|  |  |  |
| --- | --- | --- |
|  | Univerzitet u Beogradu - Elektrotehnički fakultet  Katedra za signale i sisteme |  |



**Izveštaj**

**Primena stabala odlučivanja na dijagnostiku dijabetesa**

**Kandidat**

Aleksandar Cvetković, 2018/0323

**Mentor**

Doc. dr Aleksandra Krstić

Beograd, *decembar* 2022. godine

# Sadržaj

[Sadržaj 2](#_Toc122383140)

[1 Uvod 4](#_Toc122383141)

[2 Analiza postojećih istraživanja 5](#_Toc122383142)

[3 Metodologija rada 6](#_Toc122383143)

[3.1 Podela skupa podataka 7](#_Toc122383144)

[3.2 Funkcije nečistoće 7](#_Toc122383145)

[3.2.1 Entropija – IH 8](#_Toc122383146)

[3.2.2. Gini nečistoća 9](#_Toc122383147)

[3.2.3. Greška klasifikacije – IE 10](#_Toc122383148)

[3.2.4. Hi-kvadrat test – Ichi 10](#_Toc122383149)

[3.3. Diskretizacija kontinualnih promenjivih 13](#_Toc122383150)

[3.3.1. Podela na skupove jednakih veličina 13](#_Toc122383151)

[3.3.2. Podela na skupove sa jednakim brojem primera u njima 14](#_Toc122383152)

[3.3.3. Dinamička podela atributa na skupove 14](#_Toc122383153)

[3.4. Metode za sprečavanje preobučavanja stabla 15](#_Toc122383154)

[3.4.1. Ograničavanje maksimalne dubine stabla: 16](#_Toc122383155)

[3.4.2. Ograničavanje minimalnog broja primera u čvoru za podelu: 16](#_Toc122383156)

[3.4.3. Orezivanje stabla za smanjenje greške (*Reduced Error Pruning)* 16](#_Toc122383157)

[3.4.4. Pesimistično orezivanje stabla (*Pessimistic Error Pruning*) 17](#_Toc122383158)

[3.5. Nedostajući podaci 19](#_Toc122383159)

[4 Implementacija 20](#_Toc122383160)

[4.1. Klasa *DecisionTree* 20](#_Toc122383161)

[4.1.1. Parametri klase *DecisionTree* 20](#_Toc122383162)

[4.1.2. Metode klase *DecisionTree* 21](#_Toc122383163)

[4.2. Klasa *Node* 22](#_Toc122383164)

[5 Rezultati 23](#_Toc122383165)

[6 Diskusija i zaključak 24](#_Toc122383166)

[Literatura 25](#_Toc122383167)

# 1 Uvod

# 2 Analiza postojećih istraživanja

# 3 Metodologija rada

Stabla odlučivanja su model mašinskog učenja za analizu podataka i predikciju koji se zasniva na postavljanju niza pitanja. Naziv su dobila su po tome što svojim izgledom podsećaju na okrenuta stabla iz prirode. Podaci su u stablima sadržani unutar čvorova, pri čemu se u svakom koraku čvor koji sadrži dati skup podataka grana na više čvorova u zavisnosti od vrednosti atributa podataka. Čvorove koji se granaju nazivamo **čvorovima odluke**. Grananje prestaje kada nema više atributa na osnovu kojih bismo dalje delili skup ili kada metrike performansi stabla budu dostignute. Čvorove koji se ne granaju već samo nose informaciju o klasi koju predstavljaju nazivamo **čvorovima listovima**.

Podaci za analizu predstavljeni su u vidu skupa podataka u kom svaki primer ima svoje odlike (osobine, prediktore) i labelu (oznaku) koja predstavlja ciljnu vrednost koju bismo želeli da predvidimo. Stabla odlučivanja mogu se koristiti za rešavanje problema regresije – kada je ciljna promenjiva realan broj i klasifikacije – kada ciljna promenjiva iz diskretnog skupa uzima vrednost koja predstavlja njenu klasu. Tematika ovog rada baziraće se na upotrebi stabala odlučivanja za potrebe klasifikacije.

Stabla odlučivanja spadaju u nadgledane modele mašinskog učenja, što znači da nam za svaki primer u našem skupu podataka trebaju odlike datog primera kao i labela koja označava ispravnu klasu ciljne promenjive.

Ono što čini stabla odlučivanja posebno pogodnim za primenu u medicini predstavlja njihova izuzetna interpretabilnost koja omogućava lekarima da na osnovu formiranog stabla donesu zaključke i uvide veze između odlika pacijenata kojih pre nisu bili svesni.

