Usporedba metoda za klasifikaciju podataka koristeći skup podataka o crnim vinima

Šegota, Karla Marija

Undergraduate thesis / Završni rad

2022

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Mechanical Engineering and Naval Architecture / Sveučilište u Zagrebu, Fakultet strojarstva i brodogradnje**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:235:743528

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2025-03-07**

Repository / Repozitorij:

Repository of Faculty of Mechanical Engineering and Naval Architecture University of Zagreb

SVEUČILIŠTE U ZAGREBU

FAKULTET STROJARSTVA I BRODOGRADNJE

ZAVRŠNI RAD

Mentor:

Doc.dr.sc.Tomislav Stipančić, dipl.ing.

Student:

Karla Marija Šegota

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

Zahvaljujem se mentoru doc.sc.dr. Tomislavu Stipančiću pristupačnosti i pomoći tijekom izrade rada.

Karla Marija Šegota

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 / 22 – 6 / 1**	
Ur.broj: **15 - 1703 - 22 -**	

ZAVRŠNI ZADATAK

Student: **Karla Marija Šegota** JMBAG: **0035214198**

Naslov rada na hrvatskom jeziku: **Usporedba metoda za klasifikaciju podataka koristeći skup podataka o crnim vinima**

Naslov rada na engleskom jeziku: **Comparison of methods for data classification using a red wine data set**

Opis zadatka:

U svijetu podataka i velike količine informacija algoritmi umjetne inteligencije postaju sve značajniji. U ovisnosti o vrsti i strukturi podataka, metode strojnog učenja koriste se prilikom predviđanja, odlučivanja, klasifikacije, prepoznavanja, i slično.

Koristeći nekoliko metoda za klasifikaciju podataka primijenjenih na skupu koji označava karakteristike različitih sorti crnog vina potrebno je analizirati i usporediti rezultate odabranih metoda.

U sklopu rada potrebno je:

- pripremiti podatke o vinima da budu prikladni za korištenje od strane korištene metode,
- primijeniti pet metoda za klasifikaciju na zadanom skupu podataka (decision trees, random forest, AdaBoost, Gradient Boost i XGBoost),
- načiniti analizu te usporediti rezultate primjene pojedinog klasifikatora,
- odrediti značajke koje pojedino vino čine poželjnim.

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

Zadatak zadan:

30. 11. 2021.

Zadatak zadao:

Doc. dr. sc. Tomislav Stipančić

Datum predaje rada:

1. rok: 24. 2. 2022.
2. rok (izvanredni): 6. 7. 2022.
3. rok: 22. 9. 2022.

Predviđeni datumi obrane:

1. rok: 28. 2. – 4. 3. 2022.
2. rok (izvanredni): 8. 7. 2022.
3. rok: 26. 9. – 30. 9. 2022.

Predsjednik Povjerenstva:

Prof. dr. sc. Branko Bauer

SADRŽAJ

POPIS SLIKA

SAŽETAK

Ovaj rad se bavi implementacijom klasifikacijskih metoda u područje poboljšanja kvalitete vina. Kako bi se smanjila potreba za ljudskim stručnjacima za kontrolu kvalitete vina, povoljno bi bilo što bolje unaprijed znati koje značajke utječu na njegovu kvalitetu i kako utjecati na njegovo poboljšanje. Ovaj rad koristi klasifikacijske metode kao što su decision trees, random forrest, AdaBoost, Gradient Boost i XGBoost kako bi stvorili vezu između željenih izlaznih varijabli te zavisnih značajki. Također, uspoređuju se i klasifikacijske metode međusobno kako bi imali uvid u to koja od njih nudi najbolje rješenje za ovaj problem.

Ključne riječi: algoritam, klasifikacija, predviđanje, kvaliteta vina

SUMMARY

This paper deals with the implementation of classification methods in the area of wine quality improvement. In order to reduce the need for human experts to control the quality of wine, it would be advantageous to know in advance which features affect its quality and how to influence its improvement. This work uses classification methods such as decision trees, random forrest, AdaBoost, Gradient Boost and XGBoost to create a connection between the desired output variables and the dependent features. Also, classification methods are compared with each other in order to have an insight into which of them offers the best solution for this problem.

Keywords: algorithm, classification, prediction, wine quality

1. UVOD

Mnoge tvrtke koriste certifikate kvalitete kako bi povećale vrijednost svojih proizvoda na tržištu. Provjera kvalitete proizvoda se odvijala na kraju proizvodnje što je uzimalo puno vremena i resursa. U današnje vrijeme razne vrste industrija napreduju i prihvaćaju korištenje novih tehnologija. Te tehnologije su korisne kako bi ubrzale proizvodnju i osigurale što manje pogrešaka. Međutim, još uvijek postoje područja u kojima je potrebna ljudska stručnost.

„Kvalitetu vina je lakše detektirati nego definirati.“, rekla je Maynard Amerine. Razlog je djelomično jer je kvaliteta vina subjektivna i pod velikim utjecajem vanjskih faktora. Većina poznavatelja vina se slaže da ono što smatraju kvalitetnim vinom je subjektivno ono što im se počelo sviđati kroz mnogo kušanja. Međutim, definiranje kvalitete vina u vezi sa njezinim značajkama može biti i djelomično uspješno, bar kako bi se povećala vjerojatnost da će se dobiti kvalitetno vino te bi se time i smanjili troškovi proizvodnje.

Kako bi se i u ovom području implementirale nove tehnologije, traži se veza između lako mjerljivih značajki vina i njegove kvalitete.

Kroz ovaj rad pokušat ćemo pronaći vezu tih značajki sa zadovoljavajućim rezultatom kvalitete vina implementacijom klasifikacijskih algoritama strojnog učenja kao što su: decision trees, radnom forest, AdaBoost, Gradient Boost i XGBoost te usporediti koji od tih algoritama najbolje predviđa kvalitetu vina.

2. STOJNO UČENJE (MACHINE LEARNING)

Strojno učenje je potpodručje računarstva koje se razvilo iz nauke prepoznavanja uzoraka i teorije računalnog učenja u umjetnoj inteligenciji. Istražuje konstrukciju i proučava algoritme koji mogu učiti i na temelju kojih se mogu predviđati podaci. Takvi algoritmi rade na principu izgradnje modela iz ulaznih primjera kako bi napravila predviđanja ili odluke na temelju podataka, umjesto da striktno slijede statičke programske upute. Strojno učenje je usko povezano, a i često se preklapa s računalnom statistikom, disciplina koja je također specijalizirana za predviđanje. Snažno je vezana za matematičku optimizaciju iz koje vuče metode i teoriju. Koristi se u nizu računalnih zadataka gdje je projektiranje i programiranje eksplicitnih algoritama neizvedivo. Na primjer, filtriranje neželjene pošte, optičko prepoznavanje znakova (OCR), tražilice i računalni vid. Ponekad je povezano s rudarenjem podataka (computer minig) iako se ono više fokusira na istraživačkoj analizi podataka. Kada se koriste u industijskom kontestu, metode strojnog učenja mogu se nazvati prediktivnom analitikom ili prediktivnim modeliranjem.

