Predviđanje uspešnosti ICD terapije korišćenjem klasifikacionih modela

Seminarski rad u okviru kursa Istraživanje podataka 2

Matematički fakultet u Beogradu

Profesor:
Nenad Mitić

Student:
Nikola Belaković 104/2019

Sadržaj

1. Sažetak....3
2. Uvod....3
3. Skup podataka....4
4. Preprocesiranje podataka....4
5. Opisi klasifikacionih metoda....6
    5.1. Drveta odlučivanja....6
    5.2. Random Forest....7
    5.3. SVM (Support Vector Machine)....8
    5.4. ANN (Artificial Neural Network)....9
    5.5. Bajesovski klasifikator....10
6. Konstrukcija modela....10
    6.1. Drveta odlučivanja....10
    6.2. Random Forest....14
    6.3. SVM....17
    6.4. ANN....20
    6.5. Bajesovski klasifikator....24
7. Zaključak....24
8. Pokretanje projekta....25
Literatura....26

1. Sažetak

Cilj ovog rada je modelovanje uzročnika naprasne srčane smrti koja može biti uslovljena drugim srčanim bolestima, a može se dogoditi i pacijentima bez prethodnih srčanih obolenja. Za ove potrebe koristićemo nekoliko klasifikacionih metoda i pokušati da pronađemo model koji će najbolje opisivati date podatke, koji su prikupljeni testiranjem određenog broja pacijenata.

2. Uvod

Naprasna srčana smrt je jedan od najčešćih uzroka smrti, dešava se u roku kraćem od sat vremena od pojave prvih simptoma i posledica je srčane slabosti (insuficijencije). Insuficijencija je nesposobnost srca, uprkos adekvatnoj venskoj ponudi da snabde ćelije tkiva i organa potrebnom količinom krvi (kiseonika i hranljivih materija).

Uzroci srčane insuficijencije su različiti faktori koji predstavljaju opterećenje za srčani mišić ili ga oštećuju. To su:

- koronarna bolest srca
- hipertenzivna bolest srca
- infekcije srčanog mišića
- perikardna bolest
- urođene i stečene srčane mane
- intoksikacija
- anemija

Slika preuzeta sa reference 1)

Prevencija naprasne srčane smrti se vrši ugradnjom ICD (IntraCardiac Defibrilator) uređaja koji u odgovarajućim trenucima šalje elektronske

impulse srcu da bi reaktivirao njegov rad. Iako ICD daje dobre rezultate u prevenciji naprasne srčane smrti, indikacije za njegovu ugradnju nisu precizne. Trenutni standard za predviđanje se zasniva na EF (Ejection Fraction) vrednosti – procenat krvi koji napušta srce pri svakoj kontrakciji.

3. Skup podataka

U ovom radu je korišćen skup podataka dobijen ispitivanjem određenog broja pacijenata. Skup se sastoji od 106 redova (instanci) i 51 kolone (atributa). Od 51 atributa, 25 je kategoričkog tipa (od toga 20 binarni), a ostali su numerički. Na osnovu ovih karakteristika se predviđa karakteristika ICDterapija koja govori da li se ICD uređaj aktivirao u nekom trenutku praćenja i isporučio terapiju.

Od 106 instanci, za 18 se nije aktivirao ICD uređaj, a za ostale jeste, što nas dovodi do zaključka da su podaci jako nebalansirani. Nebalansirani podaci mogu predstavljati otežavajući faktor za pronalaženje dobrog klasifikacionog modela koji treba biti rešen pre klasifikacije.

Podaci koji su korišćeni u ovom radu se nalaze u datoteci *ICD_podaci.csv,* a objašnjenje svih karakteristika datih instanci se nalazi u datoteci *objasnjenja_atributa.txt.*

4. Preprocesiranje podataka

Nakon učitavanja i posmatranja učitanih podataka, možemo uočiti postojanje nedostajućih vrednosti. Postoji nekoliko načina za rešavanje problema nedostajućih vrednosti koja ona donose, izbacivanje kolone koja sadrži nedostajuću vrednost, izbacivanje torke koja sadrži nedostajuću vrednost ili ubacivanje odgovarajuće vrednosti umesto nedostajuće, najčešće srednje vrednosti. Nakon ispitivanja statistike broja nedostajućih

vrednosti u podacima, vidimo da imamo 3 atributa koji za više od dve trećine redova nemaju vrednost, i zbog toga izbacujemo ta 3 atributa iz našeg skupa podataka.

U skupu podataka imamo i nekoliko atributa čije su vrednosti tekstulane, a koje imaju pogodno preslikavanje u numerički tip (vrednosti da, ne, I, II, III, IV…).

Za potrebe klasifikacije izdvajamo promenljivu ICDterapija od ostatka podataka, ona će biti ciljna promenljiva. Vredosti atributa ICDterapija su kategoričkog tipa, pa je potrebno transformisati ih u numerički tip (vrednosti 0 i 1).

Nakon izdvajanja ciljne promenljive iz ostatka skupa podataka su izbačene kolone koje nemaju značaj za klasifikaciju.

Daljom analizom podataka zaključujemo da imamo veliki broj podataka sa malom varijansom i to je rešeno korišcenjem metode fit_transform pozvane pomoću selektora kreiranog naredbom VarianceThreshold, kojoj se prosleđuje vrednost varijanse na osnovu koje se izbacuju podaci iz skupa podataka.

Problem nebalansiranih podataka koji se javlja na datom skupu podataka se rešava korišćenjem metode fit_resample klase SMOTE koja se nalazi u imblearn.over_sampling biblioteci.

Konačni deo pripreme podataka za klasifikaciju uključuje podelu podataka na trening i test skup, koju izvršavamo koristeći train_test_split metod iz sklearn.model_selection biblioteke. Kao parametre metode prosleđujemo odnos trening i test skupa, u ovom slučaju smo za veličinu trening skupa uzeli 60% podataka. Podela podataka će biti stratifikovana u odnosu na ciljnu promenljivu.

