Univerzitet u Beogradu

Matematički fakultet

Seminarski rad

Predviđanje toksičnosti molekula dizajniranih za funkcionalni domen CRY1 proteina

Mentor:
Prof. Dr. Nenad Mitić
Katedra za računarstvo i informatiku

Student:
Vasilije Todorović
Broj indeksa: 25/2019
Smer: Informatika

Datum: Jul 2023

Sadržaj

1 Uvod

U farmakologiji, proučavanje proteina i molekula koji igraju ključne uloge u biološkim procesima od suštinskog je značaja za razvoj novih lekova i terapija. Jedan od takvih proteina je CRY1 (Cryptochrome 1), koji je odgovoran za generisanje cirkadijalnog ritma, odnosno dnevnih ritmova koji su uslovljeni smenjivanjem dana i noći u biološkom časovniku čoveka.

Protein CRY1 pripada porodici Cryptochrome i igra ključnu ulogu u regulaciji biološkog sata kod sisara. On funkcioniše kao fotosenzitivan protein koji reaguje na svetlost, a posebno na svetlo plave boje. Aktivacija proteina CRY1 pod uticajem svetlosti pokreće biološki sat i omogućava organizmu da sinhronizuje svoje unutrašnje ritmove sa spoljnim svetlosnim ciklusima.

U cilju razumevanja i kontrolisanja ovog biološkog sistema, istraživači su dizajnirali različite molekule sa potencijalnim ulogama u interakciji sa proteinom CRY1. Ovi molekuli su konstruisani kako bi modifikovali aktivnost proteina CRY1 i tako uticali na biološki sat. Ipak, pre nego što se takvi molekuli mogu koristiti u razvoju novih lekova, neophodno je pažljivo proceniti njihovu toksičnost i bezbednost.

Upravo u cilju proučavanja toksičnosti ovih molekula dizajniranih za funkcionalni domen CRY1 proteina, ovaj rad fokusira se na primenu različitih algoritama mašinskog učenja. Pored algoritama mašinskog učenja, ovaj rad takođe za cilj ima da objasni i pokaže čitav proces prilikom pravljenja jednog modela mašinskog učenja. Kroz analizu podataka o tim molekulima i njihovoj klasifikaciji kao toksičnih ili netoksičnih, očekuje se da će se unaprediti razumevanje njihovih potencijalnih efekata na biološki časovnik.

2 Priprema i podela podataka

Podaci koji su korišćeni u ovom radu potiču sa sajta "https://archive-beta.ics.uci.edu/dataset/728/toxicity-2". Ovaj sajt je arhiva podataka Univerziteta u Kaliforniji, Irvine (UCI) i pruža pristup raznovrsnim skupovima podataka za naučne svrhe. Skup podataka "toxicity-2" sadrži informacije o 171 molekulu dizajniranih za funkcionalni domen proteina CRY1, koji je odgovoran za generisanje cirkadijalnog ritma kod ljudi. Od ovih molekula, 56 je označeno kao toksično, dok je 115 označeno kao netoksično.

Svaki molekul opisan je uz pomoć 1203 atributa koji su raznoliki i uključuju strukturne, fizičko-hemijske i topološke karakteristike. Neki od atributa uključuju informacije o atomima i vezi u molekulu, elektronskim i prostornim osobinama, kao i karakteristikama površine molekula.

2.1 Uklanjanje nedostajućih vrednosti i duplikata

Priprema podataka igra ključnu ulogu u postizanju uspešnih rezultata u analizi podataka i pravljenju modela mašinskog učenja. Jedan od najvažnijih koraka u ovom procesu je **uklanjanje nedostajućih vrednosti i duplikata** iz skupova podataka. Ovi koraci nam obezbeđuju pouzdaniju analizu i izgradnju modela tako što smanjuju potencijalne greške koje mogu dovesti do pogrešnih zaključaka.

Nedostajuće vrednosti su često neizbežan deo stvarnih skupova podataka. Međutim, rad sa nedostajućim podacima može biti problematičan, jer mnoge statističke metode i algoritmi ne mogu pravilno funkcionisati kada nedostaju ključni podaci. Iz tog razloga, uklanjanje nedostajućih vrednosti je od suštinske važnosti.

Izbacivanjem redova ili kolona koje sadrže nedostajuće vrednosti, osiguravamo da analize budu bazirane na potpunim informacijama. Ovo ne samo da povećava tačnost rezultata, već takođe smanjuje rizik da modeli budu trenirani ili ocenjeni na osnovu nepotpunih ili nepouzdanih podataka.

Duplirani podaci mogu takođe narušiti tačnost analize podataka i modela. Bez njihovog uklanjanja, isti podaci bi se mogli tretirati kao potpuno nezavisni entiteti, što bi dovelo do nepreciznih rezultata. Pored toga, ukoliko je model treniran na takvim dupliranim podacima, on može ispoljiti preteranu sklonost ka ovim podacima čime se smanjuje sposobnost generalizacije na nove podatke, što predstavlja cilj svakog modela mašinskog učenja.

Kraćom proverom se može ustanoviti da preuzeti podaci koji se koriste u ovom radu u sebi *ne sadrže* nedostajuće vrednosti niti duplirane podatke, čime je obezbeđena osnova za naredne korake pravljenja modela.

2.2 Podela na trening i test podatke

Još jedan od ključnih koraka u procesu razvoja modela mašinskog učenja jeste podela podataka na trening i test skup. Ovaj korak omogućava nam da obučimo model na jednom skupu podataka (*trening skup*) i ocenimo njegove performanse na nezavisnom skupu podataka (*test skup*), čime možemo da steknemo uvid u sposobnost modela da generalizuje nad novim, nepoznatim podacima.

Trening skup podataka se koristi kako bi model naučio šablone i zavisnosti unutar podataka. Svaki element u trening skupu se sastoji od ulaza (atributa) i ciljne promenljive (klase). Tokom treniranja, model prilagođava svoje parametre kako bi minimizovao grešku između predviđenih vrednosti i stvarnih vrednosti ciljne promenljive.

Test skup podataka je nezavisan skup podataka koji se koristi kako bi se procenile performanse modela nakon obučavanja (treniranja). Kako model nije obučavan na test podacima, procena performansi na ovom skupu pruža nam sliku o tome kako će se model ponašati na stvarnim, nepoznatim podacima. Model se koristi da bi se predviđale vrednosti za ulaze iz test skupa, a zatim se te predviđene vrednosti upoređuju sa stvarnim ciljnim vrednostima. Ova procena omogućava nam da dobijemo realan uvid u performanse modela i da identifikujemo potencijalne probleme kao što su preprilagođavanje ili nedovoljna sposobnost generalizacije.

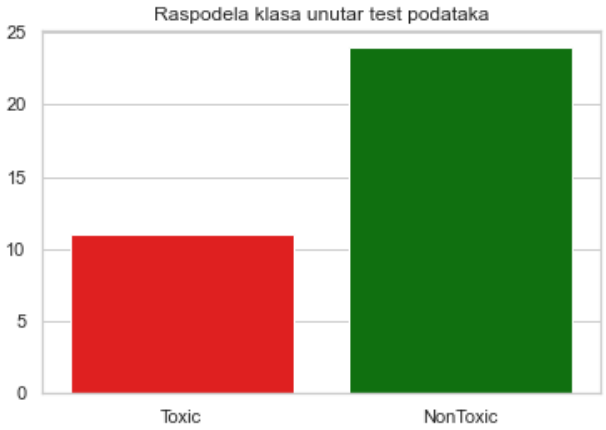
Funkcija *train_ test_ split* u programskom jeziku Python, deo biblioteke scikit-learn, koristi se za podelu skupa podataka na trening i test skupove. Ova funkcija prihvata nekoliko važnih parametara: X - skup ulaznih podataka, y - ciljna promenljiva podataka, *test_ size* - veličina test skupa u odnosu na celokupan skup podataka, *random_ state* - vrednost koja se koristi za postavljanje semena generatora slučajnih brojeva, što osigurava dosledno deljenje podataka svaki put kada se programski kod pokrene, *stratify* - parametar koji omogućava da raspodela klasa ciljne promenljive podataka bude očuvana i unutar trening i test podataka.

Pomoću pomenute funkcije izvršavamo podelu podetaka na trening i test skup. Za parametar X odabrani su svi redovi iz tabele ali je izvršena projekcija na sve atribute sem poslednjeg, dok su za y odabrani svi redovi ali je izvršena projekcija na poslednji atribut koji predstavlja ciljnu promenljivu, *test_ size* je postavljen na 0.2 što znači da će se 20% ukupnih podataka naći u test dok će se preostalih 80% naći u trening skupu, *random_ state* je postavljen na 42, a *stratify* je postavljen na y, kako bi se očuvala raspodela klasa u y.

Kao povratnu vrednost ove funkcije dobijamo 4 promenljive: *X_ train* - ulazni trening podaci, *X_ test* - ulazni test podaci, *y_ train* - trening vrednosti ciljne promenljive i *y_ test* - test vrednosti ciljne promenljive. Ukupan broj podataka koji se nalaze u trening skupu je 136, a u test skupu 35, što zapravo i odgovara postavljenom uslovu da je njihov odnos 4:1. Još jedan od uslova postavljenih unutar funkcije jeste da je raspodela klasa unutar trening skupa ista kao i unutar test skupa. Unutar trening skupa, 45 podataka pripada klasi *Toxic* a preostalih 91 pripada *NonToxic*, dok unutar test skupa, 11 podataka pripada klasi *Toxic* a preostalih 24 pripada klasi *NonToxic*. Da je raspodela očuvana može se zaključiti i na

osnovu sledeća dva histograma.

