

SVEUCILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 1

Predviđanje sportskih rezultata korištenjem neuronske mreže optimirane genetskim algoritmom

Dina Zjača

Zagreb, srpanj 2010.

Sadržaj

1. Uvod	1
2. Oblikovanje ulaznih podataka neuronske mreže	2
2.1 Dobivanje podataka i baza	2
2.2 Ulazni atributi neuronske mreže	3
3. Neuronske mreže	5
3.1 Računalni program <i>Weka</i>	5
3.2 Podešavanje parametara mreže	5
3.2.1 Determinizam neuronske mreže	5
3.2.2 Trajanje učenja	6
3.2.3 Stopa učenja	6
3.2.4 Broj čvorova srednjeg sloja	7
3.3 Podešavanje parametara podataka	8
3.3.1 Nedavne utakmice	8
3.3.2 Grupiranje sezona	9
3.3.3 Izostavljene utakmice	10
3.3.4 Ispitivanje dobivene mreže na ligi druge kvalitete	12
3.3.5 Učenje i ispitivanje mreže na ligi drugoge kvalitete	12
4. Genetski algoritam	14
4.1 Ideja	14
4.2 Implementacija	14
4.2.1 Prikaz jedinki	14
4.2.2 Operatori	15
4.2.3 Pseudokod algoritma	16
4.2.4 Računalni program	18
4.3 Podešavanje parametara genetskog algoritma	20
4.3.1 Broj generacija	21

4.3.2	Veličina populacije.....	22
4.3.3	Operatori GA	23
4.4	Analiza važnosti atributa	25
4.4.1	Analiza broja atributa koji nedostaju	25
4.4.2	Utjecaj jednog redundantnog atributa	26
4.4.3	Utjecaj dva redundantna atributa	27
4.4.4	Utjecaj više redundantnih atributa	29
4.4.5	Složenost i vrijeme izvođenja	30
5.	Primjena i nedostaci	32
6.	Zaključak.....	34
7.	Literatura	35
8.	Sažetak	36

1. Uvod

Predmet ovog diplomskog rada je korištenje umjetne inteligencije za predviđanje rezultata sportskih utakmica, te analiziranje utjecaja pojedinih statističkih značajki na ishode utakmica. Izabrana je umjetna inteligencija jer je pogodna za rješavanje problema velikog prostora ulaznih podataka, te velikog prostora pretraživanja rješenja. Za analizirani sport odabran je nogomet, vrlo velikim dijelom zbog dostupnosti veće količine podataka, potrebnih za rad s neuronskom mrežom. Cilj rada je podatke o nogometnim utakmicama svesti na problem pogodan za rješavanje neuronskoj mreži, te pomoću genetskog algoritma optimirati izbor podataka koje mreža prima. Taj se proces sastoji od nekoliko koraka:

1. prikupljanje podataka,
2. izbor bitnih značajki i oblikovanje ulaza neuronske mreže,
3. učenje neuronske mreže,
4. optimiranje parametara mreže,
5. izrada računalanog programa za genetski algoritam
6. optimiranje parametara genetskog algoritma,
7. analiza utjecaja ulaza neuronske mreže.

Dobiveni rezultati stvaraju sliku o tome koliko je pojedini izabrani parametar utjecajan na uspješnost neuronske mreže. U drugom poglavlju 'Oblikovanje ulaznih podataka neuronske mreže' opisano je kako su prikupljeni nogometni podaci, kako su od njih dobiveni ulazni podaci za neuronsku mrežu, te kako je oblikovan problem predviđanja rezultata. U trećem poglavlju 'Neuronske mreže' opisan je računalni program za neuronske mreže – Weka, provedeno je učenje i optimiranje parametara neuronske mreže, te analizirani dobiveni rezultati. U četvrtom poglavlju 'Genetski algoritam' objašnjena je uloga genetskog algoritma, predstavljene osnove napravljenog računalnog programa, optimirani njegovi parametri, te analizirani rezultati.

2. Oblikovanje ulaznih podataka neuronske mreže

Neuronske mreže su vrsta tehnike umjetne inteligencije koje oponašaju rad grupe povezanih neurona. Najčešće se sastoje od 3 sloja: ulazni, skriveni i izlazni, iako skrivenih slojeva može biti i više. Neuroni iz različitih slojeva su povezani vezama različitih težina. Ulazni neuroni primaju podatke brojanog oblika, te ih proslijeđuju neuronima s kojima su povezani, umnožene za težinu poveznice. Na isti se način proslijeđuju i podaci iz skrivenog sloja u izlazni. Prilikom učenja neuronske mreže, mreža prima veću količinu podataka, te se težine poveznica neurona prilagođavaju.

S obzirom da ulaz neuronske mreže mora biti brojanog oblika, neuronska mreža ne može učiti na nominalnim podacima (pod 'nominalnim' se smatraju podaci koji mogu poprimiti nebrojčane vrijednosti), već na brojčanim vrijednostima, koje su najčešće statističke značajke određenog problema. Primjerice, ime tima koji igra zadanu utakmicu ne može biti predano kao ulaz neuronskoj mreži.

Kako bi obične podatke pretvorili u oblik pogodan za ulaz u neuronsku mrežu, odabrane su statističke značajke koje predstavljaju svaku utakmicu na kojima se neuronska mreža uči. Iako su statistički podaci točni i objektivni, sam izbor značajki sporta koje će biti prikazane statistikom je subjektivan. Iz tog je razloga odabran poveći broj značajki, a optimizacija se vršila genetskim algoritmom, eliminacijom pojedinih značajki.

Što se tiče izlaza neuronske mreže, on je nominalan, tj. predstavlja pobjedu, poraz ili izjednačeno. Tako je problem sveden na klasifikaciju, i pogodan za rješavanje neuronskoj mreži.

2.1 Dobivanje podataka i baza

Korišteni podaci preuzeti su sa stranice sa nogometnim rezultatima [1], a sastoje se od rezultata pojedine utakmice, imena timova, lige u kojoj je utakmica odigrana, te sezone i datuma. Baza sa dobivenim podacima sastoji se od tri tablice, sa pripadajućim atributima:

- *League*: LeagueId, *LeagueName*, *Country*
- *Team*: TeamId, *TeamName*, *Country*

- *Match*: *MatchId*, *TeamAId*, *TeamBId*, *ScoreA*, *ScoreB*, *LeagueId*, *Season*, *Date*

Dotična stranica je izabrana između drugih stranica koje posjeduju podatke o minutama golova, postavi tima i sl., zbog velikog opsega liga i sezona.

2.2 Ulazni atributi neuronske mreže

Svaka utakmica predstavljena je značajkama izraženima kao statistički podatak. Odabrane značajke utakmice preuzete su iz sličnog rada [2], te označavaju:

- *HomeTeam_MatchesPlayed* (broj utakmica koje je domaćin odigrao u toj sezoni do sada)
- *HomeTeam_LeaguePoints* (koliko je domaćin skupio bodova u svojoj ligi u toj sezoni do sada, pobjeda se boduje 3 boda, a izjednačeno 1)
- *HomeTeam_Home_Won* (koliko je domaćin utakmica pobijedio u toj sezoni do sada, igrajući na domaćem terenu)
- *HomeTeam_Home_Drawn* (koliko je domaćin utakmica odigrao neriješeno u toj sezoni do sada, igrajući na domaćem terenu)
- *HomeTeam_Home_Lost* (koliko je domaćin utakmica izgubio u toj sezoni do sada, igrajući na domaćem terenu)
- *HomeTeam_Home_GoalsScored* (koliko je golova domaćin postigao u toj sezoni do sada, igrajući na domaćem terenu)
- *HomeTeam_Home_GoalsConceded* (koliko je golova domaćin primio u toj sezoni do sada, igrajući na domaćem terenu)
- *HomeTeam_Away_Won* (koliko je domaćin utakmica pobijedio u toj sezoni do sada, igrajući u gostima)
- *HomeTeam_Away_Drawn* (koliko je domaćin utakmica odigrao neriješeno u toj sezoni do sada, igrajući u gostima)
- *HomeTeam_Away_Lost* (koliko je domaćin utakmica izgubio u toj sezoni do sada, igrajući u gostima)

- *HomeTeam_Away_GoalsScored* (koliko je domaćin golova postigao u toj sezoni do sada, igrajući u gostima)
- *Hometeam_Away_GoalsConceded* (koliko je domaćin golova primio u toj sezoni do sada, igrajući u gostima)
- *HomeTeam_RecentForm_Points* (koliko je domaćin skupio bodova u toj ligi u zadnjih nekoliko utakmica, tj. u kakvoj je tim trenutno formi)

Također, isti su atributi definirani i za gostujuć tim, kao i izlazni atribut *Class*. Svi atributi su realnog tipa, osim atributa *Class* koji nominalan, te može poprimiti vrijednosti:

- 1, kada domaćin pobijedi
- 0.5, kada utakmica završi neriješeno
- 0, kada domaćin izgubi

S obzirom da je atribut *Class* nominalnog tipa svejedno je koliko pojedina vrijednost iznosi, već je bitno samo da se radi o tri različite vrijednosti.

Jedini atributi čija definicija nije u potpunosti preuzeta iz spomenutog rada [2] je *HomeTeam_RecentForm_Points* (i *AwayTeam_RecentForm_Points*), koji označavaju trenutnu formu tima domaćina, odnosno gostujućeg tima. U radu [2], forma je definirana kao bodovi skupljeni u dosadašnjoj igri u toj ligi te sezone, u zadnje 3 utakmice. Pobjeda je bodovana sa 3, izjednačen rezultat sa 1 bodom, a poraz s 0 bodova. U različitim izvorima, koji se bave nogometnom tematikom, taj je podatak slično definiran, dok razlike postoje u broju posljednjih utakmica koje ulaze u računicu. Iz tog razloga, broj posljednjih utakmica koje utječu na trenutnu formu tima se u ovom radu dinamički definira prilikom dobivanja podataka iz baze. Kasnije je empirijski utvrđeno s kojim će se brojem posljednjih utakmica izvoditi daljnja ispitivanja.

Također, prilikom optimiranja ulaznih podataka za neuronsku mrežu, ispitana je uspješnost atributa dobivenih tokom jedne i više sezona, što pak utječe na definiciju svih atributa.

Radi širokog opsega podataka, podaci korišteni za učenje i ispitivanje su iz samo jedne lige, engleske.

3. Neuronske mreže

Zadatak je prilagođen na problem klasifikacije, koji je pogodan za rješavanje neuronskom mrežom. Zbog postojanja skupa parova ulaznih i izlaznih podataka, izabrano je nadgledano učenje (engl. *supervised learning*). Izabran je jednostavan model jednosmjerne (engl. *feedforward*) mreže, točnije višeslojni perceptron (engl. *multilayered perceptron*). Za algoritam učenja mreže izabran je *backpropagation*, najistraženiji i u zadnje vrijeme najpopularniji algoritam za ovu vrstu neuronskih mreža.

3.1 Računalni program Weka

Za rad s neuronskim mrežama korišten je gotov *open source* računalni program, Weka. Nastao je na Sveučilištu Waikato (engl. *University of Waikato*) [3], a ostvaruje više različitih oblika algoritama korištenih za predviđanja, kao što su stabla odluke (engl. *Decision trees*), pravila klasifikacije (engl. *Classification rules*), neuronske mreže i sl.