Zbog postojanja razlika u opsezima vrednosti atributa, nakon traint_test_split metode odrađena je i standardizacija trening i test skupa atributa koji se koriste za klasifikaciju korišćenjem StandardScalera iz biblioteke sklearn.preprocessing, i svaki od metoda će biti testiran i na standardizovanom skupu podataka i na skupu koji nije standardizovan.

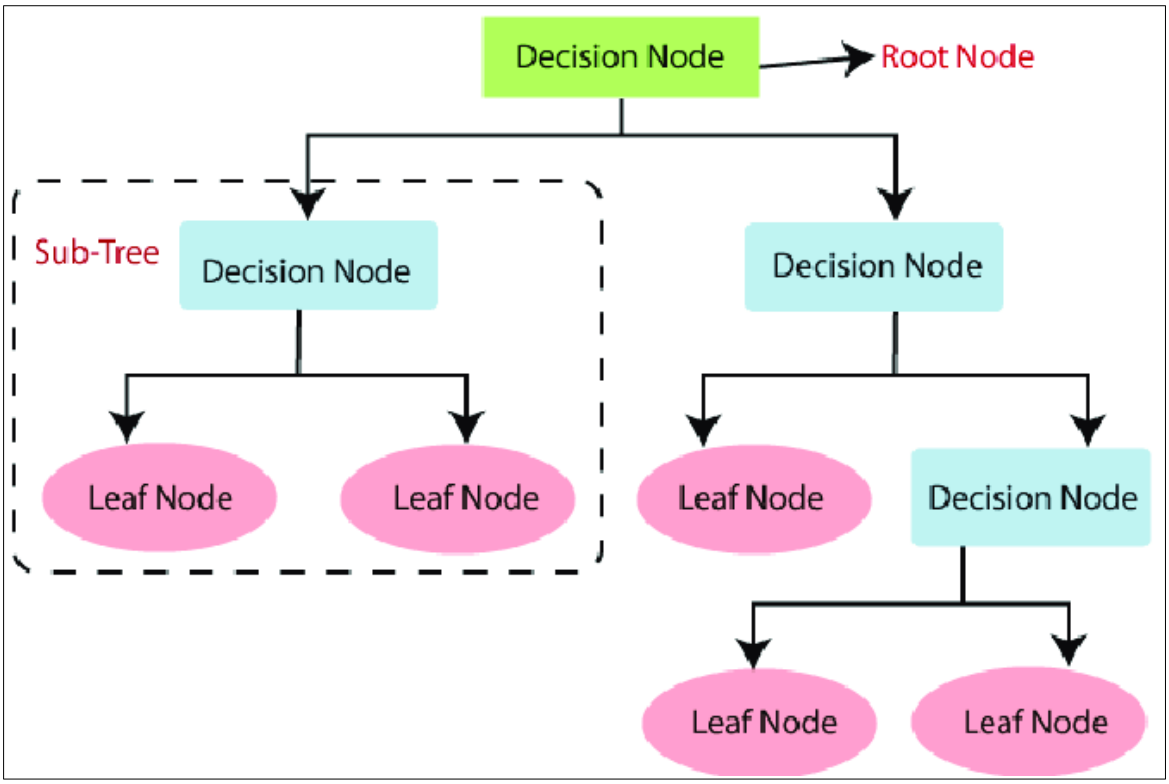
5. Opisi klasifikacionih metoda

Više informacija na referencama ispisanim u literaturi.

5.1. Drveta odlučivanja

Drveta odlučivanja je nadgledana tehnika učenja koja se može koristiti i za klasifikaciju i za regresiju, ali se najčešće koristi za klasifikaciju. To je klasifikator struktuiran u obliku drveta, u svakom ne-list čvoru se nalaze pitanja (pravila) na osnovu kojih se vrše grananja. Ulazni podaci se dele na osnovu vrednosti atributa i pitanja u čvoru. Listovi drveta određuju oznaku klase.

U ovom radu je korišćen DecisionTreeClassifier iz sklearnovog tree paketa koji simulira rad Drveta odlučivanja u Pythonu. Za algoritam je moguće podešavati maksimalnu dubinu drveta, minimalan broj uzoraka potrebnih za razdvajanje unutrašnjeg čvora.