(a) Trening skup



(b) Test skup

Figure 1: Raspodela klasa

3 Izbor atributa - Feature Selection

Važnost izbora atributa (*feature selection*) dobija poseban značaj u ovom radu jer se bavi analizom toksičnosti molekula sa čak 1203 atributa. Kako bismo dublje razumeli zašto je ovaj korak od suštinskog značaja, potrebno je sagledati izazove koje donosi visok broj atributa.

Sa velikim brojem atributa dolazi i problem dimenzionalnosti. Složenost modela raste sa brojem atributa, što čini upravljanje i analizu takvih skupova podataka prilično zahtevnim. Prevelika složenost modela može dovesti do preprilagođavanja, što uzrokuje da model "zapamti" podatke umesto da iz njih nauči određene zavisnosti, čime se smanjuje njegova sposobnost generalizacije nad novim podacima.

Pažljivim izborom atributa, ne samo da se postiže smanjenje dimenzionalnosti i izbegava preprilagođavanje već se model čini bržim i efikasnijim, čime se ubrzava proces treniranja modela. Takođe, važno je napomenuti da se pažljivim izborom atributa omogućava i dublje razumevanje podataka. Odabirom najbitnih atributa se obezbeđuje fokus na ključne karakteristike koje utiču na ciljnu promenljivu.

Postoji više algoritama i pristupa za izbor atributa, a ovi algoritmi se često grupišu u tri osnovne metode: *filter metode*, *wrapper metode* i *embedded metode*. Svaka od ovih metoda ima svoje karakteristike i prednosti, i odabir odgovarajuće metode zavisi od specifičnih zahteva analize i prirode podataka.

3.1 Filter metode

U okviru oblasti izbora atributa, filter metode predstavljaju početnu fazu u procesu sužavanja skupa atributa pre nego što se primene složeniji algoritmi. Ove metode su brze, nezavisne od modela mašinskog učenja i pružaju osnovni uvid u važnost pojedinačnih atributa za ciljnu promenljivu. Filter metode se obično sastoje iz sledećih koraka:

- **Merenje značajnosti atributa**: Proces u kojem se svakom atributu dodeljuje ocena koliki uticaj ima na ciljnu promenljivu. Svaki atribut se posmatra nezavisno, bez uzimanja u obzir međusobne interakcije između više atributa. Ocena značajnosti se određuje na osnovu različitih metrika. Na primer, mogu se koristiti statistički testovi kao što su ANOVA, χ^2 test ili korelacija.
- **Rangiranje atributa**: Nakon što se atributi ocene po značajnosti, oni se rangiraju pri čemu su najvažniji atributi na vrhu liste, a manje važni atributi su niže.
- **Odabir atributa**: Na osnovu rangiranja, određeni broj atributa se bira za dalju upotrebu. Ovde se može primeniti prag značajnosti, čime se određuje koliko atributa želimo zadržati.

3.1.1 Prednosti i mane filter metoda

Budući da se u ovom radu koriste filter metode, potrebno je istaći motivaciju za njihovo korišćenje, odnosno koje prednosti i mane takve metode nose sa sobom.

Pored brojih prednosti koje dele sa ostalim metodama izbora atributa, važno je istaći da su filter metode izuzetno efikasnije. Efikasnost ovih metoda proističe iz toga što se atributi ocenjuju nezavisno od modela učenja koji će biti korišćen. Takođe, pošto ne zavise od modela učenja, filter metode se smatraju metodama koje su manje sklone preprilagođavanju.

Filter metode imaju i nekoliko ograničenja. Prvo, ove metode ne uzimaju u obzir potencijalne međusobne veze između atributa, što može zanemariti važne informacije za bolje modeliranje. Takođe, filtriranje se vrši koristeći unapred definisane mere, što može propustiti neprimente i složene odnose među atributima. Osim toga, iako filter metode pružaju osnovnu selekciju atributa, za dublju analizu ili modeliranje često je potrebno kombinovati ih s drugim metodama izbora atributa kako bi se osigurala sveobuhvatnija i preciznija analiza podataka.

Iz ovih razloga, u našem pristupu koristimo filter metode za smanjenje broja atributa sa 1203 na manji, relevantniji skup. Nakon toga, koristićemo dodatne, naprednije metode izbora atributa, što će omogućiti da efikasno i precizno identifikujemo ključne atribute. Dve filter metode koje ćemo primeniti u ovom radu su VarianceThreshold i SelectKBest.

3.1.2 Prag varijanse - VarianceThreshold

Metoda VarianceThreshold je filter metoda za izbor atributa koja se koristi kako bi se odbacili atributi sa niskom varijansom u skupu podataka. Ako atribut ima veoma malu varijansu, to znači da je skoro konstantan ili se retko menja između instanci, što može ukazivati da taj atribut nosi malo informacija. Postupak primene metode VarianceThreshold je sledeći:

1. **Izračunavanje varijanse atributa**: Prvo se izračunava varijansa za svaki atribut tako što se računa srednja vrednost i razlika između svake vrednosti atributa i te srednje vrednosti, te se te razlike kvadriraju i srednja vrednost tih kvadrata se koristi kao mera varijanse.

2. **Postavljanje praga**: Nakon izračunavanja varijansi za sve atribute, postavlja se prag ispod kojeg se smatra da atribut ima nisku varijansu. Atributi čija je varijansa manja ili jednaka ovom pragu mogu se označiti pogodnim za odbacivanje.

3. **Odbacivanje atributa**: Atributi koji imaju nisku varijansu se odbacuju iz skupa atributa. Ovo može značiti da se uklanjaju atributi koji ne donose mnogo informacija ili variraju vrlo malo.

Kako bismo odredili prag varijanse, mozemo varirati kroz određen skup pragova i odabrati onaj koji nudi najbolju ravnotežu između smanjenja dimenzionalnosti i zadržavanja informacija o atributima. Na sledećem grafiku možemo videti kako se broj atributa menja

u zavisnosti od praga varijanse. Iz pomenutih razloga, za prag je odabrana vrednost 0.2. Nakon ove metode broj atributa je sa 1203 smanjen na 624.

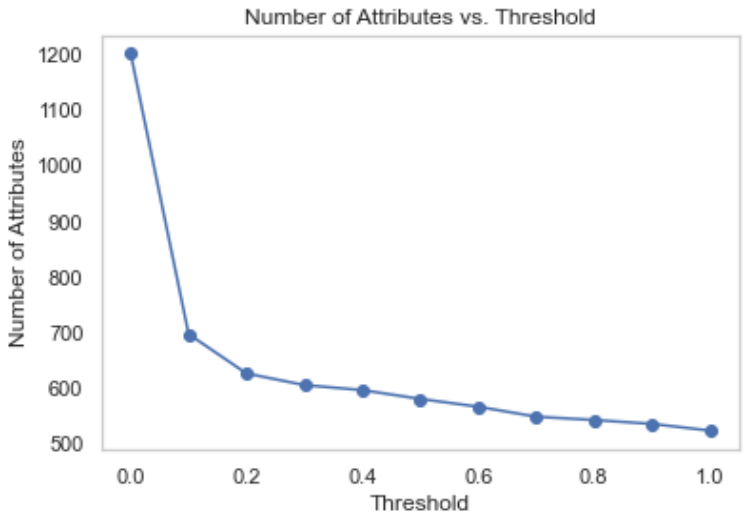


Figure 2: Uticaj praga na broj atributa nakon VarianceThreshold

3.1.3 Odabir K najboljih atributa - SelectKBest

Uzimajući u obzir da smo već primenili VarianceThreshold kako bismo izbacili atribute sa niskom varijansom i uklonili konstantne atribute, sada ćemo razmotriti SelectKBest kako bismo dalje suzili skup atributa na osnovu njihove značajnosti.

SelectKBest je metoda koja se fokusira na odabir najboljih K atributa iz datog skupa. Ideja je da se rangiraju atributi na osnovu neke metrike značajnosti. Metrika značajnosti koja je korišćena u ovom radu je F-test. Uz pomoć njega, svakom atributu pridružujemo dve vrednosti, F-statistiku i p-vrednost. Nulta hipoteza F-testa je pretpostavka da atribut nema statistički značajan uticaj na ciljnu promenljivu. Ukoliko je p-vrednost atributa dovoljno niska (obično se za prag postavlja vrednost 0.05) tada nultu hipotezu odbacujemo i atribut smatramo značajnim. F-vrednost nam pomaže da atribute rangiramo po značaju na ciljnu promenljivu, odnosno ukoliko je visoka vrednost F-statistike to znači da je i visok značaj veze tog atributa sa ciljnom promenljivom.

Bitno je napomenuti da se SelectKBest primenjuje kao korak pripreme podataka i da će rezultujući skup atributa biti ulaz za RFE (Recursive Feature Elimination), obzirom da ne radi efikasno za veliki broj atributa. RFE je metoda omotača koja će dodatno optimizovati skup atributa tako što će rekurzivno odbacivati najmanje značajne atribute, na osnovu performansi modela. Kombinacija ove dve metode omogućava nam da postignemo balans između smanjenja dimenzionalnosti i zadržavanja značajnih atributa za performanse

modela.

Parametar K za metodu SelectKBest ćemo odrediti eksperimentalno u zavinosti od rezultata koje daje RFE. Naime, parametar K želimo da odaberemo i ne izgubimo informacije o atributima a da opet rešimo problem dimenzionalnosti kako bi RFE metoda radila efikasno. Ukoliko odaberemo malu vrednost parametra K, možemo potencijalno izgubiti atribute koji posmatrani nezavisno možda nemaju veliki značaj za ciljnu promenljivu, ali je moguće vezom više takvih atributa doneti zaključke o nekim zakonitostima koje važe među podacima. Takve veze mogu biti pronađene pomoću RFE metode. S druge strane, velika vrednost parametra K može dovesti do velike neefikasnosti rada metode RFE. Kao vrednosti koje će biti odabrane za parametar K, odlučujemo se za 300, 250, 200, 150, 100 i 50. Na osnovu rezultata metode RFE odlučujemo se za tačno jednu vrednost od ponuđenih.