Prilikom uporabe neuronske mreže podešava se više parametara. U ovom su radu optimirani samo oni glavni, a ostalima su ostavljene pretpostavljene vrijednosti.

3.2 Podešavanje parametara mreže

U ovom su radu podešeni osnovni parametri neuronske mreže: vrijeme treniranja, stopa učenja, broj čvorova skrivenog sloja, kao i parametri ulaznih podataka neuronske mreže. To su broj posljednjih utakmica u kojima se boduje trenutna forma tima, broj sezona na kojima se računa svaki atribut, te broj početnih utakmica koji će biti zanemareni. S obzirom da je područje umjetne inteligencije još uvijek velikim dijelom eksperimentalno, svaki od tih faktora podešavan je ispitivanjem. S obzirom da je svaki faktor prilagođavan jedan put, faktori koji su bili podešavani među prvima prikazuju dosta loše rezultate, i nije najjasnije koju vrijednost bi trebalo odabrati za daljnja ispitivanja. Važno je napomenuti da tako dobivene vrijednosti parametara nisu nužno i najbolje.

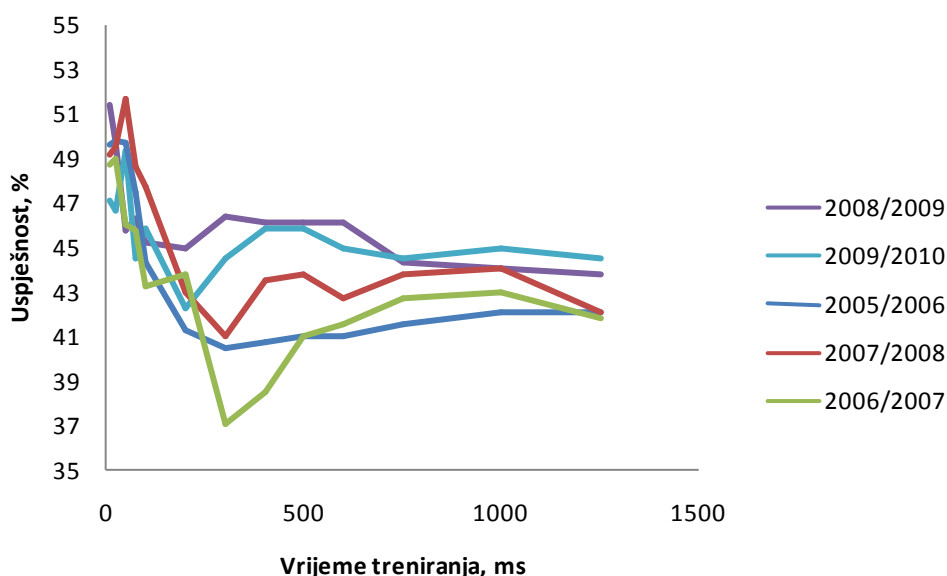
3.2.1 Determinizam neuronske mreže

S obzirom da je cilj izraditi program koji optimira neuronsku mrežu, pretpostavka je da će ona biti pokretana prilikom promjene bilo kojeg parametra genetskog algoritma. Kako bi se dobili što precizniji rezultati, jedno od prvih ispitivanja bilo je da li je neuronska mreža deterministička. U slučaju da nije, bilo bi potrebno definirati na koliko će se pokretanja mreže uspješnost

mjeriti, te da li će se koristiti najbolja ili srednja vrijednost. S obzirom da je već prilikom prvih ispitivanja uočeno kako jednako podešene neuronske mreže nakon neznatno malog broja iteracija daju jednake rezultate, to nije dalje ispitivano i neuronske mreže su tretirane kao determinističke.

3.2.2 Trajanje učenja

Vrijeme učenja označava koliko će dugo neuronska mreža podešavati težine neuronskih veza s obzirom na primjere za učenje. U programu Weka ono je izraženo u milisekundama. Opseg za ispitivanje izabran je oko pretpostavljene vrijednosti koju program sam postavlja s obzirom na broj ulaznih i izlaznih atributa. Na slici 3.1 vide se vremena učenje za 5 različitih sezona i njihove uspješnosti. Uspješnost ispitivanja se odnosi na broj ispravno klasificiranih instanci iz skupa za ispitivanje.



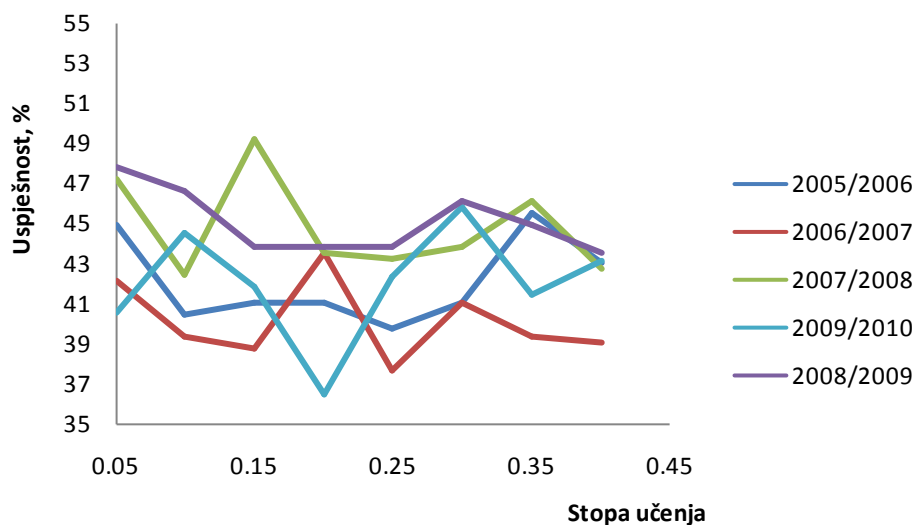
Slika 3.1 Vrijeme učenja

Ostali parametri su još uvijek nepodešavani. Za daljnja ispitivanja izabrana je vrijednost 500ms, jer se početni maksimum, vidljiv na grafu 3.1, pripisuje tome što mreža nije stigla pretjerano puno naučiti, već najčešću vrijednost izlaza dodjeljuje većini instanci.

3.2.3 Stopa učenja

Stopa učenja (engl. *learning rate*) označava kojom će se brzinom težine poveznica neurona prilagođavati pogrešci dobivenoj u usporedbi dobivenog rezultata sa stvarnim rezultatom u primjerima za učenje. Postavljajući veću stopu učenja, potencira se učenje unutar manjeg broja iteracija, ali se i

povećava mogućnost nekonvergiranja težina neuronskih veza. Ispitivanje stope učenja vidljivo je na grafu 3.2.

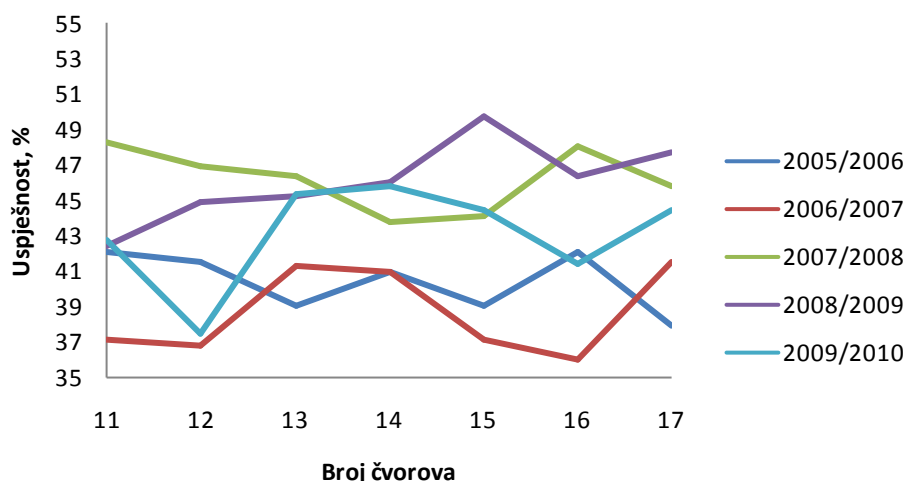


Slika 3.2 Stopa učenja

Izabrana je stopa učenja od 0.3.

3.2.4 Broj čvorova srednjeg sloja

Postavljanje broja čvorova srednjeg sloja jedna je od mnogih nedefiniranih značajki neuronskih mreža. Postoji nekoliko različitih preporuka o njihovom broju [4]. U ovom radu se uvažava preporuka da broj čvorova srednjeg sloja treba biti srednja vrijednost broja čvorova ulaznog i izlaznog sloja. S obzirom da je broj ulaznih neurona jednak broju atributa i iznosi 26, a izlaz je samo jedan, srednja vrijednost iznosi 13.5. Ispitivane su vrijednosti u okolini tog broja, a na slici 3.3 su vidljivi rezultati.



Slika 3.3 Broj čvorova

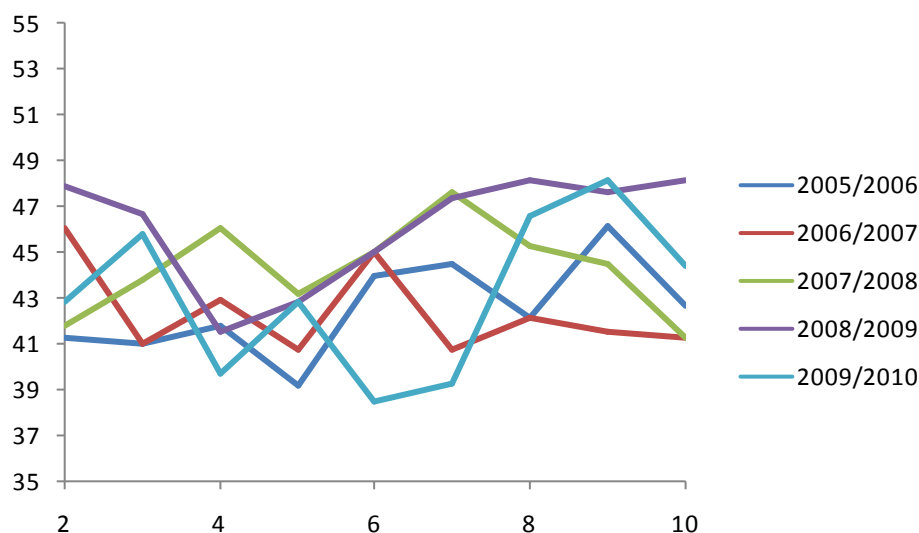
Rezultati ispitivanja pokazuju male razlike s obzirom na broj čvorova. Izabrano je 14 čvorova kako predlaže početna preporuka.

3.3 Podešavanje parametara podataka

Već je spomenuto da je izbor podataka koji čine ulaz u neuronsku mrežu subjektivan. Iz tog razloga i neki od parametara podataka ostavljeni su promijenjivi, te podešeni ispitivanjem.

