UNIVERZITET U BEOGRADU
FAKULTET ORGANIZACIONIH NAUKA

Tijana M. Vujičić

SOFTVERSKI ALAT ZA ISPITIVANJE ALGORITAMA STRUKTURNE REGRESIJE BAZIRANE NA GCRF MODELU

doktorska disertacija

Beograd, 2018.

UNIVERSITY OF BELGRADE
FACULTY OF ORGANIZATIONAL SCIENCES

Tijana M. Vujičić

SOFTWARE TOOL FOR TESTING STRUCTURED REGRESSION ALGORITHMS BASED ON GCRF MODEL

Doctoral Dissertation

Belgrade, 2018.

Mentor:

dr Vladan Devedžić, redovni profesor

Fakultet organizacionih nauka, Univerzitet u Beogradu

_____

Članovi komisije:

dr Veljko Jeremić, vanredni profesor

Fakultet organizacionih nauka, Univerzitet u Beogradu

_____

dr Zorica Stanimirović, vanredni profesor

Matematički fakultet, Univerzitet u Beogradu

_____

Datum odbrane: _____

Ovaj rad posvećujem svojim roditeljima

Izjava zahvalnosti

Ova disertacija rađena je u periodu od januara 2016. do februara 2018. godine na Fakultetu organizacionih nauka u Beogradu i na Univerzitetu Temple (Center for Data Analytics and Biomedical Informatics) u Filadelfiji. Ideja i njena realizacija su plod zajedničkog rada sa mojim mentorom, prof. dr Vladanom Devedžićem, i timom iz Centra za analizu podataka i biomedicinsku informatiku, na čelu sa prof. dr Zoranom Obradovićem.

Zahvaljujem se svim profesorima sa osnovnih i postdiplomskih studija koji su me kroz svoja predavanja, entuzijazam i posvećenost podstakli da se usavršavam i bavim naučnim radom. Prije svega se zahvaljujem profesoru dr Vladanu Devedžiću, koji je kao profesor na prvoj godini osnovnih studija kod mene probudio interesovanje za programiranje i imao najveći uticaj na moj izbor profesije. Takođe se zahvaljujem što je prihvatio ulogu mog mentora na doktorskim studijama, ukazao mi povjerenje i pružio nesebičnu stručnu pomoć i podršku u izradi doktorske disertacije. Hvala mentoru mog diplomskog i master rada, profesoru dr Draganu Đuriću, kao i profesoru dr Miomiru Anđiću, koji mi je pružio neophodne osnove i podstrek da nastavim da se bavim matematičkim i statističkim modelima. Veliku zahvalnost dugujem profesoru dr Zoranu Obradoviću i članovima njegovog tima: Fang, Jesse-u, Jeleni, Đorđu, Ivanu, Chao-u, Shoumink-u, Ani, Martinu i Branku, na gostoprimstvu, podršci i savjetima.

Hvala profesoru dr Slobodanu Backoviću koji mi je ukazao povjerenje i podstakao me da se bavim akademskim radom. Hvala dekanu Fakulteta za informacione tehnologije, Univerziteta Mediteran Podgorica, dr Adisu Baloti, i svim kolegama na razumijevanju, pomoći i saradnji.

Zahvaljujem se članovima komisije za ocjenu i odbranu ove doktorske disertacije, Veljku Jeremiću i Zorici Stanimirović, na korisnim primjedbama i sugestijama.

Hvala mojim dragim prijateljima Petru, Ivanu, Iliji i Draženu na pomoći i podršci.

Najveću zahvalnost dugujem mojoj porodici i roditeljima, kojima posvećujem ovaj rad.

Softverski alat za ispitivanje algoritama strukturne regresije bazirane na GCRF modelu

Rezime:

Predmet istraživanja ovog rada su modeli strukturne regresije, koji su dizajnirani da koriste veze između objekata prilikom predviđanja izlaznih vrijednosti. Drugim riječima, modeli strukturne regresije razmatraju atribute objekata i veze između objekata kako bi dali što tačnije predviđanje. Gaussian Conditional Random Fields (GCRF) model je jedan od najčešće korišćenih modela strukturne regresije koji integriše predikciju tradicionalnih modela nadgledanog učenja (nestrukturnih prediktora) i vezu između objekata u cilju tačnije predikcije. Glavna pretpostavka ovog modela je da su dva objekta koja su usko povezana veoma slični jedan drugom i samim tim vrijednosti njihovih izlaznih varijabli treba da budu slični. Sličnost između objekata u GCRF modelu mora da bude simetrična, ali u velikom broju realnih primjera objekti su nesimetrično povezani.

U radu je predstavljeno proširenje GCRF modela koje uzima u obzir asimetričnu sličnost između objekata (nazvan usmjereni GCRF - Directed GCRF). Na sintetičkim i realnim setovima podataka pokazano je da novi model daje tačnije predviđanje od standardnog GCRF modela i tradicionalnih nestrukturnih prediktora.

Rad obuhvata i razvoj softverskog alata otvorenog koda koji integriše različite vrste GCRF modela i omogućava treniranje i testiranje tih modela na različitim setovima podataka, preko grafičkog korisničkog interfejsa. Izvršena je evaluacija alata sa korisnicima različitih profila i različitog znanja iz oblasti mašinskog učenja. Rezultati su potvrdili da je alat je intuitivan i lak za korišćenje kako za eksperte, tako i za početnike i istraživače iz različitih domena kojima GCRF model može pomoći da dođu do željenih informacija.

Ključne riječi: inteligentni sistemi, mašinsko učenje, strukturna regresija, GCRF model, grafovi, razvoj softvera, softverski alat, softver otvorenog koda, upotrebljivost softvera.

Naučna oblast: računarske nauke

Uža naučna oblast: softversko inženjerstvo

UDK broj: 004

Software tool for testing structured
regression algorithms based on GCRF model

Abstract:

The subject of this dissertation are structured regression models that are designed to use relationships between objects for predicting output variables. In other words, structured regression models consider the attributes of objects and dependencies between objects to make predictions as accurately as possible. Gaussian Conditional Random Fields (GCRF) model is commonly used structured regression model that incorporates the outputs of traditional supervised learning models (unstructured predictors) and the correlation between output variables in order to achieve a higher prediction accuracy. A main assumption in the GCRF model is that if two objects are closely related, they should be more similar to each other and they should have similar values of the output variable. The similarity considered in GCRF is symmetric. However, in many real-world examples objects are asymmetrically linked.

This dissertation presents extension of GCRF model that considers asymmetric similarities between objects (called Directed GCRF). The effectiveness of new model is characterized on synthetic datasets and real-world datasets, on which it was more accurate than the standard GCRF model and baseline unstructured predictors.

This dissertation also presents development of an open-source software tool that integrates various GCRF methods and supports training and testing of those methods on different datasets using graphical user interface. The tool was evaluated with users with different level of knowledge in the machine learning field. Evaluation results confirmed that this tool is intuitive and easy to use for experts, as well as for beginners and researchers from different domains that can use GCRF for data prediction.

Keywords: intelligent systems, machine learning, structured regression, GCRF model, graphs, software development, software tool, open source software, software usability.

Research area: Computer science

Research focus: Software Engineering

UDK number: 004

Sadržaj

1. Uvod

1.1. Problem, predmet i cilj istraživanja

Predmet istraživanja ovog rada su modeli strukturne regresije, koji su dizajnirani da koriste veze između objekata prilikom predviđanja izlaznih vrijednosti. Drugim riječima, modeli strukturne regresije razmatraju atribute objekata i veze između objekata kako bi dali što tačnije predviđanje. Gaussian Conditional Random Fields (GCRF) (Radosavljević, Vučetić, & Obradović, 2010) model je jedan od najčešće korišćenih modela strukturne regresije koji integriše predikciju tradicionalnih modela nadgledanog učenja (nestrukturiranih prediktora) i vezu između objekata u cilju tačnije predikcije. Ovaj model je prvo primijenjen u oblasti računarske vizije (computer vision), ali je od tada korišćen u različitim oblastima i proširen za različite namjene. Glavna pretpostavka ovog modela je da su dva objekta koja su usko povezana veoma slični i samim tim se očekuje da vrijednosti njihovih izlaznih varijabli budu bliske. Sličnost između objekata u GCRF modelu mora da bude simetrična, ali u velikom broju realnih problema objekti su nesimetrično povezani (Beguerisse-Díaz, Garduno-Hernández, Vangelov, Yaliraki, & Barahona, 2014).

Jedan od ciljeva ovog rada je proširenje GCRF modela uzimajući u obzir asimetričnu povezanost objekata. Drugi cilj rada je razvoj aplikacije koja integriše različite vrste GCRF modela i omogućava treniranje i testiranje tih modela na različitim setovima podataka preko korisničkog interfejsa. Aplikacija obezbjeđuje razumljiv i jednostavan korisnički interfejs koji pojednostavljuje upotrebu GCRF modela za korisnike koji nemaju iskustva u oblasti mašinskog učenja, ali i za istraživače koji žele da uporede svoje modele sa GCRF-om. Takođe je razvijena i Java biblioteka koja omogućava istraživačima koji imaju iskustva u Java programiranju da koriste GCRF model u svom kodu i da na jednostavniji način implementiraju njegova proširenja.

1.2. Polazne hipoteze

Istraživanje je orijentisano ka analizi oblasti, analizi postojećih GCRF modela, predlogu proširenja GCRF modela za usmjerene grafove i razvoju i evaluaciji aplikacije otvorenog koda koja će integrisati različite vrste GCRF modela.

Polazne hipoteze ovog rada su:

- **H1 -** Postojeća proširenja GCRF modela se ne mogu primijeniti na usmjerene grafove.
- **H2 -** Predloženo proširenje GCRF modela za usmjerene grafove omogućava preciznije predviđanje od standardnog GCRF modela i tradicionalnih nestrukturnih prediktora.
- **H3 -** Softverski alat pojednostavljuje treniranje i testiranje GCRF modela i njegovih proširenja na različitim setovima podataka.
- **H4 -** Softverski alat koji nudi implementaciju standardnog GCRF modela i njegovih proširenja mogu koristiti kako i eksperti u oblasti mašinskog učenja, tako i početnici kojima GCRF model može pomoći da dođu do željenih informacija.

1.3. Metode istraživanja

Za analizu oblasti i postojećih GCRF modela, korišćene su opšte metode prikupljanja i analize postojećih naučnih rezultata, na osnovu dostupne literature i objavljenih radova koji prikazuju modele i njihovu primjenu u praksi.

Predložen je matematički model za proširenje GCRF modela, koji je implementiran u Java programskom jeziku i testiran na različitim vrstama sintetički generisanih usmjerenih grafova, kao i na više realnih setova podataka. Performanse i preciznost kreiranog modela upoređene su sa postojećim GCRF modelom, kao i sa tradicionalnim nestrukturnim prediktorima.

Aplikacija je razvijena u skladu sa savremenim metodama softverskog inženjerstva koje se primjenjuju prilikom razvoja softvera otvorenog koda. Cilj je da aplikacija bude intuitivna i laka za korišćenje za eksperte u oblasti mašinskog učenja, ali i za početnike i istraživače iz različitih domena kojima GCRF model može pomoći da dođu do željenih informacija. U cilju ocijene praktične primjene razvijene aplikacije, realizovana je evaluacija aplikacije sa korisnicima različitih profila i različitog znanja iz oblasti mašinskog učenja.

Rezultati istraživanja prezentovani su tekstualno i grafički, korišćenjem slika, tabela i dijagrama, a programski kod je dostupan na GitHub-u.

1.4. Pregled sadržaja rada

Nakon uvoda, u drugom poglavlju dat je pregled relevantnih oblasti za realizovano istraživanje: mašinskog učenja, strukturne regresije, GCRF modela i njegovih proširenja.

U trećem poglavlju je opisano proširenje GCRF modela za usmjerene grafove: matematički model, implementacija i evaluacija, koja uključuje testiranje tačnosti predviđanja na različitim setovima podatka, kao i analizu performansi i konveksnosti modela.