3.2 Metode omotača - Wrapper metode

Wrapper metode su napredne tehnike izbora atributa koje koriste sam model mašinsog učenja kao sredstvo za procenu i odabir atributa, umesto da koriste statističke metrike ili nezavisne testove kao filter metode. Ove metode koriste performanse modela kao merilo za izbor atributa, ali je važno napomenuti da taj model može biti potpuno nezavisan od modela koji će se konačno koristiti za klasifikaciju. Ovo omogućava veću fleksibilnost i objektivnost u odabiru atributa, jer se ne vezuje za specifičan model i njegove pretpostavke. Wrapper metode se sastoje iz sledećih koraka:

- **Generisanje podskupova atributa**: Prvo se generišu različiti podskupovi atributa. Ovo može biti svaki mogući skup atributa, ali obično se koriste heuristike kako bi se smanjila vremenska složenost.
- **Treniranje i ocena modela**: Za svaki podskup atributa, model se trenira i ocenjuje na osnovu nekog unapred definisanog metričkog sistema. To mogu na primer biti tačnost, preciznost, odziv i ostale relevantne metrike.
- **Izbor najboljeg skupa atributa**: Metoda bira skup atributa koji je dao najbolje performanse modela na validacionom skupu podataka.

3.2.1 Prednosti i mane wrapper metoda

Motivacija za korišćenje wrapper metoda predstavlja zapravo njihova glavna prednost u odnosu na filter metode, a to je uzimanje u obzir interakcije između atributa i modela. Uzimajući u obzir ove kompleksne interakcije, može se doći do preciznijih rezultata, visoke tačnosti i visokih performansi modela. Kod filter metoda su posmatrane veze između pojedinačnih atributa iz ciljne promenljive dok se kod wrapper metodama mogu posmatrati i veze više atributa sa ciljnom promenljivom, kako bi se uočile neke kompleksnije zavisnosti među podacima.

Budući da wrapper metode zahtevaju treniranje i ocenu modela za svaki podskup atributa ovo može biti izuzetno vremenski zahtevno, posebno za velike skupove atributa. Iz tog

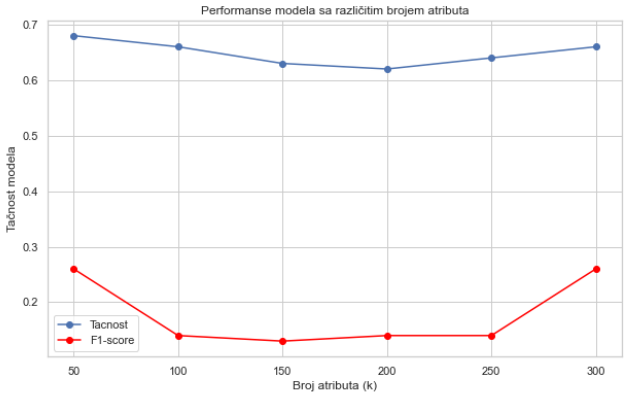
razloga smo prethodno koristili filter metode ali i dodatno pomaže u izbegavanju potencijalnih problema preprilagođavanja koji se može pojaviti kada wrapper metode rade sa velikim brojem atributa. Metoda koja je već ranije pomenuta, a koju ćemo koristiti u ovom radu jeste RFE (Recursive Feature Elimination)

3.2.2 RFE - Recursive Feature Elimination

Rekurzivna eliminacija atributa je jedna od tehnika wrapper metoda kojom se identifikuju najznačajniji atributi u skupu podataka. RFE često koristi određeni model za procenu značajnosti atributa i njihovo rangiranje. Kada se RFE primenjuje sa nasumičnom šumom (Random Forest) kao modelom, sledeći koraci su tipični:

1. **Inicijalno treniranje modela**:Nasumična šuma se trenira na celokupnom skupu atributa kako bi se procenili njihovi koeficijent značajnosti.
2. **Izbacivanje atributa**: Nakon inicijalnog treniranja, atribut sa najmanjim značajem se izbacuje iz skupa atributa. Model se zatim ponovo trenira na preostalim atributima.
3. **Ponavljanje koraka**: Nakon izbacivanja atributa, model se ponovo trenira na preostalim atributima, a postupak se ponavlja iterativno. U svakom koraku se izbacuje po jedan atribut.
4. **Ocena performansi**: Nakon svakog koraka izbacivanja, mere se performanse modela. To se može raditi na osnovu tačnosti, preciznosti, odziva ili drugih relevantnih metrika. Ovo pomaže da se prati kako se performansa modela menja sa svakim izbačenim atributom.
5. **Završetak procesa**: Postupak se ponavlja sve dok se ne dostigne željeni broj atributa ili dok performansa modela ne prestane značajno da se poboljšava.

Kao što je ranije pomenuto, u ovom radu ćemo prvo primeniti SelectKBest sa različitim vrednostima za K, a zatim na svakom skupu od K atributa primeniti RFE i pratiti kako se performanse menjaju, što možemo videti na sledećem grafiku.



Iako su tačnost i F1-skor modela za broj atributa 50 i 300 gotovo izjednačene, odlučeno je da se favorizuje model sa 300 atributa. Razlog ove odluke leži u pretpostavci da metoda RFE, usled svog prirodnog iterativnog pristupa, može očuvati dublje i suptilnije veze među atributima, koje bi mogle biti izgubljene u procesu odabira atributa metodom SelectKBest. Važno je napomenuti da se posmatrani F1 skor odnosi na klasu *Toxic*, obzirom da je F1 skor koji se odnosi na klasu *NonToxic* za različite vrednosti parametra K ujednačen i iznosi ≈ 0.78.

3.3 Konačan odabir atributa

Nakon opisanog postupka, odabrano je sledećih 15 atributa:

1. **MDEC-23**: Molekulski deskriptor baziran na strukturi, koji se koristi za kvantifikaciju elektronske distribucije u molekulu.
2. **ATSC8i**: Autokorelacija topološkog indeksa osme vrste, koji meri udaljenost između atoma i njegovih 8 suseda u molekulu.
3. **AATSC3v**: Autokorelacija topološkog indeksa treće vrste, koja se odnosi na broj atoma treće vrste koji su udaljeni tačno tri veze od datog atoma.
4. **SpDiam_Dt**: Spektralni dijametar matriksa udaljenosti, što je numerička mera maksimalne udaljenosti između atoma u molekulu na osnovu spektralnih svojstava matrice.
5. **ATS1m**: Autokorelacija topološkog indeksa za atome u molekulu. Ovaj indeks meri udaljenost između atoma i njegovih suseda prvog reda.
6. **ATSC3v**: Autokorelacija topološkog indeksa treće vrste, koja se odnosi na udaljenost između atoma i njegovih suseda treće vrste.
7. **ATSC5s**: Autokorelacija topološkog indeksa pete vrste, koja meri udaljenost između atoma i njegovih suseda petog reda.
8. **SHaaCH**: Broj kratkih H-H vodoničnih veza u molekulu.
9. **SaaCH**: Broj kratkih veza između S (sumpor) i C (ugljenik) atoma, uključujući vodonične veze.
10. **SwHBa**: Broj kratkih veza između S (sumpor) i H20 (voda) molekula, uključujući vodonične veze.
11. **ATSC5v**: Autokorelacija topološkog indeksa pete vrste, koja meri udaljenost između atoma i njegovih suseda petog reda.
12. **SpMax_Dt**: Spektralni maksimum matriksa udaljenosti, što je numerička mera maksimalne udaljenosti između atoma u molekulu na osnovu spektralnih svojstava matrice.

13. **ATSC5m**: Autokorelacija topološkog indeksa pete vrste, koja meri udaljenost između atoma i njegovih suseda petog reda.

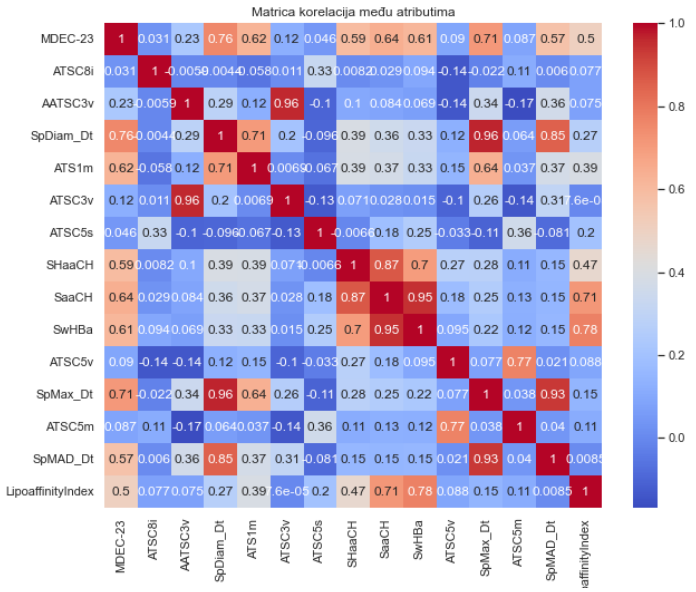
14. **SpMAD_Dt**: Spektralni srednji apsolutni odstup matriksa udaljenosti, što je numerička mera prosečnog apsolutnog odstupa između atoma u molekulu na osnovu spektralnih svojstava matrice.

15. **LipoaffinityIndex**: Indeks lipoafiniteta koji se koristi za procenu afiniteta molekula prema lipofilnoj sredini.

3.3.1 Određivanje korelacija između atributa

Kako bismo bili sigurni da nijedan od ovih atributa nije suvišan ili redundantan, bitno je detaljnije ispitati njihove korelacije.

Korelacije između atributa nam pružaju uvid u međusobne veze i zavisnosti između različitih atributa. Visoka korelacija između dva atributa može ukazivati na to da jedan atribut može pružiti slične informacije kao i drugi, što može dovesti do prekomerne složenosti modela. Korelacije koje važe između odabranih 15 atributa se mogu videti na sledećoj slici koja predstavlja matricu korelacija.