3.3.1 Nedavne utakmice

Već je spomenuto kako je atribut koji se odnosi na trenutnu formu tima (*HomeTeam_RecentForm_points* i *AwayTeam_RecentForm_Points*) ostavljen djelomično nedefiniran, te da će se se ispitivanjem utvrditi koliko će zadnjih utakmica utjecati na trenutnu formu tima. Inače, on označava broj bodova postignutih u zadnjih nekoliko utakmica, s tim da se pobjeda boduje sa 3 boda, a izjednačeno sa 1 bodom. Na grafu 3.4 vide se uspješnosti različito definiranog atributa.



Slika 3.4 Broj utakmica koje određuju trenutnu formu

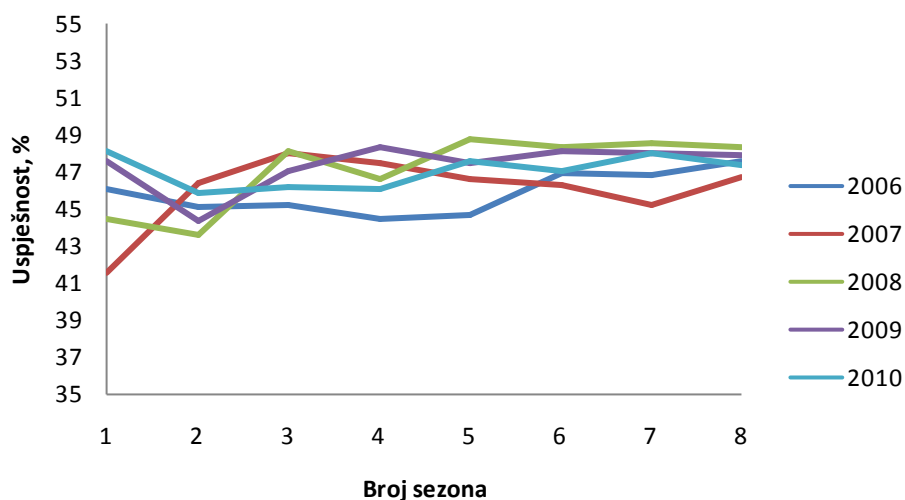
Izabrano je da će u daljnjim ispitivanjima atribut biti definiran sa posljednjih 9 utakmica. Na prvi pogled izgleda nelogično, ali preuzeta je posljednja vrijednost sa grafa i nisu rađena daljnja ispitivanja tog atributa. To je iz razloga što postoje atributi *HomeTeam_LeaguePoints* i *AwayTeam_LeaguePoints* koji označavaju sve bodove skupljne igrajući te godine, a bodovani su na jednak način kao i trenutna forma (3 boda pobjeda, 1 bod izjednačeno, a poraz 0 bodova). Što više opseg atributa *HomeTeam_RecentForm_points* i *AwayTeam_RecentForm_Points* raste, to se on više približava drugom atributu. Vrlo je vjerojatno kako bi taj atribut postizao jednako dobre rezultate i sa manjom vrijednošću, da je optimiziran nakon eliminacije početnih utakmica i grupiranja sezona. Povećavanjem broja posljednjih utakmica, grupiranjem sezona i eliminacijom početnih utakmica, postiže se veća koncentracija podataka.

Bitno je još napomenuti da zadnja sezona možda daje nešto lošije rezultate, vidljivo na grafu 3.4. To je vjerojatno posljedica najmanjeg broja utakmica. Svaka se sezona sastoji od oko 350 utakmica, dok je posljednja u bazi popunjena samo sa 220. Izrada ovog rada je počela prije završetka lige, pa zadnjih 100-tinjak utakmica u bazi nedostaje.

3.3.2 Grupiranje sezona

Drugo prilagođavanje podatkovnih parametara sastojalo se od grupiranja sezona. Na taj način postignuto je učenje na više sezona istovremeno, kao i veći broj podataka za učenje i ispitivanje. Na grafu 3.3 vidi se ovisnost

uspješnosti neuronske mreže o broju sezona ulaznih podataka na nekoliko različitih skupova godina. Godina označava zadnju godinu u skupu.

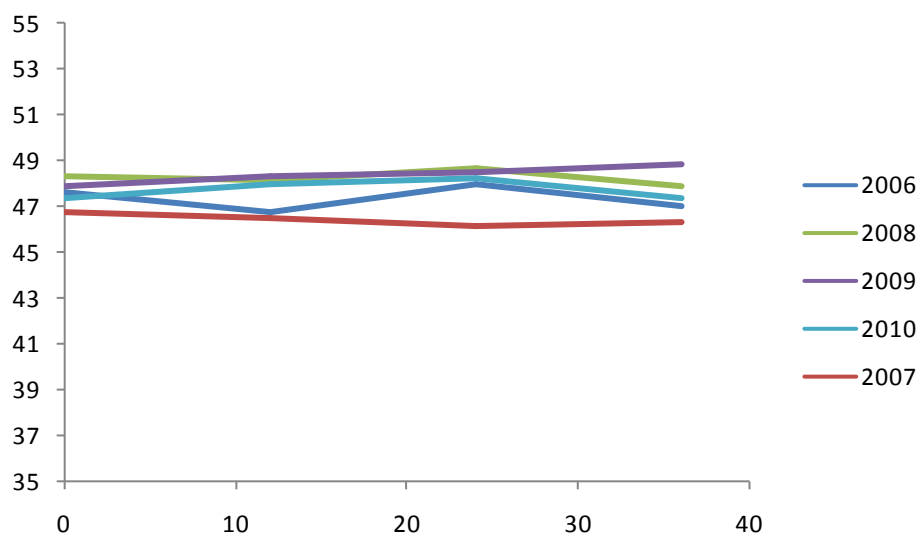


Slika 3.5 Broj sezona

Izabrano je 8 sezona za daljnje podešavanje parametara. Nije ispitivana uspješnost na više sezona, jer se kao i u podešavanju broja utakmica koje utječu na formu, smatralo da će se sličan ili bolji efekt postići sa eliminacijom početnih utakmica, koja slijedi.

3.3.3 Izostavljene utakmice

Posljednje prilagođavanje parametara podataka je izostavljanje početnih utakmica svake sezone. S obzirom da je svaka utakmica predstavljena atributima koji statistički izražavaju postignuto u toj ligi do trenutka odigravanja spomenute utakmice, početne utakmice sadrže jako malo podataka. Prvih nekoliko sadrže samo nule, i neuronskoj mreži stvaraju krivu sliku. Graf 3.6 prikazuje uspješnost mreže u ovisnosti o broju početnih utakmica koje nedostaju.



Slika 3.6 Izostavljene utakmice

Izabrano je da će u ulaznim podacima za učenje na dalje nedostajati 24 utakmice. U tablici 3.1 prikazane su vrijednosti svih parametara dobivenih optimiranjem.

Tablica 3.1 Optiirani parametri neuronske mreže

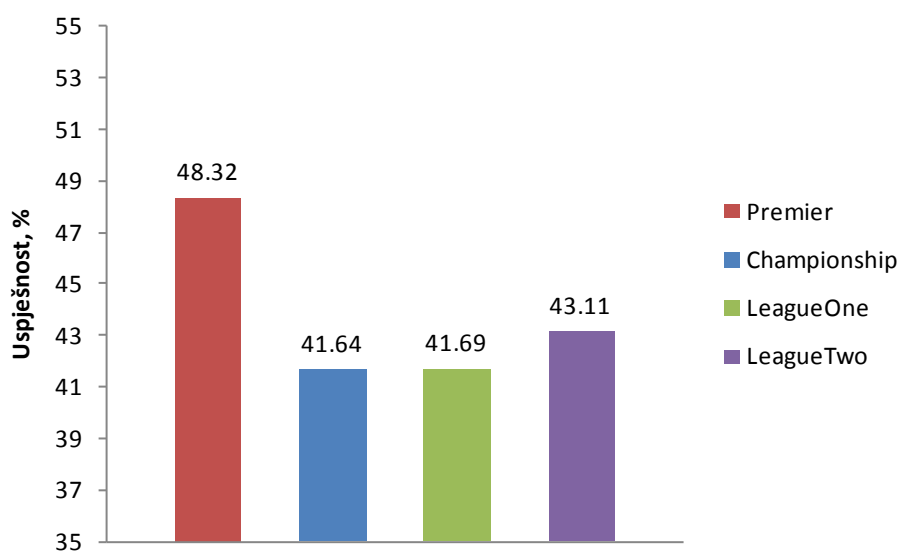
Trajanje učenja	500ms
Stopa učenja	0.3
Broj čvorova srednjeg sloja	14
Broj utakmica koje određuju trenutnu formu	9
Broj sezona	8
Izostavljeni početni mečevi	24

Sveukupnim podešavanjem parametara neuronske mreže i parametara podataka dobivena je prosječna točnost između 46 i 48%. Prije podešavanja ona je iznosila u prosjeku između 41 i 43%. Iako nije savršeno, podešavanje parametara jedan po jedan, pokazalo se korisno.

3.3.4 Ispitivanje dobivene mreže na ligi druge kvalitete

Nakon podešavanja parametara ispitivana je točnost neuronske mreže na drugim ligama. Prvo je učena na dosadašnjim podacima iz *Premier league*, te ispitivana uspješnost na drugim ligama, a zatim je učena i ispitivana točnost na istoj ligi, sa parametrima podešenim na podacima iz *Premier League*. Na taj način vidljivo je da li su mreže učene na jednoj ligi pogodne za predviđanja rezultata drugih liga, te da li su parametri podešavani ispitivanjem na jednoj ligi, iskoristivi za predviđanje rezultata drugih liga.

Na slici 3.7 prikazane su uspješnosti liga različitih kvaliteta. Korištene su engleske lige - *Premier league* se odnosi na najbolju državnu ligu, *Championship* na drugu državnu ligu, *League One* na treću, te *League Two* na četvrtu.

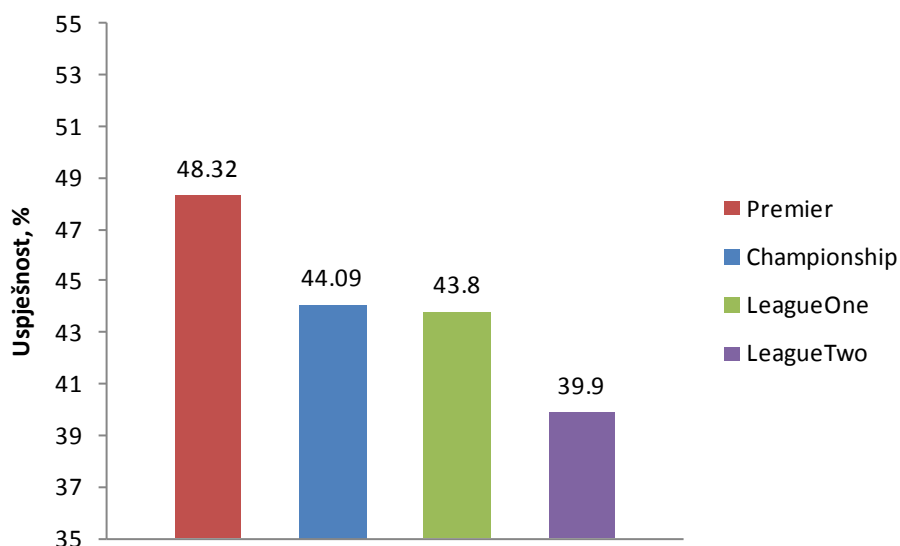


Slika 3.7 Lige učene na Premier ligi

Vidljivo je da uspješnost ispitivanja na ostalim ligama nije niti približa uspješnosti na ligi korištenoj za učenje.