U četvrtom poglavlju predstavljen je softverski alat koji nudi implementaciju standardnog GCRF modela i njegovih proširenja. Predstavljene su sve funkcionalnosti alata, njegov dizajn i način implementacije, kao i način upotrebe alata za treniranje različitih modela na različitim setovima podataka. Predstavljeni su i rezultati evaluacije alata sa korisnicima različitih profila.

U petom poglavlju opisana je implementacija i način upotrebe Java biblioteke koja sadrži osnovne klase za GCRF koncepte i implementaciju nekih konkretnih modela.

U šestom poglavlju dati su primjeri primjene usmjerenog GCRF modela, razvijenog softvera i Java biblioteke, kao i mogućnost primjene rezultata istraživanja u nastavi.

U sedmom poglavlju je predstavljena analiza prednosti i nedostataka, kao i plan daljeg razvoja.

U osmom poglavlju data su zaključna razmatranja.

2. Pregled oblasti

2.1. Mašinsko učenje

U savremenom svijetu problemi iz različitih naučnih područja rješavaju se primjenom metoda vještačke inteligencije. Vještačka inteligencija je oblast informatike koja se fokusira na razvijanje softvera koji će omogućiti računarima da se ponašaju na način koji bi se mogao okarakterisati inteligentnim. Zahvaljujući ovoj disciplini, računarski sistemi mogu biti sposobni da na osnovu ulaznih podataka, ugrađenog znanja i mehanizma rasuđivanja 'inteligentno' generišu izlaz. Za razliku od nekih oblasti računarstva, za koje se smatra da su već dostigle maksimum svojih mogućnosti, vještačka inteligencija je još uvijek nedovoljno istražena i tek treba da postigne svoje najznačajnije rezultate, uprkos tome što već postoje mnogi inteligentni sistemi koji funkcionišu veoma dobro. Navedeni razlozi čine oblast vještačke inteligencije atraktivnom za izučavanje, a konstantno pojavljivanje novih eksperimenata i istraživanja omogućavaju njenu primjenu u najrazličitijim oblastima.

Mašinsko učenje je jedna od oblasti vještačke inteligencije koja za cilj ima kreiranje sistema koji su sposobni da se prilagode novim situacijama i uče iz iskustva. Sistemi za mašinsko učenje generalizuju znanje na osnovu prethodnog iskustva (podataka o objektima koji su predmet učenja) i stečeno znanje koriste kako bi dali odgovore na pitanja za nove objekte (Samuel, 1959). U mnogim oblastima se kontinuirano prikupljaju podaci sa ciljem da se na osnovu njih donesu zaključci i steknu nova znanja. Analiza ovakvih setova podataka uz pomoć tehnika mašinskog učenja omogućava da se otkriju pravilnosti i zavisnosti koje nijesu jednostavne i očigledne.

Postoje različite tehnike i modeli učenja koje su razvijene za izvršavanje različitih zadataka i rješavanje problema iz različitih oblasti. Osnovni oblici mašinskog učenja su (Mohri, Rostamizadeh, & Talwalkar, 2012):

- nadgledano (supervised) učenje - algoritam uči na primjerima, u kojima su poznate određene ulazne veličine i očekivani izlaz za navedene ulaze.
- nenadgledano (unsupervised) učenje - algoritam uči na primjerima koji se sastoje se samo od ulaznih veličina, bez pružanja očekivanih izlaza. Koristi se kada ne postoji pouzdan način da se zna koji izlaz odgovara određenom ulazu.

Mašinsko učenje se, između ostalog, može koristiti za rješavanje problema klasifikacije i regresije. Zadatak programa je da nauči kako da novom ulaznom podatku dodijeli tačnu izlaznu vrijednost. Ukoliko je ta izlazna vrijednost labela (diskretna ili kategorična vrijednost) riječ je o klasifikaciji i problem se svodi na klasifikovanje podataka u definisan, konačan broj klasa. Ukoliko je izlaz realan broj riječ je o regresiji, tj. kontinualnom predviđanju.

GCRF model (Radosavljević, Vučetić, & Obradović, 2010) integriše predikciju tradicionalnih modela nadgledanog učenja i veze između objekata. Tradicionalne tehnike za nadgledano učenje koriste samo informacije koje se nalaze u ulaznim podacima (atribute) kako bi predvidjeli izlaznu vrijednost, ne uzimajući u obzir veze između objekata i zbog toga se nazivaju nestrukturnim prediktorima. Kao nestrukturni prediktori u ovom radu korišćene su neuronske mreže i linearna regresija.

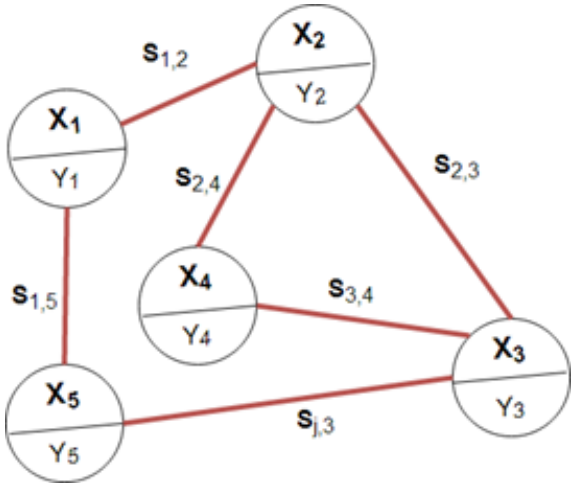
Neuronske mreže (Haykin, 2009) koriste matematičke forme i strukturu ljudskog mozga kako bi razvile strategiju procesiranja podataka. Neuronske mreže imaju sposobnost učenja na primjerima, u kojima su date određene ulazne veličine i očekivani izlaz za navedene ulaze. Kada jednom pronađe pravila za izračunavanje (vezu između ulaza i izlaza), neuronska mreža je sposobna da odredi izlaz za bilo koji ulaz, naravno uz mogućnost greške. Zbog navedenih karakteristika neuronske mreže predstavljaju sisteme koji su u stanju da stiču, čuvaju i koriste iskustveno znanje. Neuronska mreža sastoji se od neurona, koji su grupisani po nivoima. Neuroni na istom nivou vrše slične funkcije i postoje tri vrste nivoa: ulazni, izlazni i skriveni nivo.

Regresija je statistički metod da se pronađe veza između zavisne varijable i jedne ili više nezavisnih varijabli. Linearna regresija (Weisberg, 2005) je najosnovnija vrsta regresije koja je našla primjenu u oblasti mašinskog učenja, jer se može koristiti za podešavanje prediktivnog modela na osnovu posmatranog skupa podataka. U zavisnosti od broja nezavisnih varijabli koristi se jednostavna linearna regresija (Simple Linear Regression - SLR) ili višestruka linearna regresija (Multiple Linear Regression – MLR).

2.2. Strukturna regresija

Tradicionalni modeli za nadgledano učenje su moćan alat za učenje nelinearnih mapiranja, ali su oni fokusirani na predviđanje jednog izlaza, tretiraju sve parove ulaz-izlaz nezavisno i ne mogu da iskoriste veze između objekata kako bi unaprijedili predviđanje. Kao proširenje tradicionalnih modela kreirani su modeli strukturnog predviđanja (Baklr, 2007), koji su dizajnirani da koriste veze između objekata prilikom predviđanja izlaznih vrijednosti. Veze između objekata zavise od konkretne primjene i određene su unaprijed, preko domenskog znanja ili pretpostavkom. Sve je veći broj problema u oblasti računarske vizije i prepoznavanja paterna koji zahtijevaju strukturno učenje, jer postoje vremenske i prostorne zavisnosti koje se ne mogu uočiti tradicionalnim tehnikama.

Problemi koji se mogu riješiti upotrebom strukturnog učenja mogu se predstaviti kao grafovi (slika 1), u kojima svaki čvor (i, i=1, 2, …, N) predstavlja objekat koji ima jedan ili više atributa (X_i) i izlaznu vrijednost (Y_i), dok veze između čvorova zavise od konkretne primjene (mogu biti neusmjerene ili usmjerene, sa težinama ili bez, itd.). Na primjer, veza između bolnica može se bazirati na sličnosti njihovih specijalizacija, veza između naučnih radova može da se odredi na osnovu sličnosti radova koji ih citiraju, veza između dvije osobe može da predstavlja jačinu njihovog prijateljstva itd. Matrica povezanosti grafa koristi se prilikom predviđanja i naziva se matricom sličnosti (S).



Slika 1. Model grafa za strukturnu regresiju

Postoji veliki broj radova u prostornoj statistici koji imaju za cilj da istraže korelaciju u strukturnim podacima. Spatial Auto Regression model (SAR) (Anselin, 1988) je generalizacija modela linearne regresije koja je povećala tačnost predviđanja standardne linearne regresije za mnoge prostorne setove podataka. Gaussian Processes je takođe proširen za strukturne izlaze predlogom Twin Gaussian Processes (TGP) (Bo & Sminchisescu, 2009) modela koji razmatra međuzavisnost između ulaza, kao i korelacije između izlaza. Korelacije ili veze između izlaza mogu da se predstave grafičkim modelima, a najčešće korišćeni grafički modeli za učenje iz prostornih podataka su Markov Random Fields (Solberg, Taxt, & Jain, 1996.) i Conditional Random Fields (CRF) (Lafferty, McCallum, & Pereira, 2001). Većina istraživanja u ovoj oblasti odnosi se na predviđanje diskretnih izlaza (klasifikaciju), dok je primjena strukturnih modela za probleme regresije manje istražena oblast. Takođe, većina modela koristi linearne veze između ulaza i izlaza, dok u realnim situacijama veze često nijesu linearne i ne mogu biti precizno modelovane korišćenjem linearnih funkcija. Neki od postojećih modela mogu da uče veze između objekata na osnovu vrijednosti atributa (Han, Zhang, Ghalwash, Vučetić, & Obradović, 2016), (Stojković, Jelisavčić, Milutinović, & Obradović, 2017), dok neki od njih zahtijevaju da struktura bude dio ulaznih podataka (Radosavljević, Vučetić, & Obradović, 2010), (Glass & Obradović, 2017).

2.3. GCRF model

Gaussian Conditional Random Fields (GCRF) (Radosavljević, Vučetić, & Obradović, 2010) model je jedan od najčešće korišćenih modela strukturne regresije. Glavna pretpostavka ovog modela je da su dva objekta koja su usko povezana veoma slični i samim tim, vrijednosti njihovih izlaznih varijabli bi trebalo da budu slične. GCRF je CRF model vjerovatnoće koji koristi jedan ili više nestrukturnih prediktora za formiranje atributa i istraživanje veza između izlaza. Najveća prednost ovog modela je to što kombinuje izlaze iz tradicionalnih prediktora, kao što su prethodno istrenirane neuronske mreže, kao i prostornu ili vremensku korelaciju između izlaznih varijabli, što može značajno poboljšati predikciju. Model takođe omogućava i uključivanje proizvoljnih osobina ulazno-izlaznih parova u mjeru kompatibilnosti, što znači da bilo koja potencijalno relevantna osobina može biti uključena u model. Prilikom procjene

parametara se automatski određuje stvarna relevantnost svake od osobina, pomoću dodjeljivanja težina. GCRF se može primijeniti na različite probleme regresije, gdje god postoji potreba za integracijom znanja i eksploatacijom korelacije između izlaznih varijabli. Do sada je primijenjen u različitim oblastima kao što su: računarska vizija (Liu, Adelson, & Freeman, 2007), meteorologija (Radosavljević, Vučetić, & Obradović, Neural Gaussian conditional random fields, 2014), (Đurić, Radosavljević, Obradović, & Vučetić, 2015), energetika (Wytock & Kolter, 2013), (Guo, 2013), zdravstvo (Gligorijević, Stojanović, & Obradović, 2015), prepoznavanje govora (Khorram, Bahmaninezhad, & Sameti, 2014), saobraćaja (Qian, Ukkusuri, Yang, & Yan, 2017) itd. U nastavku poglavlja da je kratak opis CRF-a i GCRF-a.