Zadaci strojnog učenja obično se klasificiraju u tri šire kategorije, ovisno o prirodi „signala“ ili „povratne informacije“ učenja dostupnog sustavu učenja. To su učenje pod nadzorom (supervised learning), učenje bez nadzora (unsupervised learning) i učenje s pojačanjem (reinforcement learning). U učenju pod nadzorom računalu su dani ulazni primjeri i njiihovi željeni izlazi te je cilj naučiti opće pravilo koje ulaze preslikava u izlaze. U učenju bez nadzora ne daju mu se nikakve oznake u algoritmu učenja, ostavljajući ga da sam nađe stukturu u ulazu. U učenju s pojačanjem računalni program je u interakciji s dinamičnim okruženjem u kojem mora ostvariti određeni cilj bez da mu učitelj izričito kaže je li se približio cilju ili ne.

Postoji još jedna kategorizacija strojnog učenja kada se uzme u obzir željeni izlaz stojno učenog sustava: klasifikacija, regresija, grupiranje (clustering), procjena gustoće i smanjenje dimenzionalnosti. U klasifikaciji, ulazi se dijele na dva ili više razreda, a računalno mora proizvesti model koji dodjeljuje nevidljive ulaze jednom ili više tih razreda. Ovo se uobičajeno rješava pod nadzorom. U regresiji, koja je također nadzirani problem, izlazi su kontinuiranim, a ne diskretni. U grupiranju, skup ulaza treba podijeliti u skupine. Za razliku od klasifikacije, grupe nisu unaprijed poznate pa je ovo uobičajeno nenadzirani zadatak. Procjena gustoće pronalazi distribuciju izlaza u nekom prostoru. Smanjenje dimenzionalnosti pojednostavljuje ulaze mapirajući ih u niždimenzionalni prostor.

2.1. Učenje pod nadzorom (supervised learning)

Učenje pod nadzorom je zadatak strojnog učenja da iz označenih podataka za obuku zaključi funkciju. Podaci o obuci (training data) sastoje se od seta primjera za obuku. U učenju pod nadzorom, svaki primjer je par koji se sastoji od ulaznog objekta (obično vektor) i željenog izlaza koji se također naziva i nadzorni signal. Algoritam učenja pod nadzorom analizira podatke za obuku i proizvodi izvedenu funkciju koja se može koristiti za mapiranje novih primjera. Optimalni scenarij će omogućiti algoritmu da ispravno označi klase za neviđene instance. To zahtjeva da algoritam učenja iz podataka za obuku generalizira na neviđene situacije na „razuman“ način. Paralelan zadatak u ljudskom i životinjskoj psihologiji se često naziva konceptualno učenje.

Kako bi riješio dati problem naziranog učenja, trebaju se izvršiti slijedeći koraci:

1. Odrediti vrstu primjera za obuku. Prije nego šta se radi išta drugo, korisnik treba odlučiti koja vrsta podataka će se koristiti u setu za obuku.

2. Prikupiti set za obuku. Set za obuku treba predstavljati ulogu funkcije u stvarnom svijetu. Zato se od ljudskih stručnjaka ili iz mjerenja prikupljaju skup ulaznih objekata i odgovarajući izlazi.

3. Odrediti prikaz značajki ulaza naučene funkcije. Točnost naučene funkcije snažno ovisi o tome kako je predstavljen uzlazni objekt. Obično se ulazni objekt transformira u vektor obilježja (feature vector) koji sadrži niz obilježja koji opisuju predmet. Broj značajki ne smije biti prevelik zbog dimenzionalnosti, ali treba sadržavati dovoljno informacija za točno predviđanje rezultata.

4. Odrediti strukturu naučene funkcije i odgovarajući algoritam učenja.

5. Dovršiti dizajn. Provesti algoritam učenja na prikupljenom setu za obuku. Neki algoritmi nadziranog učenja zahtijevaju od korisnika da odredi određene kontrolne parametre. Ti parametri se mogu prilagoditi optimiziranjem izvedbe na podskupu skupa za obuku ili unakrsnom provjerom valjanosti.

6. Ocijeniti točnost naučene funkcije. Nakon podešavanja parametara i učenja, izvedba rezultirajuće bi se trebala mjeriti na testnom setu koji je odvojen od skupa za obuku.

Ne postoji jedinstveni algoritam učenja koji najbolje funkcionira na svim problemima učenja pod nadzorom.

2.1.1. Klasifikacija

Klasifikacija se definira kao proces prepoznavanja, razumijevanja i grupiranja objekata i ideja u unaprijed postavljene kategorije tzv. "pod-populacije". Uz pomoć ovih unaprijed kategoriziranih skupova podataka za obuku, klasifikacija u programima strojnog učenja koristi širok raspon algoritama za klasificiranje budućih skupova podataka u odgovarajuće i relevantne kategorije.

Klasifikacijski algoritmi koji se koriste u strojnom učenju koriste ulazne podatke za obuku u svrhu predviđanja vjerojatnosti da će podaci koji slijede pripasti u jednu od unaprijed određenih kategorija. Jedna od najčešćih primjena klasifikacije je filtriranje e-pošte u "spam" ili "non-spam", kako ga koriste današnji vrhunski pružatelji usluga e-pošte.

Ukratko, klasifikacija je oblik "prepoznavanja uzoraka". Ovdje algoritmi klasifikacije primijenjeni na podatke o obuci pronalaze isti obrazac (slične nizove brojeva, riječi ili osjećaje i slično) u budućim skupovima podataka.

U ovom radu će se usporediti 5 klasifikacijskih algoritama: stabla odluke (decision trees), random forest, AdaBoost, Gradient Boost i XGBoost.

2.1.1.1. Decision trees

Stablo odlučivanja je često korištena neparametarska tehnika modeliranja strojnog učenja za probleme regresije i klasifikacije. Kako bi pronašlo rješenje, stablo odlučivanja donosi sekvencijalne hijerarhijske odluke o izlaznoj varijabli na temelju podataka za predviđanje. Hijerarhijski znači da je model definiran nizom pitanja koja vode do klase ili vrijednosti primijenjen na bilo koje opažanje. Jednom kada je postavljen, model se ponaša poput protokola u nizu uvjeta „ako se ovo dogodi onda se ovo dogodi“ koji proizvodi određeni rezultat iz ulaznih podataka. Neparametarska metoda znači da ne postoje temeljne pretpostavke o distribuciji pogrešaka ili podataka. To u osnovi znači da je model konstruiran na temelju promatranih podataka.

Modeli stabla odlučivanja gdje ciljna varijabla koristi diskretni skup vrijednosti klasificiraju se kao stabla klasifikacije. U tim stablima svaki čvor ili list, predstavlja oznake klase dok grane predstavljaju veze značajki koje vode do oznaka klase. Stablo odlučivanja gdje ciljna varijabla ima kontinuiranu vrijednost, obično brojeve, naziva se regresijsko stablo. Ova dva tipa se obično zajednički nazivaju CART (Classification and Regression Tree).

Svaki CART model je slučaj usmjerenog acikličkog grafa. Ovi grafovi imaju čvorove koji predstavljaju točke odlučivanja o glavnoj varijabli s obzirom na prediktor, a linije su veze između čvorova.

Kako je cilj stabla odlučivanja napraviti optimalan izbor na kraju svakog čvora, potreban mu je algoritam koji je sposoban učiniti upravo to. Taj algoritam je poznat kao Huntov algoritam, koji na svakom koraku donosi najoptimalnije odluke i veće probleme dijeli na manja i rješava ih na isti način. Odluka o podjeli na svakom čvoru donosi se prema metrici koja se zove čistoća. Čvor je 100% nečist kada je čvor ravnomjerno podijeljen 50/50 i 100% čist kada svi njegovi podaci pripadaju jednoj klasi.