3.3.5 Učenje i ispitivanje mreže na ligi drugoge kvalitete

Nakon toga napravljeno je učenje i ispitivanje na podacima svake lige zasebno. Rezultati su vidljivi na slici 3.8.



Slika 3.8 Lige učene i ispitvane zasebno

Vidljivo je kako je točnost mreža za nekoliko postotaka bolja kada je mreža učena i ispitivana na utakmicama iz iste lige, što je i logično. Također, na slici 3.8 vidljivo je da je uspješnost lige ovisna i od njene kvalitete – što je liga bolje kvalitete, to joj je i bolja točnost. Iako, nije sasvim jasno zašto do toga dolazi, pretpostavlja se da to može biti iz dva razloga.

Prvi je zbog parametara podešavanih na *Premier league*. Logično je da svaka liga postigne najbolju točnost na parametrima podešavanim na njoj samoj, a lige bolje kvalitete imaju više sličnosti sa *Premier league*.

Drugi razlog je da lige lošije kvalitete jednostavno imaju nepredvidljivije rezultate.

Ja sam se sklona prikloniti prvoj tezi, iz razloga što su točnosti liga nešto lošije od onih dobivenih na *Premier league*. I smatram kako bi svaka liga imala nešto bolju točnost da su parametri podešavani na njenom učenju.

Koji god da je razlog lošije točnosti ostalih liga, vidljivo je kako ni korištenje mreža učenih na jednoj ligi, ni parametara podešavanih na jednoj ligi, nije pogodno za ostale lige.

4. Genetski algoritam

4.1 Ideja

Već je spomenuto kako neuronska mreža ima 26 ulaznih atributa, koji predstavljaju odabrane značajke za svaku utakmicu. Odabir značajki je subjektivan, te ih je zato izabran velik broj, kako se ne bi neka bitna karakteristika izostavila. Potom je genetskim algoritam optimiran izbor značajki eliminirajući pojedine značajke. Svaka jedinka genetskog algoritma zapravo je popis eliminiranih ulaza neuronske mreže. Može nedostajati između 0 i 8 ulaznih atributa, to čini prostor od približno 20 milijuna jedinki.

Parametri genetskog algoritma su optimirani jedan po jedan, a zatim je analizirano koji ulazni atributi najčešće nedostaju kod najboljih jedinki.

4.2 Implementacija

Odabran je jednostavan genetski algoritam, sa generacijskom selekcijom uz odabir pomoću kotača ruleta (engl. *roulette wheel*), nekoliko različitih operatora križanja i mutacije, te opcija poput elitizma i translacije (engl. *windowing*)[5]. Program je pisan u programskom jeziku Java, te alatu Eclipse.

4.2.1 Prikaz jedinki

Jedinke su kodirane poprilično jednostavno, na dva međusobno izmjenjiva načina. Prvi se sastojao od liste indeksa atributa koji nedostaju 4.1, a drugi od liste nula i jedinica 4.2, gdje nula označava nedostajanje pojedinog atributa. Prvi način nastao je iz potrebe za ocjenjivanjem jedinke u već gotovom računalnom programu za neuronske mreže, a drugi radi jednostavnije primjene binarnih operatora. U oba primjera 4.1 i 4.2, jedinki nedostaju isti atributi.

5	6	12	18	20	24
---	---	----	----	----	----

Slika 4.1 Lista nedostajućih atributa nebinarno kodirane jedinke

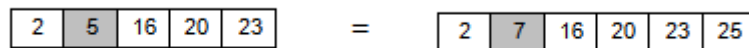
1	1	1	1	0	0	1	1	1	1	1	0	1	1	1	1	0	1	0	1	1	1	0	1	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

Slika 4.2 Binarno kodirana jedinka

4.2.2 Operatori

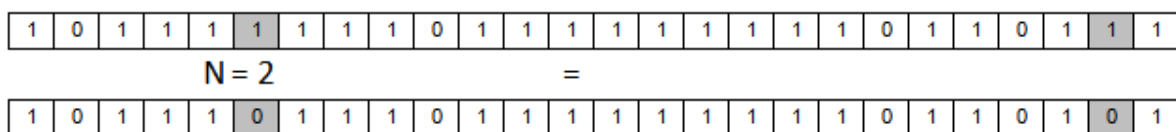
U okviru rada napravljene su dvije vrste mutacije, te tri vrste križanja.

MissingList mutacija mutira jedinke kodirane kao liste atributa. Slučajnim odabirom traži se broj atributa koji će nedostajati u novonastaloj jedinki. Ako je taj broj veći od broja atributa stare jedinke, razlika broja atributa se nadoknađuje odabirom novih atributa. Zatim se, bez obzira na veličinu jedinke, mijenja nasumični gen, tj. jedan od atributa. Primjer takve mutacije prikazan je na slici 4.3.



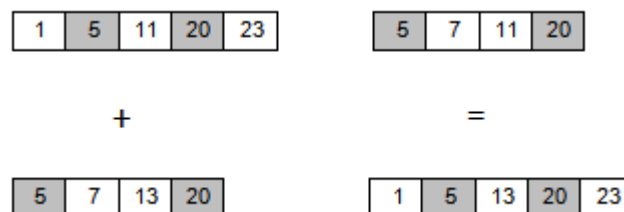
Slika 4.3 MissingList mutacija

PointMutation mutira jedinke kodirane binarno. Ukoliko je izabrana ova mutacija, prije pokretanja programa postavlja se N , broj gena koji će biti mutirani. Postupak se sastoji od slučajnog odabira N gena koji će biti promijenjeni. Postupak je prikazan na slici 4.4.



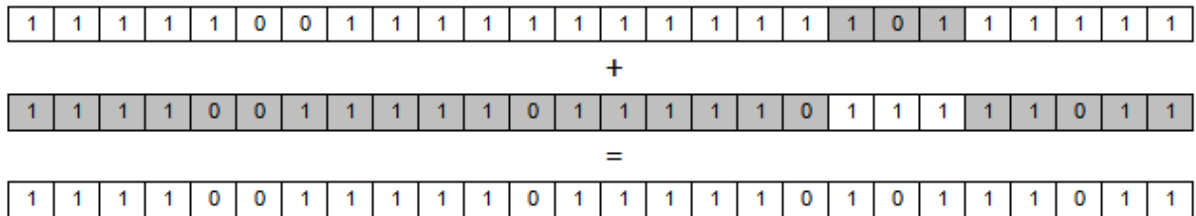
Slika 4.4 PointMutation mutacija

MissingList križanje napravljeno je sa jedinkama kodiranim kao liste nedostajućih atributa. Od dva roditelja nastaje dvoje djece. Postupak se sastoji od toga da oba djeteta naslijede nedostajuće attribute zajedničke roditeljima, te im se slučajnim odabirom dodjeljuju ostali nedostajući atributi roditelja. Primjer jednog takovog križanja vidljiv je na slici 4.5.



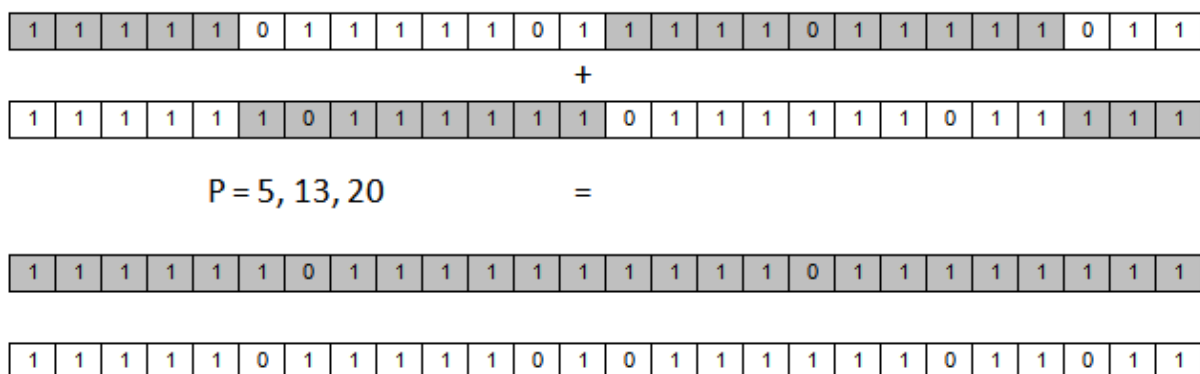
Slika 4.5 MissingList križanje

BinaryUniformCross križanje je jednostavno uniformno križanje. Također je napravljeno za binarno kodirane jedinice. Od dva roditelja, nasumičnim odabirom svakog gena nastaje jedno dijete. Postupak je dosta jednostavan, i prikazan na slici 4.6.



Slika 4.6 BinaryUniformCross križanje

PointCross je također križanje za binarno kodirane jedinice. Prije pokretanja programa postavlja se broj točki prekida. Postupak je takav da od dva roditelja nastaje dvoje djece. Prvo se slučajnim odabirom generiraju točke prekida. Svaka točka prekida označava indeks jednog gena. U početku svako diejte nasljeđuje gene od jednog roditelja, dok ne dođe do točke prekida. Tada se roditelji zamijene, i tako na svakoj točki prekida. Postupak je prikazan na slici 4.7.



Slika 4.7 pointCross_križanje

4.2.3 Pseudokod algoritma

Pseudokod genetskog algoritma prikazan je na slici 4.8.