U Conditional Random Field (CRF) modelu (Lafferty, McCallum, & Pereira, 2001), atributi x_j utiču na svaki od izlaza y_i direktno i nezavisno jedan od drugog. Funkcija vjerovatnoće CRF-a definisana je na sledeći način:

$$P(y|\mathrm{x}) = \frac{1}{Z(\mathrm{x},\alpha,\beta)}\exp\Big(\sum_{i=1}^{N} A(\alpha, y_i, \mathrm{x}) + \sum_{i,j:\, i\sim j} I(\beta, y_i, y_j)\Big) \tag{1}$$

Izlaz y_i povezan je sa vektorom ulaza x (x = (x_1, x_2, ..., x_N), gdje je N broj čvorova u grafu) preko funkcije A, koja se naziva potencijal povezanosti (engl. association potential) i u kojoj α predstavlja set parametra dimenzije K. Veća vrijednost funkcije A znači da je zavisnost između izlaza y_i i atributa x veća. Za modelovanje međusobne povezanosti izlaza koristiti se funkcija I, koja se naziva potencijal interakcije (engl. interaction potential) i u kojoj β predstavlja set parametra dimenzije L. Veća vrijednost funkcije I znači da je zavisnost između izlaza y_i i izlaza y_j veća. Oznaka j~i predstavlja povezanost između izlaza y_i i y_j, a Z(x,α,β) je funkcija normalizacije. Ukoliko se pretpostavi da postoji K nestrukturnih prediktora koji predviđaju jedan izlaz y_i, i L matrica sličnosti koje predstavljaju različite vrste veza među izlazima, funkcije A i I mogu biti predstavljene sledećim formulama:

$$A(\alpha, y_i, \mathrm{x}) = -\sum_{k=1}^{K} \alpha_k (y_i - R_{i,k}(\mathrm{x}))^2 \tag{2}$$

$$I(\beta, y_i, y_j) = -\sum_{l=1}^{L} \beta_l S_{ij}^{l} (y_i - y_j)^2 \tag{3}$$

$R_{i,k}$ je predviđena vrijednost dobijena pomoću prediktora k za objekat i, dok je $S^l_{i,j}$ vrijednost veze između objekata i i j u l-toj matrici sličnosti (l = 1 , ... , L).

GCRF je CRF model sa kvadratnim atributima i kvadratnim funkcijama interakcije, pa može direktno da se prevede u Gausovu raspodjelu vjerovatnoće:

$$P(y|\mathrm{x}) = \frac{1}{2\pi|\sum|^{\frac{1}{2}}}\exp(-\frac{1}{2}(y-\mu)^T Q(y-\mu)) \tag{4}$$

gdje je μ matematičko očekivanje, y zavisna promjenjiva, Q matrica tačnosti, ∑ matrica kovarijanse, a $|\sum|$ determinanta matrice kovarijanse. Nakon što se izjednače modeli uslovne vjerovatnoće koji su označeni sa (1) i (4), dobija se matricu tačnosti (Q) definisana na osnovu pouzdanosti ulaza prediktora (koja je izmjerena parametrom α) i važnosti interne strukture (koja je izmjerena parametrom β). Cilj učenja je da se nađu vektori sa parametrima α=(α_1, α_2, ..., α_K) i β=(β_1, β_2, ..., β_L) sa maksimalnom vjerodostojnosti (engl. maximum likelihood) na setu podatka za trening.

K predstavlja broj nestrukturnih prediktora koji se koriste za predviđanje jednog izlaza ($R_{i,k}$), a L predstavlja broj matrica sličnosti (S) koje predstavljaju različite vrste veza među izlazima. Matrica tačnosti se može izračunati po sledećoj formuli:

$$Q = \sum_{k=1}^{K} \alpha_k I + \sum_{l=1}^{L} \beta_l L_l \tag{5}$$

gdje je L_l Laplasova matrica za S_l matricu.

Ukoliko se ulazne vrijednosti dobijene od prediktora označe sa R, formula za konačno predviđanje može biti napisana na sledeći način:

$$\mu = Q^{-1}R\alpha \tag{6}$$

Algoritam koji se koristi za učenje je metod penjanja u pravcu gradijenata (engl. gradient ascent) koji je opisan u knjizi (Shalev-Shwartz & S., 2014). Izvodi logaritamske funkcije vjerodostojnosti (engl. log-likelihood), označene sa l, i promjene α i β vrijednosti prilikom primjene metoda penjanja u pravcu gradijenata predstavljeni su sledećim formulama:

$$\log \alpha_k^{new} = \log \alpha_k^{old} + \eta \frac{\partial l}{\partial \log \alpha_k} \tag{7}$$

$$\log \beta_l^{new} = \log \beta_l^{old} + \eta \frac{\partial l}{\partial \log \beta_l} \tag{8}$$

U izrazima (7) i (8) stopa učenja (engl. learning rate) je označena sa η. Jednostavnost zaključivanja (predviđanje je jednako srednjoj vrijednosti η raspodjele) je u kontrastu sa opštim CRF modelom, koji obično zahtijeva napredne pristupe zaključivanja kao što je Markov Chain Monte Carlo.

2.4. Proširenja GCRF modela i njihova primjena

Od kad se pojavio GCRF model je zainteresovao naučnike iz oblasti mašinskog učenja, primijenjen je u različitim oblastima, ali i proširen za različite namjene. Najznačajnija proširenja GCRF modela su:

- **Unimodal GCRF (UmGCRF)** (Glass, Ghalwash, Vukićević, & Obradović, 2016) metod omogućava modeliranje pozitivnih i negativnih uticaja. Standardni GCRF model je ograničen na pozitivne težine veza u grafu, što znači da se pretpostavlja da je uticaj jednog objekta na drugi uvijek pozitivan, što nije slučaj u realnim primjerima. UmGCRF proširuje opseg za pretragu parametara kako bi dozvolio negativne težine veza. Ovaj model je takođe poboljšao efikasnost računanja što je dovelo do značajnog ubrzanja. Evaluacija UmGCRF izvršena je na problemu predviđanja mjesečnog prijema u bolnicima za 189 klasa bolesti u Kaliforniji (California HCUP baza podataka) (HCUP, 2005-2009). U toku 9 godina prikupljene su informacije o više od 35 miliona pacijenata za 253 različite bolesti na osnovu CSS šeme za kodifikaciju. Autori rada kreirali su mjesečne grafove bolesti (čvor u grafu predstavlja jednu bolest) i svaki od ovih grafova imao je 189 čvorava, zbog nepotpunih informacija tokom vremena. Veza između bolesti bazira se na sličnosti (na skali do 0 do 1) koja je kreirana na osnovu bolest-simptom mreže sličnosti koja je definisana u (Zhou, Menche, Barabasi, & Sharma, 2014). UmGCRF je postigao poboljšanje tačnosti od 12% do 17% u odnosu na standardni GCRF, a nova matematička formulacija je

značajno unaprijedila brzinu i skalabilnost standardnog GCRF modela i dostigla skoro iste performanse kao tehnike aproksimacije.

- **Marginalized GCRF (m-GCRF)** (Stojanović, J., Gligorijević, & Obradović, 2015) metod rješava realan problem nedostatka podataka u djelimično posmatranim vremenskim grafovima. Za standardni GCRF je neophodno obraditi podatke u fazi pretprocesiranja (kako bi se riješio problem sa podacima koji nedostaju), dok m-GCRF može da obradi nepotpune podatke. Prednosti m-GCRF modela pokazane su na primjeru predviđanja padavina na osnovu parcijalnih opservacija klimatskih varijabli u vremenskim grafovima koji obuhvataju kontinentalni dio SAD-a (Menne, Williams, & Vose, 2009). Svaki graf ima 1218 čvorova, pri čemu svaki čvor u grafu predstavlja meteorološku stanicu. Prostorna udaljenost iskorišćena je da se izračuna sličnost (veza) između stanica, i za svaku od stanica poznate su padavine i još 6 atributa. Nije bilo nedostataka u ulaznim podacima, ali oko 5% zavisnih podataka (padavine) je nedostajalo. Eksperiment na ovim podacima pokazao je da m-GCRF dovodi do povećanja preciznosti od 5% u odnosu na neuronske mreže.
- **Uncertainty Propagation GCRF (up-GCRF)** (Gligorijević, Stojanović, & Obradović, 2016) metod kreira predikciju, ali i procjenjuje njenu pouzdanost. Kreiran kako bi olakšao primjenu strukturne regresije za dugoročno donošenje odluka. Up-GCRF uzima u obzir neizvjesnost koja dolazi iz podataka kada procjenjuje neizvjesnost svoje predikcije. Evaluacija up-GCRF modela je takođe izvršena na California HCUP podacima (HCUP, 2005-2009). Cilj je bio da se predvidi prijem u bolnice i stopa smrtnosti na osnovu podataka o pacijentima. U svim eksperimentima up-GCRF je nadmašio ostale modele i po preciznosti i po procjeni pouzdanosti.
- **Representation Learning based Structured Regression (RLSR)** (Han, Zhang, Ghalwash, Vučetić, & Obradović, 2016) metod je sposoban da paralelno uči skrivene ulaze i veze između izlaza. Cilj ovog modela je da se unaprijede GCRF modeli uvođenjem skrivenih varijabli koje su nelinearne funkcije ulaznih varijabli. Model može da uči strukturu (veze između objekata) iz podataka, što znači da nije neophodno unaprijed definisati matricu sličnosti za set podataka za trening. Jedna od primjena RLSR metoda je predviđanje dnevnog prihoda od

solarne energije u 98 država iz Oklahoma Mesonet mreže[1]. Na ovim podacima, RLSR metod je nadmašio sve ostale za najmanje 50%.

Analizom dostupne literature i objavljenih radova koji prikazuju GCRF modele i njihovu primjenu u praksi, ustanovljeno je da ne postoji proširenje GCRF modela koje se može primijeniti na usmjerene grafove, što potvrđuje polaznu hipotezu (H1).

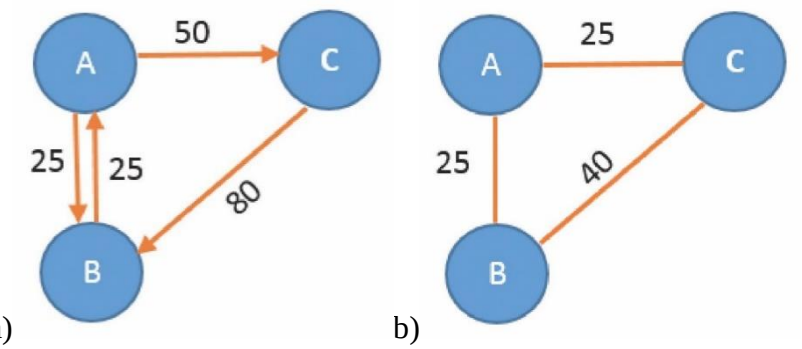
[1] https://www.kaggle.com/c/ams-2014-solar-energy-prediction-contest

3. Usmjereni GCRF

3.1. Analiza problema

Podaci o različitim vrstama mreža (kao što su društvene mreže, saobraćajne mreže, informacione mreže, itd.) mogu se predstaviti preko grafova, u kojima su objekti predstavljeni čvorovima, a odnosi među objektima odgovaraju granama (vezama) između čvorova. Često je neophodno koristiti usmjerene veze (grane) između čvorova u grafu, u skladu sa konkretnom situacijom iz prakse koja podrazumijeva asimetrične veze između objekata. Na primjer, jačina prijateljstva često nije simetrična. U empirijskim studijama (Michell & Amos, 1997), (Snijders, Van de Bunt, & Steglich, 2010) od ispitanika se traži da navedu svoje prijatelje i da odrede koliko su sa njima bliski, što rezultira usmjerenim grafom u kom često postoje jednosmjerna prijateljstva. Još jedan primjer usmjerenog grafa su društvene mreže Twitter, Instagram ili GitHub, u kojima relacija „follower-followee“ ne mora da bude (i često nije) dvosmjerna. Takođe, sistem za razmjenu elektronske pošte (e-mail) može se predstaviti usmjerenim grafom u kom svaki čvor prestavlja adresu elektronske pošte, dok svaka veza sadrži broj poslatih poruka kao težinu.