Na slici 1 možemo da vidimo primer stabla koje se koristi za dijagnostiku dijabetesa:

Diagram

Description automatically generated

Slika 1. Primer stabla odlučivanja za dijagnostiku dijabetesa

## 3.1 Podela skupa podataka

Cilj stabala odlučivanja je konstrukcija stabla koje na datom skupu podataka ostvaruje najbolju moguću tačnost. Teoretski, ukoliko bismo dozvolili stablu odlučivanja da se neograničeno grana ono bi moglo da postigne stopostotnu tačnost tako što bi dugom serijom pitanja savršeno razvojila primerke različitih klasa u posebne čvorove (ovo je moguće ukoliko u skupu podataka nema izuzetnih slučajeva gde su dva primera istih odlika označeni različitom klasom). Ipak, iako bi ovakvo stablo postiglo zavidan nivo tačnosti, ta tačnost bi se zasnivala na prevelikom prilagođavanju primerima u skupu podataka umesto na izvlačenju generalnih zaključaka koji bi nam koristili i za do tad neviđene primere. Ova pojava se u mašinskom učenju naziva **preobučenost**. Da bismo detektovali preobučenost nakon konstrukcije stabla, skup podataka delimo na dva dela: jedan skup nad kojim će se vršiti konstrukcija stabla (treningu) i drugi skup koji neće učestvovati u treningu već će nam služiti za proveru performansi stabla na primerima koji do tada nisu viđeni. Ove skupove podataka nazivamo trening skup i validacioni skup i glavni indikator preobučenosti biće velika tačnost modela na trening skupu i mala tačnost modela na validacionom skupu podataka.

## 3.2 Funkcije nečistoće

U saglasnosti sa načinom podele predstavljenim u prethodnom poglavlju cilj rada modela jeste stablo koje ostvaruje maksimalnu tačnost na validacionom skupu podataka. Da bismo obezbedili visoke performanse na validacionom skupu i sprečili preobučavanje potrebno je veliku pažnju posvetiti redosledu odlika na osnovu kojih se vrši grananje unutar stabla. Jedan od načina da izaberemo redosled odlika za grananje je da probamo svaku moguću kombinaciju redosleda odlika. Iako bi ovakav postupak sa sigurnošću doveo do optimalnog rešenja on je izuzetno vremenski skup i postao bi nemoguć za skupove podataka sa velikim brojem odlika.

Umesto isprobavanja svake moguće kombinacije, stabla odluke u svakom koraku biraju odliku koja odvaja primere u grane koje su u najvećoj meri „**čiste**” tj. sastoje se iz većeg dela primera iste klase. Za procenu „čistoće” grana koriste se funkcije koje se nazivaju **funkcije čistoće**. U daljem delu biće predstavljene najčešće korišćene funkcije čistoće u stablima odlučivanja.

### 3.2.1 Entropija – IH

U toeriji informacija, entropija predstavlja meru neodređenosti slučajne promenjive. Primenjena na stabla odlučivanja, onda predstavlja meru nečistoće nakon grananja i definisana je na sledeći način:

gde **t** predstavlja oznaku čvora, **p(k|t)** predstavlja proporciju primera koji pripadaju klasi **k** u čvoru **t**, dok **C** predstavlja ukupan broj klasa.

Shodno ovoj definiciji, entropija je minimalna i iznosi 0 kada svi primeri u čvoru pripadaju istoj klasi (tj. čvor je „čist“). Entropija ima maksimalnu vrednost kada u čvoru ima podjednako mnogo primera različitih klasa. Koristeći entropiju kao funkciju čistoće, minimizacija entropije direktno bi obezbedila maksimalnu čistoću listova nakon grananja.

Za binarnu promenjivu koja uzima vrednost 1 sa verovatnoćom p dat je sledeći grafik vrednosti entropije:

A picture containing diagram

Description automatically generated

Slika 2. Vrednost entropije u zavisnosti od verovatnoće p binarne promenjive. Izvor: [wikipedia](https://en.wikipedia.org/wiki/Binary_entropy_function)

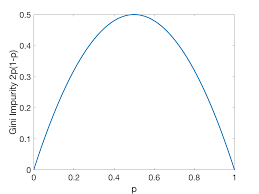
### 3.2.2. Gini nečistoća

Gini nečistoća je funkcija nečistoće koja je jako slična entropiji i takođe se zasniva na proporciji primera različitih klasa unutar čvora. Definisana je na sledeći način:

gde su oznake iste kao u jednačini entropije.

Gini nečistoća takođe ima minimalnu vrednost kada se u listu nalaze primeri samo jedne klase, dok je maksimalna kada se u listu nalazi jednak broj jedinki različitih klasa. Po vrednostima funkcija, Gini nečistoća je veoma slična skaliranoj verziji entropije i često prelazak sa jedne funkcije nečistoće na drugu ne dovodi ni do kakvih promena. Međutim, zbog lakšeg izračunavanja Gini nečistoće od entropije (za koju je neophodno nalaziti vrednosti logaritma sa osnovom 2) Gini nečistoća se češće primenjuje.

Za binarnu promenjivu koja uzima vrednost 1 sa verovatnoćom p dat je sledeći grafik vrednosti Gini nečistoće (skaliran sa 2):



Slika 3. Vrednost Gini nečistoće u zavisnosti od verovatnoće p binarne promenjive. Izvor: [cornell](https://www.cs.cornell.edu/courses/cs4780/2022sp/notes/Notes20.pdf)

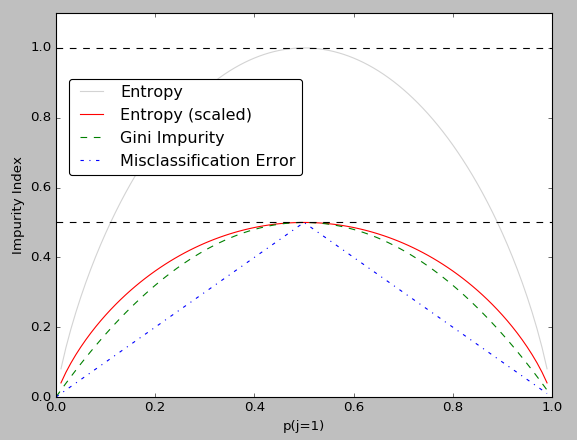
### 3.2.3. Greška klasifikacije – IE

Greška klasifikacije je funkcija nečistoće koja uzima u obzir samo proporciju primera najčešće klase u čvoru. Pošto će list koji nije čist imati oznaku upravo najčešće klase u njemu, onda će proporcija primera ostalih klasa predstavljati **grešku klasifikacije**.