Parametri genetskog algoritma:

- veličina_populacije – broj jedinki u populaciji

- broj_generacija – maksimalni broj stvaranja novih generacija, jedan od uvijeta zaustavljanja
- vjerojatnost_mutacije – vjerojatnost za svaku jedinku da će biti mutirana
- selekcijski_pritisak – postotak populacije koji sudjeluje u stvaranju populacije roditelja
- križanje – vrsta križanja
- mutacija – vrsta mutacije
- uvijet_zauustavljanja – skup uvijeta koji određuju da li će se GA zaustaviti, najvažniji je broj generacija

```

Genetski_algoritam(veličina_populacije, broj_generacija,
vjerojatnost_mutacije, selekcijski_pritisak, križanje, mutacija,
uvijet_zauustavljanja){

    populacija = stvori_generaciju(veličina_populacije);
    dok(!uvijet_zauustavljanja){
        ocijeni svaku jedinku populacije;
        stvori kotač odabira;
        populacija_roditeља;
        dok(populacija_roditeља.veličina < (selekcijski_pritisak *
        veličina_populacije)){
            slučajni_odabir pomoću kotača_odabira;
            populacija_roditeља.dodaj(
            populacija.element(služajni_odabir));
        }
        nova_populacija;
        dok(nova_populacija < veličina_populacije){
            roditelj1 = populacija_roditeља(služajnost);
            roditelj2 = populacija_roditeља(služajnost2);
            nova_populacija.dodaj(križanje(roditelj1, roditelj2));

        }
        mutacija(nova_populacija, vjerojatnost_mutacije);
        broj_generacija --;
        populacija = nova_populacija;
    }
}

```

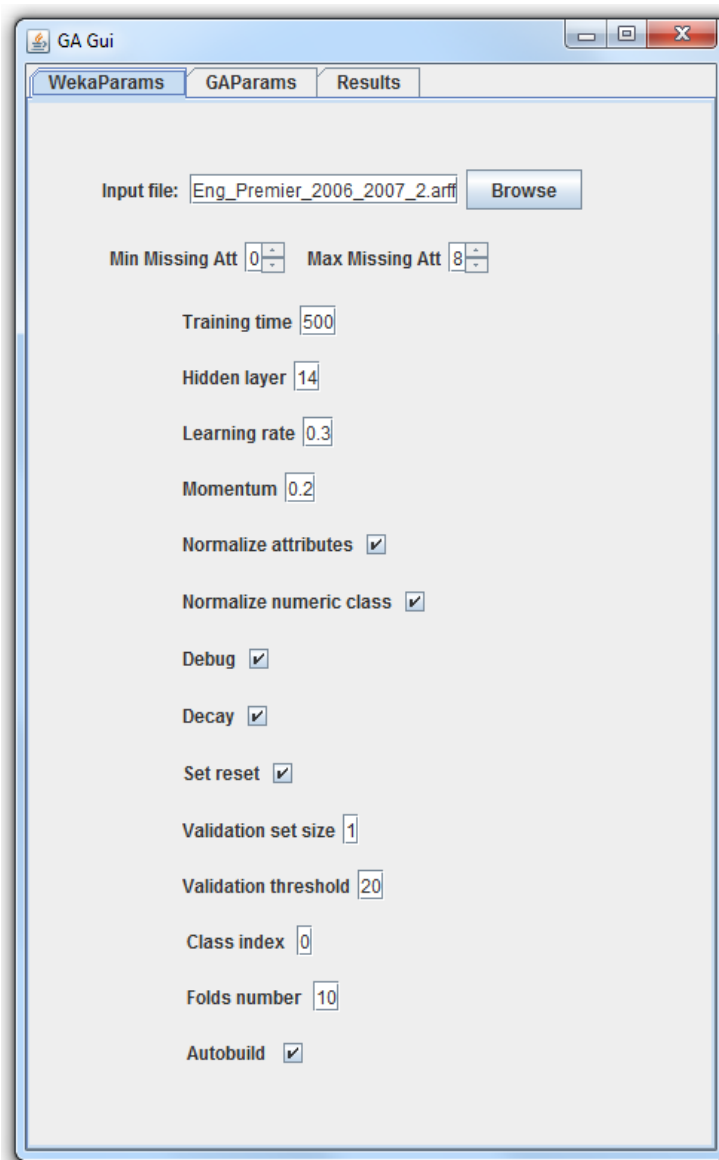
Slika 4.8 Pseudokod

Dobrota jedinke definirana je kao uspješnost neuronske mreže. S obzirom da se jedinka genetskog algoritma sastoji od popisa ulaznih atributa neuronske

mreže, za postavljanje dobrote svake jedinice trenirana je neuronska mreža s pripadajućim atributima. Uspješnost naučene neuronske mreže izražena u postotcima postavljena je za dobrotu jedinice genetskog algoritma.

4.2.4 Računalni program

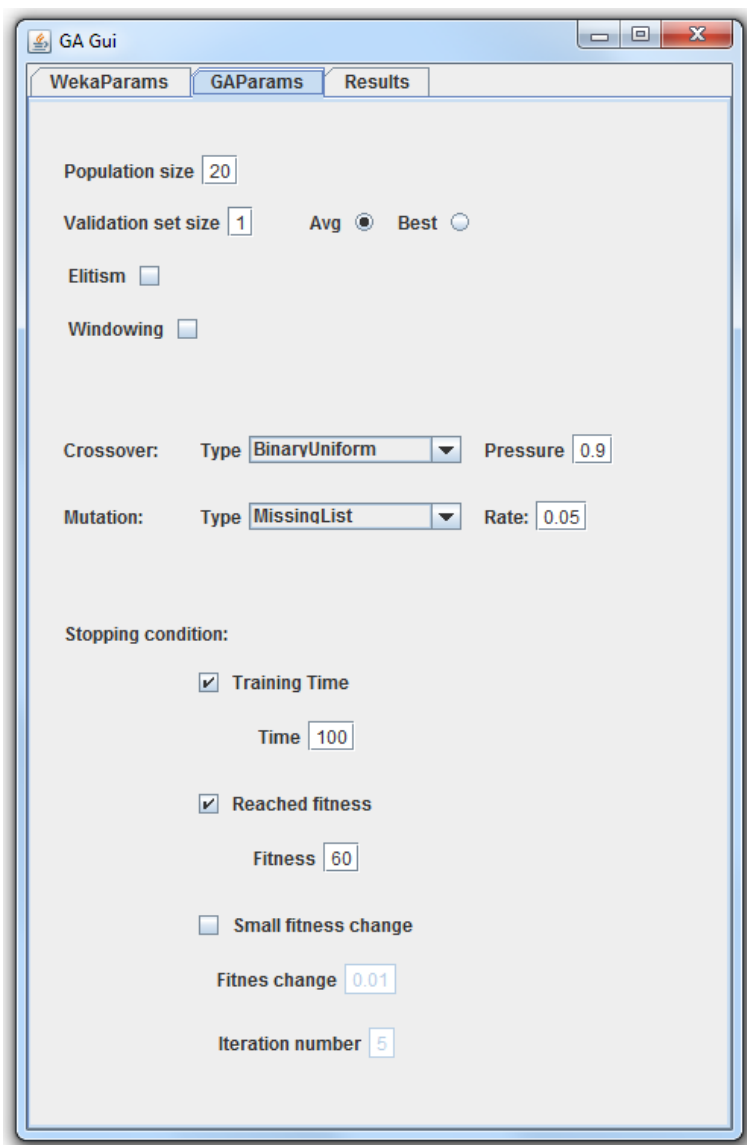
Prema pseudokodu, napravljen je računalni program za provođenje genetskog algoritma. Osim ostvarivanja pseudokoda, u njemu se zadaju parametri.



4.9 Sučelje za parametere neuronske mreže

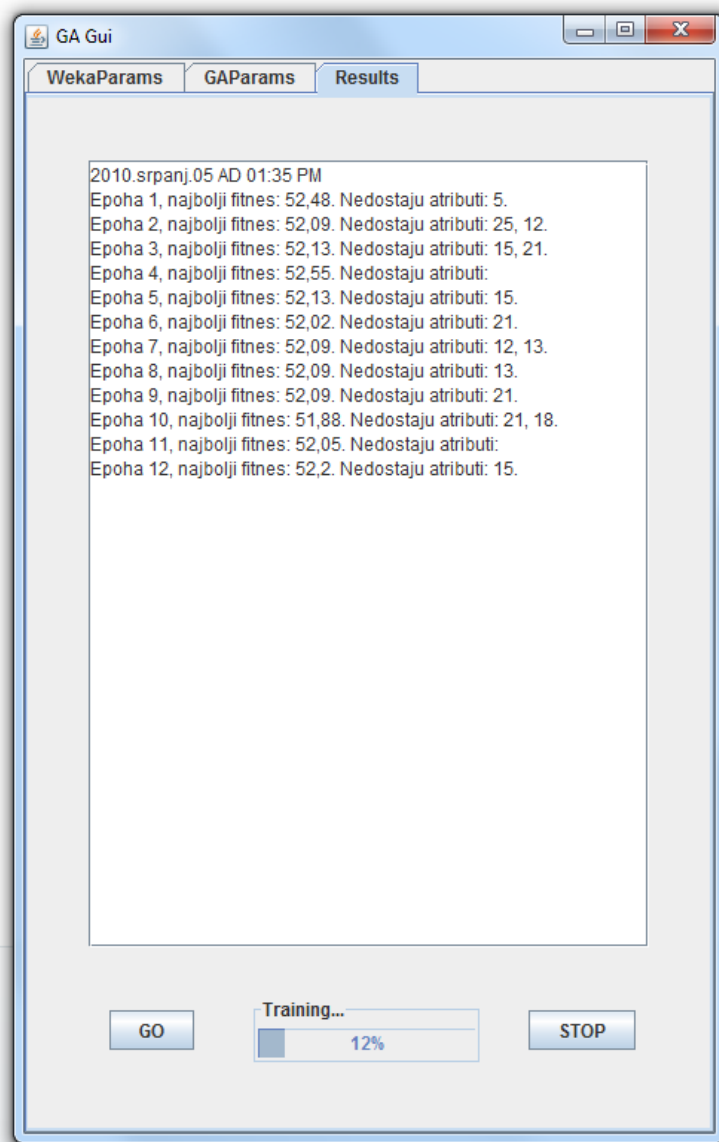
Na slici 4.9 prikazano je sučelje za podešavanje parametara neuronske mreže. Svi prikazani parametri odnose se na već spomenuti računalni program Weka.

Slika 4.10 prikazuje sučelje za podešavanje parametara genetskog algoritma. Većina će biti pojašnjena prilikom optimizacije svakog pojedinog parametra.



4.10 Sučelje parametre genetskog algoritma

Rezultati provedenog programa vidljivi su na slici 4.11. Za svaku je generaciju ispisana dobrota najbolje jedinke, te koji atributi nedostaju toj jedinki.



4.11 Prikaz rezulatat genetskog algoritma

4.3 Podešavanje parametara genetskog algoritma

Isto kao i s neuronskom mrežom, parametri genetskog algoritma trebaju biti optimirani kako bi se dobila što bolja uspješnost neuronske mreže. Podešavanje parametara napravljeno je slično kao i s neuronskom mrežom. Optimizirani su jedan po jedan: broj generacija, veličina populacije, tip selekcije, tip mutacije. Elitizam i translacija su optimirani zajedno sa brojem generacija. Sva danja ispitivanja provedena su na isim ulaznim datotekama neuronske mreže. To je zadnjih 8 sezona *Premier League*, zaključno s 2008/2009 sezonom, s obzirom da je ona posljednja potpuna. S dobivenim parametrima ispitivano je koliko i kojih atributa nedostaje u najboljim

jedinkama. U svim ispitivanjima za prikaz uspješnosti odabrana je dobrota najbolje jedinice dobivene algoritmom unutar svih generacija.

Zbog dugotrajnosti izvođenja programa, koje je opisano u zasebnom poglavlju 'Složenost i vrijeme izvođenja', neka su ispitivanja provedena na manjem broju uzorka izvođenja programa. Iz toga razloga ne daju optimalne rezultate, te nisu prikazane maksimalne, minimalne i prosječne vrijednosti najbolje jedinice. Ta ispitivanja su okvirna i služe samo kao procjena za daljnja ispitivanja.