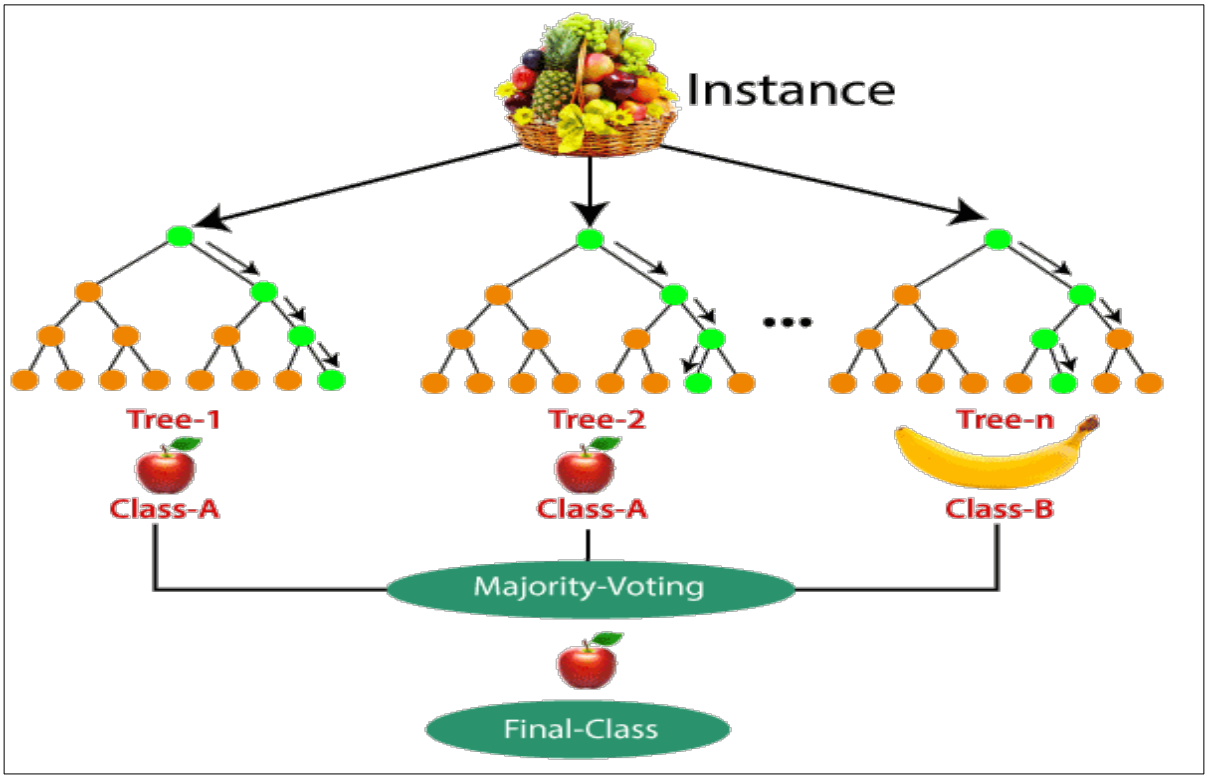


Slika preuzeta sa reference 2)

5.2. Random Forest

Random forest (nasumična šuma) je jedan od najkorišćenijih algoritama mašinskog učenja. Algoritam se može koristiti i za regresiju i za klasifikaciju. Nasumična šuma je posebno konstruisana metoda za ansambl drveta odlučivanja. Svako drvo koristi slučajan vektor atributa generisan sa fiksnom distribucijom raspodele i vraća rezultat svog rada. Na kraju se rezultati svakog drveta kombinuju i dobije se konačan rezultat Random forest algoritma. Ideja korišćenja ansambl metoda je veća preciznost i pouzdanost u odnosu na svaki pojedinačan model.

U ovom radu je korišćen RandomForestClassifier iz sklearnovog ensemble paketa koji simulira rad Random forest algoritma u Pythonu. Neki od najznačajnijih parametara ovog metoda su: maksimalna dubina šume,broj drveta u šumi…

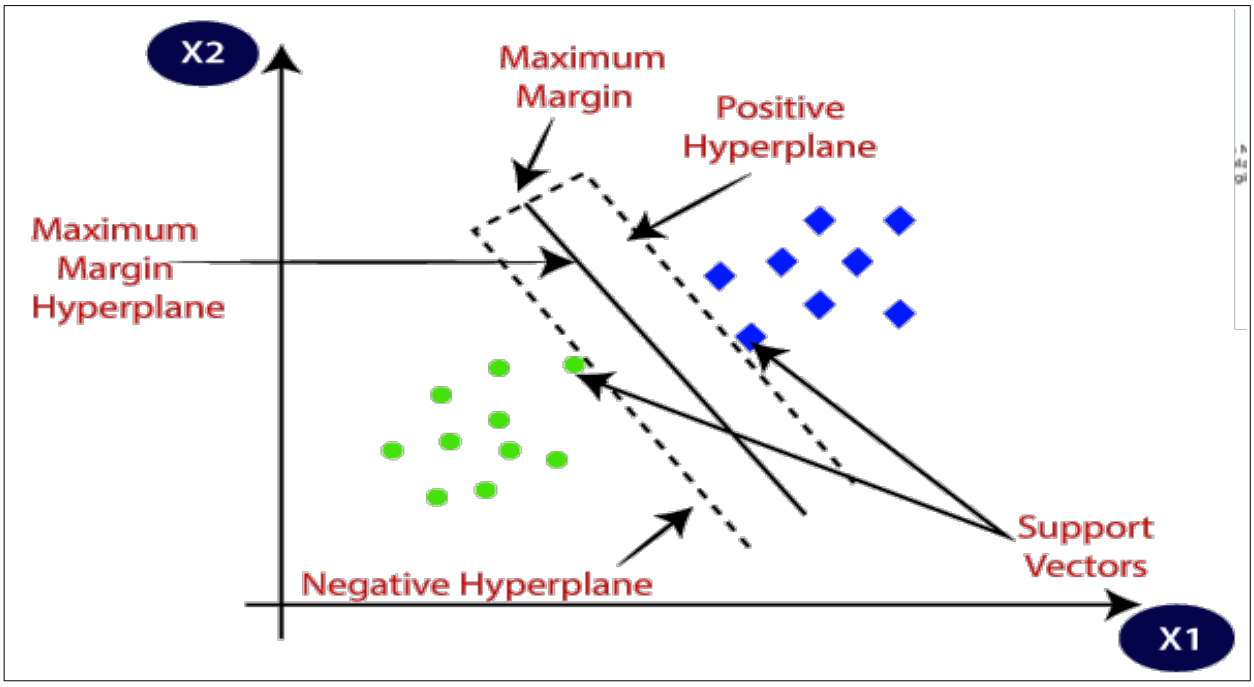


Slika preuzeta sa reference 4)

5.3. SVM (Support Vector Machine)

SVM (metoda potpornih vektora) je metoda nenadgledanog mašinskog učenja za klasifikaciju i regresiju. Često se koristi za visokodimenzione i nebalansirane podatke, jer daje dobre rezultate i u tim situacijama. Model kod SVMa je formula (klasa se izračunava). Koristi se samo za numeričke podatke, kategoričke je potrebno transformisati u numeričke. Ideja algoritma je pronalaženje razdvajajuće hiper-ravni koja maksimizuje razdaljinu između instanci jedne i druge klase. Ova hiper-ravan je potpuno određena podskupom trenirajućih podataka, koji se zovu potporni vektori.