Kako bismo optimizirali naš model, moramo postići maksimalnu čistoću i izbjeći nečistoće. Da bismo to izmjerili koristimo Ginijevu nečistoću (Gini impurity), koja mjeri koliko je često nasumično odabrani element netočno označen ako je nasumično označen prema distribuciji. Izračunava se davanjem vjerojatnosti, π, stavke s oznakom, i, pomnožene s vjerojatnošću $(1–\pi)$ pogreške koja kategorizira vrijeme. Naš cilj je da dosegne vrijednost 0 gdje će biti minimalno nečist i maksimalno čist spadajući u jednu kategoriju.

Druga metrika koja se koristi je dobitak informacija, koji se koristi za odlučivanje koje će se obilježje podijeliti na svakom koraku u stablu. Ovo se izračunava na sljedeći način:

Dobitak informacija = Entropija (roditelj) - Ponderirani zbroj entropije (Djeca).

Iako je ovo sjajan model, on predstavlja veliki problem jer rezultira modelom koji se zaustavlja samo kada su sve informacije u jednoj klasi ili atributu. Odstupanja za ovaj model su ogromna i sigurno će dovesti do pretjeranog uklapanja. "Stabla odlučivanja mogu stvoriti presložena stabla koja se ne mogu dobro generalizirati iz podataka o obuci." Kako bi se borili protiv toga možemo postaviti maksimalnu dubinu stabla odlučivanja (tj. koliko će čvorova duboko ići i/ili je alternativa navesti minimalni broj podatkovnih točaka potrebnih za podjelu svake odluke.

Međutim, stablo odlučivanja ima i nedostataka. Lokalno je optimizirano korištenjem „greedy“ algoritma pri čemu ne možemo jamčiti povratak na globalno optimalno stablo odlučivanja. To je nevjerojatno pristran model osim ako skup podataka nije uravnotežen prije stavljanja u stablo. Iako postoje nedostaci, stabla odlučivanja imaju mnoge prednosti. Nevjerojatno su jednostavni za razumijevanje zbog svojeg vizualnog prikaza, zahtijevaju vrlo malo podataka, mogu rukovati kvalitativnim i kvantitativnim podacima, mogu se provjeriti korištenjem statističkih skupova, mogu rukovati velikim količinama podataka i prilično su računski jeftini.

2.1.1.2. Random forest

Nasumične šume vrsta su algoritma strojnog učenja koji se koristi za klasifikaciju i regresiju. Model klasifikatora uzima ulazne podatke i dodjeljuje ih jednoj od nekoliko kategorija. Algoritam slučajne šume radi tako da stvara više stabala odlučivanja, od kojih se svako temelji na slučajnom podskupu podataka. Stabla odlučivanja vrsta su algoritma koji predviđaju gledajući ulazne podatke i određujući kojoj kategoriji pripadaju. Nasumične šume idu korak dalje stvarajući više stabala odlučivanja i zatim uzima srednju vrijednost njihovih rezultata. To pomaže u smanjenju mogućnosti pretjeranog prilagođavanja, što je slučaj kada algoritam dobro radi samo na podacima za obuku, a ne na novim podacima. Nasumične šume moćan su alat za strojno učenje i mogu se koristiti za razne zadatke kao što su prepoznavanje lica, otkrivanje prijevara, predviđanje ponašanja potrošača i predviđanja tržišta dionica.

Slučajna šuma može se smatrati skupom nekoliko stabala odlučivanja. Ideja je agregirati ishod predviđanja višestrukih stabala odlučivanja i stvoriti konačni ishod temeljen na mehanizmu izračunavanja prosjeka (glasovanje većine). Pomaže modelu obučenom korištenjem slučajne šume da bolje generalizira s većom populacijom. Osim toga, model postaje manje osjetljiv na prekomjerno opremanje / veliku varijancu.

Nasumične šume točnije su od stabala odlučivanja jer smanjuju varijancu modela i manja je vjerojatnost preopremanja. To radi izračunavanjem prosjeka predviđanja pojedinačnih stabala. Nasumične šume također se mogu nositi s vrijednostima koje nedostaju (missing values) i „ouliers“ bolje od stabala odlučivanja. Lakše ih me podesiti nego stabla odlučivanja. Međutim i ona imaju svojih nedostataka. Potrebno im je više vremena za obuku, ali točnost i fleksibilnost modela čine ih vrijednima dodatnog ulaganja vremena. Također, može ih biti teško interpretirati.

2.1.1.3. AdaBoost

AdaBoost je adaptivni algoritam učenja, koji uči pozivanjem algoritama slabog učenja više puta uzastopno. Prilagođava se stopama pogreške pojedinačnih slabih hipoteza. To je osnova njegovog imena - "ada" što znači prilagodljiv. To je skupni model pojačanja (boosting model). Boosting je tehnika kombiniranja skupa slabih učenika (weak learners) u dobre učenike (strong learner). „Weak learner“ je klasifikator čija je izvedba loša (točnost je malo bolja od slučajnog pogađanja).U suprotnosti tome, „strong learner“ je klasifikator visoke točnosti. Ideja poboljšanja je trenirati slabe učenike uzastopno, pri čemu svaki pokušava ispraviti svog prethodnika. To znači da će algoritam uvijek naučiti nešto što nije uvijek potpuno točno, ali uzastopnim ispravljanjem pogrešaka poboljšava snagu predviđanja. Način na koji se slabi učenici odabiru i kombiniraju u boostingu ovisi o algoritmu. npr. AdaBoost, Gradientboost, XGboost.

AdaBoost algoritam funkcionira tako da prvo prilagodi slabi klasifikator izvornom skupu podataka stvarajući izlaznu hipotezu, a zatim iterativno ponovno odmjerava pogrešno klasificirane podatke kako bi odgovarali sljedećem slabom klasifikatoru. Svakom slabom učeniku dodijeljen je koeficijent tako da je zbroj pogrešaka uvježbavanja rezultirajućeg pojačanog klasifikatora minimiziran. Obuka AdaBoost klasifikatora sastoji se od iterativnog učenja slabih klasifikatora koji su ponderirani na takav način da su povezani s učinkom slabog učenika i dodavanja njih konačnom jakom klasifikatoru. Nakon dodavanja slabog učenika, težine ulaznih podataka se prilagođavaju, poznato kao "ponovno ponderiranje". Ponovno ponderiranje znači da bi ulazni podaci koji su pogrešno klasificirani dobili veću težinu, a ispravno klasificirani podaci izgubili na težini. Stoga se sljedeći slabiji učenici više fokusiraju na podatke koje je prethodni slabi učenik pogrešno klasificirao. Često korišteni slabi učenici u AdaBoost klasifikatoru su stabla odlučivanja s jednim dijeljenjem koja se nazivaju panjevi (stumps). AdaBoost trenira niz klasifikatora (weak learners) s ponovno ponderiranim podacima uzorka, generirajući koeficijente α za pojedinačne klasifikatore na temelju pogrešaka. Ovaj koeficijent određuje koliko ovaj slabi klasifikator doprinosi konačnom kombiniranom klasifikatoru. α promatra pogreške koje čini klasifikator. Kada je pogreška velika, α je manji, tj. manja je važnost klasifikatora s pogreškama u glasanju. Kada je greška mala, α je velika, što znači veću važnost u glasanju.