Ona je definisana sledećom formulom:

Iako je jednostavna za računanje, greška klasifikacije nije dobar indikator čistoće čvora zbog manje osetljivost na promene u klasnim proporcijama unutar čvora. Ipak, ona svoju primenu nalazi prilikom postupaka **orezivanja** stabala o čemu će biti reči u daljim poglavljima.

Poređenje funkcija nečistoće: entropije, Gini i greške klasifikacije dato je na sledećoj slici:



Slika 4. Vrednosti funkcija nečistoće u zavisnosti od verovatnoće p binarne promenjive. Izvor: [scikit-learn](https://www.bogotobogo.com/python/scikit-learn/scikt_machine_learning_Decision_Tree_Learning_Informatioin_Gain_IG_Impurity_Entropy_Gini_Classification_Error.php)

### 3.2.4. Hi-kvadrat test – Ichi

Hi-kvadrat test predstavlja funkciju za poređenje grupa u odnosu na kategorijalnu zavisnu promenjivu. Ova funkcija opisuje razliku između posmatrane frekvencije - **fo** i očekivane frekvencije - **ft.** Očekivanu frekvenciju definišemo kao frekvenciju koja bi mogla da se očekuje potpuno slučajno, u skladu sa uslovnim verovatnoćama svakog ishoda.

Očekivanu frekvenciju za svaku od ćelija tabele određujemo proizvodom marginalnih frekvencija (sume odgovarajućeg reda i kolone) podeljenim ukupnom veličinom uzorka tj.:

gde **r** predstavlja odgovarajući red, **k** odgovarajuću kolonu, a **N** ukupan broj uzoraka.

Vrednost Hi-kvadrat testa određuje se kao suma odstupanja očekivanih od osmotrenih vrednosti frekvencija u svakoj ćeliji tj.:

gde **fo** predstavlja posmatranu frekvenciju, a **ft** očekivanu frekvenciju

Posmatrajmo sledeću sliku:

Chart, bubble chart

Description automatically generated

Slika 5. Tabela zadovoljstva studenata uslugama studentskog restorana gde nema razlike u raspodeli. Izvor: [filozofski fakultet](http://psihologija.ff.uns.ac.rs/viz/hi-kvadrat-test)

Na slici je data tabela čiji redovi predstavljaju fakultete: Filozofski fakultet (FF) i Prirodno-matematički fakultet (PMF). U kolonama je izraženo zadovoljstvo studenata uslugama studentskog restorana: 1-nezadovoljni, 2-zadovoljni, 3-vrlo zadovoljni. Iako je više studenata sa FF-a učestvovalo u anketi možemo da primetimo da su međusobni odnosi broja studenata u sve tri kategorije zadovoljstva jednaki.

Iz datog primera možemo da vidimo da ni za jednu ćeliju tabele nema razlike između posmatrane i očekivane frekvencije. Na primer, za ćeliju FF1 očekivana frekvencija bila bi 15x70/105=10 što se upravo slaže sa sadržajem ćelije. Samim tim suma odstupanja vrednosti očekivanih i osmotrenih frekvencija u datoj tabeli je 0 što bi značilo da ne postoje statistički značajne razlike između distribucija studenata sa FF-a i PMF-a.

U kontekstu stabala odluke ovakva situacija odgovarala bi slučaju gde smo nakon izvršene podele po zadovoljstvu studenata uslugama restorana uvideli da novonastale grupe imaju podjednak odnos frekvencija studenata sa FF-a i PMF-a kao što je ona bila i pre podele tj. grupe nastale podelom po zadovoljstvu nisu postale čistije. Iz ovog primera možemo da izvučemo zaključak da bi manje vrednosti Hi-kvadrat testa odgovarale lošijoj podeli po datom atributu u stablu odlučivanja.

Na sledećoj slici dat je primer kada između grupa postoje znatne statističke razlike u distribuciji studenata:

Chart, bubble chart

Description automatically generated

Slika 6. Slika 5. Tabela zadovoljstva studenata uslugama studentskog restorana gde ima razlike u raspodeli. Izvor: [filozofski fakultet](http://psihologija.ff.uns.ac.rs/viz/hi-kvadrat-test)

Na slici 6. možemo da vidimo da u ćelijama u prvoj i trećoj koloni postoje znatne statističke razlike u distribuciji studenata. Na primer očekivana frekvencija studenata u ćeliji FF1 iznosi: 50x60/120=25 ali je njena stvarna vrednost 10. Ovakve razlike akumuliraju se u Hi-kvadrat testu čija vrednost sada iznosi 36. U kontekstu stabala odluke ovakva situacija odgovarala bi slučaju gde smo nakon izvršene podele po zadovoljstvu studenata uslugama restorana uvideli da novonastale grupe veoma različit odnos frekvencija studenata sa FF-a i PMF-a nego što je ona bila i pre podele tj. grupe nastale podelom po zadovoljstvu su **čistije**. Iz ovog primera možemo da izvučemo zaključak da bi veće vrednosti Hi-kvadrat testa odgovarale boljoj podeli po datom atributu u stablu odlučivanja.