U sklearnovoj biblioteci postoji paket svm u kojem je implementirana metoda SVC koja se koristi za klasifikaciju podataka korišćenjem SVM algoritma u Pythonu



Slika preuzeta sa reference 6)

5.4. ANN (Artificial Neural Network)

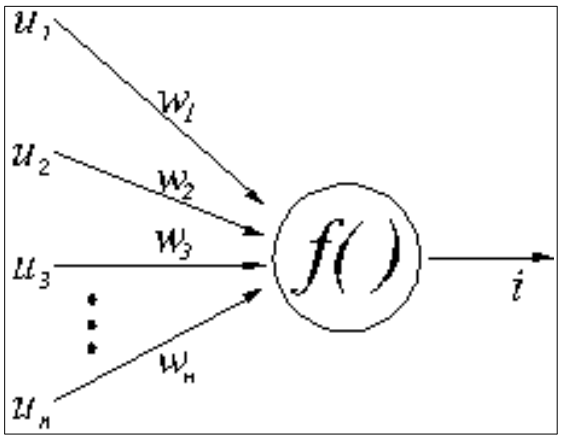
ANN (Veštačke neuronske mreže) je računarski sistem inspirisan biološkim neuronskim mrežama. Ideja veštačke neuronske mreže je simulacija rada biološkog nervnog sistema. Sastavljen je od povezanih čvorova (veštačkih neurona) koji su, slično biološkim neuronima, povezani svojim vezama koje sadrže propusne (težinske) koeficijente, koje su po ulozi slične sinapsama.

Učenje ANN se odvija promenom vrednosti težinskih koeficijenata sve dok se ne sinhronizuju ulazno/izlazne zavisnosti podataka.

Po arhitekturi, neuronske mreže se razlikuju prema broju neuronskih slojeva. Svaki sloj prima ulaze iz prethodnog sloja, a svoje izlaze šalje narednom sloju. Prvi sloj se naziva ulazni, poslednji je izlazni, a ostali slojevi se nazivaju skrivenim slojevima.

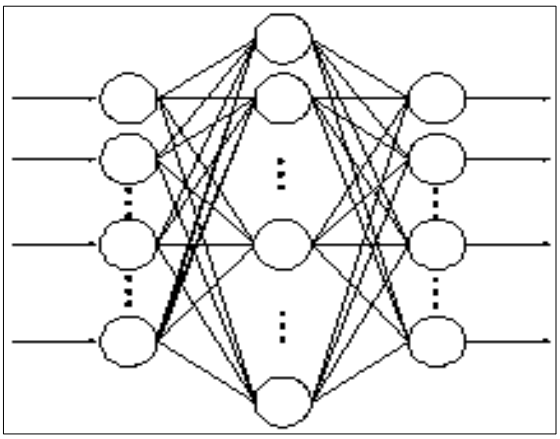
Za potrebe ovog projekta korišćena je tensorflow biblioteka, koja u svojim paketima ima sve potrebne metode za simulaciju rada ANN.

Model veštačkog neurona



Slika preuzeta sa reference 8)

Model neuronske mreže



Slika preuzeta sa reference 8)

5.5. Bajesovski klasifikator

Naivni Bajesovski klasifikator je metoda nenadgledanog učenja zasnovana na Bajesovoj teoremi sa pretpostavkom o nezavisnosti između svakog para karakteristika date klasne promenljive. Zasnovan na teoriji verovatnoće.

Bajesova teorema: $P(y|x_1,\dots,x_n)=\frac{P(y)P(x_1,\dots,x_n|y)}{P(x_1,\dots,x_n)}$

Postoji nekoliko vrsta naivnih Bajesovskih klasifikatora i osnovna razlika im je pretpostavka koju prave u vezi sa rapodelom $P(x_i|y)$.

U ovom radu je korišćen Multinomijalni Bajesovski klasifikator (MultinomialNB) iz sklearnovog paketa naive_bayes.

6. Konstrukcija modela

U ovom delu ćemo se baviti pravljenjem modela koji će najbolje klasifikovati naše podatke. Koristićemo metode koje su navedene ranije u tekstu i uporediti njihove rezultate u potrazi za najboljim.

6.1. Drveta odlučivanja

Algoritam DecisionTreeClassifier je testiran korišćenjem tehnike unakrsne validacije (GridSearchCV) za različite vrednosti parametara criterion koji predstavlja meru nečistoće koja se koristi i parametra max_depth koji predstavlja maksimalnu dubinu drveta. Vrednosti testirane za criterion su ‘gini’ i ‘entropy’, a max_depth je testiran za vrednosti od 4 do 12.

U narednim tabelama su prikazani rezultati rada DecisionTreeClassifier algoritma na trening skupu, bez i sa standardizacijom, za različite vrednosti

parametara navedenih u prethodnom pasusu, kao i diskusija dobijenih rezultata.

Criterion	Max_depth	Mean_test_score	Std_test_score
gini	4	0.571429	0.079682
gini	5	0.580952	0.076190
gini	6	0.628571	0.101686
gini	7	0.628571	0.063174
gini	8	0.628571	0.121964
gini	9	0.609524	0.132651
gini	10	0.600000	0.083027
gini	11	0.609524	0.097124
gini	12	0.609524	0.106053
entorpy	4	0.561905	0.076190
entorpy	5	0.639095	0.077372
entorpy	6	0.619048	0.085184
entorpy	7	0.676190	0.106053
entorpy	8	0.628571	0.092337
entorpy	9	0.685714	0.071270
entorpy	10	0.676190	0.118187
entorpy	11	0.704762	0.069985
entorpy	12	0.619048	0.079682