Predviđanje konačnog kombiniranog klasifikatora vrši se ponderiranim glasovanjem. Kada se testni podaci provuku kroz svaki klasifikator, predviđeni rezultat bio bi maksimum zbroja koeficijenata svakog klasifikatora koji klasificira podatke u određenu klasu.

2.1.1.4. Gradient Boosting

Gradient Boosting definira boosting kao problem numeričke optimizacije gdje je cilj minimizirati funkciju gubitka modela dodavanjem slabih učenika korištenjem spuštanja gradijenta (gradient descent). Gradijentni spust je iterativni optimizacijski algoritam prvog reda za pronalaženje lokalnog minimuma diferencijabilne funkcije. Budući da se povećanje gradijenta temelji na minimiziranju funkcije gubitka, mogu se koristiti različite vrste funkcija gubitka, što rezultira fleksibilnom tehnikom koja se može primijeniti na regresiju, višeklasnu klasifikaciju itd. Intuitivno, povećanje gradijenta je postupno aditivni model koji generira učenike tijekom procesa učenja (tj. stabla se dodaju jedno po jedno, a postojeća stabla u modelu se ne mijenjaju). Doprinos slabog učenika ansamblu temelji se na procesu optimizacije gradijentnog spuštanja. Izračunati doprinos svakog stabla temelji se na minimiziranju ukupne pogreške dobrog učenika. Pojačavanje gradijenta ne mijenja distribuciju uzorka jer slabi učenici vježbaju na preostalim rezidualnim pogreškama jakog učenika (tj. pseudoreziduali). Uvježbavanjem reziduala modela, ovo je alternativni način da se da veća važnost pogrešno klasificiranim opažanjima. Intuitivno se dodaju novi slabi učenici kako bi se koncentrirali na područja u kojima postojeći učenici imaju slabe rezultate. Doprinos svakog slabog učenika (weak learner) konačnom predviđanju temelji se na procesu optimizacije gradijenta kako bi se smanjila ukupna pogreška dobrog učenika (strong learner).

2.1.1.5. XGBoost

XGBoost je kratica za Extreme Gradient Boosting. XGBoost je specifična implementacija metode Gradient Boosting koja daje točnije aproksimacije korištenjem derivacije drugog reda funkcije gubitka, L1 i L2 regulacije i paralelnog računanja. XGBoost je regulariziraniji oblik Gradient Boostinga. Koristi naprednu regularizaciju (L1 & L2), koja poboljšava mogućnosti generalizacije modela. Pruža visoke performanse u usporedbi s Gradient Boosting. Njegova obuka je vrlo brza i može se paralelizirati/distribuirati po klasterima. Izračunava gradijente drugog reda, tj. drugu parcijalnu derivaciju funkcije gubitka, što pruža više informacija o smjeru gradijenata i kako doći do minimuma naše funkcije gubitka. Također obrađuje vrijednosti koje nedostaju u skupu podataka.

3. BAZA PODATAKA

Za ovaj projekt, koristi se Kaggle Red Wine Quality skup podataka kako bi različitim klasifikacijskim modelima predvidjelo je li neko crno vino dobre kvalitete ili ne. Svakom vinu u skupu podataka je dana ocjena kvalitete između 0 i 10.

Kvaliteta vina određena je sa 11 ulaznih varijabli:

1. fiksna kiselost - većina kiselina uključenih u vino / fiksne / nehlapljive

2. hlapljiva kiselost - količina octene kiseline u vinu koja, pri visokim razinama, može dovesti do neugodnog okusa octa

3. limunska kiselina - u malim količina i može dodati „svježinu“ i okus vinima

4. preostali šećer - količina šećera koji je ostao nakon završetka fermentacije, rijetko je pronaći vina s manje od 1g/L

5. kloridi - količina soli u vinu

6. slobodni sumpor dioksid - slobodni sumporov dioksid u ravnoteži između molekularnog sumporovog dioksida i bisulfitnog iona

7. ukupni sumporni dioksid - ukupan oblik slobodnog i vezanog oblika sumporovog dioksida

8. gustoća - gustoća je slična gustoći vode, ali ovisi o udjelu alkohola i šećera

9. pH vrijednost - opisuje koliko je vino kiselo (0) ili lužnato (14)

10. sulfati - aditiv za vina koji može doprinijeti razinama plina sumporovog dioksida

11. alkohol - udio alkohola u vinu

index	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.7	0	1.9	0.076	11	34	0.9978	3.51	0.56	9.4	5
1	7.8	0.88	0	2.6	0.098	25	67	0.9968	3.2	0.68	9.8	5
2	7.8	0.76	0.04	2.3	0.092	15	54	0.997	3.26	0.65	9.8	5
3	11.2	0.28	0.56	1.9	0.075	17	60	0.998	3.16	0.58	9.8	6
4	7.4	0.7	0	1.9	0.076	11	34	0.9978	3.51	0.56	9.4	5
5	7.4	0.66	0	1.8	0.075	13	40	0.9978	3.51	0.56	9.4	5
6	7.9	0.6	0.06	1.6	0.069	15	59	0.9964	3.3	0.46	9.4	5
7	7.3	0.65	0	1.2	0.065	15	21	0.9946	3.39	0.47	10	7
8	7.8	0.58	0.02	2	0.073	9	18	0.9968	3.36	0.57	9.5	7
9	7.5	0.5	0.36	6.1	0.071	17	102	0.9978	3.35	0.8	10.5	5
10	6.7	0.58	0.08	1.8	0.097	15	65	0.9959	3.28	0.54	9.2	5
11	7.5	0.5	0.36	6.1	0.071	17	102	0.9978	3.35	0.8	10.5	5

Slika 1. Baza podataka (1)

1584	6.7	0.32	0.44	2.4	0.061	24	34	0.99484	3.29	0.8	11.6	7
1585	7.2	0.39	0.44	2.6	0.066	22	48	0.99494	3.3	0.84	11.5	6
1586	7.5	0.31	0.41	2.4	0.065	34	60	0.99492	3.34	0.85	11.4	6
1587	5.8	0.61	0.11	1.8	0.066	18	28	0.99483	3.55	0.66	10.9	6
1588	7.2	0.66	0.33	2.5	0.068	34	102	0.99414	3.27	0.78	12.8	6
1589	6.6	0.725	0.2	7.8	0.073	29	79	0.9977	3.29	0.54	9.2	5
1590	6.3	0.55	0.15	1.8	0.077	26	35	0.99314	3.32	0.82	11.6	6
1591	5.4	0.74	0.09	1.7	0.089	16	26	0.99402	3.67	0.56	11.6	6
1592	6.3	0.51	0.13	2.3	0.076	29	40	0.99574	3.42	0.75	11	6
1593	6.8	0.62	0.08	1.9	0.068	28	38	0.99651	3.42	0.82	9.5	6
1594	6.2	0.6	0.08	2	0.09	32	44	0.9949	3.45	0.58	10.5	5
1595	5.9	0.55	0.1	2.2	0.062	39	51	0.99512	3.52	0.76	11.2	6
1596	6.3	0.51	0.13	2.3	0.076	29	40	0.99574	3.42	0.75	11	6
1597	5.9	0.645	0.12	2	0.075	32	44	0.99547	3.57	0.71	10.2	5
1598	6	0.31	0.47	3.6	0.067	18	42	0.99549	3.39	0.66	11	6

Slika 2. Baza podataka (2)