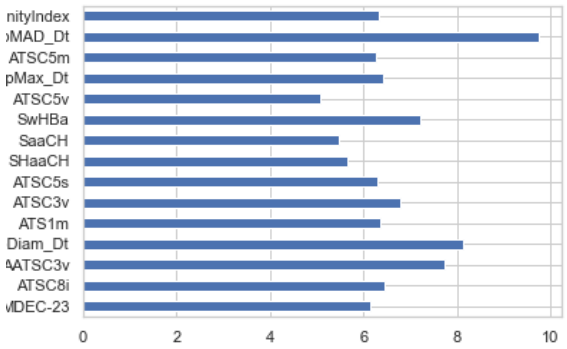


Da bismo odlučili koje atribute da izbacimo na osnovu analize korelacija, koristićemo pristup koji će nam omogućiti da zadržimo najvažnije atribute, dok istovremeno izbegavamo visoku kolinearnost i redundantnost. Ovo su koraci koje ćemo preduzeti:

1. **Definisanje praga korelacije**: Prvi korak je da definišemo prag za visoku korelaciju između atributa.
2. **Identifikacija visoko korelisanih atributa**: Korišćenjem definisanog praga, identifikovaćemo sve parove atributa koji imaju visoku korelaciju.
3. **Analiza značajnosti atributa**: Nakon identifikacije visoko koreliranih atributa, razmotrićemo značajnost svakog od njih u kontekstu našeg modela. Ovo možemo postići analiziranjem koeficijenata značajnosti atributa unutar modela korišćenog za metodu RFE.
4. **Odabir atributa za izbacivanje**: Na osnovu analize značajnosti i korelacija, donosimo odluku o tome koje atribute treba izbaciti. Atribut koji ima manju značajnost ili koji je manje važan za model možemo izbaciti kako bismo smanjili kolinearnost i unapredili performanse modela.
5. **Ponovna ocena modela**: Nakon izbacivanja određenih atributa, ponovno treniramo model koristeći novi skup atributa i ocenjujemo njegove performanse. Ovo nam omogućava da utvrdimo da li je izbor atributa doveo do povećanja performansi modela.

Posmatrajući matricu korelacije, prag za visoku korelaciju između atributa postavljamo na 0.93. Na osnovu toga uočavamo sledeće visokokorelisane parove atributa: **AATSC3v** i **ATSC3v** sa korelacijom 0.96, **SpDiam_Dt** i **SpMax_Dt** sa korelacijom 0.96, **SaaCH** i **SwHBa** sa korelacijom 0.95 i **SpMax_Dt** i **SpMAD_Dt** sa korelacijom 0.93.

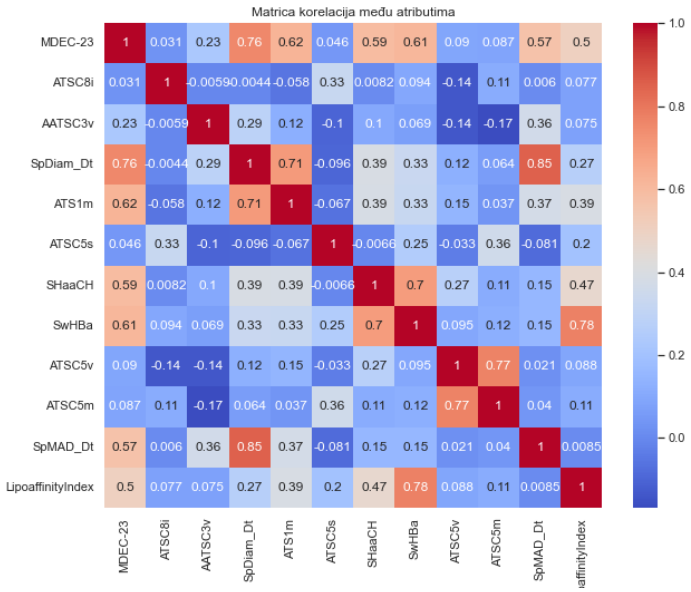
Na sledećem grafiku prikazani su koeficijenti značajnosti atributa dobijenih iz modela prilikom metode RFE. Ovo nam može pomoći pri odabiru koji atribut iz visokokorelisanog para treba razmatrati za odbacivanje.



Prvi atribut koji možemo izbaciti iz trenutnog skupa atributa jeste **SpMax_Dt**, obzirom da se nalazi u dva para visokokorelisanih atributa i u oba slučaja njegov koeficijent značajnosti je manji. Takođe, tumačeći značenje ovih atributa, stvara se jasna slika zbog čega je ovaj atribut izbačen. Ponovnom procenom modela dobijamo sledeće rezultate na test podacima: tačnost 0.66, F1-skor za klasu *Toxic* 0.27, F1-skor za *NonToxic* 0.8. Ovi rezultati su prihvatljivi i dopuštamo odbacivanje razmatranog atributa.

Sledeći atribut koji možemo eliminisati jeste **ATSC3v**, obzirom da je manje značajnosti od svog viskokorelisanog para i da se njegovim uklanjanjem tačnost modela na test podacima povećava (sa 0.66 na 0.69), dok F1-skor ostaje isti. Iz istih razloga možemo izbaciti i atribut **SaaCH**, čime smo ukupan broj atributa smanjili na 12.

Matrica korelacije nakon konačnog odabira atributa izgleda ovako:



4 SMOTE algoritam

SMOTE (Synthetic Minority Over-sampling Technique) je tehnika za balansiranje klasa u skupu podataka koja je posebno korisna kada postoji neravnoteža između različitih klasa. Ova tehnika se često primenjuje u problemima klasifikacije gde je jedna klasa značajno manje zastupljena od druge. Uvođenje ovog algoritma predstavlja važan korak u unapređenju performansi modela za klasifikaciju toksičnih molekula u okviru ovog rada, obzirom da smo već primetili nizak F1-skor za klasu *Toxic*.

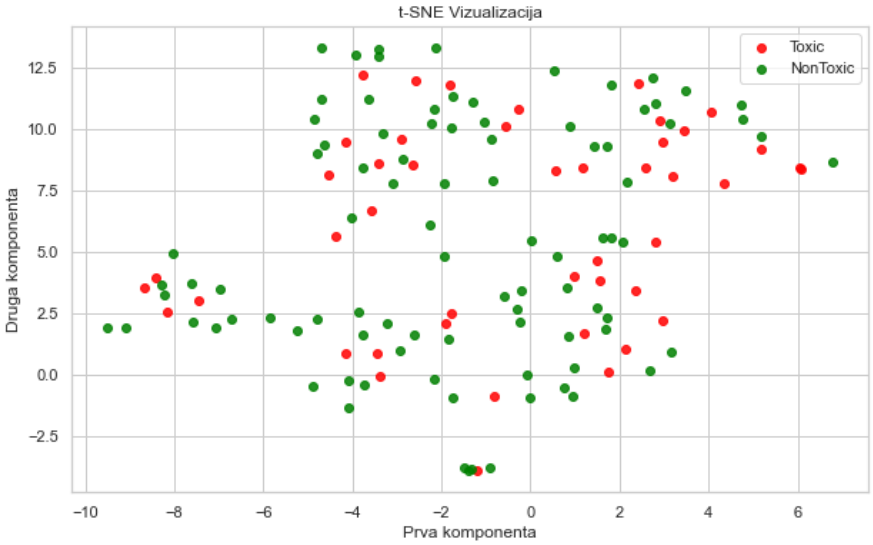
Osnovna ideja SMOTE algoritma je generisanje novih, sintetičkih uzoraka za manje zastupljenu klasu, tako da se poveća broj uzoraka i postigne bolja ravnoteža između klasa. Glavni koraci SMOTE algoritma su:

1. **Izbor referentnih uzoraka**: Za svaki uzorak iz manje zastupljene klase, SMOTE bira nekoliko najbližih suseda iz iste klase. Ovi susedi će se koristiti za generisanje sintetičkih uzoraka.
2. **Generisanje sintetičkih uzoraka**: SMOTE bira nasumično jednog od suseda koje je odabrao u prethodnom koraku. Zatim, za svaki atribut u uzorku, SMOTE bira nasumično tačku između uzorka i odabranog suseda. Na taj način se generiše novi sintetički uzorak.
3. **Ponavljanje procesa**: Ovaj proces generisanja sintetičkih uzoraka se ponavlja više puta za svaki uzorak iz manjinske klase kako bi se postigao željeni nivo balansiranja između klasa.

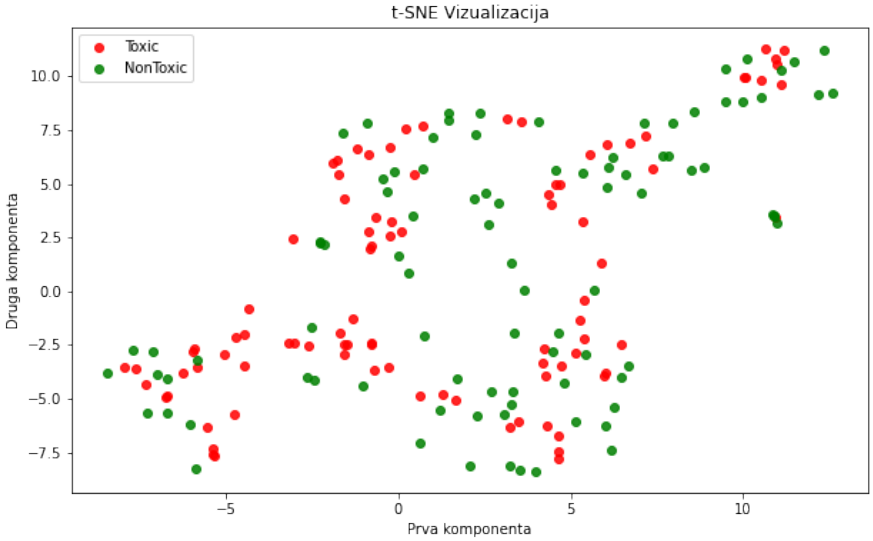
Izražena neravnoteža između broja toksičnih i netoksičnih molekula može uzrokovati da model neuspešno nauči karakteristike manje zastupljene klase. SMOTE algoritam popravlja ovu neravnotežu generisanjem novih, sintetičkih uzoraka toksičnih molekula. Na taj način, SMOTE pomaže da se poboljša sposobnost modela da prepozna šablone i karakteristike toksičnih molekula, čime se smanjuje tendencija modela da favorizuje netoksične molekule zbog njihove veće zastupljenosti.