Tabela rezultata bez standardizacije

Criterion	Max_depth	Mean_test_score	Std_test_score
gini	4	0.609524	0.035635
gini	5	0.580952	0.035635
gini	6	0.657143	0.046657
gini	7	0.590476	0.038095
gini	8	0.609524	0.081927
gini	9	0.638095	0.064594
gini	10	0.638095	0.122706
gini	11	0.609524	0.019048
gini	12	0.600000	0.071270
entorpy	4	0.561905	0.081927
entorpy	5	0.609524	0.081927
entorpy	6	0.619048	0.095238
entorpy	7	0.628571	0.092337
entorpy	8	0.666667	0.108588
entorpy	9	0.666667	0.067344
entorpy	10	0.657143	0.101686
entorpy	11	0.657143	0.097124
entorpy	12	0.646719	0.077372

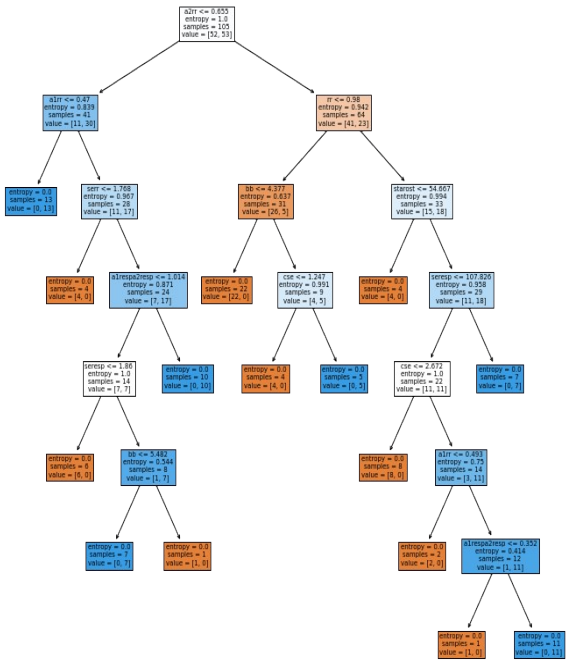
Tabela rezultata sa standardizacijom

Proverom kvaliteta dobijenih rezultata zaključujemo da najbolji rezultat bez standardizacije dobijamo za vrednosti criterion=’entropy’ i max_depth=11 (slika 1), a prilikom korišćenja standardizacije najbolji rezultat se dobija za vrednosti criterion=’entropy’ i max_depth=8 (slika 2).

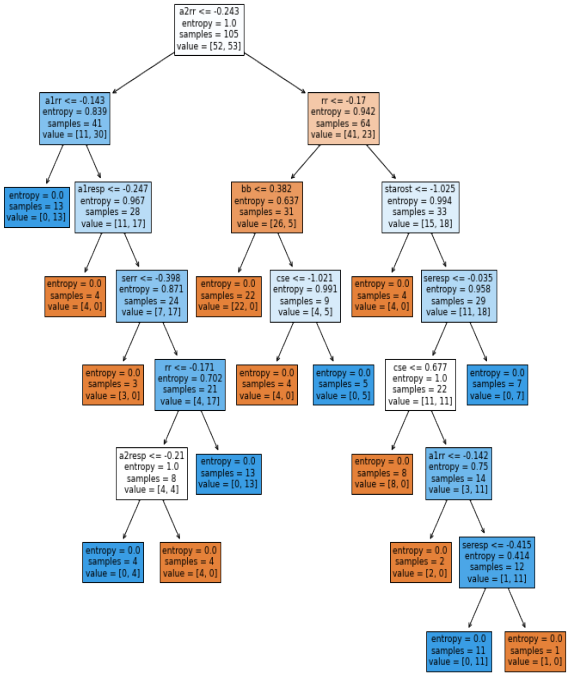
Primenom ova dva dobijena klasifikatora na test skup podataka dobijamo sledeće rezultate:

Standardizacija	Criterion	Max depth	Accuracy score	Confusion matrix		Precision	Recall	F1-score
ne	entropy	11	0.704225	24	9	0 : 0.73	0 : 0.67	0 : 0.70
				12	26	1 : 0.68	1 : 0.74	1 : 0.71
da	entropy	8	0.746479	26	8	0 : 0.76	0 : 0.72	0 : 0.74
				10	27	1 : 0.73	1 : 0.77	1 : 0.75

Iz prethodne tabele zaključujemo da se algoritam Drveta odlučivanja bolje snalazi sa standardizovanim podacima, ali da razlika u kvalitetu sa standardizacijom nije mnogo velika. Takođe, primećujemo da je model bez standardizacije pokazao skoro iste rezultate na trening i test skupu, dok je model sa standardizacijom mnogo bolje rezultate dao na test nego na trening skupu što nam govori da nije došlo do preprilagođavanja trening skupu podataka. Pošto je cilj naše klasifikacije da odredimo kada je potrebno ugraditi ICD uređaj, najbitnije nam je da imamo što više True Positive vrednosti u matrici konfuzije, a što manje False Negative vrednosti. Ondos ove dve vrednosti je predstavljena kao recall (odziv) vrednost za klasu 1 koji se izračunava kao TP/(TP+FN), za koju primećujemo da je slična za podatke sa i bez standardizacije.



Slika 1



Slika 2

6.2. Random Forest

Algoritam RandomForestClassifier je testiran korišćenjem tehnike unakrsne validacije (GridSearchCV) za različite vrednosti parametara n_estimators koji predstavlja broj drva u šumi, parametra max_depth koji predstavlja maksimalnu dubinu drveta, parametra criterion koji predstavlja meru nečistoće koja se koristi i parametra max_features koji predstavlja broj atributa koji se posmatraju kada se traži najbolja podela. Vrednosti testirane za criterion su ‘gini’ i ‘entropy’, max_depth je testiran za vrednosti 4,6 i 8,n_estimators za vrednosti 200 i 500, a max_features za vrednosti ‘sqrt’ i ‘log2’.

U narednim tabelama su prikazani rezultati rada RandomForestClassifier algoritma na trening skupu, bez i sa standardizacijom, za različite vrednosti parametara navedenih u prethodom pasusu.