3.1. Učitavanje biblioteka

Početni skup podataka koji će se koristiti za predviđanje je velik i neuređen. S obzirom da će ga trebati prilagoditi kako bi ga mogli koristiti, prvo se u Python skriptu učitavaju potrebne biblioteke i module kao što je prikazano na slici 3. Biblioteke koriste prethodno napisane kodove kako ne bi trebali pisati kod iz nule te tako skratimo vrijeme programiranja. Instaliraju ne na način da se u terminal upiše „pip install [paket]“.

```
In [1]: import numpy as np
   ...: import pandas as pd
   ...: import matplotlib as plt
   ...: import seaborn as sns
   ...: import plotly.express as px
```

Slika 3. Učitavanje biblioteka

3.1.1. Numpy

Numpy je Python biblioteka otvorenog koda koja se upotrebljava u matematičkim i numeričkim proračunima, znanstvenom i inženjerskom programiranju te programiranju podataka. Ime je nastalo kao akronim punog naziva Numerical Python. Ključna je biblioteka za izvođenje matematičkih i statističkih operacija, a izvrsna je i za rad sa višedimenzionalnim nizovima, množenje i preoblikovanje matrica te strojno učenje. Ima učinkovitu memoriju, što znači da može pristupiti golemoj količini podataka puno brže od drugih biblioteka. To je posljedica činjenice da su nizovi pohranjeni na jednom kontinuiranom mjestu u memoriji za razliku od popisa, što rezultira jednostavnim pristupanjem i manipulacijom njima.

3.1.2. Pandas

Pandas je Python biblioteka koja se koristi za rad sa skupovima podataka. Uključuje funkcije za analizu, čišćenje, istraživanje i manipulaciju podacima. Ime dolazi od dva naziva: „Panel Dana“ i „Python Data Analysis“. Omogućuje nam analizu velikih podataka i donošenje zaključaka na temelju statističkih teorija. Može očistiti neuredne skupove podataka i učiniti ih čitljivim i relevantnim. Također nam daje informacije o podacima: korelacije između stupaca, prosječna, maksimalna i minimalna vrijednost. Između ostalog može i izbrisati retke koji nisu relevantni ili sadrže pogrešne vrijednosti, kao što su prazne ili null vrijednosti.

3.1.3. Matplotlib

Matplotlib je biblioteka otvorenog koda koja pruža različite alate za vizualizaciju podataka u Pythonu i komponenta je Numpy biblioteke te stvara 2D dijagrame iz podataka u nizovima. Jedna od najvećih prednosti vizualizacije je da nam omogućuje vizualni pristup ogromnim količinama podataka kako bi smo ih lakše razumjeli. Vizualizira podatke koristeći niz različitih vrsta dijagrama kao što su raspršeni dijagrami, histogrami, grafovi pogrešaka itd.

3.1.4. Seaborn

Seaborn je biblioteka za izradu statističkih grafikona u Pythonu. Izgrađena je na matplotlibu i blisko integrirana s pandas strukturama podataka. Vizualizacija je središnji dio koji pomaže istraživanju i razumijevanju podataka. Neke od funkcionalnosti koje nudi su: API orijentiran na skup podataka za određivanje odnosa između varijabli, automatska procjena i crtanje grafa linearne regresije, vizualizacija univarijantne i bivarijantne distribucije itd. To se sve može prikazati u širokom opsegu različitih grafova kao što su distribucijski dijagrami, tortni ili trakasti dijagrami, raspršeni grafovi i toplinske mape.

3.1.5. Plotly.express

Modul plotly.express otvorenog koda sadrži funkcije koje mogu stvoriti cijele grafove odjednom, a naziva se Plotly Express ili PX. Dio je plotly biblioteke inspiriran Seabornom i ggplot2om. Posebno je dizajniran da ima sažet, dosljedan i jednostavan API za učenje: jednim učitavanje mogu se napraviti interaktivni dijagrami jednim pozivom funkcije, uključujući karte, animacije itd. Dolazi s ugrađenim skupovima podataka, ljestvicama boja i temama. Također nam omogućuje pronalaženje obrazaca u skupu podataka prije njegova modeliranja.

3.2. Učitavanje podataka

CSV je tekstualna datoteka koja koristi zareze kako bi odvojila podatke. CSV datoteke se koriste kako bi jednostavno mogli učitati velike skupove podataka. Kako bi podatke iz te datoteke mogli učitati u naš DataFrame koristi se funkcija read_csv koja može čitati takve datoteke.

```
In [2]: df = pd.read_csv("C:\\Users\\Karla Marija\\Documents\\7. semestar\\Završni rad\\winequality-red.csv")
```

Slika 4. Učitavanje podataka

S obzirom da datoteka ima veliku količinu podataka, provjerit će se s kakvim podacima se radi. Funkcija df.shape vraća dimenzije DataFramea. Funkcija df.head prikazuje prvih n redaka DataFramea. Ako nije zadan n, prikazuje prvih 5 redaka jer mu je to zadana vrijednost.

```
In [3]: # See the number of rows and columns
   ...: print("Rows, columns: " + str(df.shape))
   ...: # See the first five rows of the dataset
   ...: df.head()
Rows, columns: (1599, 12)
Out[3]:
   fixed acidity  volatile acidity  citric acid  ...  sulphates  alcohol  quality
0            7.4              0.70         0.00  ...       0.56      9.4        5
1            7.8              0.88         0.00  ...       0.68      9.8        5
2            7.8              0.76         0.04  ...       0.65      9.8        5
3           11.2              0.28         0.56  ...       0.58      9.8        6
4            7.4              0.70         0.00  ...       0.56      9.4        5

[5 rows x 12 columns]
```

Slika 5. Prvih 5 redaka DateFramea

Skup podataka izgleda uredno, ali svejedno treba se provjeriti nedostaju li negdje neke vrijednosti ("NaN"). Funkcijom *isna* otkriva se nedostaju li negdje vrijednosti, a funkcijom sum se zbroje. U ovom skupu podataka ne nedostaje niti jedna vrijednost.

```
In [4]: # Missing Values
   ...: print(df.isna().sum())
fixed acidity           0
volatile acidity        0
citric acid             0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide    0
density                 0
pH                      0
sulphates               0
alcohol                 0
quality                 0
dtype: int64
```

Slika 6. Missing values

4. PRIPREMA PODATAKA

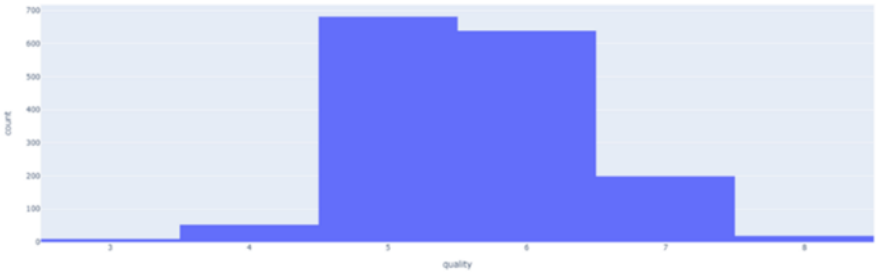
4.1. Korelacijska matrica

Prvo će se prikazati distribucija varijable kvalitete kako bi se provjerilo ima li dovoljno reprezentativnih vina dobre i loše kvalitete.