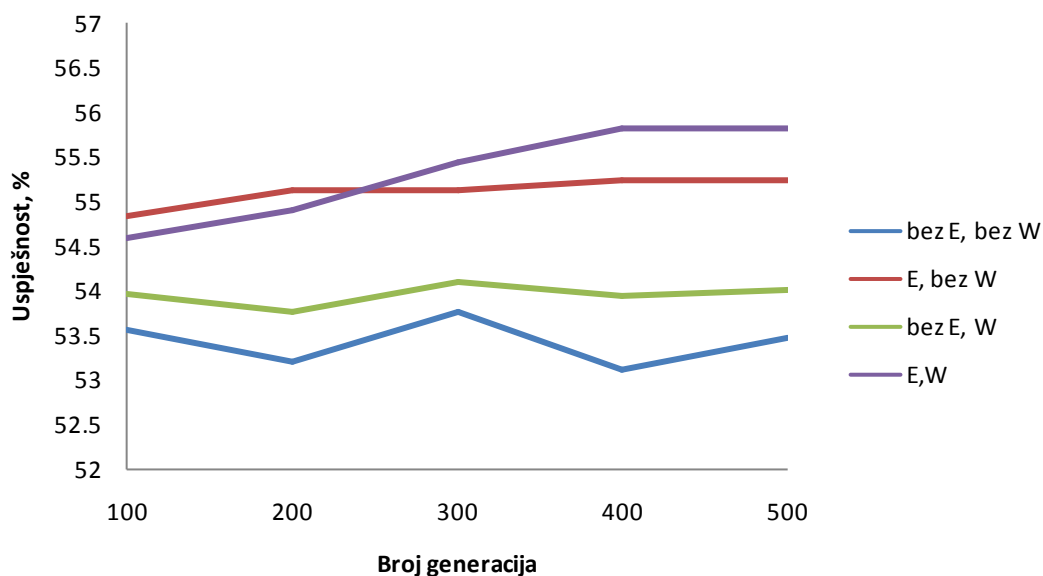
4.3.1 Broj generacija

Broj generacija označava koliko je dugo genetski algoritam traži optimalnu jedinku. Prije ovog ispitivanja nije niti jedan drugi parametar optimiran. Korištena je populacija veličine 100 jedinki, selekcija *MissingList*, selekcijski pritisak 0.9, mutacija *MissingList*, vjerojatnost mutacije 0.05. Ispitivane su i opcije elitizam, koji označava preživljavanje najbolje jedinice u svakoj epohi, te translacija, koja označava koji će se dio dobrote preslikati u kotač odabira prije odabira jedinki roditelja. Za translaciju je korištena jednostavna formula prikazana u jednadžbi 4.1.

$$F_i = 1 + (F_{\max} - F_i) / (F_{\max} - F_{\min}) \quad (4.1)$$

Prametri jednadžbe su:

- F_i = dobrota pojedine jedinice
- $F_{\max}$ = najveća dobrota u populaciji
- $F_{\min}$ = najmanja dobrota u populaciji

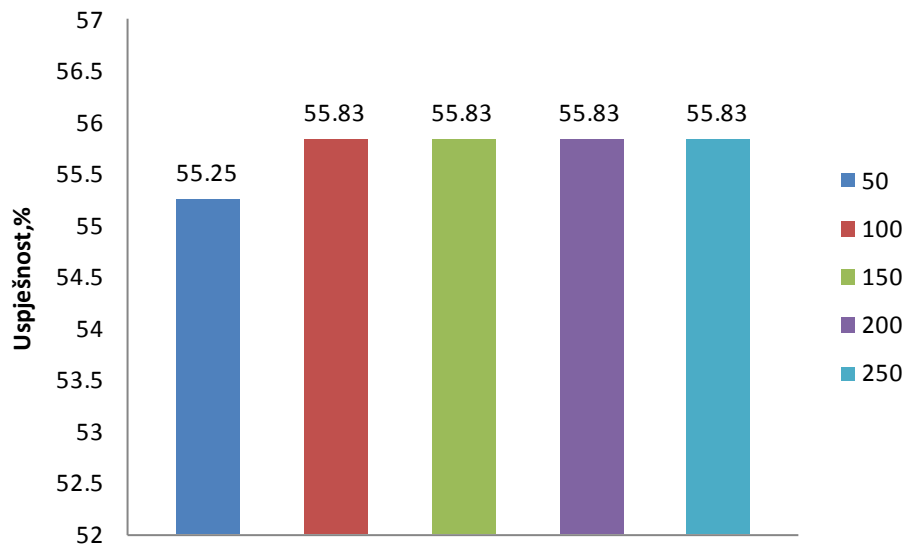


Slika 4.12 Broj generacija

Što se tiče elitizma i translacije, ispitivane su sve četiri mogućnosti: bez elitizma i bez translacije, sa elitizmom bez translacije, bez elitizma sa translacijom, te sa elitizmom i translacijom. Dobiveni rezultati vidljivi su na slici 4.12. Za svaku generaciju prikazana je jedinka najbolje dobrote. Najuzlaznija i najuspješnija je linija koja prikazuje ispitivanje sa elitizmom i translacijom. Izabrano je 400 generacija, te translacija i elitizam.

4.3.2 Veličina populacije

Drugi optimirani parametar je veličina populacije. Jedini optimirani parametar je broj generacija koji iznosi 400. Ostali su korišteni selekcija *MissingList*, selekcijski pritisak 0.9, mutacija *MissingList*, vjerojatnost mutacije 0.05, te elitizam i windowing. Rezultati ispitivanja vidljivi su na slici 4.13. Za prikaz svake populacije odabrana je generacija sa najboljom jedinkom (s obzirom da je korišten elitizam, to je posljednja generacija).

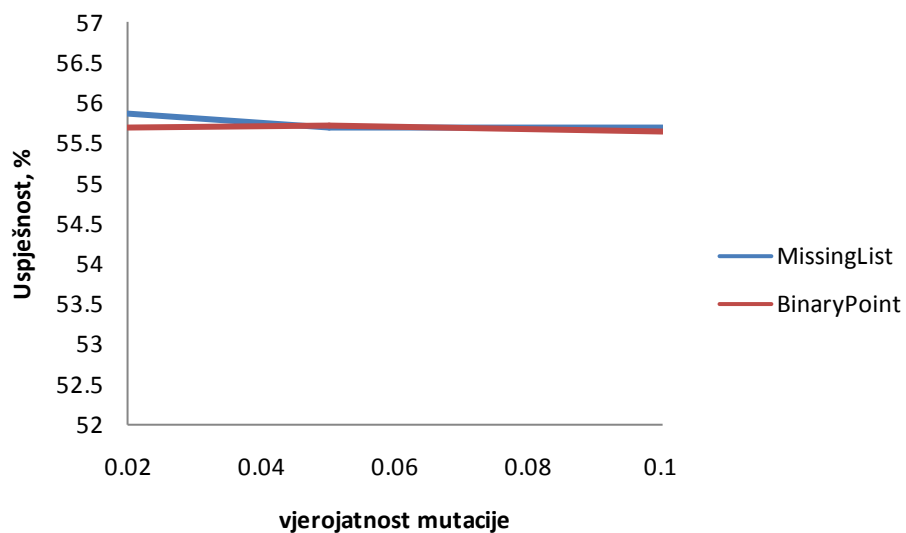


Slika 4.13 Veličina populacije

Nije se pokazalo da određena veličina populacije postiže posebno dobru točnost, no barem je jednostavno izabrati veličinu populacije od 100 jedinki.

4.3.3 Operatori GA

Sljedeća optimirana je vjerojatnost mutacije, te tip mutacije.



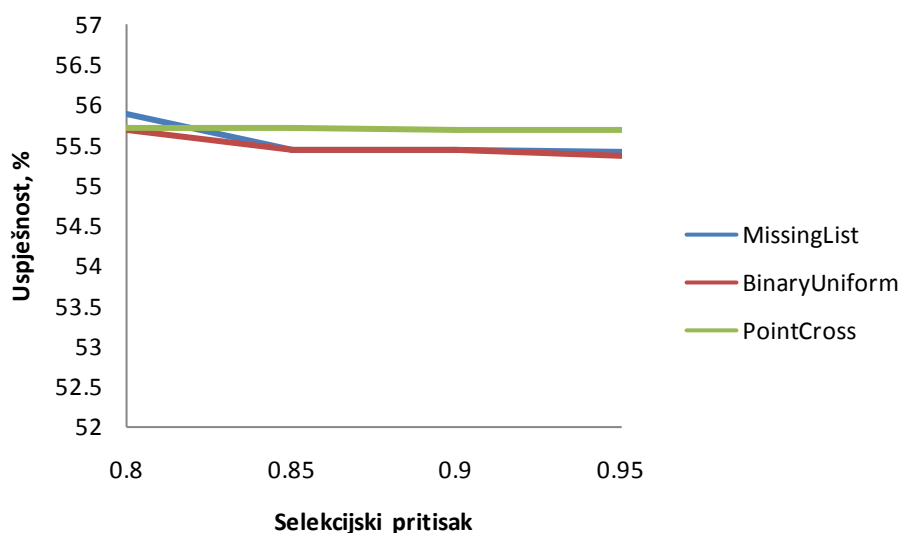
Slika 4.14 Vjerojatnost mutacije

Vjerojatnost mutacije određuje koliko će jedinki dobivenih križanjem biti mutirano. Iz tog razloga vjerojatnosti mutacije su obično vrlo mali brojevi. Nakon ispitivanja, koje je vidljivo na grafu 4.14, Prikazane su dobrote

najuspješnije jedinke. izabrana je *MissingList* mutacija, te vrijednost vjerojatnosti mutacije od 0.02.

Nakon toga optimiran je selekcijski pritisak i tip križanja. Selekcijski pritisak označava koliko jedinki iz stare generacije sudjeluje u stvaranju nove generacije. Križanje i selekcijski pritisak nisu direktno povezani, križanjem nastaju potomci od odabrane populacije roditelja.

Selekcijski pritisak je obično veći postotak, ali je ovisan i o veličini populacije. Bitno je da premala populacija roditelja ne generira preveliku količinu djece. S druge strane, ako prevelika populacija roditelja generira potomke, veća je vjerojatnost za lošije jedinke. Iz tog razloga izabran je opseg ispitivanja između 0.8 i 0.95. Rezultati podešavanja selekcijskog pritiska su vidljivi na grafu 4.15.



Slika 4.15 Selekcijski pritisak

Izabrano je *MissingList* križanje, jer osim i nešto bolje točnosti djeluje i stabilnije, te selekcijski pritisak od 80%.

Vrijednosti svih optimiranih parametara genetskog algoritma vidljivi su u tablici 4.1.