Criterion	Max depth	Max features	N estimator	Mean_test score	Std_test score
gini	4	sqrt	200	0.742857	0.098054
gini	4	sqrt	500	0.761905	0.085184
gini	4	log2	200	0.733333	0.098054
gini	4	log2	500	0.733333	0.111066
gini	6	sqrt	200	0.771429	0.139321
gini	6	sqrt	500	0.733333	0.088320
gini	6	log2	200	0.723810	0.118187
gini	6	log2	500	0.752381	0.106053
gini	8	sqrt	200	0.790476	0.115077
gini	8	sqrt	500	0.752381	0.092337
gini	8	log2	200	0.790476	0.115077
gini	8	log2	500	0.752381	0.110246

entorpy	4	sqrt	200	0.742857	0.077372
entorpy	4	sqrt	500	0.704762	0.087287
entorpy	4	log2	200	0.733333	0.115077
entorpy	4	log2	500	0.723810	0.069985
entorpy	6	sqrt	200	0.790476	0.057143
entorpy	6	sqrt	500	0.733333	0.083027
entorpy	6	log2	200	0.761905	0.067344
entorpy	6	log2	500	0.752381	0.110246
entorpy	8	sqrt	200	0.790476	0.064594
entorpy	8	sqrt	500	0.752381	0.076190
entorpy	8	log2	200	0.733333	0.088320
entorpy	8	log2	500	0.723810	0.118187

Tabela rezultata bez standardizacije

Criterion	**Max depth**	**Max features**	**N estimator**	**Mean_test score**	**Std_test score**
gini	4	sqrt	200	0.723810	0.118187
gini	4	sqrt	500	0.752381	0.118187
gini	4	log2	200	0.733333	0.106904
gini	4	log2	500	0.723810	0.081927
gini	6	sqrt	200	0.714286	0.067344
gini	6	sqrt	500	0.742857	0.098054
gini	6	log2	200	0.752381	0.101686
gini	6	log2	500	0.723810	0.081927
gini	8	sqrt	200	0.780952	0.088320
gini	8	sqrt	500	0.761905	0.099887
gini	8	log2	200	0.723810	0.092337
gini	8	log2	500	0.780952	0.122706

entorpy	4	sqrt	200	0.733333	0.122706
entorpy	4	sqrt	500	0.723810	0.097124
entorpy	4	log2	200	0.714286	0.095238
entorpy	4	log2	500	0.742857	0.088320
entorpy	6	sqrt	200	0.752381	0.110246
entorpy	6	sqrt	500	0.704762	0.092337
entorpy	6	log2	200	0.733333	0.111066
entorpy	6	log2	500	0.733333	0.106904
entorpy	8	sqrt	200	0.733333	0.102575
entorpy	8	sqrt	500	0.742857	0.115077
entorpy	8	log2	200	0.752381	0.118187
entorpy	8	log2	500	0.771429	0.101686

Tabela rezultata sa standardizacijom

Proverom kvaliteta dobijenih rezultata zaključujemo da najbolji rezultat bez standardizacije dobijamo za vrednosti criterion=’entropy’, max_depth=8, max_features=’sqrt’ i n_estimators=200, a prilikom korišćenja standardizacije najbolji rezultat se dobija za vrednosti criterion=’gini’, max_depth=8, max_features=’sqrt’ i n_estimators=200. Iako je možda bilo očekivano da rezultat bude bolji korišćenjem većeg broja drveta u šumi, iz prikazanog vidimo da to nije slučaj i da se bolji rezultati dobijaju za manju količinu drveta.

Primenom ova dva dobijena klasifikatora na test skup podataka dobijamo sledeće rezultate:

Standa-rdizacija	**Criterion**	**Max depth**	**Max features**	**N estimator**	**Accuracy score**	**Confusion matrix**		**Precision**	**Recall**	**F1-score**
ne	entropy	8	sqrt	200	0.816901	26	3	0 : 0.90	0 : 0.72	0 : 0.80
						10	32	1 : 0.76	1 : 0.91	1 : 0.83
da	gini	8	sqrt	200	0.830986	27	3	0 : 0.90	0 : 0.75	0 : 0.82
						9	32	1 : 0.78	1 : 0.91	1 : 0.84

Iz prethodne tabele zaključujemo da se algoritam Nasumične šume bolje snalazi sa standardizovanim podacima, ali da razlika u kvalitetu sa standardizacijom nije mnogo velika. Primećujemo i da su oba modela dala bolje rezultate na test nego na trening skupu što nam pokazuje da nije došlo do preprilagođavanja trening skupu podataka. Iz matrice konfuzije i recall vrednosti primećujemo da su modeli isti za vrednosti sa i bez standardizacije. Takođe, zaključujemo da bi ova dva modela napravila bitnu grešku samo za 3 osobe iz datog test skupa, koja bi dovela do velikih posledica, što je znatno manje nego kod algoritma Drveta odlučivanja. Ipak, postoji veći broj False Positive instanci, što može dovesti do većeg broja bespotrebno ugrađenih ICD uređaja, prilikom korišćenja ovog modela klasifikacije.

6.3. SVM

Algoritam SVC je testiran korišćenjem tehnike unakrsne validacije (GridSearchCV) za različite vrednosti parametara kernel koji predstavlja tip kernela koji će se koristiti u algoritmu, C koji predstavlja parametar regularizacije i parametra gamma koji predstavlja koeficijent kernela i postoji samo za neke tipove kernela. Vrednosti koje su testirane: kernel={‘linear’, ‘rbf’}, C={0.01, 0.1, 1, 10} i gamma={0.01, 0.1, 1, 10}, samo za ‘rbf’ kernel.

U narednim tabelama su prikazani rezultati rada SVC algoritma na trening skupu, bez i sa standardizacijom, za različite vrednosti parametara navedenih u prethodom pasusu.