U skupu podataka koji je učitan, posljednji stupac predstavlja izlaznu varijablu kvalitete vina. Kvaliteta je podijeljena na 7 kategorija s time da su vina kvalitete 1 najlošija, a kvalitete 7 najbolja. Funkcijom histogram će se prikazati koliko je vina dodijeljeno svakoj razini kvalitete. Na x-osi će biti postavljena razina kvalitete, a na y-osi broj vina u pojedinoj kategoriji.

```
In [5]: fig = px.histogram(df,x='quality')
   ...: fig.show()
```

Slika 7. Stvaranje histograma



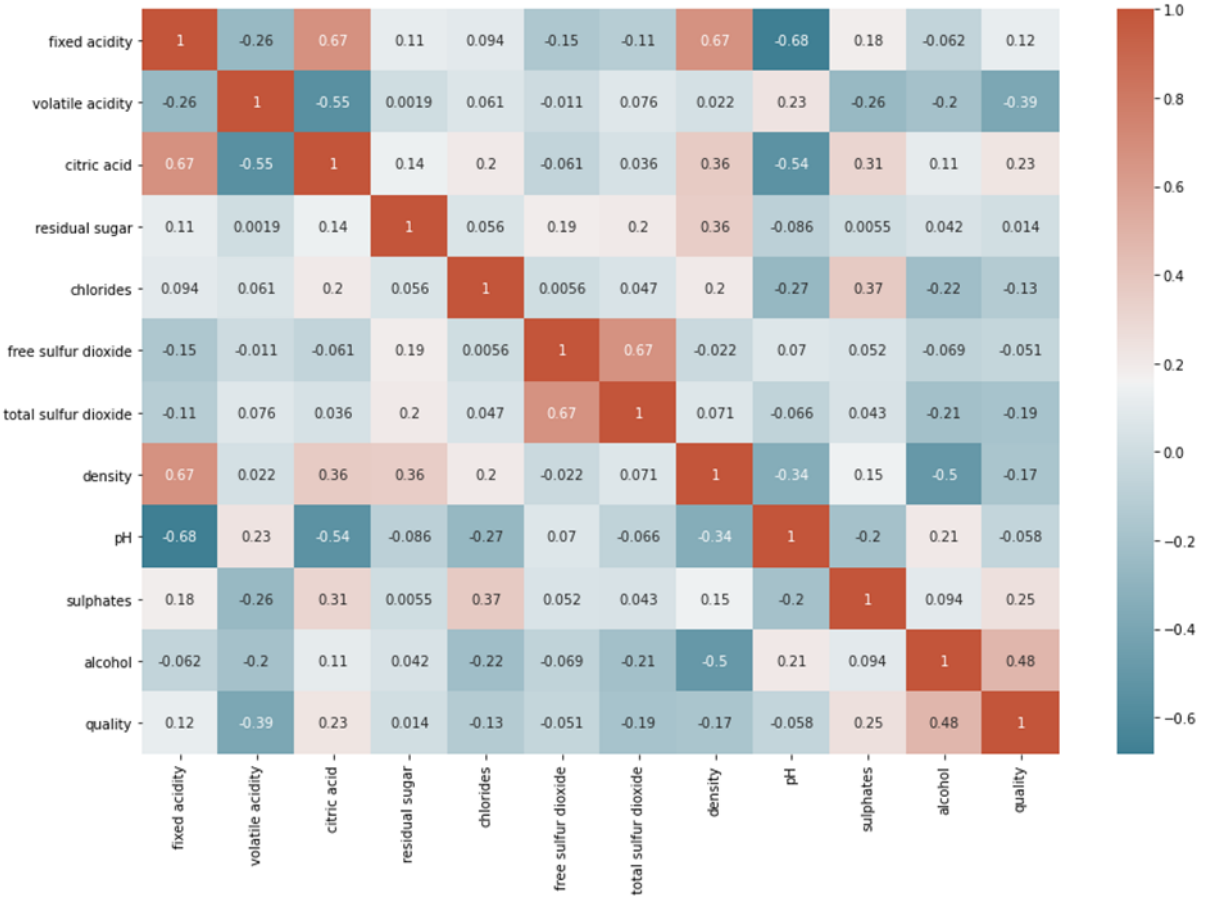
Slika 8. Histogram

Korelacijska matrica je tablica koja pokazuje međusobnu povezanost između dvije varijable, pa poznavanjem jedna možemo odrediti vrijednost druge varijable. Retci i stupci matrice predstavljaju promatrane varijable, a podatak na presjeku nekog retka i stupca predstavlja Pearsonov koeficijent korelacije između varijabli. Na dijagonali je koeficijent korelacije 1 zato što se tu preklapaju stupci i retci varijable sa samom sobom pa je i u potpunoj korelaciji sa sobom. Također jer simetrična po dijagonali jer su i iznad i ispod isti parovi varijabli, tako da ju ne treba cijelu promatrati nego samo njezin jedan dio.

Za određivanje korelacija između varijabli koristi se funkcija corr(). A prikazuje ju se pomoću heatmape.

```
In [6]: corr = df.corr()
   ...: matplotlib.pyplot.subplots(figsize=(15,10))
   ...: sns.heatmap(corr, xticklabels=corr.columns, yticklabels=corr.columns, annot=True, cmap=sns.diverging_palette(220, 20,
as_cmap=True))
```

Slika 9. Stvaranje korelacijske matrice



Slika 10. Korelacijska matrica

4.2. Prilagodba podataka za klasifikacijske probleme

Ideja ovog rada je različitim klasifikacijskim metodama predvidjeti je li vino dobre kvalitete ili ne, tj. potrebna je podjela kvalitete vina na dva razreda. Kvaliteta vina, iz baze podataka koju smo prethodno učitali, je klasificirana u 10 razreda. S obzirom da vino treba klasificirati kao „dobro“ ili „loše“, tj. binarno klasificirati, klasifikaciju u 10 razreda treba prilagoditi. Binarna klasifikacija podataka o kvaliteti vina će se napraviti tako da vina s kvalitetom 7 ili više definiramo kao „dobra vina“ (1), a ona s manjom od 7 kao „loša vina“.

```
In [7]: # Create Classification version of target variable
   ...: df['goodquality'] = [1 if x >= 7 else 0 for x in df['quality']]
```

Slika 11. Klasifikacija ciljne varijable

Nakon što je binarno klasificirana kvaliteta vina, stvorit ćemo dva nova skupa podataka X i Y, pomoću kojih će se razdvojiti varijable značajki i ciljne varijable. Od varijabli značajki iz skupa podataka df će se stvori novi skup podataka X koji će se koristiti kao ulazni podatci za određivanje kvalitete vina. Novi skup podataka X koji sadržava varijable značajki, stvorit će se izbacivanjem stupaca ['quality'] i ['goodquality'] iz skupa podataka df pomoću funkcije df.drop. Ciljana varijabla tj. binarno klasificirani podaci o kvaliteti vina, nalaze se u stupcu ['goodquality'] u skupu podataka df, pa će se uzimanjem tog stupca stvoriti novi skup podataka Y

```
...: # Separate feature variables and target variable
...: X = df.drop(['quality','goodquality'], axis = 1)
...: y = df['goodquality']
```

Slika 12. Razdvajanje varijabli značajki i ciljnih varijabli

Kako bi bili sigurni da se predviđanje provodi s reprezentativnim brojem „dobrih vina“ (1), funkcijom df.value_counts() će se prebrojati koliko ih je u usporedbi s „lošim vinima“ (0). S obzirom na dobivene rezultate, zaključuje se da postoji dovoljan broj „dobrih vina“.

```
In [8]: # See proportion of good vs bad wines
   ...: df['goodquality'].value_counts()
Out[8]:
0    1382
1     217
Name: goodquality, dtype: int64
```