Tablica 4.1 Optimirani parametri genetskog algoritma

Broj generacija	400
Elitizam	da
Translacija	da

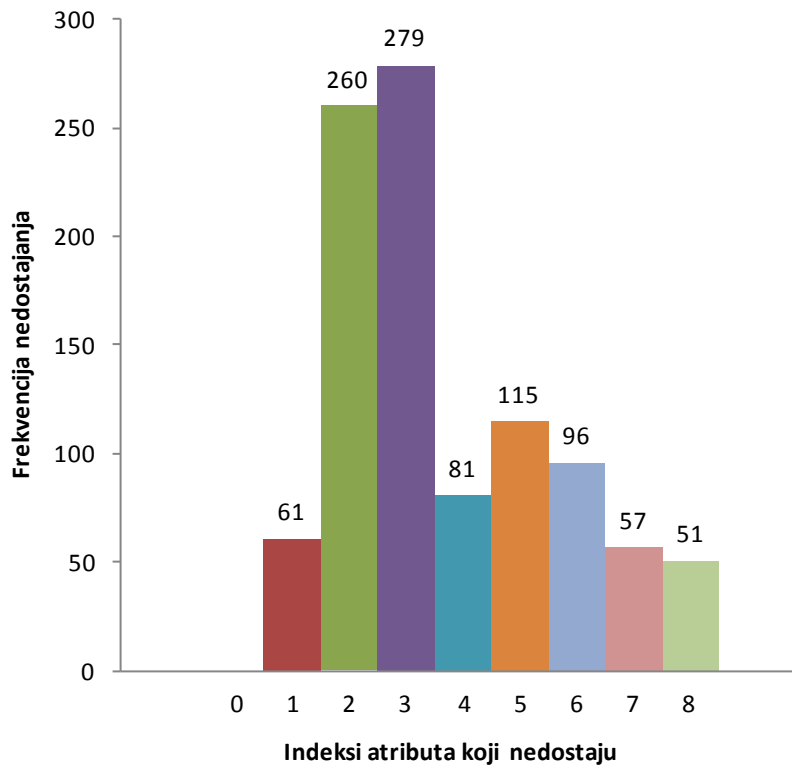
Veličina populacije	100
Vjerojatnost mutacije	0.02
Tip mutacije	MissingList mutacija
Selekcijski pritisak	0.8
Tip križanja	MissingList križanje

4.4 Analiza važnosti atributa

Osim dobivanja što bolje točnosti, cilj rada je i analiza značajki koje su bile odabrane kao ulaz u neuronsku mrežu. Već je objašnjeno kako genetski algoritam eliminira pojedine ulaze mreže te pretražuje prostor takvih jedinki. Na taj se način može primijetiti koliko je pojedini atribut utjecajan, te čak i ako se ne koristi umjetna inteligencija za predviđanje rezultata, to može koristiti u nekim drugim metodama, u kojima odabir značajki ima ulogu. Nije korištena opcija elitizma, iz razloga što je primarni cilj ovog ispitivanja otkrivanje utjecaja različitih atributa, a ne nužno postizanje najbolje točnosti. S obzirom da su svi ostali parametri već optimizirani, radi se i o vrlo malim razlikama u točnosti, koje trenutno nisu najbitnije.

4.4.1 Analiza broja atributa koji nedostaju

Prvo je napravljeno ispitivanje količine atributa koji nedostaju. Prije pokretanja programa, definiran je opseg od 0 do maksimalno 8 atributa koji mogu nedostajati. Ispitivano je na 400 generacija genetskog algoritma, a iz svake epohe uzeta je najbolja jedinka. Rezultati su vidljivi na grafu 4.16.

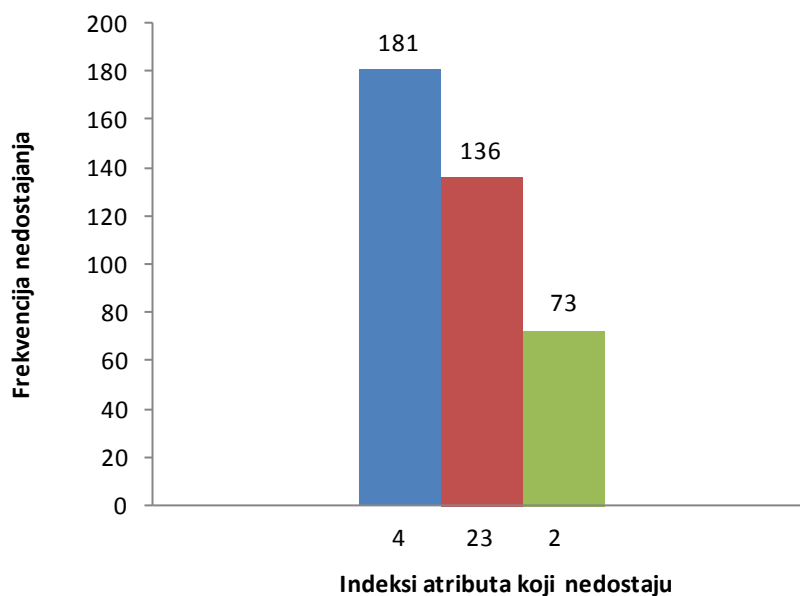


Slika 4.16 Broj atributa koji nedostaju (0-8)

Graf 4.16 prikazuje kako su najčešće jedinice kojima nedostaje 2 ili 3 atributa. Jedinke kojima ne nedostaje niti jedan atribut imaju frekvenciju nula. To znači da niti u jednoj od 1000 epoha najbolja jedinka nije bila ona koja sadrži sve ulaze neuronske mreže. Iz toga se pak može zaključiti da ulaza u neuronsku mrežu ima previše, te postoje atributi, ili skupine atributa koji loše utječu na točnost. Daljnja ispitivanja određuju koji su najčešći atributi koji nedostaju najboljim jedinkama.

4.4.2 Utjecaj jednog redundantnog atributa

Za početak, ispitivano je nedostajanje točno jednog atributa. Minimalan i maksimalan broj atributa koji nedostaju postavljen je na 1, te je praćeno koji od atributa najčešće nedostaju. Kao i u prethodnom ispitivanju, izabrane su najbolje jedinice iz svake generacije, a provedeno je 400 generacija. Graf 4.17 prikazuje rezultate.

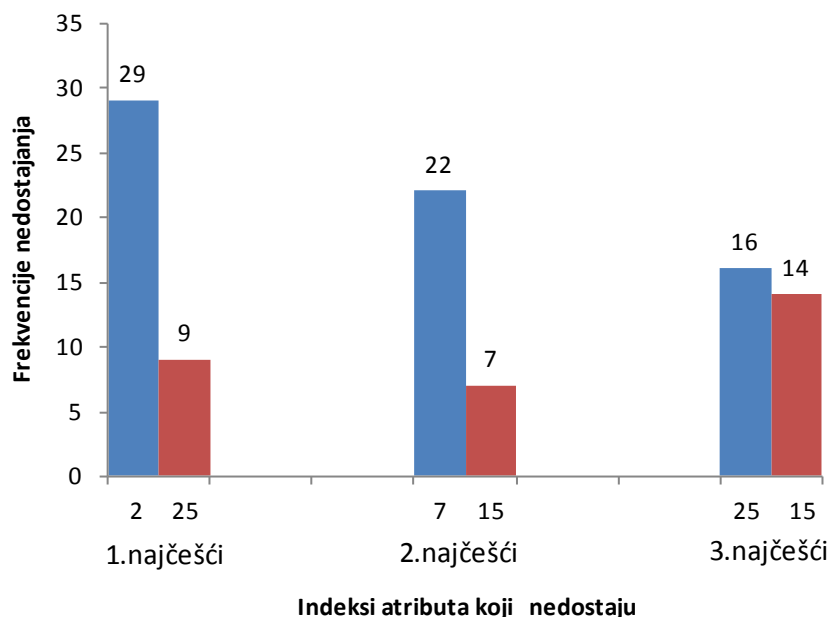


Slika 4.17 Broj atributa koji nedostaju (1)

Na grafu 4.17 prikazano je koja 3 atributa najčešće nedostaju u jedinkama u kojima može nedostajati točno jedan atribut. To su atributi s indeksima: 4, 23 i 2. Atribut s indeksom 4 je *HomeTeam_HomeDrawn*, tj. koliko je domaćin do sada poraza imao na domaćem terenu. Drugi atribut koji najčešće nedostaje je s indeksom 23, i zove se *AwayTeam_AwayLost*, tj. koliko je poraza doživio gostujući tim, igrajući u gostima. Treći najčešći nedostajući atribut je s indeksom 2, imena *HomeTeam_LeaguePoints*, tj. koliko je bodova u toj ligi do sada skupio domaći tim.

4.4.3 Utjecaj dva redundantna atributa

Sljedeće ispitivanje je dosta slično prethodnom, odnosi se samo na točno 2 nedostajuća atributa. S obzirom da u svakoj jedinki nedostaju 2 atributa, na grafu 4.18 su prikazani indeksi atributa koji najčešće nedostaju, te koji je atribut koji najčešće nedostaje uz njih.



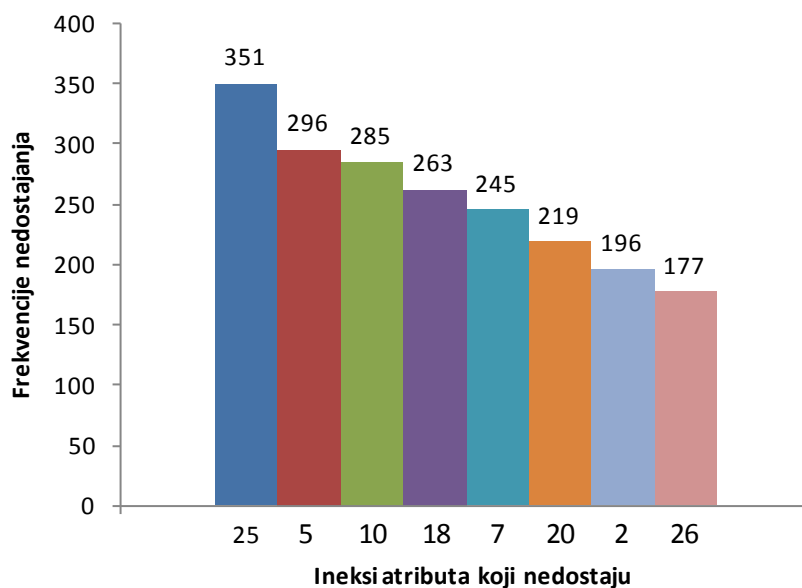
Slika 4.18 Broj atributa koji nedostaju (2)

Vidljivo je kako najčešće nedostaje atribut indeksa 2, a to je *HomeTeam_LeaguePoints*, tj. koliko je bodova u toj ligi do sada skupio domaći tim. Taj se atribut pojavio i u analizi sa jednim nedostajućim atributom, te je vrlo vjerojatno da loše utječe na točnost. Atribut koji uz njega najčešće nedostaje je sa indeksom 25, tj. *AwayTeam_Away_GoalsConceded*, koji označava koliko je gostujući tim do sada golova primio igrajući gostima. Drugi atribut po broju nedostajanja je onaj s indeksom 7, naziva *HomeTeam_Home_GoalsConceded*, tj. koliko je domaćin golova primio na domaćem terenu. Uz njega najčešće nedostaje atribut indeksa 15, *Awayteam_LeaguePoints*, koji označava koliko je gostujući tim skupio bodova do sada u ligi. Treći najčešći atribut koji nedostaje je s indeksom 25, koji je već spomenut da najčešće nedostaje uz prvi najčešće nedostajući atribut. A u z njega najčešće nedostaje atribut indeksa 15, koji je također već spomenut uz drugi najčešće nedostajući atribut.