Kernel	C	Gamma	Mean_test_score	Std_test_score
linear	0.01	-	0.514286	0.055533
linear	0.1	-	0.600000	0.064594
linear	1	-	0.600000	0.048562
linear	10	-	0.571429	0.067344

rbf	0.01	0.01	0.571429	0.079682
rbf	0.01	0.1	0.561905	0.101686
rbf	0.01	1	0.514286	0.055533
rbf	0.01	10	0.504762	0.038095
rbf	0.1	0.01	0.571429	0.079682
rbf	0.1	0.1	0.561905	0.106053
rbf	0.1	1	0.514286	0.055533
rbf	0.1	10	0.504762	0.038095
rbf	1	0.01	0.590476	0.077372
rbf	1	0.1	0.619048	0.067344
rbf	1	1	0.628571	0.076190
rbf	1	10	0.523810	0.052164
rbf	10	0.01	0.628571	0.081927
rbf	10	0.1	0.628571	0.055533
rbf	10	1	0.628571	0.076190
rbf	10	10	0.523810	0.052164

Tabela rezultata bez standardizacije

Kernel	**C**	**Gamma**	**Mean_test_score**	**Std_test_score**
linear	0.01	-	0.542857	0.071270
linear	0.1	-	0.609524	0.076190
linear	1	-	0.609524	0.121964
linear	10	-	0.600000	0.111066
rbf	0.01	0.01	0.552381	0.077372
rbf	0.01	0.1	0.542857	0.106904
rbf	0.01	1	0.590476	0.098054
rbf	0.01	10	0.533333	0.076190

rbf	0.1	0.01	0.552381	0.077372
rbf	0.1	0.1	0.542857	0.106904
rbf	0.1	1	0.590476	0.098054
rbf	0.1	10	0.533333	0.076190
rbf	1	0.01	0.552381	0.083027
rbf	1	0.1	0.657143	0.081927
rbf	1	1	0.780952	0.077372
rbf	1	10	0.646719	0.064594
rbf	10	0.01	0.600000	0.077372
rbf	10	0.1	0.676190	0.110246
rbf	10	1	0.838095	0.088320
rbf	10	10	0.647619	0.064594

Tabela rezultata sa standardizacijom

Proverom kvaliteta dobijenih rezultata zaključujemo da najbolji rezultat bez standardizacije dobijamo za vrednosti kernel=’rbf’, C=1 i gamma=1, a prilikom korišćenja standardizacije najbolji rezultat se dobija za vrednosti kernel=’rbf’, C=10 i gamma=1.

Primenom ova dva dobijena klasifikatora na test skup podataka dobijamo sledeće rezultate:

Standardizacija	Kernel	C	Gamma	Accuracy score	Confusion matrix		Precision	Recall	F1-score
ne	rbf	1	1	0.704225	35	20	0 : 0.64	0 : 0.97	0 : 0.77
					1	15	1 : 0.94	1 : 0.43	1 : 0.59
da	rbf	10	1	0.929577	34	3	0 : 0.92	0 : 0.94	0 : 0.93
					2	32	1 : 0.94	1 : 0.91	1 : 0.93

Posmatranjem rezultata iz prethodnih tabela možemo zaključiti da se kod SVM algoritama rezultati dosta razlikuju za standardizovane i

nestandardizovane podatke, što nas dovodi do zaključka da je za SVM algoritam jako bitno da se odradi standardizacija podataka.

Za nestandardizovane podatke možemo videti jako malu recall vrednost za klasu 1, što nam govori da ovaj model nije dovoljno dobar, jer ima problema u prepoznavanju i klasifikaciji stvarnih pozitivnih instanci klase 1 (koji su nama najbitniji). Zbog mnogo boljih rezultata na test skupu u odnosu na trening skup, možemo primetiti da se model bolje prilagodio test skupu, što može ukazivati na dobru generalizaciju.

Rezultati kod standardizovanih podataka nam pokazuju jako dobre rezultate korišćenog modela koji je podjednako dobro klasifikovao i instance klase 0 i instance klase 1, i za koji su skoro isti rezultati dobijeni na trening i test skupu.

6.4. ANN

Algoritam ANN je testiran korišćenjem tehnike unakrsne validacije (GridSearchCV) za različite vrednosti parametara batch_size koji predstavlja broj instanci koje uzima prilikom jednog koraka gradijentnog spusta i epochs koji predstavlja broj prolazaka za trening skup. Vrednosti koje su testirane: batch_size={32, 50} i epochs={20, 50, 100}. Za kreiranje neuronske mreže korišćen je KerasClassifier iz scikeras.wrappers biblioteke.

U narednim tabelama su prikazani rezultati rada ANN algoritma na trening skupu, bez i sa standardizacijom, za različite vrednosti parametara navedenih u prethodom pasusu.

Batch_size	Epochs	Mean_test_score	Std_test_score
32	20	0.495238	0.013469
32	50	0.495238	0.013469
32	100	0.609524	0.035635

50	20	0.495238	0.013469
50	50	0.466667	0.013469
50	100	0.504762	0.058709

Tabela rezultata bez standardizacije

Batch_size	Epochs	Mean_test_score	Std_test_score
32	20	0.476190	0.035635
32	50	0.476190	0.013469
32	100	0.533333	0.035635
50	20	0.466667	0.035635
50	50	0.457143	0.000000
50	100	0.485714	0.000000

Tabela rezultata sa standardizacijom

Proverom kvaliteta dobijenih rezultata zaključujemo da najbolji rezultat bez standardizacije i sa standardizacijom dobijamo za vrednosti batch_size=32 i epochs=100.