Očekuje se da će uvođenje SMOTE algoritma rezultirati značajnim poboljšanjem performansi modela, naročito u pogledu tačnosti predviđanja toksičnih molekula. Povećanje F1-scora za klasu *Toxic* može ukazati na bolje usklađivanje modela sa stvarnim karakteristikama toksičnih molekula, čime se povećava njegova pouzdanost u proceni toksičnosti.

Nakon što smo izvršili SMOTE algoritam nad isključivo trening podacima, odnos klasa je ujednačen tako da im pripada po 91 instanca. Ovime smo završili proces pripreme podataka i sada modeli imaju veću mogućnost da nauče toksične molekule iz novih podataka. Na sledećim graficima su vizualizovani podaci pomoću t-SNE tehnike pre i posle upotrebe SMOTE algoritma.



(a) Pre SMOTE



(b) Nakon SMOTE

Figure 3: Vizualizacija podataka

5 Algoritmi klasifikacije

Algoritmi klasifikacije su ključni deo procesa mašinskog učenja, a njihov zadatak je da nauče kako da razvrstavaju podatke u različite klase na osnovu njihovih atributa. U kontekstu problema klasifikacije molekula kao *Toxic* ili *NonToxic*, istražujemo različite algoritme klasifikacije kako bismo utvrdili koji od njih najbolje odgovara našem zadatku. Naši podaci su već pripremljeni i uravnoteženi primenom SMOTE algoritma, što znači da smo rešili problem nebalansiranih klasa i da su podaci spremni za klasifikaciju.

Prilikom ocenjivanja performansi različitih algoritama klasifikacije, koristimo različite metrike kako bismo kvantifikovali njihovu efikasnost. Dve od najčešće korišćenih metrika su:

- **Tačnost (Accuracy):** Ova metrika meri ukupan procenat tačno klasifikovanih instanci u odnosu na ukupan broj instanci u testnom setu. Iako je tačnost važna metrika, ne daje potpunu sliku performansi, posebno u slučajevima sa nebalansiranim klasama.
- **F1-skor:** F1-skor je harmonijska sredina između preciznosti (precision) i odziva (recall). Preciznost meri koliko je pozitivnih instanci klasifikovano tačno, dok odziv meri koliko pozitivnih instanci je identifikovano. F1-skor je posebno koristan kada imamo problema sa nebalansiranim klasama, jer daje bolju procenu performansi modela.

Primenjujući ove metrike na sve modele možemo utvrditi koji daje najbolje rezultate i koji je najbolji za naše podatke.

5.1 Drvo odlučivanja

Drvo odlučivanja (Decision Tree) predstavlja jedan od algoritama klasifikacije. Ideja ovog algoritma jeste formiranje drveta koje će u svojim listovima imati podatke koji su klasifikovani jednom od mogućih klasa. Koraci u izgradnji drveta odlučivanja su sledeći:

1. **Izbor atributa u čvoru**: Cilj ovog koraka je odabrati atribut pomoću kojeg se vrši najbolja podela podataka posmatrajući ciljnu promenljivu. Idealan slučaj je da se skup podataka podeli u dva disjunktna skupa tako da jednom skupu pripadaju instance samo prve klase a drugom skupu instance samo druge klase. Metrika pomoću koje ćemo ocenjivati koliko je podela dobra ("čista") zove se Ginijev indeks.

   Veće vrednosti Ginijevog indeksa ukazuju na veću nečistotu unutar podataka, odnosno među podacima ima instanci obe klase. Nad skupom podataka koji se nalazi u tekućem čvoru se računa Ginijev indeks za svaki mogući atribut i za atribut podele u datom čvoru bira se onaj koji ima najmanji Ginijev indeks, odnosno obezbeđuje najčistiju podelu podataka.

2. **Podela čvora**: Podaci se dele na podskupove na osnovu vrednosti izabranog atributa podele, čime nastaju novi deca-čvorovi.

3. **Rekurzivni korak**: Postupak izbora atributa i podele se rekurzivno ponavlja za svaki od podskupova u čvorovima. Ovo se nastavlja dok se ne ispune određeni uslovi zaustavljanja, kao što su maksimalna dubina drveta, minimalni broj uzoraka u čvoru ili druge metrike za zaustavljanje.

4. **Klasifikacija**: Kada se postignu uslovi zaustavljanja, čvorovi postaju listovi stabla. List se klasifikuje klasom koja preovladava medju podacima tog čvora.

Nakon formiranog stabla, klasifikacija novog podatka dobija se prolaskom kroz stablo tako što se posmatra vrednost odgovarajućeg atributa u tekućem čvoru, sve dok se ne dođe do lista, čime je određuje kojoj klasi podatak pripada.

Prilikom konstruisanja stabla odlučivanja bitno je uspostaviti kriterijume zaustavljanja, kako stablo ne bi bilo preveliko čime bi se moglo preprilagoditi trening podacima. Odabir ovih hiperparametara u Python-u je podržan pomoću funkcije GridSearchCV.

5.1.1 GridSearchCV

Optimizacija hiperparametara je ključni korak ka stvaranju modela koji daju najbolje performanse. Hiperparametri su parametri modela koji se ne uče tokom treniranja, već se postavljaju pre njega. Ovi parametri utiču na ponašanje modela i mogu značajno uticati na njegovu tačnost i generalizaciju.

GridSearchCV (Grid Search Cross-Validation) je alat omogućava sistematično ispitivanje različitih kombinacija hiperparametara kako bi se pronašla najbolja kombinacija za konkretni problem. GridSearchCV radi na sledeći način:

1. **Definisanje hiperparametara**: Prvo definišemo koje hiperparametre želimo da optimizujemo i koje vrednosti želimo da ispitamo.

2. **Kreiranje skupa kombinacija**: U ovom koraku GridSearchCV će generisati sve moguće kombinacije odabranih hiperparametara.

3. **Učenje i validacija**: Za svaku kombinaciju hiperparametara, GridSearchCV će izvesti unakrsnu validaciju na trening skupu podataka kako bi ocenio tačnost modela. Ovo će se ponoviti za svaku kombinaciju.

4. **Odabir najboljeg modela**: Nakon završetka pretrage, GridSearchCV će identifikovati kombinaciju hiperparametara koja je dala najbolje performanse na validacionom skupu.

5. **Treniranje konačnog modela**: Kada je najbolja kombinacija hiperparametara identifikovana, možemo trenirati konačni model sa ovim vrednostima hiperparametara na celom trening skupu podataka.

Važno je napomenuti, ukoliko je odabrano previše parametara i vrednosti, može doći do preprilagođavanja na trening podacima. Prostor hiperparametara postaje ogroman što znači da će GridSearchCV morati da izvrši veliki broj treninga i validacija, što može dovesti do dubokog pretraživanja i prilagođavanja trening podacima. Takođe, ukoliko ima mnogo kombinacija hiperparametara, postoji veća verovatnoća da će GridSearchCV slučajno pronaći kombinaciju koja se savršeno uklapa sa trening podacima, iako je to samo slučajnost.

Kako bi se zaštitili od preprilagođavanja dobro je pratiti sledeće smernice:

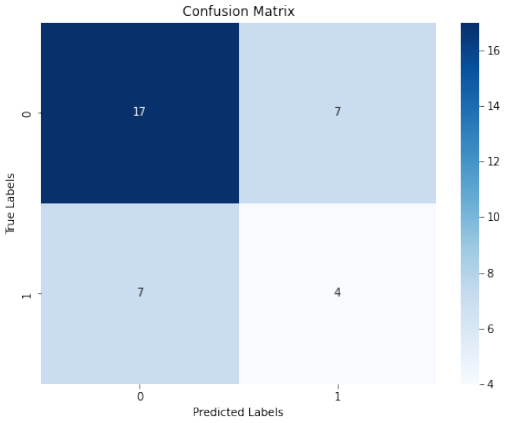
- Smanjivanje broja parametara i vrednosti
- Pravilno postavljanje unakrsne validacije pomoću `cv` parametra u GridSearchCV.
- Dodavanje regularizacije na hiperparametre (npr. ograničenje opsega vrednosti).

5.1.2 Odabir hiperparametara

Prateći navedene korake za optimizaciju hiperparametara, prvo biramo parametre i vrednosti koje će biti razmatrane u GridSearchCV alatu. Parametri i vrednosti koje su odabrane su:

- *max_depth* - maksimalna dubina stabla. Odabran skup vrednosti - [4,5,6,7,8,9,10,13]
- *min_samples_split* - minimalan broj instanci u čvoru potreban da bi čvor mogao da se podeli. Odabran skup vrednosti - [2,3,4,5,6,7,8]
- *min_samples_leaf* - minimalan broj instanci u čvoru kako bi se on smatrao listom. Odabran skup vrednosti - [1,3,5,7,9,11]

Unakrsnom validacijom dobijeni su hiperparametri koji daju najbolju tačnost na trening podacima. Najbolji hiperparametri su: *max_depth*: 13, *min_samples_leaf*: 1, *min_samples_split*: 2 i daju tačnost 0.76 na validacionom skupu (parametar cv je postavljen na 4). Sledeći korak je da model istreniramo na svim trening podacima sa pomenutim parametrima i analiziramo dobijene rezultate.
Nakon treniranja modela na svim podacima, dobijena je tačnost 1 na trening i 0.63 na test podacima, što je jasan znak da je došlo do preprilagođavanja trening podacima.