Slika 13. Proporcije dobrih i loših vina

4.3. Priprema podataka za modeliranje

4.3.1. Standardizacija varijabli značajki

Nakon što si podaci provjereni, spremni su za pripremu podataka za modeliranje. Prvi potrebni korak je standardizacija podataka. Standardiziranje podataka je transformiranje podataka tako da njegova distribucija ima srednju vrijednost 0 i standardu devijaciju 1. To se radi kako bi izjednačili opseg podataka jer bi u protivnom naglasak bio na podacima koji prirodno imaju veće vrijednosti.

```
In [11]: # Normalize feature variables
    ...: from sklearn.preprocessing import StandardScaler
    ...: X_features = X
    ...: X = StandardScaler().fit_transform(X)
```

Slika 14. Standardizacija varijabli značajki

4.3.2. Podjela podataka

Drugi korak je podjela podataka u skup za obuku (training set) i skup za testiranje (testing set) kako bi unakrsno provjerili modele i odredili njihovu učinkovitost.

```
In [12]: # Splitting the data
    ...: from sklearn.model_selection import train_test_split
    ...: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=.25, random_state=0)
```

Slika 15. Podjela podataka na skup za obuku i skup za testiranje

4.4. Klasifikacijske metode

Nakon što su napravljeni skupovi za obuku i testiranje, može ih se koristiti u svih pet modela strojnog učenja koji se uspoređuju: decision trees, random forests, AdaBoost, Gradient Boost i XGBoost. Uspješnost njihovih rezultata temelji se na točnosti (accuracy). Svaka pojedina metoda uči se koristeći funkciju *fit.* S obzirom da sve ove klasifikacijske metode pripadaju nadgledanom učenju funkcija prima dva argumenta, X_train i y_test, gdje prvi argument predstavlja početne značajke, a drugi željeno rješenje. Nakon što je model naučio na setu za obuku, pomoću testnog seta X_test, predviđa sam konačna rješenja y_pred. Nakon toga usporede se prava konačna rješenja seta za testiranje, y_test, sa rješenjima koja je sam model proizveo, y_pred. To se može vidjeti pomoću *classification_report* koji ujedno prikaže i točnost modela, a što je potrebno za daljnu usporedbu s ostalim modelima.

```
In [13]: from sklearn.metrics import classification_report
    ...: from sklearn.tree import DecisionTreeClassifier
    ...: model1 = DecisionTreeClassifier(random_state=1)
    ...: model1.fit(X_train, y_train)
    ...: y_pred1 = model1.predict(X_test)
    ...: print(classification_report(y_test, y_pred1))
              precision    recall  f1-score   support

           0       0.96      0.92      0.94       355
           1       0.53      0.73      0.62        45

    accuracy                           0.90       400
   macro avg       0.75      0.83      0.78       400
weighted avg       0.92      0.90      0.90       400
```

Slika 16. Decision trees

```
In [14]: from sklearn.ensemble import RandomForestClassifier
    ...: model2 = RandomForestClassifier(random_state=1)
    ...: model2.fit(X_train, y_train)
    ...: y_pred2 = model2.predict(X_test)
    ...: print(classification_report(y_test, y_pred2))
              precision    recall  f1-score   support

           0       0.95      0.97      0.96       355
           1       0.68      0.58      0.63        45

    accuracy                           0.92       400
   macro avg       0.82      0.77      0.79       400
weighted avg       0.92      0.92      0.92       400
```

Slika 17. Random Forrest

```
In [15]: from sklearn.ensemble import AdaBoostClassifier
    ...: model3 = AdaBoostClassifier(random_state=1)
    ...: model3.fit(X_train, y_train)
    ...: y_pred3 = model3.predict(X_test)
    ...: print(classification_report(y_test, y_pred3))
              precision    recall  f1-score   support

           0       0.94      0.94      0.94       355
           1       0.51      0.49      0.50        45

    accuracy                           0.89       400
   macro avg       0.72      0.71      0.72       400
weighted avg       0.89      0.89      0.89       400
```

Slika 18. AdaBoost

```
In [16]: from sklearn.ensemble import GradientBoostingClassifier
    ...: model4 = GradientBoostingClassifier(random_state=1)
    ...: model4.fit(X_train, y_train)
    ...: y_pred4 = model4.predict(X_test)
    ...: print(classification_report(y_test, y_pred4))
              precision    recall  f1-score   support

           0       0.94      0.94      0.94       355
           1       0.52      0.51      0.52        45

    accuracy                           0.89       400
   macro avg       0.73      0.73      0.73       400
weighted avg       0.89      0.89      0.89       400
```

Slika 19. Gradient Boosting

```
In [20]: import xgboost as xgb
    ...: model5 = xgb.XGBClassifier(random_state=1)
    ...: model5.fit(X_train, y_train)
    ...: y_pred5 = model5.predict(X_test)
    ...: print(classification_report(y_test, y_pred5))
C:\Anaconda\lib\site-packages\xgboost\sklearn.py:1224: U

The use of label encoder in XGBClassifier is deprecated
Pass option use_label_encoder=False when constructing XG
1, 2, ..., [num_class - 1].

[08:56:29] WARNING: C:/Users/Administrator/workspace/xgb
evaluation metric used with the objective 'binary:logist
restore the old behavior.
              precision    recall  f1-score   support

           0       0.96      0.95      0.95       355
           1       0.62      0.69      0.65        45

    accuracy                           0.92       400
   macro avg       0.79      0.82      0.80       400
weighted avg       0.92      0.92      0.92       400
```

Slika 20. XGBoost

Iz *classification_report* se očitavaju točnosti svakog modela:

Decision tree - 0,90

Random forest – 0,92

AdaBoost – 0,89

Gradient Boost – 0,89

XGBoost – 0,92.

Prema ovim rezultatima, random forrest i XGBoost model su jednako dobro predvidjeli rješenja na temelju zadanog testnog seta. Ako se usporede po f1-score za predviđanje dobre kvalitete vina (1), random forest će imati rezultat 0,63, a XGBoost 0,65. Kako XGBoost model ima bolji rezultat u f1-score, daje mu se prednost kao najboljem između ovih 5 modela.

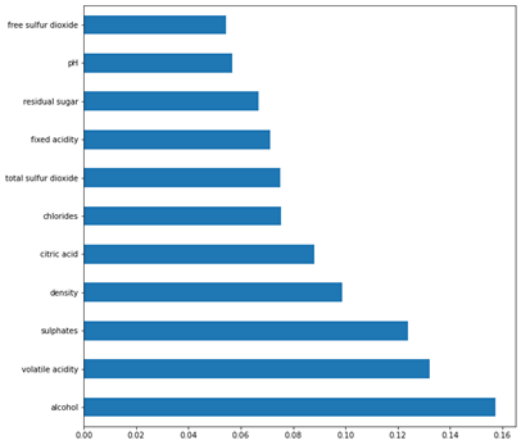
5. VAŽNOST ZNAČAJKI

5.1. Važnost značajki

Drugi cilj ovog rada je ustanoviti koje početne značajke naviše utječu na kvalitetu vina što se će se provjeriti pomoću dva najbolja modela: random forest i XGBoost. Važnosti značajki se dobiju pomoću atributa *feature_imporeances_* i računaju se kao srednja vrijednost i standardna devijacija akumulacije smanjenja nečistoća svakog stabla.

```
In [16]: feat_importances = pd.Series(model2.feature_importances_, index=X_features.columns)
    ...: feat_importances.nlargest(25).plot(kind='barh',figsize=(10,10))
```