Kao što smo videli iz datih primera Hi-kvadrat test može se efikasno koristiti da opiše kvalitet podele po određenom atributu u stablu, pri čemu veće vrednosti označavaju bolju podelu po datom atributu. Hi-kvadrat test se, osim za izbor atributa za podelu tokom konstrukcije stabla može koristiti i za orezivanje stabala o čemu će biti reči u daljim poglavljima.

## 3.3. Diskretizacija kontinualnih promenjivih

Pri grananju skupa na delove na osnovu vrednosti atributa podrazumevali smo da su te vrednosti kategoričke tj. da uzimaju vrednosti iz diskretnog skupa: kao što studenti mogu da se izjasne za jednu od tačno tri vrednosti zadovoljstva: *nezadovoljan*, *zadovoljan*, *vrlo* *zadovoljan*. Međutim, prilikom primena stabla odlučivanja na medicinske podatke atributi često mogu imati vrednosti koje predstavljaju cele ili realne brojeve: broj godina, krvni pritisak, visina, težina, koncentracija glukoze u krvi itd.. Posmatranje ovih vrednosti na isti način kao i kategoričke vrednosti dovelo bi do toga da skoro svaki pacijent ima svoju vrednost datog atributa što bi dovelo do toga da se model prilagođava svakom posebnom pacijentu, umesto da sagledava generalnu sliku. Ovakav postupak doveo bi do preobučavanja. Da bismo sprečili preobučavanje, potrebno je da realnu vrednost pretvorimo u kategoričku primenom nekog od algoritama diskretizacije. Na primer umesto da visinu izražavamo u centimetrima mogli bismo da je podelimo na dve kategorije ili podeoke: niži od 180cm i viši od 180cm. U daljem delu ovog poglavlja biće objašnjeni načini za diskretizaciju kontinualnih promenjivih.

### 3.3.1. Podela na skupove jednakih veličina

Prvi i najjednostavniji vid podele zasniva se na tome da podelimo skup mogućih vrednosti na delove jednakih veličina. Na primeru visine izražene u centimetrima podela na skupove jednakih veličina bila bi određena na sledeći način:

cm

190

210

170

150

Visina ispitanika kreće se u granicama od 150cm do 210cm. Odabraćemo da broj skupova za podelu bude 3. Na ovaj način podela će biti izvršena na skupove: niži od 170cm, između 170 i 190cm i viši od 190cm.

Ovaj način podele je lak za implementaciju i izuzetno brz za realizaciju ali se može dogoditi da za pojedine podeoke ne postoji ni jedan primer koji im pripada (u našem primeru nije se pojavio ni jedan ispitanik visine između 170 i 190). Verovatnoća javljanja ovakvog podeoka raste sa povećanjem broja podeokai smanjenjem broja primera (što se dešava sa povećanjem dubine čvora). Takođe, neophodno je precizno podesiti broj podeoka za svaki atribut.

### 3.3.2. Podela na skupove sa jednakim brojem primera u skupu

Ovaj način podele podrazumeva da skup vrednosti bude podeljen na podeoke od kojih će svaki sadržati približno jednak broj vrednosti. Na primeru visine izražene u centimetrima podela na skupove sa jednakim brojem primera bila bi određena na sledeći način:

cm

170

160

210

150

Visina ispitanika kreće se u granicama od 150cm do 210cm. Odabraćemo da broj skupova za podelu bude 3. Na ovaj način podela će biti izvršena na skupove: niži od 160cm, između 160 i 170 i viši od 170cm.

Na ovaj način nema skupova koji u sebi ne sadrže ni jedan primer ali je i dalje potrebno podesiti broj podeoka za svaki atribut.

### 3.3.3. Dinamička podela atributa na skupove

Dinamička podela atributa na skupove rešava problem izbora broja podeoka time što sama određuje podeoke na koje skup podataka treba da bude podeljen. Ovaj vid podele skupa pored vrednosti atributa primera uzima u obzir i njihovu klasu.

Algoritam deljenja funkcioniše na sledeći način:

* Inicijalno svaki od primera u datom skupu predstavlja po jedan podeok – ovi podeoci nazicaju se „**slabim**”
* Zatim se slabi podeoci iste klase koji se nalaze jedan pored drugog grupišu u veće podeoke koji se nazivaju „**jakim**” podeocima
* Na kraju se iterativno svi slabi podeoci koji se nalaze između dva jaka podeoka iste klase grupišu zajedno sa ta 2 jaka podeoka u još veći jaki podeok

Rad algoritma za dinamičku podelu predstavljen je po koracima na sledećoj slici:

Diagram

Description automatically generated

Slika 7. Primer rada algoritma za dinamičku podelu kontinualnog skupa vrednosti. Izvor: rad – dodati

Velika prednost ovog načina podele je što nema potrebe za eksplicitnim podešavanjem parametra broja podeoka. Ipak, ovaj algoritam pokazuje slabije rezultate u višeklasnoj klasifikaciji gde je broj klasa veliki jer uzrokuje manje podeoke.