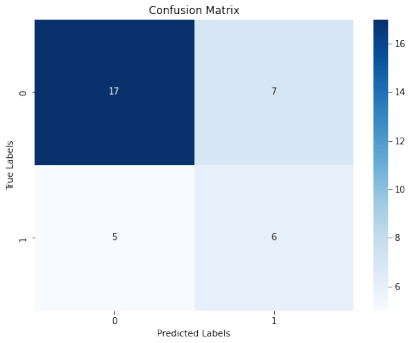
Posmatrajući matricu konfuzije sa prethodne slike, primećujemo da model ispravno klasifikuje 21 test instancu od ukupnih 35. Ukupno 17 netoksičnih molekula je ispravno klasifikovano, dok je preostalih 7 klasifikovano pogrešno kao toksični. S druge strane, od ukupno 11 toksičnih molekula, čak 7 je klasifikovano pogrešno kao netoksični, što predstavlja problem i naš model čini lošijim. Naime, manja je cena greške u realnom životu ukoliko netoksičan molekul proglasimo toksičnim nego da toksičan molekul proglasimo netoksičnim, jer bi to moglo da izazove veće posledice po organizam čoveka. Iz navedenog, jasno je da pored tačnosti modela moramo obratiti pažnju i na F1 skor. Trenutni F1-skor za klasu netoksičnih molekula iznosi 0.72, a za klasu toksičnih 0.43.

Kako bismo smanjili preprilagođavanje modela i potencijalno povećali F1-skor i tačnost na test podacima, smanjićemo broj vrednosti koje uzimaju odabrani parametri. Kako bismo znali kako da smanjimo opseg vrednosti koji svaki parametar nosi, moramo razumeti kako parametri s određenim vrednostima utiču na izgradnju stabla.

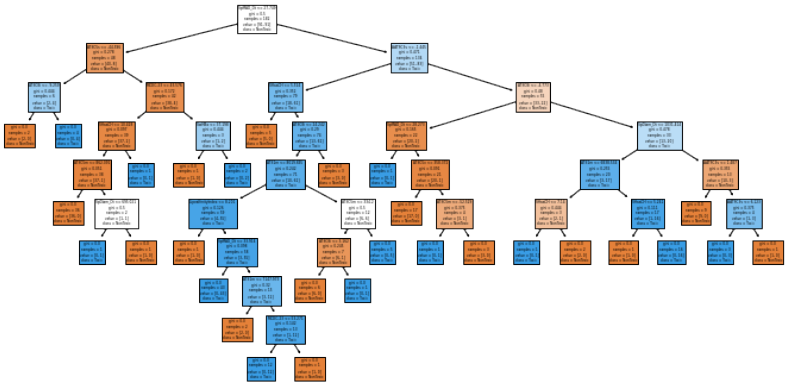
Maksimalna dubina stabla (*max_depth*) je jedan od najvažnijih faktora koji utiču na preprilagođavanje. Veća dubina može omogućiti modelu da nauči složenije obrasce u trening podacima, ali isto tako može dovesti do preprilagođavanja. Kako bismo smanjili preprilagođavanje, vrednosti za maksimalnu dubinu stabla ćemo smanjiti.**Minimalan broj instanci potreban za grananje čvora** (*min_samples_split*) može dovesti do preprilagođavanja ukoliko su vrednosti koje razmatra male, jer model može granati čvorove čak i ako postoji malo podataka u njima. Postavljanjem većeg praga za grananje čvora može se smanjiti preprilagođavanje. Isto važi i za **minimalan broj instanci u listu** (*min_samples_leaf*), veće vrednosti mogu smanjiti preprilagođavanje.

Postepenim uklanjanjem niskih i visokih vrednosti na pomenuti način, kao najbolji hiperparametri pomoću GridSearchCV odabrani su *max_depth* : 5, *min_samples_leaf*: 7, *min_samples_split*: 7. Tačnost modela na trening podacima je sa 1 smanjena na 0.85, a tačnost na test podacima je sa 0.60 povećana na 0.66. F1-skor za klasu *NonToxic* ostaje nepromenjena dok je za klasu *Toxic* povećana na 0.5. Na sledećoj slici prikazana je ma-

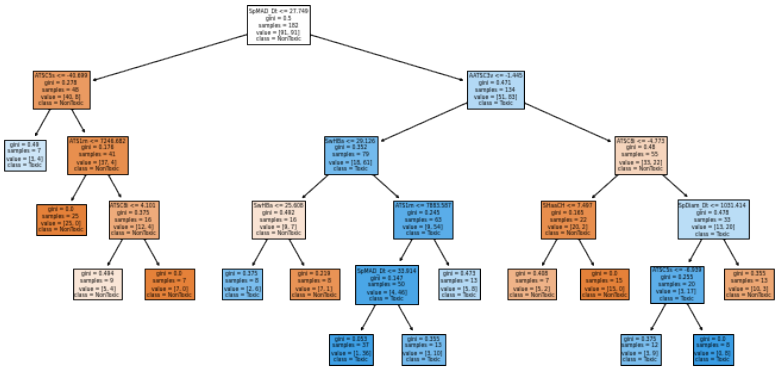
trica konfuzije novodobijenog modela. Možemo primetiti kako je sada više molekula koji su toksični tačno klasifikovano, kao i da je preprilagođavanje smanjeno ali i dalje prisutno.



Na sledećim slikama možemo videti izgled konstruisanog stabla pre i posle regulisanja vrednosti koje uzimaju parametri. Prvo što možemo videti jeste da je dubina stabla duplo manja nakon regularizacije, što je najviše doprinelo pojednostavljivanju modela. Takođe zahvaljujući povećanju vrednosti za druga dva parametra uočavamo da novodobijeno stablo ima znatno manje čvorova. Dubina stabla je smanjena sa 9 na 5, broj listova sa 25 na 14, odnosno ukupan broj čvorova sa 56 na 27.



(a) Pre regularizacije



(b) Nakon regularizacije

Figure 4: Vizualizacija stabla odlučivanja

5.2 Metoda potpornih vektora

Metoda potpornih vektora (Support Vector Machine - SVM) je još jedan od modela mašinskog učenja i može se koristiti i za klasifikaciju i regresiju. Ideja SVM jeste da pronađe hiperravan koja najbolje razdvaja podatke na osnovu njihovih klasa. Podaci koji se razdvajaju uopšte ne moraju biti linearno razdvojivi tj. ne mora postojati jasna linearna granica između različitih klasa.

Postupak traženja hiperravni sam po sebi predstavlja optimizacioni problem. Naime, potrebno je pronaći hiperravan tako da se maksimizuje rastojanje između njenih margina. Margina se definiše kao udaljenost između najbliže tačke svake klase i hiperravni, a te najbliže tačke se definišu kao potporni vektori.

Podaci s kojima mi radimo su linearno nerazdvojivi, stoga se koristi tehnika *Kernel Trick*. Ova tehnika mapira podatke u višedimenzioni prostor, gde se podaci mogu lakše razdvojiti linearno. Mapiranje se vrši tako što se na svaki par podataka primeni takozvana kernel funkcija, čime od svakog para nastaje nova tačka u tom višedimenzionom prostoru. Ovime je omogućeno da SVM radi sa nelinearnim podacima kao sa linearnim. U višedimenzionalnom prostoru, SVM pokušava pronaći hiperpovrš (hiperravan ili hiperprostor) koja najbolje razdvaja te tačke. Ova hiperpovrš postaje granica između klasa u tom višedimenzionom prostoru i potrebno je tu granicu mapirati nazad u originalan prostor. Ukoliko su podaci u originalnom prostoru nelinearni i granica će biti nelinearna.

Postoje različite vrste **kernel funkcija**, a neke od najčešće korišćenih su:

- **Linearni kernel**: Koristi se kada su podaci linearno razdvojivi.
- **Polinomijalni kernel**: Koristi se za nelinearne podatke i funkcija koja mapira podatke u višedimenzioni prostor je u obliku polinoma. **Parametrom degree** se može definisati kog je stepena polinom pomoću kojeg mapiramo podatke.
- **RBF (Radial Basis Function) kernel**: Često se koristi za nelinearne podatke i omogućava mapiranje u beskonačno dimenzionalni prostor.
- **Sigmoidni kernel**: Koristi se za podatke koji se ne ponašaju ni linearno ni polinomijalno, već kao sigmoidna funkcija.

Još jedan od parametara koji je bitan prilikom pravljenja SVM jeste **parametar C**. On predstavlja kompromis između tačnosti modela i širine margina. Niže vrednosti (između 0 i 1) ovog parametra favorizuju šire margine čime može doći do nešto niže tačnosti modela ali može sprečiti preprilagođavanje trening podacima za razliku od većih vrednosti ovog parametra. Više vrednosti (>100) favorizuju visoku tačnost modela na trening podacima, tj. da sve trening instance budu tačno razdvojene što dovodi do kompleksnijih modela sa uskim marginama. Vrednosti između 1 i 100 mogu biti dobar izbor ako ne postoje jaki zahtevi za širinom margine i ako je potrebno postići dobar balans između tačnosti i generalizacije.

Parametar gamma utiče na sve kernel funkcije u SVM modelu, ali njegov uticaj je posebno izražen kada se koristi RBF funkcija. Male vrednosti gamma (npr. 0.01 i manje)

čini RBF kernel "širokim" i funkcija jezgre postaje glatka, dok veliki gamma (npr. 10 i više) čini RBF kernel "uskim" i funkcija jezgre postaje oštra. Uticaj gamma na model što se tiče kompromisa između tačnosti i širine margina je isti kao kod parametra C.