Grafovi koji su navedeni u ovim primjerima mogu se predstaviti preko asimetrične matrice povezanosti, što znači da GCRF model (koji zahtjeva simetričnu matricu) ne može biti direktno primijenjen za rješavanje ovih problema. Moguće rješenje je da se prije primjene modela asimetrična matrica transformiše u simetričnu, što će vrlo vjerovatno prouzrokovati gubitak preciznosti modela. Primjer grafa koji je dat na slici 2 ilustruje ovaj problem. Slika 2a predstavlja graf sa tri čvora (označena sa A, B i C) čije međusobne veze predstavljaju uticaj jednog čvora na drugi (veća težina veze predstavlja veći uticaj). Sa slike 2a se može zaključiti da na čvor B više utiče čvor C (80) nego čvor A (25). Čvor B nema nikav uticaj na čvor C, dok je čvor C pod uticajem čvora A sa težinom 50. Međusobni uticaj čvorova A i B je jednak 25 u oba smjera. Konvertovanje ovog grafa u neusmjereni (primjenom principa prosječne vrijednosti) daje graf koji je predstavljen na slici 2b. Analiza dobijenog neusmjerenog grafa vodi ka potpuno drugačijim zaključcima, jer je sada uticaj dvosmjeran, što znači da su čvorovi međusobno povezani uticajima iste težine. Uticaji na relacijama A-B i A-C sada su isti, dok uticaj na relaciji B-C ima veliku težinu.

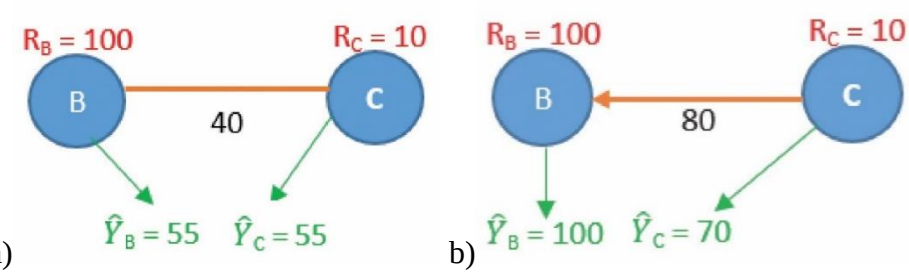


a) b)

Slika 2. Primjer grafa koji predstavlja uticaj između objekata:

a) usmjereni graf b) neusmjereni graf

Uticaj transformacije asimetrične matrice u simetričnu na rezultat predviđanja može se vidjeti na slici 3, na primjeru čvorova B i C. Pretpostavka je da je nestrukturni prediktor predvidio sledeće vrijednosti: 100 za čvor B (R_B) 100 i 10 za čvor C (R_C). Na slici 3a ovi čvorovi su nesimetrično povezani, što znači da će predviđeni izlaz ($\hat{Y}_C$) za čvor C biti blizak vrijednosti koja je predviđena za čvor B ($\hat{Y}_B$). Sa druge strane, na slici 3b je uticaj simetričan, što znači da će vrijednosti predviđenog izlaza za čvorove B i C da se približe jedna drugoj.



a) b)

Slika 3. Ilustrativni primjer grafa koji predstavlja uticaj između objekata B i C, sa odgovarajućim vrijednostima za predviđeni izlaz ($\hat{Y}_B$ i $\hat{Y}_C$)

a) usmjereni graf b) neusmjereni graf

U narednim odjeljcima je predložen novi model, pod nazivom usmjereni GCRF (engl. Directed Gaussian Conditional Random Fields - DirGCRF), koji proširuje standardni GCRF uzimajući u obzir asimetričnu korelaciju između objekata. Opisan je matematički model i način implementacije, prikazani su rezultati testiranja preciznosti modela na sintetičkim i realnim setovima podataka, kao i analiza performansi modela i njegove konveksnosti.

3.2. Matematički model

Činjenica da veze između objekata mogu biti usmjerene narušava jednu od glavnih pretpostavki GCRF modela (Radosavljević, Vučetić, & Obradović, 2010). Iz tog razloga je potrebno izvesti formule za novi model, objasniti u čemu se novi model razlikuje od polaznog modela i predstaviti novi oblik matrice Q. U prvom koraku je potrebno dokazati da Gausova normalna forma (Gaussian Normal Form - GNF) može biti jednaka CRF modelu pod određenim uslovima. Matematička formulacija CRF modela data u izrazu (1) može biti zapisana i na sledeći način, koji prikazuje ekvivalentnost CRF sa GNF-om:

$$\sum_{i=1}^{N} A(\alpha, y_i, \mathrm{x}) + \sum_{i,j:\, j\sim i} I(\beta, y_i, y_j) = -\sum_{i=1}^{N}\sum_{k=1}^{K} \alpha_k \left(y_i - R_{i,k}(\mathrm{x})\right)^2 - \sum_{l=1}^{L}\sum_{i,j:\, i\sim j} \beta_l S_{ij}^l (y_i - y_j)^2 \tag{9}$$

Suma težina svih veza se ne mijenja, i ako se pretpostavi da su sve težine jednake 0 formulacija (9) može biti zapisana u obliku koji ne zahtjeva informacije o strukturi, što omogućava grupisanje sume nezavisnih linearnih i kvadratnih komponenti:

$$-\sum_{i=1}^{N}\sum_{k=1}^{K} \alpha_k y_i^2 + \sum_{i=1}^{N}\sum_{k=1}^{K} 2\alpha_k y_i R_{i,k}(\mathrm{x}) - \sum_{i=1}^{N}\sum_{k=1}^{K} \alpha_k (R_{i,k}(\mathrm{x}))^2 + \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{l=1}^{L} \beta_l S_{ij}^l y_i y_j - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{l=1}^{L} \beta_l \left(S_{ij}^l + S_{ji}^l\right) y_i^2 \tag{10}$$

Jednačina se može segmentirati tako da se izdvoje kvadratna, linearna i konstantna komponenta. Navedene promjenjive su iste za oba modela, bez obzira što se koeficijenti razlikuju. Time se omogućavaju jednaki uslovi za CRF i GNF. Jasniji prikaz dobija se ukoliko se GNF napiše u obliku suma:

$$P(y) = -\frac{1}{2} y^T Q y + y^T b + c = \sum_{i=1}^{N}\sum_{j=1}^{N} Q_{ij} y_i y_j + \sum_{j=1}^{N} y_i b_i + c \tag{11}$$

U izrazu (11) Q, b, i c redom predstavljaju matricu (dimenzija NxN), vektor (dimenzije N) i skalar za GNF. Glavna razlika između standardnog i usmjerenog GCRF modela je da kod usmjerenog GRCF modela zbir težina po redovima u matrici S, u oznaci

rowsum(S), nije jednak zbiru težina po kolonama, u oznaci colsum(S), tako da kvadratna komponenta u izrazu (11) može biti zapisana na sledeći način:

$$\sum_{i=1}^{N}\sum_{j=1}^{N} Q_{ij}y_i y_j = \sum_{k=1}^{K}\alpha_k \sum_{i=1}^{N} y_i^2 + \sum_{i=1}^{N}\sum_{l=1}^{L}\big(rowsum(S_l) + colsum(S_l)\big)\,\beta_l y_i^2 - \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{j=1}^{L}\beta_l S_{ij}^{l} y_i y_j \tag{12}$$

Na glavnoj dijagonali matrice Q nalaze se elementi y_i^2, dok su $y_i y_j$ ostali elementi matrice. Dakle, matrica Q može biti predstavljena kao zbir dijagonalne matrice D i matrice povezanosti A:

$$Q = D + \mathrm{A} \tag{13}$$

$$D_{ii} = \sum_{k=1}^{K}\alpha_k + \sum_{l=1}^{L}\beta_l \frac{1}{2}\big(rowsum(S_l) + colsum(S_l)\big) \tag{14}$$

$$A_{ij} = -\sum_{l=1}^{L}\beta_l S_{ij}^{l} \tag{15}$$

Laplasova matrica za asimetričnu matricu se može definisati na sledećim izrazom:

$$L = -\frac{1}{2}\big(diag\big(rowsum(S) + colsum(S)\big) - 2S\big) \tag{16}$$

gdje je $diag$ dijagonalna matrica u kojoj su elementi dijagonale jednaki elementima vektora koji se dobija kao rezultat izraza $rowsum(S) + colsum(S)$.

Nova matrica Q se može predstaviti izrazom:

$$Q = \left(\sum_{k=1}^{K}\alpha_k\right) I + \sum_{l=1}^{L}\beta_l L_l \tag{17}$$

Prilikom mapiranja linearnih koeficijenata iz CRF-a u GNF, promjena strukture ne mijenja mapiranja iz polaznog GCRF modela, što znači da je b i dalje jednako proizvodu R i α (b = Rα).

Funkcija vjerodostojnosti (18) može se izjednačiti sa GNF-om (19) ukoliko su dati isti uslovi (20) i (21).

$$P(y-\mu)=\frac{1}{2\pi|\sum|^{\frac{1}{2}}}\exp(-\frac{1}{2}(y-\mu)^T\sum^{-1}(y-\mu)) \quad (18)$$

$$P(y)=\frac{1}{2\pi|\sum|^{\frac{1}{2}}}\exp(-y^T\sum^{-1}y+b^Ty+c) \quad (19)$$

$$c=-\mu^T\sum^{-1}\mu \quad (20)$$

$$\mu=\sum b \quad (21)$$

gdje μ predstavlja optimalno predviđanje, ukoliko je data matrica kovarijanse (Σ) i linerana komponenata za GNF (b). ∑ je matrica kovarijanse, a $\sum^{-1}$ označava inverznu matricu kovarijanse.

DirGCRF koristi navedene formule i metod penjanja u pravcu gradijenata kako bi pronašao optimalne vrijednosti parametara α_i i β_i. Neophodno je još odrediti parcijalne izvode prvog reda logaritamske funkcije vjerodostojnosti po α_i i β_i, koji su neophodni za primjenu algoritma penjanja u pravcu gradijenata. Izraz za logaritamsku funkciju vjerodostojnosti je:

$$l=-\frac{1}{2}(y-\mu)^T\sum^{-1}(y-\mu)-\frac{1}{2}\log\sum \quad (22)$$

Iz GNF-a μ se može napisati kao:

$$\mu=Q^{-1}b \quad (23)$$

S obzirom da se μ nalazi u logaritamskoj funkciji vjerodostojnosti, funkcija l se može predstaviti izrazom:

$$l=-\frac{1}{2}(y^TQy-y^TQ\mu-\mu^TQy+\mu^TQ\mu)-\log|Q^{-1}|^2 \quad (24)$$

Q se računa po formuli (17) i zavisi od vektora parametara α i β. Parcijalni izvodi prvog reda logaritamske funkcije vjerodostojnosti po α_i i β_i su:

$$\frac{\partial l}{\partial\alpha_i}=-\frac{1}{2}\left[(y-\mu)^T(y-\mu)+(R_i-\mu)^T[I+Q^{-1}Q](\mu-y)\right]+\frac{1}{2}Tr(Q^{-1}) \quad (25)$$