## 3.4. Metode za sprečavanje preobučavanja stabla

Kao što je bilo reči u poglavlju 3.1. stabla odlučivanja čiju konstrukciju ne ograničimo razgranaće se tako da će se povinovati svakom dostupnom primeru u skupu podataka za trening. Ovaj postupak će učiniti da greška na trening skupu bude mala, jer mu je stablo prilagođeno, ali će greška na validacionom skupu biti velika. S’ obzirom da validacioni skup podataka predstavlja primere koje stablo nije videlo prilikom procesa treninga ovaj skup mnogo bolje oslikava dešavanja pri praktičnoj primeni stabla, kada dobijemo podatke od novog pacijenta kakve stablo do tada nije videlo. Zbog osobine stabla da se prilagođava svakom trening skupu koji dobije, stabla će se veoma razlikovati po svojoj konstrukciji u zavisnosti od toga kakvi podaci se nalaze u trening skupu. Zbog velike razlike u izgledu stabla u odnosu na podatke u trening skupu kažemo da ovaj algoritam ima **veliku varijansu.**

U daljem delu ovog poglavlja biće reči o tehnikama za rešavanje problema preobučavanja stabla. Prvo će biti izložene tehnike za ograničavanje grananja stabla tokom njegove konstrukcije, a zatim će biti obrađeni postupci za smanjenje dubine stabla nakon njegove konstrukcije metodama orezivanja (potkresivanja).

Sprečavanjem stabla da raste neograničeno, listovi stabla se više neće sastojati od primera samo jedne klase čime će prestati da budu čisti. Za ovakve listove klasa će biti određena klasom primera kojih ima najviše u datom listu.

### 3.4.1. Ograničavanje maksimalne dubine stabla:

U poglavlju 3.4. objašnjena je tendencija stabla odlučivanja da se grana sve dok se ne povinuje svakom pojedinačnom primeru iz trening skupa, što dovodi do preobučavanja. Ideja ograničavanja maksimalne dubine stabla je da spreči neograničeno grananje stabla čime će smanjiti preobučavanje. Ovo ograničavanje sprovodi se preko parametra **max\_depth,** koji označava maksimalnu dubinu do koje puštamo stablo da raste. Na ovaj način podela će biti izvršena samo po najinformativnijih **max\_depth** atributa, što sprečava da stablo nakon toga krene da donosi odluke po neinformativnim atributima čije vrednosti nemaju direktne veze sa izborom klase.

### 3.4.2. Ograničavanje minimalnog broja primera u čvoru za podelu:

Slično kao ograničavanje maksimalne dubine stabla, i ova metoda pokušava da spreči grananje na osnovu neinformativnih atributa. Primenom ove metode, prestaće se sa daljim grananjem čvora koji sadrži manje od parametra koji označava minimalni broj primera. Ovo će dovesti do toga da se primeri koje nije moguće „razumno“ podeliti na posebne čvorove ne dele dalje koristeći neinformativne atribute čije vrednosti nemaju direktne veze sa izborom klase.

### 3.4.3. Orezivanje stabla za smanjenje greške (*Reduced Error Pruning)*

Orezivanje stabla za smanjenje greške je tehnika orezivanja za koju je neophodno obezbediti poseban skup podataka uz pomoć kog će se vršiti orezivanje. Podaci iz ovog skupa se ne smeju koristiti za trening stabla. Shodno tome trening skup podataka koji stablo dobije podeliće se na podatke koji će stvarno biti korišćeni za trening i na podatke koji će biti korišćeni za orezivanje.

Postoji više verzija ovog algoritama koje su nastale kao implementacija opisa algoritma iz rada (dodati rad). U ovom radu biće korišćena verzija algoritma koja je u radu (dodati rad) bila ocenjena kao jedina konzistentna sa opisima algoritma u radu (dodati rad) ali i u ostalim radovima (dodati autora).

Algoritam funkcioniše na sledeći način:

* Čvorovi se obilaze u post-order redosledu obilaska
* Listovi se ne obrađuju
* Za svaki čvor koji nije list (čvor odluke) radi se sledeće:
  + Određuje se greška koju trenutno stablo pravi na skupu podataka za orezivanje
  + Čvor se proglašava za list, njegova deca se orezuju i za klasu čvora uzima se klasa primera kojih u datom čvoru ima najviše
  + Ispituje se kolika je greška koju stablo pravi na skupu podataka za orezivanje nakon orezivanja
  + Ako je greška nakon orezivanja manja nego pre orezivanja onda se orezivanje zadržava, u suprotnom stablo se vraća na prethodno stanje

Ovakav vid orezivanja stabla obezbeđuje da rezultujuće stablo postiže minimalnu grešku na skupu podataka za orezivanje i da od svih stabala koje postižu ovu minimalnu grešku ovo stablo bude sa minimalnim brojem čvorova. Problem ovog načina orezivanja je to što je za njegovu implementaciju potreban poseban skup podataka za orezivanje.

### 3.4.4. Pesimistično orezivanje stabla (*Pessimistic Error Pruning*)

Jedna od navedenih mana orezivanja stabla za smanjenje greške je to što nam je za njegovu implementaciju potreban poseban skup podataka na kome će biti testirane performanse pre i posle orezivanja.