Iako gamma utiče na sve kernel funkcije, njegov efekat može biti manje izražen u drugim kernelima, poput linearnog ili polinomijalnog kernela. U tim slučajevima, gamma će uticati na oblik funkcije jezgra, ali će tačnost modela više zavisiti od drugih hiperparametara, kao što su stepen polinoma u polinomijalnom kernelu ili vrednost C.

5.2.1 Skaliranje podataka

Skaliranje podataka može poboljšati tačnost SVM modela. Ukoliko podaci nisu skalirani, atributi sa većim vrednostima mogu dominirati u odnosu na one sa manjim vrednostima prilikom donošenja odluka o razdvajanju klasa. Ovo može dovesti do neujednačenog uticaja pojedinih atributa na klasifikaciju.

Takođe, skalirani podaci mogu ubrzati konvergenciju tokom treniranja SVM modela. Ako podaci nisu skalirani, optimizacija može zahtevati više iteracija kako bi konvergirala, jer će algoritam morati raditi s različitim domenima atributa.

U ovom radu su podaci skalirani tako da im srednja vrednost bude 0 a standardna devijacija 1.

Kako bismo pokazali važnost skaliranja podataka, istrenirali smo model sa podrazumevanim parametrima nad neskaliranim podacima i postigli tačnost 0.58 na trening i 0.48 na test podacima. Nakon skaliranja tačnost nad trening podacima iznosi 0.83 a nad test podacima 0.71.

5.2.2 Odabir hiperparametara

Svi pomenuti parametri će biti korišćeni prilikom nalaženja optimalnih hiperparametara unutar GridSearchCV. Oni će uzimati sledeće vrednosti:

- **kernel**: ['rbf','sigmoid','poly','linear']
- **C**:[0.01, 0.1,0.5,0.8,1]
- **gamma**: [0.001,0.1,1,10]
- **degree**: [2,3,4,5]

Sada ćemo uporediti rezultate dobijenih modela u zavisnosti od odabranog kernela, tj. za svaki tip kernela ćemo odabrati putem GridSearchCV najoptimalnije hiperparametre.Nakon unakrsne validacije kao najoptimalniji hiperparametri odabrani su: C: 0.5, gamma: 0.1, kernel: 'rbf'. Model sa datim hiperparametrima nakon treniranja nad svim trening podacima daje sledeće rezultate: tačnost na trening podacima 0.83, tačnost na trening podacima 0.71. F1-skor za klasu *NonToxic* iznosi 0.8, a za klasu *Toxic* 0.44. Možemo primetiti kako je tačnost modela na test instancama veća nego u slučaju stabla

odlučivanja, ali je F1-skor za klasu *Toxic* lošiji, zbog čega ne možemo reći da je ovaj model bolji u odnosu na stablo odlučivanja.

5.3 Ansambl metode

Ansamble metode predstavljaju tehnike za poboljšanje tačnosti klasifikacije tako što kombinuju predviđanja više klasifikatora, zbog čega se ove metode takođe nazivaju **metode kombinovanih klasifikatora** (*classifier combination methods*). Ansambl metoda konstruiše skup baznih klasifikatora pomoću trening podataka i klasifikuje podatke putem glasanja na osnovu predviđanja svakog baznog klasifikatora. Glasanje se odvija tako što svaki bazni klasifikator predviđa klasu za svaki podatak, potom ansambl metoda svaki podatak klasifikuje onom klasom kojom ga je većina baznih klasifikatora klasifikovala.

5.3.1 Motivacija za korišćenje ansambl metoda

Ansamble metode obično pružaju bolje performanse u poređenju sa pojedinačnim klasifikatorima što predstavlja glavnu motivaciju za njihovo korišćenje. Ukoliko bismo konstruisali ansamlb metod od 20 binarnih klasifikatora od kojih svaki klasifikator ima tačnost 0.7 (stopa greške $\epsilon = 0.3$), možemo razmatrati sledeća dva slučaja:

1. **Svi bazni klasifikatori su identični**: U ovom slučaju, svi bazni klasifikatori prave iste greške, jer su identični. Kako svaki od njih ima stopu greške od 0.3, ansambl metoda od 20 klasifikatora takođe će imati stopu greške od 0.3.

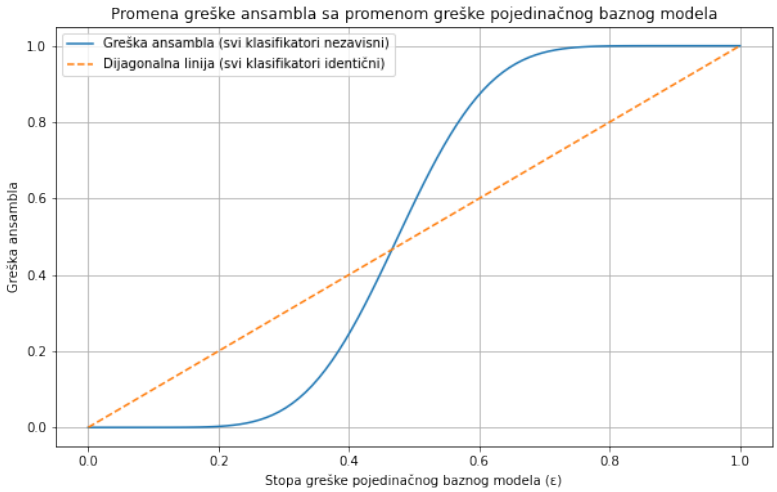
2. **Bazni klasifikatori su nezavisni**: Kada su bazni klasifikatori nezavisni, njihove greške su takođe nezavisne. Zbog predviđanja glasanjem, ansambl metoda čini pogrešnu predikciju podatka samo ako više od polovine baznih klasifikatora pogrešno predviđa. Kako imamo 20 baznih klasifikatora, da bismo postigli pogrešnu predikciju, mora biti više od 10 njih koji pogrešno predviđaju. Izračunaćemo verovatnoću da se to desi.

$$\epsilon_{ensemble} = \sum_{i=11}^{20} \frac{20!}{i!(20-i)!}(\epsilon^i)(((1-\epsilon)^{20-i}) \approx 0.0343$$

   Ovde se koristi binomna distribucija kako bi se izračunala verovatnoća da će više od 10 od 20 klasifikatora pogrešiti. Kada se ovo izračuna, dobijamo vrednost od 0.0343, što je stopa greške ansambla u ovom scenariju.

Zaključak je da kada su osnovni klasifikatori nezavisni, ansambl ima znatno nižu stopu greške u odnosu na bazne klasifikatore, ali samo ukoliko je stopa greške manja od 0.5.

Na sledećem grafiku možemo videti kako se stopa greške ansambl metode menja u odnosu na stopu greške pojedinačnih baznih klasifikatora. Dijagonalna linija predstavlja slučaj kada su bazni klasifikatori identični, dok puna linija predstavlja slučaj kada su bazni klasifikatori nezavisni. Primećujemo da ansambl klasifikatora daje lošije rezultate od baznih klasifikatora kada je stopa greške veća od 0.5.



Ovim primerom smo ilustrovali dva neophodna uslova za to da ansambl metoda klasifikatora radi bolje od pojedinačnog klasifikatora:

1. Osnovni klasifikatori treba da budu nezavisni jedni od drugih.
2. Osnovni klasifikatori treba da postižu bolje rezultate od klasifikatora koji vrši slučajno predviđanje.

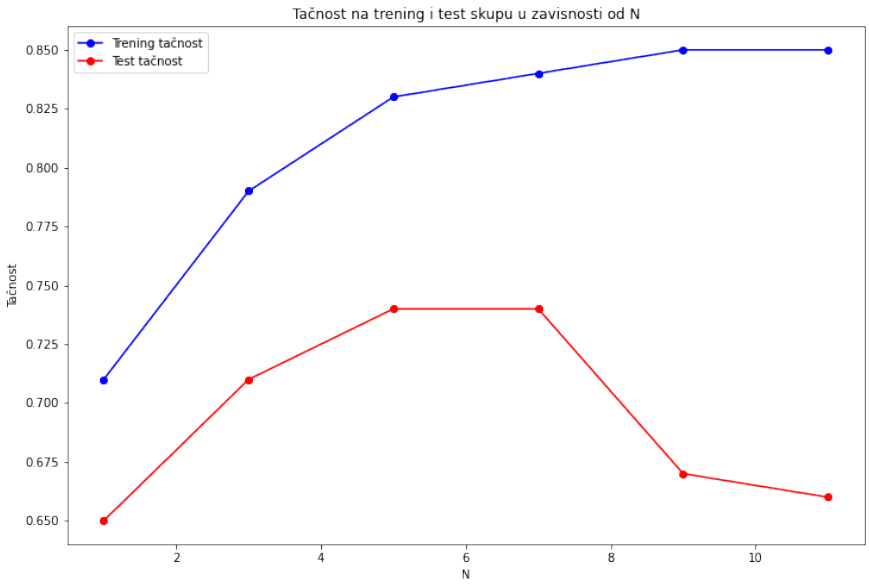
U praksi, potpuna nezavisnost među baznim klasifikatorima je teško postići. Ipak, uočene su poboljšane tačnosti klasifikacije u ansambl metodama u kojima su bazni klasifikatori delimično korelirani.

5.3.2 Pakovanje - Bagging

Pakovanje (*bagging*) predstavlja jednu od ansambl metoda, zasnovana na uzorkovanju podataka sa ponavljanjem iz originalnog skupa podataka prema uniformnoj raspodeli. Ideja ove tehnike je da svaki bazni model ima svoje trening podatke dobijene uzorkovanjem iz originalnih podataka tako da su iste dimenzije kao originalni podaci. Pošto se vrši uzorkovanje sa ponavljanem, u svakom skupu podataka se mogu javiti duplikati. Naime, u svakom skupu podataka naći će se u proseku 63% originalnih podataka zbog uniformne raspodele.