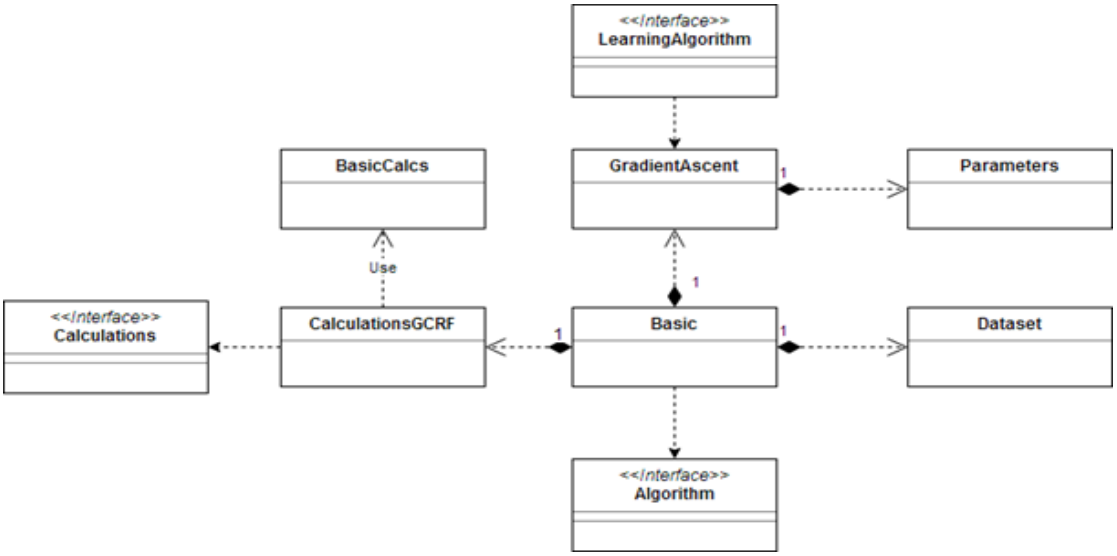
$$\frac{\partial l}{\partial\beta_i}=-\frac{1}{2}\left[y^TL_iy-(-Q^{-1}L_i\mu)^TQy-\mu^TL_iy+(-Q^{-1}L_i\mu)^TQ\mu\right]+\frac{1}{2}Tr(L_iQ^{-1}) \quad (26)$$

Tr predstavlja trag matrice, tj. zbir elemenata na glavnoj dijagonali matrice.

3.3. Implementacija modela

Model je implemetiran u Java programskom jeziku, u Eclipse razvojnom okruženju. U cilju jednostavnije implementacije DirGCRF modela i lakšeg poređenja sa standardnim GCRF-om, prvo je kreiran skup osnovnih klasa koje obezbjeđuju generalnu strukturu i logičke komponente koje su neophodne za sve modele bazirane na GCRF-u. Nakon toga je DirGCRF model implementiran preko nasleđivanja osnovinih klasa i redefinisanja postojećih i dodavanja novih metoda. Osnovne klase (slika 4) su:

- ***BasicCalcs*** – klasa koja sadrži statičke metode za neophodne matematičke proračune, kao što je množenje vektora, množenje matrica, računanje inverzne matrice itd.
- ***Calculations*** – interfejs koji specificira listu metoda za pravila izračunavanja koje svaki GCRF model mora da sadrži
- ***CalculationsGCRF*** – klasa koja implementira interfejs *Calculations* koristeći pravila standardnog GCRF modela.
- ***Dataset*** – klasa koja se koristi za specifikaciju i kreiranje setova podatka
- ***LearningAlgorithm*** - interfejs koji specificira listu metoda koje mora da sadrži svaki algoritam za učenje
- ***GradientAscent*** - klasa koja implementira interfejs *LearningAlgorithm Calculations* koristeći pravila algoritma penjanja u pravcu gradijenata
- ***Parameters*** – klasa koja specificira sve parametre koji su potrebni algoritmu penjanja u pravcu gradijenata
- ***Algorithm*** – interfejs koji specificira listu metoda koje svaki GCRF model mora da sadrži
- ***Basic*** – osnovna klasa za GCRF model (algoritam)



Slika 4. Klasni dijagram koji predstavlja osnovne klase za implementaciju GCRF modela

Nakon implementacije standardnog GCRF-a na jednostavan način se može implemetirati bilo koje od njegovih proširenja. Glavna razlika između GCRF i DirGCRF modela je što je matrica S sada asimetrična, pa je neophodno definisati nove formule za računanje matrice Q i Laplasove matrice, kao i parcijalne izvode prvog reda funkcije vjerodostojnosti po α i β parametrima. Zbog toga je kreirana nova klasa (pod nazivom *CalculationsDirGCRF*), koja nasleđuje klasu *CalculationsGCRF* i redefiniše sledeće metode:

- l – metoda koja računa Laplasovu matricu
- q – metoda koja računa matricu tačnosti (Q)
- dervativeAlpha – metoda koja računa izvod po α parametru
- dervativeBeta - metoda koja računa izvod po β parametru

Takođe je kreirana nova klasa pod nazivom DirGCRF, koja nasleđuje klasu *Basic* i definiše da se koriste pravila za izračunavanje definisana u klasi *CalculationsDirGCRF* i algoritam za učenje definisan u klasi *GradientAscent*.

3.4. Testiranje modela

Preciznost DirGCRF modela na različitim realnim setovima podataka upoređena je sa sledećim modelima:

- **Standardni GCRF** (Radosavljević, Vučetić, & Obradović, 2010)**:** Kako bi se standardni GCRF mogao primijeniti na usmjerene grafove, S matrica se konvertuje iz asimetrične u simetričnu. U simetričnoj matrici svaki par čvorova povezan je sa jednom neusmjerenom vezom čija je težina jednaka prosjeku težina iz odgovarajuće asimetrične matrice. Neuronske mreže se koriste kao nestrukturni prediktor i za usmjereni i za standardni GCRF.
- **Neuronske mreže (NN)** (Haykin, 2009)**:** Broj neurona u ulaznom nivou je isti kao broj varijabli u razmatranom setu podataka, a broj izlaznih neurona je 1 za sve setove. Broj neurona u skrivenom sloju je određen na osnovu preciznosti na podacima za trening.
- **Linearna regresija (LR) ili višestruka linearna regresija (MLR)** (Weisberg, 2005)**:** U zavisnosti od broja varijabli u razmatranom setu podataka koristi se obična ili višestruka linearna regresija. Koeficijenti se dobijaju na osnovu vrijednosti varijabli u podacima za trening i zatim se primjenjuju na podatke za testiranje u cilju dobijanja predikcije.
- **Metod posljednje vrijednosti (Last):** U realnim setovima podataka grafovi evoluiraju i zbog toga se razmatra jednostavan metod posljednje vrijednosti, u kom se predviđa da će izlazna promjenjiva imati istu vrijednosti koju je imala u prethodnoj tački uzorkovanja.
- **Metod prosjeka (Average):** Jednostavna metoda koja računa da će izlazna promjenjiva imati vrijednost koja je jednaka prosjeku vrijednosti iz prethodnih tački uzorkovanja.

Za računanje preciznosti svih metoda korišćen je R^2 koeficijent odlučnosti, koji mjeri koliko se izlaz modela poklapa sa očekivanom vrijednosti. R^2 se računa po sledećoj formuli:

$$R^2 = 1 - \sum_{i=1}^{N} \frac{(y_i - \hat{y}_i)^2}{(y_i - y_{avg})^2}$$

gdje je $\hat{y}$ predviđena vrijednost, y_i stvarna (očekivana) vrijednost, y_{avg} prosjek y vrijednosti za N primjera. Vrijednosti R^2 koeficijenata su najčešće u opsegu od 0 do 1. Prilikom perfektnog poklapanja rezultat će biti 1, dok u slučaju nekih veoma loših prediktora vrijednost može da bude i negativna.

3.4.1. Testiranje modela na sintetičkim podacima

Cilj eksperimenata na sintetičkim podacima je da se predloženi model testira na različitim vrstama grafova, pod kontrolisanim uslovima. Model je testiran sa sledećim vrstama grafova:

- **Potpuni povezan usmjereni graf:** Svaki par različitih čvorova povezan je sa dvije veze (po jedna u svakom smjeru) koje imaju različite težine.
- **Usmjereni graf sa *p* vjerovatnoćom veze:** Usmjereni grafovi različitih gustina. Za svaki par različitih čvorova slučajno se generiše broj između 0 i 1. Ukoliko je taj broj veći od *p* odabrani par će biti povezan vezom slučajno izabrane težine.
- **Usmjereni graf bez povratnih veza:** Svaki par različitih čvorova povezan je samo jednom vezom, čiji se smjer bira slučajno. Na primjer, ukoliko postoji veza od čvora A do čvora B, onda ne može postojati veza od čvora B do čvora A.
- **Usmjereni aciklični graf:** Graf u kom nema zatvorenih petlji (ciklusa). Na primjer, ne postoji put koji počinje u čvoru A i prateći usmjerenu sekvencu veza dolazi nazad u čvor A.
- **Lanac:** Svi čvorovi su povezani u jednu sekvencu.
- **Binarno stablo:** Graf koji ima strukturu stabla, u kom svaki čvor smije imati maksimalno dva potomka.

Težine veza (matrica S) i vrijednosti koje je predvidio nestrukturni prediktor (R) se takođe slučajno generišu. Težine veza u grafu (vrijednosti u matrici S) su u opsegu od 0 do 1, dok vektor R sadrži decimalne brojeve između 0 i neke zadate vrijednosti (za ove setove podatka korišćena je vrijednost 5). Generisani S i R se koriste za izračunavanje očekivane vrijednosti izlazne varijable y za svaki čvor, u skladu sa izrazom (6), uz dodavanje slučajno generisanog odstupanja. Kako bi se izračunao vektor y, neophodno je da se izaberu vrijednosti α_i i β_i parametara. Prilikom testiranja modela korišćen je jedan nestrukturni prediktor (jedan α parametar) i jedna matrica sličnosti (jedan β parametar). U tim slučajevima (kada postoje jedan α i jedan β parametar) za modele bazirane na GCRF-u bitan je samo odnos između vrijednosti parametara, a ne i njihova stvarna vrijednost. Veća vrijednost parametra α znači da na formiranje modela više

utiču vrijednosti koje je predvidio nestrukturni prediktor (R), dok veća vrijednost parametra β znači da na formiranje modela više utiče struktura (S).

Prilikom poređenja preciznosti usmjerenog i standardnog GCRF izvršeno je testiranje svih navedenih vrsta grafova. Vrijednost parametra α je podešena na 5, a parametra β na 1. Jedan graf je korišćen za treniranje i pet grafova za testiranje modela, a svaki od njih je imao 200 čvorova. Prosječna vrijednost R^2 koeficijenta i standardna devijacija za oba modela predstavljeni su tabeli 1.