Pesimistično orezivanje stabla je postupak u kom možemo da procenimo performanse stabla pre i posle orezivanja na samom skupu za trening, bez potrebe da uvodimo poseban skup podataka. Međutim, korišćenjem trening seta za ispitivanje performansi na uobičajen način do orezivanja nikada ne bi došlo jer se treningom maksimizuju performanse uravo na datom trening skupu. U svom radu (dodati rad) (dodati autora) tvrdi kako je performansa stabla na trening setu previše optimistična zbog toga što je stablo videlo date primere u toku treninga. Samim tim ovu procenu je potrebno učiniti *pesimističnijom.* U pesimističnom orezivanju ovo je postignuto dodavanjem korekcije kontinuiteta za binomnu raspodelu što dovodi do bolje procene greške na trening skupu.

Neka je za čvor **t** broj primera u datom čvoru predstavljen sa **N(t)** i neka je **e(t)** broj pogrešno klasifikovanih primera u datom čvoru. Tada je greška u datom čvoru data sa:

Kada se u ovu procenu uključi i korekcija kontinuiteta dobija se:

Shodno tome, greška u klasifikaciji za podstablo T definisano je sa:

gde suma po **i** ide po svim listovima u datom podstablu**,** pri čemu je ∑N(i) = N(t)

Pa će greška u klasifikaciji za podstablo T uz korekciju biti data sa:

gde je sa **Nt** označen broj listova u podstablu **T**

Kako je ∑N(i) = N(t) imenioci izraza za r’(t) i r’(T) biće isti pa je dovoljno upoređivati brojioce koji predstavljaju broj pogrešno klasifikovanih primera (sa korekcijom) označenih sa **n(t)**:

Iz ovih jednačina se vidi da kada orežemo stablo mi ćemo broju pogrešno klasifikovanih primera dodati ½ samo jednom, dok ćemo u slučaju kada imamo podstablo mi dodati ½ za svaki list. Dakle postojanje svakog lista se dodatno naplaćuje, ovim je obezbeđeno da se stablo oreže ukoliko postojanje datih listova nije obrazloženo preciznijom klasifikacijom.

Shodno sa prethodnim definicijama podstablo ne bi trebalo orezati ako je n’(T) < n(t) tj. ako podstablo ima bolje performanse od orezanog čvora. Međutim, i pored korišćenja korekcije, greška podstabla je i dalje optimistična u odnosu na grešku koju bismo dobili kada bismo stablo primenili na podatke koji nisu viđeni u treningu. Zato ćemo sačuvati podstablo samo ako bismo njegovim očuvanjem postigli performanse bolje za jednu standardnu grešku bolje u odnosu na orezani čvor. Tj. stablo nećemo orezati ako je **n’(T) + SE(n’(t)) < n(t)** pri čemu je standardna greška definisana sa:

I pored ove korekcije procena greške podstabla je i dalje blago optimistična u odnosu na stvarne rezultate koje bi stablo postiglo na do tad neviđenim primerima.

Za razliku od orezivanja stabla za smanjenje greške, ovaj algoritam kreće rezanje od vrha stabla ka dnu. Ovo ga čini mnogo bržim jer kada se oreže podstablo koje je bliže vrhu više neće biti ispitivanja njegovih podstabala. Ipak, iteriranje kroz sva podstabla je nekad korisno jer se ekspertima može ponuditi serija stabala od kojih bi oni izabrali ono koje ima najviše fizičkog smisla.

## 3.5. Nedostajući podaci

Problem sa nedostajućim podacima javlja se u svim sferama analize podataka i mašinskog učenja. U oblasti primene modela u medicini nedostatak podataka je znatno češća pojava zbog toga što u specifičnim situacijama nije moguće obaviti sva neophodna merenja zbog fizičkog stanja pacijenta ili je vremenski nemoguće obaviti merenje. Postoje razni načini da se nedostajuće vrednosti procene iz ostatka podataka, ali je u ovom radu primenjen drugačiji pristup. Kada se u odluke desi situacija da nedostaje vrednost atributa u odnosu na koji se vrši podela u grane, put kroz stablo se nastavlja na sledeći način: grana za nastavak puta bira se nasumično, tako da grane koje sadrže veći broj primera imaju veću verovatnoću da budu odabrane.

Posmatrajmo sledeću sliku:

Diagram

Description automatically generated

Slika 8. Primer podstabla

Pretpostavimo da pokušavamo da odredimo klasu primera koji je došao do podstabla na slici 8. Neka za taj primer nedostaje informacija o indeksu telesne mase (*body weight mass – mbi).* Čvor **bmi** ima 3 sina sa ukupno 26+44+55=125 potomka.Verovatnoća nastavka puta do levog sina je 26/125, za srednjeg sina 44/125i za desnog sina 55/125.

# 4 Implementacija

Implementacija stabla odlučivanja rađena je u programskom jeziku *Python*. Implementacija je rađena „od nule“ pri čemu su jedino korišćene: biblioteka za analizu baza podataka *pandas* iz koje je iskorišćena funkcija za učitavanje fajla u *.csv* formatu i biblioteka *numpy* iz koje su korišćene funkcije za obradu nizova i matrica.

Po funkcionalnosti, implementacija je podeljena u dve klase:

* klasa ***DecisionTree*** koja predstavlja klasu stabla odlučivanja koju korisnik koristi
* klasa ***Node*** koja predstavlja klasu jednog čvora u stablu odlučivanja i čija je funkcionalnost skrivena od korisnika

U daljim poglavljima biće objašnjena funkcionalnost ove dve klase.