Nekoliko atributa ponavlja se u analizi, atributi s indeksima 15 dva puta se pojavljuju kao drugi nedostajući atribut, a atribut s indeksom 25 je drugi uopće koji nedostaje i drugi najčešći uz prvi nedostajući. Također, atribut koji najčešće nedostaje, onaj s indeksom 2, pojavio se u analizi jednog nedostajućeg atributa, kao treći nedostajući generalno. Iz tih ponavljanja, iako na prvi pogled djeluju malo zbunjujuće, nije toliko zanimljivo koji su to točno atributi, već da postoji neki uzorak ponavljanja i da je korištenje genetskog algoritma imalo smisla.

4.4.4 Utjecaj više redundantnih atributa

U sljedećoj analizi nije ograničavan broj nedostajućih atributa, postavljen je na minimalno 0 i maksimalno 8 atributa. Graf sa brojem atributa koji nedostaju vidljiv je na slici 4.16, a nakon nje analizirano je koji su to točno atributi koji najčešće nedostaju. Izabrano je 8 atributa s najvećim frekvencijama nedostajanja, te su oni prikazani na grafu 4.19, točnije njihovi indeksi. Učenje je izvedeno na 1000 epoha, te je u svakoj epohi izabrana najbolja jedinka.



Slika 4.19 Broj atributa koji nedostaju (0-8)

Imena atributa, poredanih po učestalosti nedostajanja, čiji se indeksi mogu vidjeti u legendi:

- 25 = *AwayTeam_Away_GoalsConceded* () = koliko je gostujući tim golova primio igrajući u gostima
- 5 = *HomeTeam_Home_Lost* () = koliko je domaćin utakmica izgubio igrajući na domaćem terenu
- 10 = *HomeTeam_Away_Lost* () = koliko je domaćin utakmica izgubio igrajući u gostima
- 18 = *AwayTeam_Home_Lost* () = koliko je gostujući tim utakmica izgubio igrajući kod kuće
- 7 = *HomeTeam_Home_GoalsConceded* () = koliko je domaći tim golova primio igrajući kod kuće

- $20 = \text{AwayTeam_Home_GoalsConceded} ()$ = koliko je gostujući tim golova primio igrajući na domaćem terenu
- $2 = \text{HomeTeam_LeaguePoints} ()$ = koliko je domaćin sakupio bodova u ligi
- $26 = \text{Awayteam_RecentForm_Points} ()$ = koliko je gostujući tim bodova sakupio u zadnjih 9 utakmica

Vidljivo je kako se neki od atributa češće ponavljaju. Iako analiza nije provedena pretjerano ciljano, vidljivo je kako je moguće utvrditi da pojedini atributi češće nedostaju kod najboljih jedinki. Ta značajka se čini bitna i korisna, kod definiranja utjecaja svakog pojedinog atributa za bilo koji drugi program, te također prilikom definiranja bilo kojeg potencijalno subjektivnog skupa značajki.

4.4.5 Složenost i vrijeme izvođenja

Složenost izvođenja cijelog genetskog algoritma je poprilično komplicirana, i izražena je samo približna formula 4.2, kako bi se dobio što bolji osjećaj za vrijeme izvođenja.

$$O \approx N \times \text{Weka} + \text{It_num} \times (N \times \text{Weka} + \text{Rw} \times N(\text{parents}) + \text{Rw} \times N(\text{children}) + M + \text{Cr}) \quad (4.2)$$

Parametri označavaju:

- **O** - približna složenost
- **N** – veličina populacije
- **Weka** - vrijeme učenja neuronske mreže
- **It_num** - broj iteracija genetskog algoritma
- **Rw** – roulette wheel
- **N(parents)** - veličina populacije roditelja
- **N(children)** – veličina populacije djece
- **M** - mutacija
- **Cr** – križanje

Valja uzeti u obzir da algoritam ima mnogo podkoraka, i da su uzeti okvirni, pojednostavljeni koraci. Primjerice, broj iteracija genetskog algoritma je uzet kao broj epoha genetskog algoritma. To nije najpreciznije, jer je on samo jedan od uvjeta zaustavljanja, pored postizanja određenog iznosa dobrote jedinke ili premale promjene dobrote u zadanom broju uzastopnih iteracija. Također nije

uzeta u obzir složenost računanja svakog operatora, poput mutacije i križanja, ponajviše zbog nemjerljivosti s ocjenjivanjem dobrote cijele populacije. S obzirom da učenje neuronske mreže i ocjenjivanje u Weki traje u prosjeku između 30 i 50s, ovisno od brzine procesora, te koliko je instanci programa pokrenuto, vremenski najzahtjevniji član formule 4.2 je umnožak $(It_num \times N \times Weka)$.

4.24.2. Iz podešavanja parametara u prethodnim poglavljima dobiveno je kako je optimalno učenje genetskog algoritma na 400 epoha, te veličina populacije od 100 jedinki. Uzimajući prosječnu vrijednost učenja neuronske mreže i navedene vrijednosti parametara dobiva se umnožak $40s \times 400epoha \times 100jedinki$, koji iznosi 1 600 000s, tj. 2 667minuta, tj. 444minuta, tj. 18 dana. Ako se na tu vrijednost dodaju svi zanemareni članovi, vrijednost se slobodno može promatrati kao 20 dana izvođenja. Mislim da je svim kompjuterima u mojoj blizini drago da je ono gotovo.

5. Primjena i nedostaci

Nastanak ovog rada bio je isključivo motiviran ispitivanjem mogućnosti može li se umjetna inteligencija koristiti prilikom kladjenja. Iako su dobivene točnosti veće od 50%, prostora za poboljšanje još uvijek ima. U sličnom radu Alana McCabea [6], o korištenju neuronskih mreža za predviđanje rezultata, dobivene su točnosti do 68.1%. Korištena je slična konfiguracije mreže, sa različitim ulaznim atributima. Između ostalog i jačina postave tima, tako da su igrači bili rangirani s obzirom na težinu natjecanja u kojima su sudjelovali, te bi njihov zbroj činio jačinu tima. Naravno da to nije optimalno, ali čini se da barem dijelom utjecalo na poboljšanje predviđanja.

Drugi nedostatak je potrošnja vremena. Dobiveni rezultati su pokazali kako su parametri mreže ovisni i od podataka, te korištenje neuronske mreže učenje na jednoj ligi nije preporučljivo za predviđanje rezultata druge lige. Iz tog razloga bilo bi potrebno raditi odvojene optimizacije parametara za predviđanje svake lige, što bi bilo dugotrajno.

Treći, također značajan nedostatak je predviđanje samo rezultata, a ne vjerojatnosti pobjede. S obzirom kako je već zaključeno da je neuronska mreža deterministička, vjerojatnost pojedinog rezultata nije moguće dobiti većim brojem učenja i ispitivanja mreže. Razlog tomu što bi vjerojatnost bila korisniji podatak je postojanje koeficijenata. Primjerice, ako mreža predviđa za pojedinu utakmicu da će pobijediti tim A, a koeficijent kladionice izražava vjerojatnost od 95% da pobijedi tim A, a sveukupna uspješnost te neuronske mreže iznosi 55%, onda ona zasigurno nije otkrila toplu vodu. Iz tog razloga, bilo bi bolje postaviti koeficijent, ili vjerojatnost pobjede pojedinog tima kao izlaz. To nije učinjeno sada zbog toga što se podaci o detaljima utakmice, i o koeficijentima, većinom nalaze na različitim web stranicama, te su stranice s koeficijentima manjeg opsega liga.

Četvrti razlog je već natuknut a to je izbor značajki koje definiraju jednu utakmicu. Ne samo da je on subjektivan, već je i neprecizan. U bazi podataka ne nalaze se podaci o postavama timova, te minutama postignutih pogodaka. Neki od nogometnih navijača vjerojatno bi se složili kako je postava igrača jedan od bitnijih faktora.

Ipak, postoje i pozitivne strane. Dobivena točnost veća od 50% svakako je ohrabrujuća. Vidljivo je kako je moguće korištenje umjetne inteligencije za sportske problematike. Vrlo velik dio problematike leži u postavljanju početnog problema.

Također, analiziranjem nedostajanja pojedinih atributa, moguće je analizirati utjecaj specifičnih značajki nogometnih utakmica na rezultat.

6. Zaključak

Cilj ovog rada bio je vidjeti može li se umjetna inteligencija koristiti za predviđanje sportskih rezultata. Usprkos nedostacima, dobivene su točnosti neuronske mreže preko 50%, te je pokazano kako je moguće optimizacijom parametara i podataka postupno popravljati točnost. To smatram veoma bitnom značajkom. Mnogo je značajnije moći utjecati na dobivenu točnost, nego dobiti statičku točnost koja se ne može popravljati. Vjerujem kako se prikupljanjem opširnijih podataka, postave timova, minuta postignutih pogotaka, te predviđanjem koeficijenata stvara šire područje pretraživanja, a time i mogućnost optimiranja većeg broja parametara podataka.

Vjerujem kako je moguće koristiti umjetnu inteligenciju za predviđanje sportskih rezultata i kako veći dio problematike leži u definiranju problema, dok neuronske mreže i genetski algoritmi rade samo usmjerena pretraživanja velikih prostora.

7. Literatura

- [1] www.soccerway.com
- [2] Soccer Match Result Prediction using Neural Networks, (CS534 Project Report), Radha-Krishna Balla
- [3] <http://www.cs.waikato.ac.nz/ml/weka/>
- [4] www.heatonresearch.com/articles/5/page2.html
- [5] http://www.zemris.fer.hr/~golub/ga/ga_skripta1.pdf
- [6] Artificial Intelligence in Sports Prediction, Alan McCabe, MAIT Technologies

8. Sažetak

Svrha ovog rada je ispitivanje da li se umjetna inteligencija može koristiti za predviđanje sportskih rezultata, te analiziranje značaja pojedinih odabranih sportskih značajki. Korištene su neuronske mreže, genetski algoritam, a odabrani sport za analiziranje bio je nogomet. Odabrane su značajke svake nogometne utakmice kao ulaz u neuronsku mrežu, a izlaz je predviđanje pobjede, poraza ili izjednačenog rezultata. Nakon učenja i ispitivanja parametri neuronske mreže podešavani su empirijski. Dobivena je točnost između 45 i 48%. Nakon toga neuronske su mreže optimirane genetskim algoritmom, eliminacijom pojedinih ulaznih značajki. Nakon podešavanja parametara empirijski, dobivena je točnost oko 55%. Primijećeno je kako pojedini ulazi neuronske mreže loše utječu na njenu točnost.

Abstract

Point of this work was investigating can an artificial intelligence be used for predicting sport's results, and analyzing the influence of specific sport characteristics. That was achieved using neural networks and genetic algorithm, and the analyzed sport was football. Football characteristics were chosen for each football game as an input in a neural network, and output was predicting winning, loosing or a draw. After training and testing of the neural network, parameters were adjusted empirically. Successfull rate was between 45 and 48%. After that neural networks were optimized using genetic algorithm, eliminating some of the neural network's input. After the tuning parameters empirically, achieved successfull rate was about 55%. It was concluded that some of the input of the neural network were not influencing the successfull rate very well.