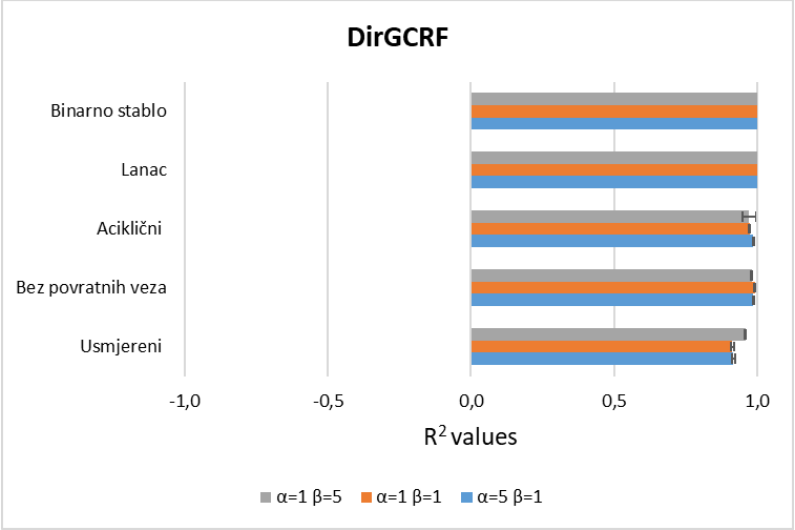
Tabela 1. Prosječna vrijednost R^2 koeficijenta (± standardna devijacija) DirGCRF i GCRF modela za različite vrste sintetički generisanih usmjerenih grafova, sa vrijednostima parametra $\alpha = 5$ i $\beta = 1$

Vrsta grafa	**DirGCRF**	**GCRF**
Potpuno povezan usmjereni graf	0,9176 (±0,00625)	0,5893 (±0,02680)
Usmjereni graf sa vjerovatnoćom veze p=0,5	0,9799 (±0,00332)	0,6582 (±0,06063)
Usmjereni graf sa vjerovatnoćom veze p=0,2	0,9951 (±0,00074)	0,8880 (±0,00846)
Usmjereni graf bez povratnih veza	0,9865 (±0,00084)	0,4608 (±0,03497)
Usmjereni aciklični graf	0,9881 (±0,00019)	0,2580 (±0,03584)
Lanac	0,9965 (±0,00104)	0,6992 (±0,03949)
Binarno stablo	0,9983 (±0,00023)	0,8348 (±0,04647)

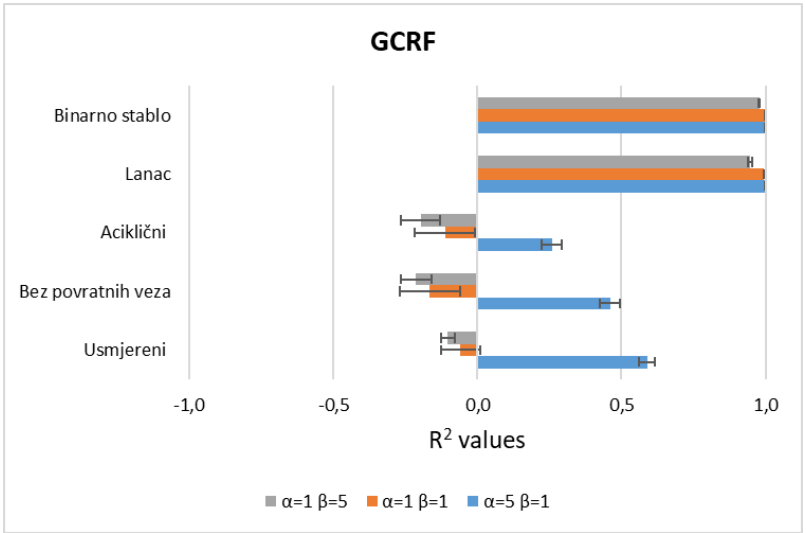
Rezultati pokazuju da DirGCRF ima veću preciznost nego standardni GCRF za sve vrste usmjerenih grafova. Za potpuno povezan usmjeren graf DirGCRF ima za 0,33 veću preciznost nego GCRF. Sa smanjenjem vjerovatnoće postojanja veze graf postaje rjeđi i razlika u preciznosti između DirGCRF i GCRF modela se smanjuje. Za grafove koji nemaju direktne povratne veze ili cikluse, DirGCRF se pokazao mnogo preciznijim, sa većim R^2 koeficijentom za 0,53, odnosno 0,73, što dokazuje superiornost ovog modela za usmjerene grafove. Takođe se primjećuje da u svim eksperimentima R^2 koeficijent DirGCRF modela ima veoma malu standardnu devijaciju (od 0,007 do 0,00004). Jedini izuzetak su rezultati za lanac i binarno stablo, gdje su oba algoritma pokazala sličnu preciznost, što je očekivano, s obzirom na to da su ove strukture veoma rijetke i da na svaki čvor mogu uticati najviše dva druga čvora.

U cilju analize uticaja α i β vrijednosti (odabranih prilikom generisanja podataka) na preciznost DirGCRF i GCRF modela testirane su tri različite kombinacije:

1. Veća vrijednost parametra α (α = 5, β = 1), što znači da veći uticaj imaju vrijednosti koje je predvidio nestrukturni prediktor.
2. Veća vrijednost parametra β (α = 1, β = 5), što znači da veći uticaj ima struktura.
3. Ista vrijednost za oba parametra (α = 1, β = 1), što znači da obje informacije imaju jednak uticaj.



Slika 5. Prosječne vrijednosti R^2 koeficijenta za DirGCRF model za različite vrste sintetički generisanih usmjerenih grafova, sa različitim vrijednostima α i β parametara



Slika 6. Prosječne vrijednosti R^2 koeficijenta za GCRF model za različite vrste sintetički generisanih usmjerenih grafova, sa različitim vrijednostima α i β parametara

Iz rezultata koji su predstavljeni na slici 5 može se vidjeti da je razlika u vrijednostima R^2 koeficijenta za sve tipove grafova veoma mala ako se koristi DirGCRF. Sa druge strane, rezultati sa slike 6 pokazuju da za standardni GCRF postoje velike razlike, pogotovo za usmjerene grafove i usmjerene grafove bez povratnih veza ili ciklusa. Na primjer, za usmjerene grafove povećanje vrijednosti β parametra je prouzrokovalo malo povećanje preciznost DirGCRF-a (sa 0,92 na 0,96), ali veoma veliki pad preciznosti kod GCRF-a (sa 0,59 na -0,1). Navedeno jasno pokazuje da standardni GCRF ne može da pruži dobre rezultate sa podacima koji imaju asimetričnu strukturu, pogotovo za setove podatka u kojima je struktura važnija i korisnija.

3.4.2. Testiranje modela na realnim podacima

Predloženi model je testiran na četiri realna seta podataka: Delinquency (Snijders, Van de Bunt, & Steglich, 2010), Teenagers (Michell & Amos, 1997), Glasgow (Bush, West, & Michell, 1997) i Geostep (Šćepanovic, Vujičić, Matijević, & Radunović, 2015). Prva tri seta sadrže podatke o navikama mladih ljudi (npr. podatke o konzumaciji cigareta i alkohola) i mrežu njihovog prijateljstva u različitim vremenskim periodima, dok Geostep sadrži podatke o igrama lova na blago. Atributi čvorova, težine veza i vrijednosti izlaznih varijabli su izvedeni iz podataka, a sve vrijednosti su normalizovane da budu u opsegu od 0 do 1. Detaljnije informacije o svim setovima podataka nalaze se u tabeli 2.

Tabela 2. Informacije o realnim setovima podataka

Set podataka (čvorovi)	**Broj tački uzorko-vanja**	**Atributi (x)**	**Izlazna varijabla (y)**	**Sličnost (S)**
Delinquency (26 učenika)	4	1. nivo delikvencije u prethodnoj tački 2.konzumiranje alkohola	nivo delikvencije	prijateljstvo
Teenagers (50 tinejdžera)	4	1. konzumiranje alkohola u prethodnoj tački	konzumiranje alkohola	prijateljstvo

Glasgow (129 učenika)	3	1. konzumiranje alkohola 2. konzumiranje kanabisa 3. da je u ljubavnoj vezi 4. iznos mjesečnog džeparca	konzumiranje cigareta	prijateljstvo
Geostep (50 igara)	/	1. broj tragova u kategoriji „društvo“ 2. broj tragova u kategoriji „biznis“ 3. broj tragova u kategoriji „putovanja“ 4. broj tragova u kategoriji „ostalo“ 5. privatnost igre 6. trajanje igre	relevantnost za turističke svrhe	sličnost između igara

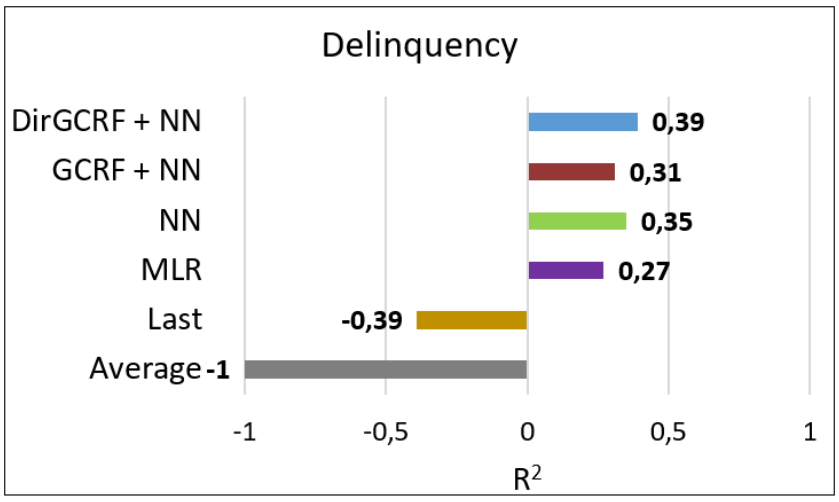
Delinquency set podataka

Delinquency[2] set sadrži podatke o 26 đaka (od 11 do 13 godina) jedne njemačke škole, koji su prikupljeni kroz četiri periodična ispitivanja, između septembra 2003. i juna 2004. godine. U svakoj tački uzorkovanja za svakog studenta poznati su i nivo delikvencije i uzimanja alkohola, koji mogu imati vrijednost od 1 do 5, a cilj je da se predvidi nivo delikvencije. Takođe, za svako ispitivanje je data matrica prijateljstva, koja je formirana tako što je svaki đak birao do 12 najboljih prijatelja. Ukupan broj veza u ovim matricama je između 88 i 133 (gustina od 13% do 20%). U prosjeku je 49% prijateljstva bilo jednosmjerno. Sličnost studenta *i* sa studentom *j* u određenoj tački uzorkovanja *t* (S^t_{ij}) računa se na osnovu postojanja prijateljstava u trenutnoj tački uzorkovanja, ali i u svim prethodnim:

$$S_{ij}^{l} = \frac{1}{t}\sum_{k=1}^{l} S_{ij}^{k}$$

[2] https://www.stats.ox.ac.uk/~snijders/siena/tutorial2010_data.htm.

Za x vrijednosti koristi se podatak o konzumiranju alkohola, kao i prethodni nivo delikvencije. Treniranje je obavljeno nad podacima iz druge i treće tačke, a model je testiran nad podacima iz četvrte tačke uzorkovanja.



Slika 7. Rezultati za Delinquency set podataka

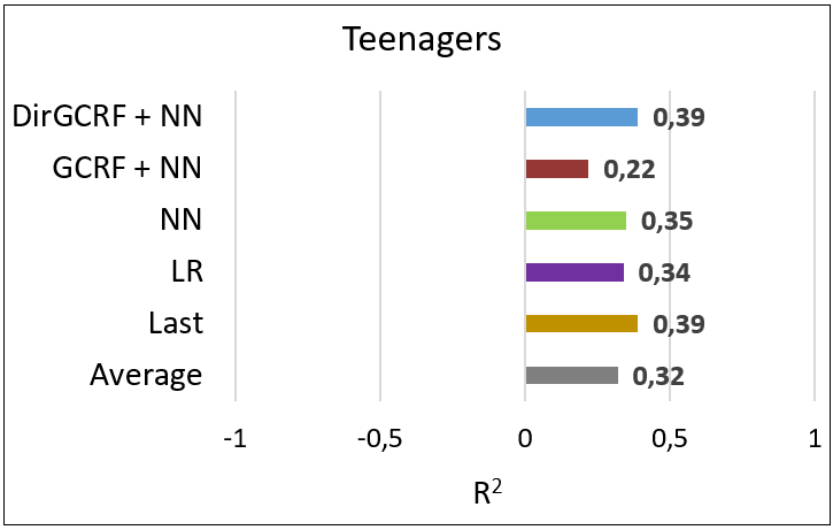
Eksperiment je pokazao da DirGCRF ima veću preciznost nego svi ostali konkurentski modeli (slika 7). DirGCRF ima za 8% veću preciznost nego običan GCRF, a 4% veću nego neuronska mreža. Sledeći metod po preciznosti je neuronska mreža, dok je MLR precizniji od Last i Average metoda, koje imaju negativan R^2 koeficijent. GCRF ima manji R^2 nego neuronska mreža, što znači da korišćenje simetrične matrica prijateljstva (koja je dobijena konvertovanjem) nije moglo da poboljša predikciju.