## 4.1. Klasa *DecisionTree*

Ova klasa predstavlja klasu stabla odlučivanja koje se daje korisniku na korišćenje. Kroz ovu klasu korisnik bira parametre koji se tiču detalja funkcionisanja stabla, može da vrši trening stabla, upit stabla za klasifikaciju novih podataka i vizuelizaciju stabla.

Samo stablo odlučivanja predstavljeno je u ovoj klasi pomoću reference na koren stabla koje je tipa klase *Node.* Svi upiti, treninzi i modifikacije vezane za stablo odrađene su preko ove reference.

U narednim poglavljima detaljnije su opisani parametri ove klase, njene metode i atributi.

### 4.1.1. Parametri klase *DecisionTree*

Kroz parametre klase *DecisionTree* korisnik definiše detalje funkcionalnosti stabla odlučivanja koji se tiču načina podele čvora na grane, metode diskretizacije kontinualnih vrednosti i vrste orezivanja stabla. U datom delu dat je kompletan opis parametara sa njihovim opisima:

* **impurity\_funciton**: definiše funkciju nečistoće koja će biti korišćena prilikom grananja čvora. Moguće vrednosti su
  + „*gini*“ – koristi se *Gini Impurity*
  + „*entropy*“ – koristi se *entropija*
  + „*chi*“ – koristi se *Hi-kvadrat test*
  + „*classification\_error*“ – koristi se greška klasifikacije
* **discretization\_method**: definiše način na koji će odlike sa kontinualnim vrednostima biti diskretizovane. Moguće vrednosti:
  + „*same\_length*“ – koristi se podela na skupove jednakih veličina
  + „*same\_size*“ – koristi se podela na skupove sa jednakim brojem primera u njima
  + „*dynamic*“ – koristi se dinamička podela na skupove
* **pruning\_method**: definiše način na koji će stablo biti orezano nakon treninga Moguće vrednost:
  + „*reduced\_error*“ – koristi se orezivanje za smanjenje greške
  + „*pessimistic\_error*“ – koristi se pesimističko orezivanje stabla
* max\_depth: definiše maksimalnu dubinu stabla. Na ovoj dubini čvorovi se više ne granaju i klasa čvora postaje klasa koja je predstavljena najvećim brojem primera u tom čvoru
* **min\_leaf\_size**: minimalan broj primera u čvoru za koji je dozvoljeno grananje. Čvorovi sa manjim brojem primera od ovog se više ne granaju i klasa čvora postaje klasa koja je predstavljena najvećim brojem primera u tom čvoru
* **number\_of\_discr\_classes**: broj klasa na koje će kontinualni atributi biti podeljeni

### 4.1.2. Metode klase *DecisionTree*

* **train\_tree(features, labels)**: ova metoda koristi se za treniranje stabla odluke i ovu metodu je neophodno pozvati pre svih ostalih metoda. Ova metoda kao argumente prima odlike (***features***) i labele (***labels***) u obliku *pandas* tabela.
* **get\_predictions(features)**: ova metoda koristi se za klasifikaciju primera koji su dati u argumentu **features**
* **get\_error(predictions, labels)**: ova metoda vraća procenat greške u klasifikaciji između predikcija **predictions** i labela **labels**
* **display\_tree(path):** Čuva sliku istreniranog stabla odlučivanja na putanji datoj argumentom **path**

## 4.2. Klasa *Node*

Ova klasa predstavlja apstrakciju jednog čvora u stablu odlučivanja, pri čemu to može biti čvor odluke (grana se na druge čvorove) ili čvor list (ne grana se već sadrži oznaku klase). Postojanje ove klase sakriveno je od korisnika ali je njena implementacija ključna za rad samog stabla i s’ toga će arhitektura ove klase biti detaljnije predstavljena u nastavku.

Svaki čvor sadrži primere odlika i labela iz skupa podataka za trening. Čvoru koji predstavlja koren stabla dat je ceo skup podataka za trening, i pri svakom grananju čvoru odgovarajuće grane prosleđuju se samo primeri koji pripadaju toj grani.

Svaki čvor odluke ima definisan atribut u odnosu na koji vrši grananje, dok svaki čvor list ima definisanu klasu klasifikacije koju predstavlja. Kada u čvoru odluke dođe do grananja u njemu će se zapamtiti informacija o odgovarajućim granama u vidu rečnika. Ključ rečnika je u tom slučaju vrednost atributa u odnosu na koji se vrši grananje, a vrednost rečnika za dati ključ je čvor do kog dolazi data grana.

Svaki čvor ima informaciju o dubini na kojoj se nalazi kako bi se moglo ograničiti grananje po dubini parametrom *max\_depth.*

Čvorovi grananja koriste funkciju ***split\_by\_best\_feature*** za određivanje atributa koji će se koristiti za grananje tako da odgovarajuća metrika nečistoće bude minimalna i zatim grananje na odgovarajuće grane pri čemu se čvorovima u granama prosleđuju atributi samo onih primera sa odgovarajućim vrednostima datog atributa.

Čvorovi listovi koriste funkciju ***set\_class\_to\_most\_common\_label*** koja postavlja klasu datog čvora na klasu primera kojih u datom čvoru ima najviše.