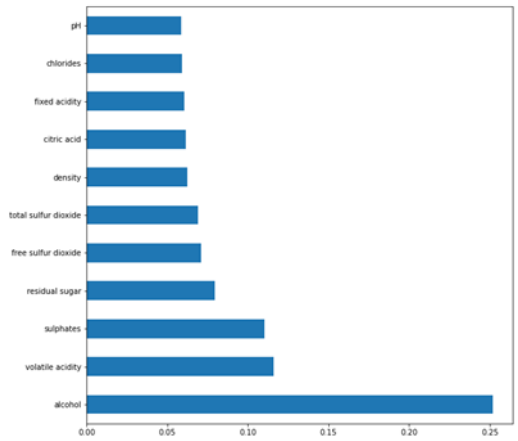
Slika 21. Određivanje najvažnijih značajki za kvalitetu pomoću random forest modela



Slika 22. Važnost značajki s obzirom na random forest algoritam

```
In [17]: feat_importances = pd.Series(model5.feature_importances_, index=X_features.columns)
    ...: feat_importances.nlargest(25).plot(kind='barh',figsize=(10,10))
```

Slika 23. Određivanje najvažnijih značajki za kvalitetu pomoću XGBoost modela



Slika 24. Važnost značajki s obzirom na XGBoost algoritam

Iz grafova (slika 22. i slika 24.) vidi se kako važnosti značajki variraju, ali tri najvažnije značajke su iste u oba modela, a to su alkohol, hlapljiva kiselost i sulfati.

5.2. Usporedba top značajki

Detaljnijim promatranjem vidi se da vina bolje kvalitete imaju prosječno više razine alkohola, nižu prosječnu hlapljivu kiselost, višu prosječnu razinu sulfata i višu prosječnu razinu zaostalog šećera.

```
In [18]: # Filtering df for only good quality
    ...: df_temp = df[df['goodquality']==1]
    ...: df_temp.describe()
    ...: # Filtering df for only bad quality
    ...: df_temp2 = df[df['goodquality']==0]
    ...: df_temp2.describe()
```

Slika 25. Filtriranje vina dobre i loše kvalitete

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality	goodquality
count	217.000000	217.000000	217.000000	217.000000	217.000000	217.000000	217.000000	217.000000	217.000000	217.000000	217.000000	217.000000	217.0
mean	8.847005	0.405530	0.376498	2.708756	0.075912	13.981567	34.889401	0.996030	3.288802	0.743456	11.518049	7.082949	1.0
std	1.999977	0.144963	0.194438	1.363026	0.028480	10.234615	32.572238	0.002201	0.154478	0.134038	0.998153	0.276443	0.0
min	4.900000	0.120000	0.000000	1.200000	0.012000	3.000000	7.000000	0.990640	2.880000	0.390000	9.200000	7.000000	1.0
25%	7.400000	0.300000	0.300000	2.000000	0.062000	6.000000	17.000000	0.994700	3.200000	0.650000	10.800000	7.000000	1.0
50%	8.700000	0.370000	0.400000	2.300000	0.073000	11.000000	27.000000	0.995720	3.270000	0.740000	11.600000	7.000000	1.0
75%	10.100000	0.490000	0.490000	2.700000	0.085000	18.000000	43.000000	0.997350	3.380000	0.820000	12.200000	7.000000	1.0
max	15.600000	0.915000	0.760000	8.900000	0.358000	54.000000	289.000000	1.003200	3.780000	1.360000	14.000000	8.000000	1.0

Slika 26. Dobra vina

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality	goodquality
count	1382.000000	1382.000000	1382.000000	1382.000000	1382.000000	1382.000000	1382.000000	1382.000000	1382.000000	1382.000000	1382.000000	1382.000000	1382.0
mean	8.236831	0.547022	0.254407	2.512120	0.089281	16.172214	48.285818	0.996859	3.314616	0.644754	10.251037	5.408828	0.0
std	1.682726	0.176337	0.189665	1.415778	0.049113	10.467685	32.585604	0.001808	0.154135	0.170629	0.959664	0.601719	0.0
min	4.600000	0.160000	0.000000	0.900000	0.034000	1.000000	6.000000	0.990070	2.740000	0.330000	8.400000	3.000000	0.0
25%	7.100000	0.420000	0.082500	1.900000	0.071000	8.000000	23.000000	0.995785	3.210000	0.540000	9.500000	5.000000	0.0
50%	7.800000	0.540000	0.240000	2.200000	0.080000	14.000000	39.500000	0.996800	3.310000	0.600000	10.000000	5.000000	0.0
75%	9.100000	0.650000	0.400000	2.600000	0.091000	22.000000	65.000000	0.997900	3.410000	0.700000	10.900000	6.000000	0.0
max	15.900000	1.580000	1.000000	15.500000	0.611000	72.000000	165.000000	1.003690	4.010000	2.000000	14.900000	6.000000	0.0

Slika 27. Loša vina

6. ZAKLJUČAK

Posljednjih godina povećan je interes za vinarsku industriju što zahtijeva napredak u ovoj industriji. Stoga tvrtke ulažu u nove tehnologije kako bi unaprijedile proizvodnju i prodaju vina. U tom smjeru, certifikacija kvalitete vina igra vrlo važnu ulogu za oba procesa i zahtijeva ispitivanje vina od strane ljudskih stručnjaka. Ovaj rad istražuje korištenje tehnika strojnog učenja kako bi se učinkovito povezale utjecajne čimbenike za stvaranje vina i njegovu kvalitetu.

Nakon što smo početni skup podataka o kvaliteti vina i njegovih značajki prilagodili za klasifikaciju i klasificirali usporedili smo točnosti svih algoritama. Decision trees algoritam imao je točnost od 90%, radnom forest 92%, AdaBoost 89%, Gradient Boost 89% i XGBoost 92%. S obzirom na rezultate najbolji algoritmi za klasifikaciju vina su decision tree i XGBoost, međutim kako je XGBoost brži i obrađuje vrijednosti koje nedostaju u skupu, dajemo mu prednost kao najbolji algoritam za ovaj problem.

LITERATURA

[1] Meliz Yuvalli, Belma Yaman, Ozgur Tosun: Classification Comparison of Machine Learning Algorithms Using Two Independen CAD Datasets, 2022.

[2] Vishal Sharma: Survey of Classification Algorithms and Various Modle Selection Methods

[3] Urtar Pradesh: Selection of important features and predicting wine quality usig machine learning techniques, 2017.

[4] Rohan Dilip Kothawasw: Wine quality predcition modle using machine learning techniques, 2021.

[5] Gregory D.Nelson: Red and white wind data analysis predict quality of wine

[6] Aized Amin Soofi, Arshad Awan: Classification techniques in machine learning: applications and issues

[7] https://www.kaggle.com/code/denizaytacc/beginner-wine-quality-binary-classification-92/notebook

[8] https://scikit-learn.org/stable/modules/cross_validation.html

[9] https://towardsdatascience.com/what-is-a-decision-tree-22975f00f3e1

[10] https://www.mygreatlearning.com/blog/gradient-boosting/

[11] https://www.sciencedirect.com/topics/food-science/wine-quality

PRILOG

I. Link za pristup kodu za predviđanje kvalitete vina
https://towardsdatascience.com/predicting-wine-quality-with-several-classification-techniques-179038ea6434