Teenagers set podataka

Teenagers[3] set sadrži podatke o 50 tinejdžera (13 godina) iz jedne škotske škole, koji su prikupljeni kroz tri periodična ispitivanja (od 1995 do 1997). Kao i u slučaju Delinquency seta podataka, tinejdžeri su birali do 12 najboljih prijatelja. Ukupan broj veza u matricama je između 113 i 122 (gustina oko 5%). U prosjeku je 60% prijateljstva jednosmjerno. Za računanje matrice sličnosti korišćen je isti princip kao i za Delinquency set podataka. Dat je i podatak o konzumiranju alkohola (na skali od 1 do

[3] https://www.stats.ox.ac.uk/~snijders/siena/s50_data.htm.

5), a cilj je predvidjeti vrijednost za konzumiranje alkohola u trećoj tački uzorkovanja, na osnovu podataka iz prethodne dvije tačke.



Slika 8. Rezultati za Teenagers set podataka

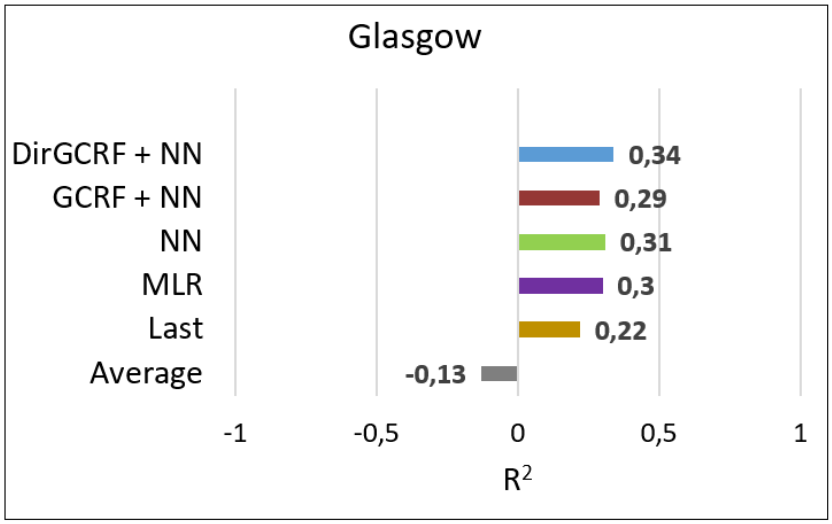
Na ovom setu podataka DirGCRF je za 17% precizniji od standardnog GCRF-a, a za 4% od neuronske mreže (slika 8). Neuronska mreža i linearna regresija imaju sličnu preciznost za ovaj set. Jednostavan Last metod ima veći vrijednost R^2 koeficijenta u odnosu na oba nestrukturna prediktora i isti kao DirGCRF, dok i Average metod ima visoku preciznost. Ovakvi rezultati prouzrokovani su time što ne postoje dodatne varijable, već se za predikciju koristi samo prethodna vrijednost izlazne varijable (y).

Glasgow set podataka

Glasgow[4] set sadrži podatke o 160 đaka iz jedne srednje škole u Glazgovu, koji su prikupljeni kroz tri periodična ispitivanja u toku dvije godine. Za ovaj eksperiment korišćeni su podaci o 129 studenta, koji su bili prisutni u toku sva tri ispitivanja. Đaci su birali do 6 prijatelja i ocjenjivali ih ocjenom od 0 do 2, na sledeći način: 1 - najbolji prijatelj, 2 - samo prijatelj, 0 - nije prijatelj. Ukupan broj veza u matricama je bio oko 362 (gustina 2%). U prosjeku je 72% prijateljstva jednosmjerno. Cilj je da se predvidi konzumiranje cigareta, a korišćene su sledeće varijable (x):

[4] https://www.stats.ox.ac.uk/~snijders/siena/Glasgow_data.zip.

- Konzumiranje alkohola (od 1 do 5)
- Konzumiranje kanabisa (od 1 do 4)
- Ljubavna veza (označava da li je osoba u vezi ili ne)
- Iznos mjesečnog džeparca



Slika 9. Rezultati za Glasgow set podataka

Model je testiran nad podacima iz treće tačke uzorkovanja, dok su podaci iz prve dvije tačke korišćeni za treniranje. Sa slike 9 se vidi da DirGCRF ima najveću preciznost, ali i da skoro svi konkurentski modeli (osim jednostavnih Last i Average) imaju bliske vrijednosti R^2 koeficijenta. Postoji primjetna razlika između upotrebe asimetrične i simetrične strukture, DirGCRF je 5% precizniji od standardnog GCRF-a.

Geostep set podataka

Geostep[5] set sadrži podatke o 50 igara lova na blago. Svaka igra može da ima najviše 10 tragova i svaki trag pripada jednoj od 4 kategorije. Cilj je da se predvidi da li se igra može koristi u turističke svrhe. Varijable koje su korišćene kao x vrijednosti su: broj tragova u svakoj od kategorija (biznis, društvo, putovanja i ostalo), privatnost igre i trajanje igre. Slučajno je odabrano 25 igara za treniranje modela, dok su ostale korišćene za testiranje. Matrica sličnosti kreirana je na osnovu podataka o igri, tako što je sličnost igre *i* sa igrom *j* (S_{ij}) definisana kao zbir zajedničkog broja tragova u svakoj

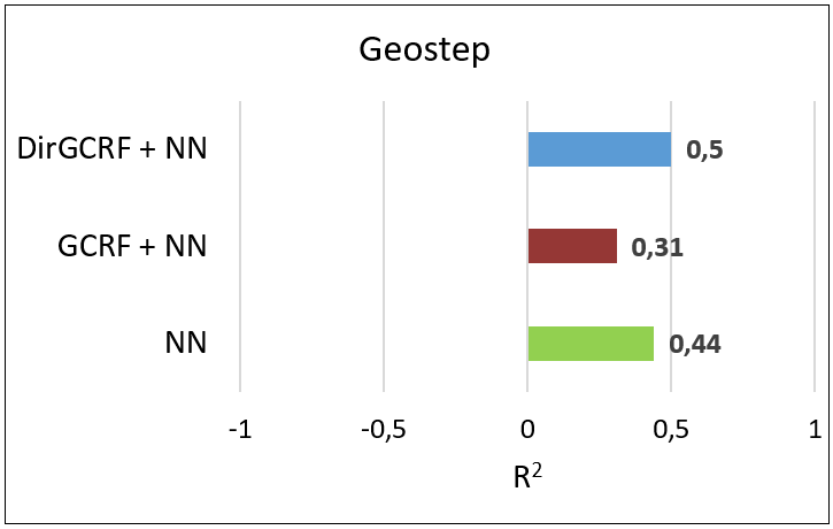
[5] http://www.geostep.me/.

od kategorija podijeljen sa ukupnim brojem tragova u igri *i*. Formula za računanje sličnosti:

$$S_{ij}^{l} = \frac{\sum_{k=1}^{4} min(C_i^k, C_j^k)}{\sum_{k=1}^{4} C_j^k}$$

gdje C^k_i predstavlja broj tragova kategorije *k* u igri *i*.

Na osnovu rezultata koji su predstavljeni na slici 10 se vidi da je upotreba asimetrične strukture značajno poboljšala rezultate neuronske mreže. S druge strane, razlika između GCRF-a i neuronske mreže je najveća za ovaj set podataka (preciznost GCRF-a je mnja za 13 %), što znači da je upotreba simetrične matrice sličnosti pogoršala predikciju. Za ovaj set podatka nije bilo moguće primijeniti Last i Average metode, jer podaci nemaju temporalan karakter.



Slika 10. Rezultati za Geostep set podataka

Na osnovu rezultata testiranja za sva četiri seta podataka (slike 7 - 10) se može zaključiti da su rezultati DirGCRF-a, GCRF-a i neuronske mreže konzistentni. U svakom eksperimentu DirGCRF ima najveću preciznost, dok GCRF ima manju preciznost od neuronske mreže. Ovi rezultati potvrđuju hipotezu da će predloženo proširenje GCRF modela omogućiti preciznije predviđanje od standardnog GCRF modela i tradicionalnih nestrukturnih prediktora (H2).

3.4.3. Performanse modela

Kompleksnost DirGCRF modela je ista kao i kompleksnost standardnog GCRF-a (Radosavljević, Vučetić, & Obradović, 2010). Ukoliko set za trening ima N čvorova u i učenje traje T iteracija, biće potrebno $O(TN^3)$ vremena da se model istrenira. Najzahtijevniji korak u treniranju je računanje inverzne matrice.

Brzina modela testirana je na sintetički generisanim potpuno povezanim usmjerim grafovima sa različitim brojem čvorova: 500, 1.000, 5.000, 10.000 i 15.000. Treniranje je trajalo 50 iteracija i eksperimenti su izvršeni na računaru sa Windows operativnim sistemom koji ima 32GB memorije (28GB za JVM) i 3,4 GHz CPU (tabela 3). Sa povećanjem broja čvorova model značajno usporava, što je prouzrokovano objektno-orjentisanom prirodom Java jezika, koji zahtjeva više vremena i više memorije za izvršavanje različitih računskih operacija sa velikim matricama.

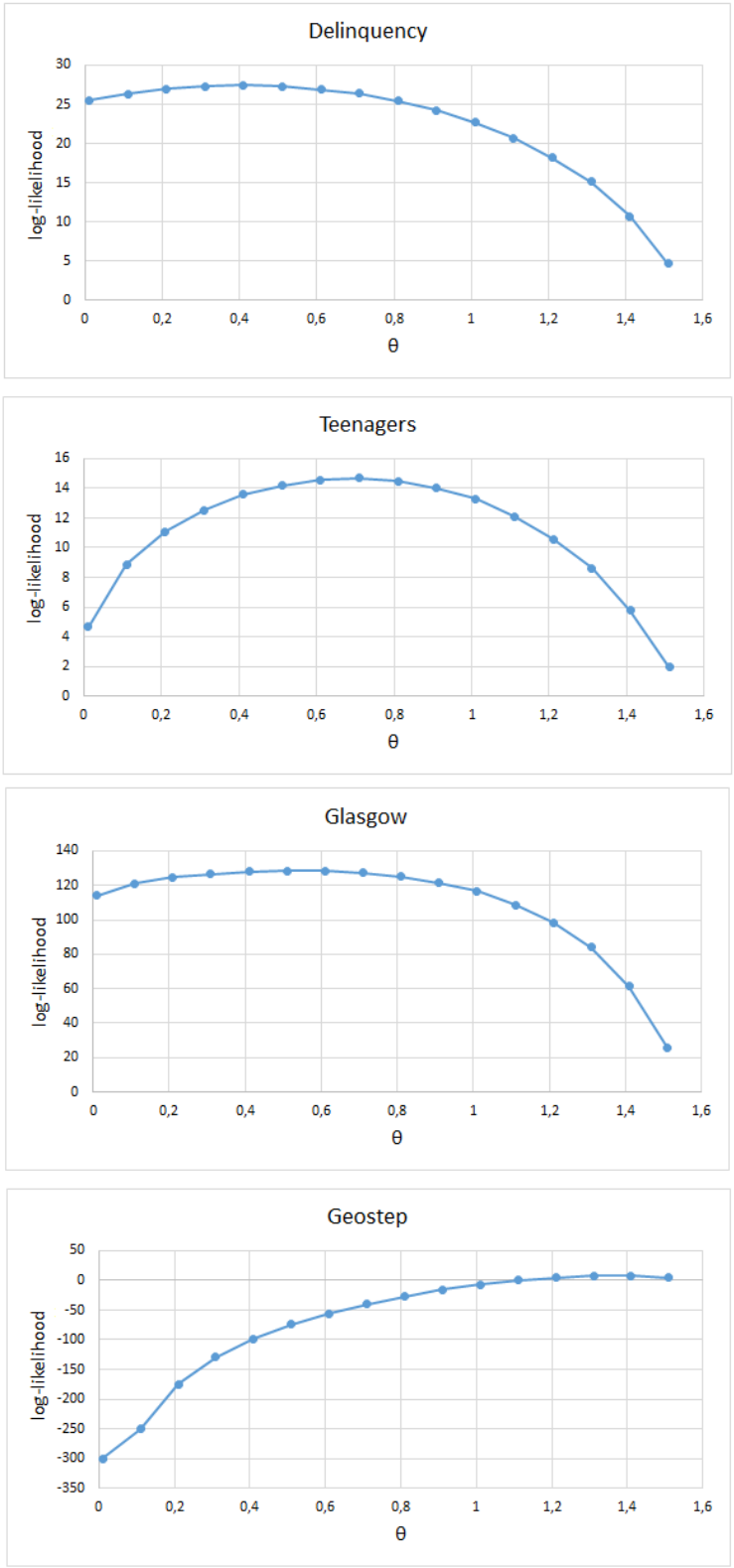
Tabela 3. Vrijeme izvršavanja DirGCRF za potpuno povezan usmjeren graf sa različitim brojem čvorova