Svaki bazni model se trenira nad različitim trening podacima, čime se postiže to da je ansambl metoda manje podložna preprilagođavanju i ima bolju sposobnost generalizacije nad novim podacima. Kao tip baznog modela koji će biti korišćen u ovoj ansambl metodi, odabrano je drvo odlučivanja. Za svako drvo odlučivanja će biti odabrani najbolji hiperparametri za njegove trening podatke, putem tehnike GridSearchCV. Parametri koji će biti razmatrani su maksimalna dubina stabla, minimalan broj instanci za podelu čvora, minimalan broj instanci u listu i kriterijum za računanje nečistoće u svakom čvoru. Kako

bismo našli koji je najbolji broj baznih klasifikatora, ispitaćemo tačnost ansambl metode za 1,3,5,7,9 i 11 baznih klasifikatora. Na sledećem grafiku možemo uočiti kako se menja tačnost na trening i test skupu u zavisnosti od broja baznih klasifikatora.



Možemo primetiti da tačnost ansambl metode na trening skupu raste kako povećavamo broj baznih klasifikatora, ali tačnost na test skupu počinje da opada ukoliko ih ima više od 7, gde počinje da se pojačava efekat preprilagođavanja. Najbolje rezultate dobijamo kada u ansambl metodi ukljucimo 5 drveta odlučivanja. Svaki od tih baznih klasifikatora bio je treniran na posebnom trening skupu i u zavisnosti od njega su mu dodeljeni hiperparametri koji najbolje klasifikuju podatke na tom skupu. Kako bi izbegli preprilagođavanje, sva drveta u ovoj metodi su jednostavna po svojoj strukturi i imaju prosečnu tačnost 0.71 na trening i 0.65 na test skupu, čime je ispunjen uslov da svaki bazni klasifikator ima bolju tačnost od nasumičnog predviđanja.

Vrednosti koje je svaki parametar mogao da uzme su pažljivo birani baš kako bi svako stablo bilo jednostavno, čime bi se smanjio efekat preprilagođavanja. Za maksimalnu dubinu stabla razmatrane su vrednosti 3 i 4, za minimalan broj instanci u čvoru za podelu razmatrane su vrednosti 6,7,8, iste ove vrednosti su razmatrane i za minimalan broj instanci u listu, dok je za kriterijum merenja nečistoće u čvoru razmatran Ginijev indeks i entropija.

Nakon izvršavanja celokupnog algoritma pakovanja sa 5 baznih klasifikatora, dobijena su sledeća drveta odlučivanja:

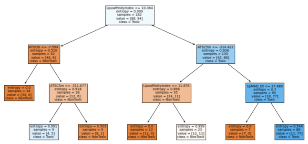
1. drvo : maksimalna dubina = 3, minimalan broj instanci u čvoru za podelu = 7, minimalan broj instanci u listu = 7, kriterijum za merenje nečistoće u čvoru = entropija

2. drvo : maksimalna dubina = 4, minimalan broj instanci u čvoru za podelu = 6, minimalan broj instanci u listu = 7, kriterijum za merenje nečistoće u čvoru = Ginijev indeks

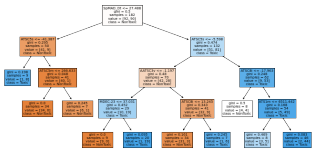
3. drvo : maksimalna dubina = 4, minimalan broj instanci u čvoru za podelu = 8, minimalan broj instanci u listu = 6, kriterijum za merenje nečistoće u čvoru = entropija

4. drvo : maksimalna dubina = 4, minimalan broj instanci u čvoru za podelu = 6, minimalan broj instanci u listu = 6, kriterijum za merenje nečistoće u čvoru = entropija

5. drvo : maksimalna dubina = 4, minimalan broj instanci u čvoru za podelu = 7, minimalan broj instanci u listu = 8, kriterijum za merenje nečistoće u čvoru = entropija



(a) 1. drvo



(b) 2. drvo

(c) 3. drvo

(d) 4. drvo

(e) 5. drvo

Kao što se može primetiti na prethodnim slikama, bazni klasifikatori se razlikuju po strukturi drveta, što možemo videti i na osnovu njihovih parametara. Ovo doprinosi drugom važnom uslovu koji želimo da postignemo prilikom izgradnje ansambl metoda - bazni klasifikatori trebaju biti što više nezavisni.

Prosečna tačnost ove ansambl metode na trening podacima iznosi 0.83, dok na test podacima iznosi 0.74. Prosečna vrednost F1-skora za klasu *NonToxic* iznosi 0.79 a za klasu *Toxic* 0.55. Možemo primetiti da nam je ova ansambl metoda pomogla da dobijemo bolje rezultate u odnosu na model u kojem smo koristili samo jedno drvo odlučivanja. Efekat preprilagođavanja je i dalje prisutan ali je njegov uticaj smanjen, čime je postignuta nešto veća sposobnost generalizacije. Takođe F1-skor za klasu *Toxic* je povećan u odnosu na prethodne modele, čime zaključujemo da ovim modelom bolje uočavamo toksične molekule.

6 Upoređivanje rezultata različitih modela

U ovom odeljku ćemo sumirati rezultate svih korišćenih modela u ovom radu, i prokomentarisati koji od njih najbolje klasifikuje naše podatke. Tačnost i F1-skor za klasu *Toxic* ćemo smatrati kao najbitnije metrike prilikom upoređivanja ovih modela. Takođe, kako bismo uočili važnost tehnike SMOTE, razmotrićemo i rezultate ovih modela treniranih na nebalansiranim podacima.

Na sledećoj tabeli možemo videti rezultate modela treniranih nad *balansiranim podacima*:

	Trening Acc	Test Acc	F1 Toxic	F1 NonToxic
DT	0.85	0.66	0.5	0.72
SVM	0.83	0.71	0.44	0.8
Bagging	0.83	0.74	0.57	0.79

Na osnovu ove tabele možemo zaključiti da je model koji daje najbolje rezultate model dobijen ansambl metodom pakovanja, što je donekle i očekivano. Pored najbolje prosečne tačnosti na test skupu koja iznosi 0.74, ovaj model najbolje detektuje toksične molekule jer je F1-skor za klasu *Toxic* 0.57, što smo ranije napomenuli kao bitnu metriku, obzirom da nije ista cena greške kada se pogrešno klasifikuje toksičan i netoksičan molekul. Pored ovoga, još jedna stavka koja ide u prilog ovom modelu je ta što je efekat preprilagođavanja, koji je prisutan kod svih modela, u ovom modelu najmanji.

Sada ćemo razmotriti i rezultate modela treniranih nad podacima nad kojima nije izvršena tehnika SMOTE, odnosno posmatraćemo rezultate nad *nebalansiranim podacima*:

	Trening Acc	Test Acc	F1 Toxic	F1 NonToxic
DT	0.8	0.68	0.3	0.9
SVM	0.73	0.66	0.0	0.79
Bagging	0.86	0.69	0.35	0.78

Posmatrajući ove rezultate, možemo primetiti da su rezultati lošiji, što opravdava razlog zbog čega smo koristili tehniku SMOTE. Pošto podaci u ovom slučaju nisu balansirani, modelima je teže da uoče toksične molekule, jer u podacima preovladavaju netoksični molekuli. Možemo videti kako je za svaki model F1-skor za klasu *Toxic* opao, dok je F1-skor za klasu *NonToxic* ostao isti ili je povećan. Da je od presudne važnosti korišćenje tehnike SMOTE vidimo i na osnovu F1-skora klase *Toxic* u modelu SVM, gde taj skor iznosi čak 0.0, što znači da ovaj model uopšte ne prepoznaje toksične molekule u test podacima, već zbog netoksičnih molekula koji preovladavaju u trening podacima, svaki test podatak klasifikuje klasom *NonToxic*. U modelu dobijenim stablom odlučivanja vidimo malo bolju tačnost na test podacima, ali je F1-skor za klasu *Toxic* znatno lošiji, čime se

i ove opravdava korišćenje SMOTE algoritma.

Zaključak koji se može dovesti na osnovu analize prikazanih rezultata jeste da je model koji najbolje klasifikuje podatke model dobijen ansambl metodom pakovanja, sa 5 baznih klasifikatora, čiji su podaci balansirani uz pomoć tehnike SMOTE. Ono što je važno napomenuti jeste da je algoritam pakovanja u ovom radu implementiran ručno bez korišćenja ugrađene klase *BaggingClassifier* u programskom jeziku Python. Razlog za to je što se korišenjem ugrađene funkcije ne dobijaju nezavisni bazni klasifikatori, što je najbitniji uslov prilikom konstruisanja ansambl metoda, već veći broj klasifikatora sa istim parametrima, čime nisu dobijeni zadovoljavajući rezultati. Našom implementacijom dobijeni su klasifikatori koji imaju različite vrednosti parametara čime se dobijaju prikazani rezultati. Takođe važno je istaći da su iz istog razloga ručno implementirane i ansambl metode pojačavanje *(boosting)* i nasumična šuma (*random forest*), ali i sa ručnom implementacijom rezultati nisu zadovoljavajući pa nisu pomenuti u ovom radu.

7 Literatura

1. Pang-Ning Tan, Michael Steinbach, Anuj Karpatne, Vipin Kumar. "Introduction to Data Mining", 2019.
2. Xin-She Yang. "Introduction to Algorithms for Data Mining and Machine Learning", 2019.
3. Verónica Bolón-Canedo, Noelia Sánchez-Maroño, Amparo Alonso-Betanzos. "Feature Selection for High-Dimensional Data", 2015.
4. CRY1 Gene Information - The Human Protein Atlas, https://www.proteinatlas.org/ENSG00000008405-CRY1/summary/gene, n.d.