Primenom ova dva dobijena klasifikatora na test skup podataka dobijamo sledeće rezultate:

Standardizacija	Batch size	Epochs	Accuracy score	Confusion matrix		Precision	Recall	F1-score
ne	32	100	0.521127	20	18	0 : 0.53	0 : 0.56	0 : 0.54
				16	17	1 : 0.52	1 : 0.49	1 : 0.50
da	32	100	0.619718	22	13	0 : 0.63	0 : 0.62	0 : 0.62
				14	22	1 : 0.61	1 : 0.62	1 : 0.62

Iz dobijenih rezultata zaključujemo da kod nestandardizovanih podataka imamo preprilagođavanje trening skupu podataka, jer su rezultati dobijeni na trening skupu znatno bolji od rezultata na test skupu podataka. Primećujemo i da je model algoritma ANN nedovoljno dobar za ove podatke.

Bolji rezultati su dobijeni za standardizovane podatke, ali i dalje ti rezultati nisu dovoljno dobri da bi ovaj model bio korišćen.

Nakon ovoga je testirano koji je najbolji optimizer od nekoliko izabranih i da li će se promenom optimizera dobiti bolji rezultati. Korišćen je isti model i vrednosti parametara batch_size i epochs za koje je određeno da su najbolji.

Dobijeni su sledeći rezultati:

Optimizer	Mean_test_score	Std_test_score
SGD	0.504762	0.088320
RMSprop	0.514286	0.061721
Adagrad	0.523810	0.071270
Adadelta	0.542857	0.046657
Adam	0.571429	0.101686
Adamax	0.523810	0.048562
Nadam	0.561905	0.013469

Tabela rezultata bez standardizacije

Optimizer	Mean_test_score	Std_test_score
SGD	0.628571	0.000000
RMSprop	0.666667	0.142539
Adagrad	0.657143	0.061721
Adadelta	0.590476	0.067344

Adam	0.590476	0.094281
Adamax	0.676190	0.094281
Nadam	0.685714	0.061721

Tabela rezultata sa standardizacijom

Proverom kvaliteta dobijenih rezultata zaključujemo da najbolji rezultat bez standardizacije dobijamo za optimizer=’Adam’, a sa standardizacijom najbolji rezultat dobijamo za optimizer=’Nadam’.

Primenom ova dva dobijena klasifikatora na test skup podataka dobijamo sledeće rezultate:

Standardizacija	**Optimizer**	**Accuracy score**	**Confusion matrix**		**Precision**	**Recall**	**F1-score**
ne	Adam	0.535211	18	15	0 : 0.55	0 : 0.50	0 : 0.52
			18	20	1 : 0.53	1 : 0.57	1 : 0.55
da	Nadam	0.718310	25	9	0 : 0.74	0 : 0.69	0 : 0.71
			11	26	1 : 0.70	1 : 0.74	1 : 0.72

Primećujemo da su rezultati dobijeni korišćenjem drugih tipova optimizera bolji za podatke sa standardizacijom, ali da je kod nestandardizovanih podataka kvalitet ostao isti promenom optimizera. Međutim, ti rezultati i dalje nisu dovoljno kvalitetni da bi se ovaj model koristio, jer je SVM algoritam postigao značajno bolje rezultate.

6.5. Bajesovski klasifikator

Primenom MultinomialNB algoritma dobijemo model čiji je score nad trening podacima bez standardizacije jednak 0.676190.
Rezultati nad test skupom su prikazani u narednoj tabeli:

Accuracy score	Confusion matrix		Precision	Recall	F1-score
0.619718	11	2	0 : 0.31	0 : 0.85	0 : 0.45
	25	33	1 : 0.94	1 : 0.57	1 : 0.71

Iz dobijenih rezultata možemo zaključiti da je došlo do malog preprilagođavanja trening skupu.Vidimo da je precision veliki za klasu 1, što nam govori da kada model označi nešto kao klasu 1, to je gotovo sigurno tačno, ali je problem u niskom recallu što nam sugeriše da postoji značajan broj pozitivnih instanci koje model ne uspeva ispravno prepoznati, što može predstavljati veliki problem u našem zadatku.

7. Zaključak

Testirane su 5 različitih klasifikacionih metoda (SVM, Random forest, Stablo odlučivanja, Naive Bayes i ANN), na skupu standardizovanih i nestandardizovanih podataka. Ponovnim pregledanjem svih dobijenih modela, možemo zaključiti da je standardizacija dala bolje rezultate kod svakog klasifikacionog metoda i da je najbolji korišćen metod bio SVM sa parametrima: kernel=’rbf’, C=10, gamma=1.

8. Pokretanje projekta

Kompletan projekat je napisan u programskom jeziku Python (verzija 3.9.5), u okruženju Jupyter Notebook, da bi kod bio prenosiv i izvršiv na različitim sistemima. Da bi mogli da pokrenete ovaj projekat potrebno je instalirati Python.

Potrebno je instalirati i biblioteke korišćene na ovom projektu:

1. Biblioteku pandas komandom: **pip3 install pandas**
2. Biblioteku numpy komandom: **pip3 install numpy**
3. Biblioteku matplotlib komandom: **pip3 install matplotlib**
4. Biblioteku imblearn komandom: **pip3 install imblearn**
5. Biblioteku tensorflow komandom: **pip3 install tensorflow**
6. Biblioteku scikeras komandom: **pip3 install scikeras**
7. Biblioteku seaborn komandom: **pip3 install seaborn**

Pre pokretanja programa u okruženju Jupyter Notebook potrebno je izmeniti liniju koda *data=pd.read_csv(“*putanja*”),*tako da putanja odgovara putanji na kojoj se podaci nalaze na korisničkom računaru.

Literatura

1) https://kardiologija.in.rs/srcana_slabost.htm
2) https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm
3) https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html
4) https://www.javatpoint.com/machine-learning-random-forest-algorithm
5) https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
6) https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm
7) https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
8) http://solair.eunet.rs/~ilicv/neuro.html#Sta su to NM
9) https://www.tensorflow.org/tutorials
10) https://www.javatpoint.com/machine-learning-naive-bayes-classifier
11) https://scikit-learn.org/stable/modules/naive_bayes.html