Čvorovi oba tipa (grananja i listovi) koriste funkciju ***get\_node*** koja u slučaju čvora grananja vraća čvor u grani sa odgovarajućom vrednošću atributa, dok u slučaju čvora lista vraća klasu datog čvora.

# 5 Skup podataka

Izabrani skup podataka odnosi se na klasifikaciju bolesti srca i preuzet je sa linka <https://archive.ics.uci.edu/ml/datasets/heart+Disease>. Originalan skup podataka sadrži 75 atributa pri čemu se u naučnim radovima aktivno koristi 13 odabranih atributa. Radi lakšeg poređenja rezultata u ovom radu uzeće se u obzir samo vrednosti 13 odabranih atributa (odlika).

## 5.1 Opisi korišćenih odlika

Odlike koje će biti opisane u daljem delu mogu se podeliti u grupe na osnovu skupa vrednosti koje mogu da imaju:

* binarne odlike: jedino mogu imati vrednosti 0 ili 1 (npr. da li pacijent ima bolest srca (1) ili ne (0)
* nominalne odlike: mogu uzimati vrednosti iz diskretnom i konačnog skupa, pri čemu poredak tih vrednosti nije bitan (npr. tip bola u grudima koji pacijent oseća: 0 – tipičan, 1 – atipičan, 2 – nema bola, 3 – asimptomatičan)
* ordinalne odlike: takođe uzimaju vrednosti iz diskretnog i konačnog skupa ali je poredak vrednosti bitan (npr. ocena na ispitu: 5, 6, 7, 8, 9, 10)
* kontinualne odlike: uzimaju vrednosti iz kontinualnog skupa koji može ali ne mora biti ograničen

Odlike na osnovu kojih se donosi odluka su:

1. godine (kontinualna): broj godina pacijenta
2. pol (binarna): pol pacijenta: 1 – muški pol, 0 - ženski pol
3. cp (ordinalna): tip bola u grudima
   * + 1. tipičan bol
       2. netipičan bol
       3. bol grudnog koša
       4. asimptomatičan
4. trestbps (kontinualna): krvni pritisak tokom mirovanja
5. chol (kontinualna): nivo holesterola u mg/dl
6. fbs (binarna): nivo šećera pre jela: 0 – manji od 120 mg/dl, 1 – veći od 120 mg/dl
7. restecg (nominalna): EKG u stanju mirovanja:

* normalan
* abnormalnost ST-T talasa
* leva ventrikularna hipertrofija

1. thalach (kontinualna): maksimalni dostignuti broj otkucaja srca
2. exang (binarna): bol u grudina tokom napora: 0 – Ne, 1 - Da
3. oldpeak (kontinualna): razlika u ST vrednostima tokom napora i u mirovanju
4. slope (nominalna): nagib u ST segmentu tokom maksimalnog napora
   * + 1. nagib naviše
       2. ravan
       3. nagib naniže
5. ca (ordinalna): broj većih krvnih sudova vidljivih tokom fluoroskopije
6. thal (nominalna): thalassemia – poremećaj nivoa hemoglobina u krvi
   * + 1. nema poremećaja
       2. trajni poremećaj
       3. izlečiv poremećaj

## 5.2. Karakteristike korišćenih odlika

Dati skup podataka sadrži vrednosti ovih atributa za 297 pacijenata, pri čemu nema nedostajućih vrednosti ni za jedan atribut. Od 297 pacijenata 160 (54%) njih nije ispoljilo bolesti srca, dok je 137 (46%) ispoljilo bolest srca. Shodno ovome skup podataka je balansiran u odnosu na ispoljavanje bolesti srca.

Chart, bar chart

Description automatically generated

Slika 9. Podela pacijenata u odnosu na prisustvo bolesti srca

U sledećem delu prikazane srednje vrednosti i standardne devijacije kontinualnih atributa podeljenih po vrednosti labele (bolesti srca).

Chart

Description automatically generatedChart

Description automatically generated

Chart, box and whisker chart

Description automatically generatedChart

Description automatically generated

Chart, box and whisker chart

Description automatically generated

Slika 10. Predstavljanje srednje vrednosti i standardne devijacije atributa u zavisnosti od klase

U sledećem delu biće prikazani histogrami atributa.

Chart, bar chart

Description automatically generated Chart, histogram

Description automatically generatedChart, bar chart

Description automatically generated Chart, histogram

Description automatically generatedChart, histogram

Description automatically generated Chart, histogram

Description automatically generatedChart, bar chart, histogram

Description automatically generated Chart, bar chart, histogram

Description automatically generatedChart, histogram

Description automatically generated Chart, histogram

Description automatically generatedChart, histogram

Description automatically generated Chart, bar chart

Description automatically generatedChart, bar chart

Description automatically generated

Slika 11. Histogrami atributa u zavisnosti od klase

Na narednoj slici prikazana je tabela koja prikazuje koeficijente korelacije između atributa. Tamnijom bojom obojene su ćelije u preseku dva atributa koja su jako korelisana dok su slabijom bojom obojene ćelije u preseku dva atributa koja su slabo korelisana. Nijansom plave boje označena je negativna korelacija, dok se crvena boja koristi da označi pozitivnu korelaciju.

A screenshot of a computer

Description automatically generated with medium confidence

Slika 12. Koeficijenti korelacije između atributa

# 5 Rezultati

# 6 Diskusija i zaključak

# Literatura