comlekcioglu), pristupljeno 16.07.2023.

[4] https://hrcak.srce.hr/pretraga?q=numpy+library (hrcak-kljucne rijeci pandas numpy-M.Lovric) pristupljeno 16.07.2023.

[5] https://www.bigcommerce.com/ecommerce-answers/what-csv-file-and-what-does-it-mean-my-ecommerce-business/ pristupljeno 15.08.2023.

[6] Pandas DataFrame describe() Method (w3schools.com) Python Machine Learning Percentiles (w3schools.com) pristupljeno 23.07.2023.

[7] https://towardsdatascience.com/classification-lets-understand-the-basics-78baa6fbff48 pristupljeno 03.08.2023.

[8] https://www.datacamp.com/tutorial/tutorial-datails-on-correlation pristupljeno 06.07.2023.

[9] https://www.geeksforgeeks.org/python-pandas-dataframe-corr/ pristupljeno 02.09.2023.

[10] Applied statistics for the behavioral sciences, Boston,Mass: Houghton Mifflin, London; Hi Marketing, Dennis E. Hinkle, William Wiersma, Stephen G. Jurs, pristupljeno 10.08.2023.

[11] Shahriyari, L. Effect of normalization methods on the performance of supervised learning algorithms applied to HTSeq-FPKMUQ data sets: 7SK RNA expression as a predictor of survival in patients with colon adenocarcinoma. Briefings Bioinform. 2019, 20, 985–994, pristupljeno 04.07.2023.

[12] Analysis of the Effect of Data Scaling on the Performance of Machine Learning Algorithm for Plant Identificaton, Agus Ambarwari, Qadhli Jafar Adrian, Yeni Herdiyeni, pristupljeno 01.07.2023.

[13] https://www.simplilearn.com/automated-recruiting-in-companies-article pristupljeno 21.07.2023.

[14] Strojno učenje (fer.hr), pristupljeno 23.08.2023.

[15] Introduction to machine learning, third edition, Ethem Alpaydin, The MIT Press Cambridge, Massachusetts London, England, pristupljeno 11.07.2023.

[16] Machine learning:Trends, perspectives and prospects, M. I. Jordan, T. M. Mitchell, pristupljeno 02.07.2023.

[17] Machine Learning Algorithms – A Review, Batta Mahesh, International Journal of Science and Research (IJSR), pristupljeno 22.07.2023.

[18] Machine Learning, Tom M. Mitchell, McGraw-Hill Science/Engineering/Math;March 1.1997, pristupljeno 05.07.2023.

[19] Vjerojatnost i statistika Predavanja, Sanja Singer, Zagreb 2009., pristupljeno 26.07.2023.

[20] Logistic Regression Models, Joseph M. Hilbe, CRC Press Taylor & Francis Group, pristupljeno 07.09.2023.

[21] Logistic regression, Joseph M. Hilbe, Arizona State University, pristupljeno 07.07.2023.

[22] https://medium.com/@kohlishivam5522/understanding-a-classification-report-for-your-machine-learning-model-88815e2ce397, pristupljeno 16.06.2023.

[23] https://medium.com/@kennymiyasato/classification-report-precision-recall-f1-score-accuracy-16a245a437a5, pristupljeno 11.06.2023.

[24] Machine Learning for dummiess, John Paul Mueller, Luca Massaron, John Wiley & Sons, Inc 2016, pristupljeno 06.05.2023.

[25] Semi-Supervised Learning, Olivier Chapelle, Bernhard Scholkopf and Alexander Zien, The MIT Press Cambridge, Massachusetts London, England, pristupljeno 16.07.2023.

[26] https://e-ucenje.fsb.hr/pluginfile.php/166033/mod_resource/content/4/01_metode_u%C4%8Denja_prilago%C4%91eno_prof_jerbic_UI_m%C5%A1_%20-%20sve.pdf, pristupljeno 16.05.2023.

[27] Machine learning: A review of classification and combining techniques, Article in Artificial Intelligence Review, November 2006., pristupljeno 18.06.2023.

[28] The research of regression model in machine learning field, Shen Rong, Zhang Bao-wen, pristupljeno 20.06.2023.

[29] https://www.upgrad.com/blog/gaussian-naive-bayes/, pristupljeno 06.07.2023.

[30] Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). Applied Logistic Regression (3. izdanje). Wiley., pristupljeno 24.05.2023.

[31] Agresti, A. (2015). Foundations of Linear and Generalized Linear Models. Wiley., pristupljeno 01.06.2023.