Broj čvorova	**Potrebno vrijeme**
500	8 s
1000	48 s
5000	2 h
10.000	17 h
15.000	2,2 dana

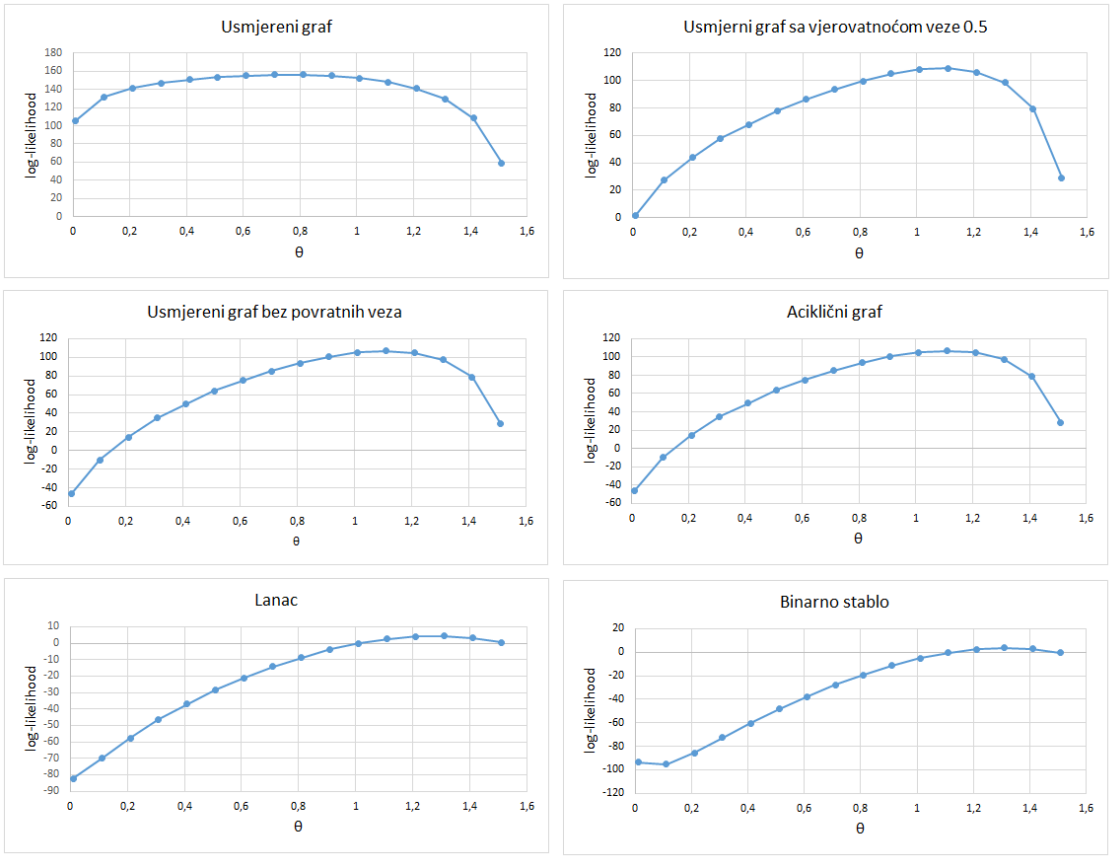
3.4.4. Konveksnost modela

Eksperimentalni rezultati koji su predstavljeni u prethodnim sekcijama dobijeni su za specifičnu vrijednost hiperparametra θ. Dodatni eksperimenti sprovedeni su kako bi empirijski pokazala konveksnost modela. U ovom eksperimentu, θ se povećava od 0 do $\pi/2$, sa korakom 0,1. Za svako θ računaju se α i β kao $\alpha = \sin(\theta)$ i $\beta = \cos(\theta)$, a nakon toga se računa logaritamska funkcija vjerodostojnosti sa tim parametrima i grafički prikazuju vrijednosti. Rezultati, koji su predstavljeni na slikama 11 i 12, pokazuju da je funkcija vjerodostojnosti konveksna sa parametrima α i β i da njena optimizacija vodi do globalnog optimuma. Jedini izuzetak je binarno stablo, za koje udubljenje na lijevoj strani krive narušava konveksnost. Ipak, procedura optimizacije uspjeva da pronađe globalni maksimum, čak i kada počne blizu nekog lokalnog maksimuma, koji nije

globalni. Činjenica da je moguće grafički prikazati cijelu funkciju vjerodostojnosti garantuje da se uspješno pronalazi globalni maksimum.



Slika 11. Eksperimentalni dokaz konveksnosti modela za realne setove podataka



Slika 12. Eksperimentalni dokaz konveksnosti modela za sintetičke setove podataka

4. Softverski alat GCRF GUI TOOL

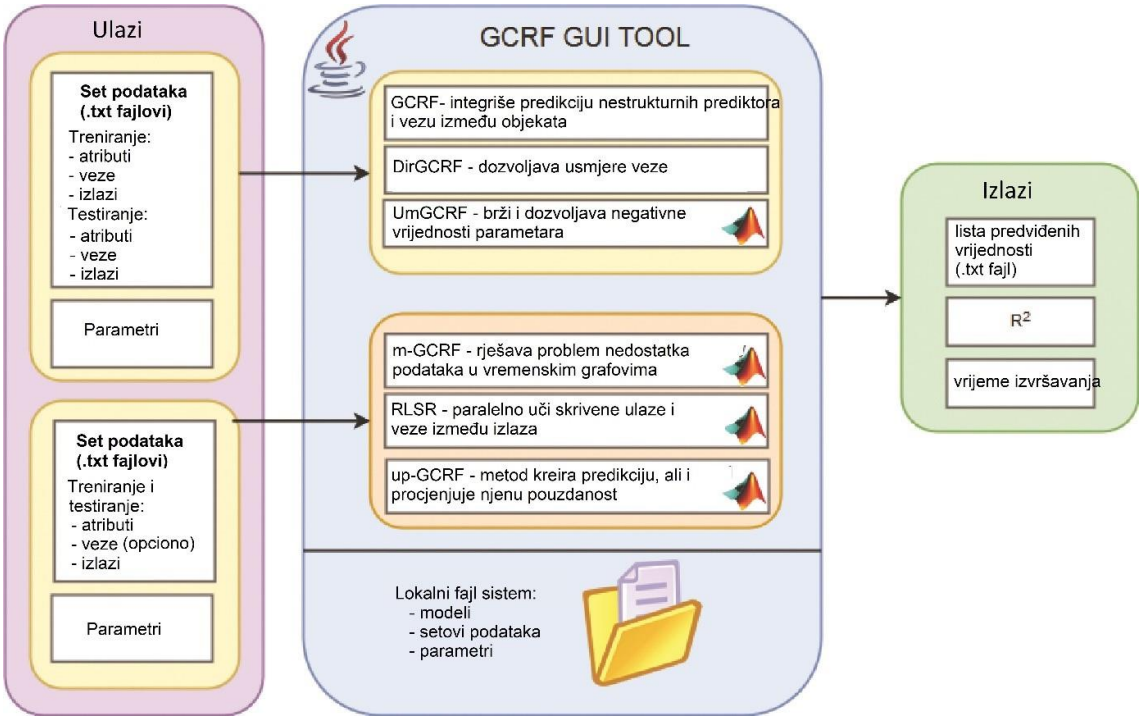
4.1. Pregled sistema

GCRF GUI TOOL može da se koristi za treniranje i testiranje GCRF modela i njegovih proširenja na setovima podataka iz različitih oblasti. Uobičajena procedura za rješavanje problema strukturne regresije modelima koji su bazirani na GCRF-u sastoji se od sledećih koraka:

1. priprema podataka za treniranje i testiranje modela
2. odabir i treniranje nestrukturnog prediktora
3. treniranje modela
4. testiranje modela
5. preuzimanje predviđenih vrijednosti
6. računanje preciznosti

Nakon što u koraku 1 dobije podatke, aplikacija GCRF GUI TOOL može da završi sve preostale korake. Softver sadrži predefinisane setove podatke, ali i korisnici mogu da dodaju svoje setove. Neophodno je da korisnik pripremi podatke u obliku tekstualnih fajlova (u formatu koji softver zahtjeva) i proslijedi ih aplikaciji. Podaci moraju da sadrže atribute, očekivane izlaze i veze među čvorovima. Nakon što proslijedi podatke, korisnik treba da izabere da li će nad svojim podacima primijeniti standardni GCRF, DirGCRF ili neko od proširenja GCRF-a koja su opisana u odjeljku 2.4. Aplikacija trenira izabrani model kako bi dobila vrijednosti parametara, dobijene vrijednosti čuva na lokalnom fajl sistemu i kasnije ih koristi za testiranje modela. Proces treniranja uključuje i treniranje nestrukturnih prediktora, i korisnik može da izabere između neuronske mreže i linerane regresije. Nakon testiranja, korisnik može da preuzme predviđene vrijednosti i da vidi performanse modela (preciznost, izraženu preko R^2 koeficijenta odlučnosti, kao i vrijeme izvršavanja).

Aplikacija je razvijena u Javi, uz upotrebu Swing GUI toolkit-a za implementaciju komponenti grafičkog korisničkog interfejsa. S obzirom da su postojeća proširenja GCRF-a implemetirana su u MatLab-u, bilo je neophodno povezati Java aplikaciju sa MatLab-om, omogućiti automatsko prosleđivanje argumenata i preuzimanja rezultata bez intervencije korisnika. Slika 13 predstavlja arhitekturu sistema.



Slika 13. Arhitektura softverskog alata GCRF GUI TOOL

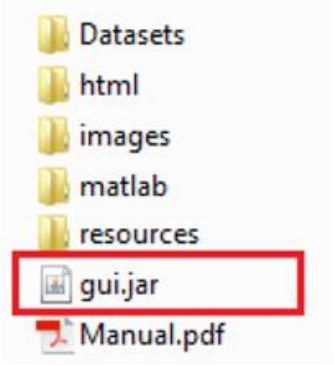
4.2. Funkcionalni opis

4.2.1. Instalacija

Kako bi mogao da koristi GCRF GUI TOOL, korisnik mora preuzeti zip fajl sa Interneta[6] i raspakovati ga na željenoj lokaciji. Struktura raspakovanog foldera prikazana je na slici 14. Izvršni fajl (gui.jar) i korisničko uputstvo (Manual.pdf) nalaze se direktno u glavnom folderu. Softver zahtjeva da na računaru budu instalirani Java 8 i Matlab (ukoliko korisnik želi da koristi metode koje su implementirane u Matlab-u). Ostali folderi sadrže:

- Datasets - primjeri setova podataka koji se mogu koristi za treniranje i testiranje modela (primjeri su dati preko .txt fajlova, u formatu koji softver zahtjeva)
- html - fajlovi za Help
- images - slike za ikonice
- matlab - izvorni kod za sve metode koje su implemetirane u Matlab-u
- resources – neophodne biblioteke (.jar fajlovi)

[6] https://drive.google.com/file/d/0B_vOEFyds9xYZXNlaHE0Zk9MYjA/view



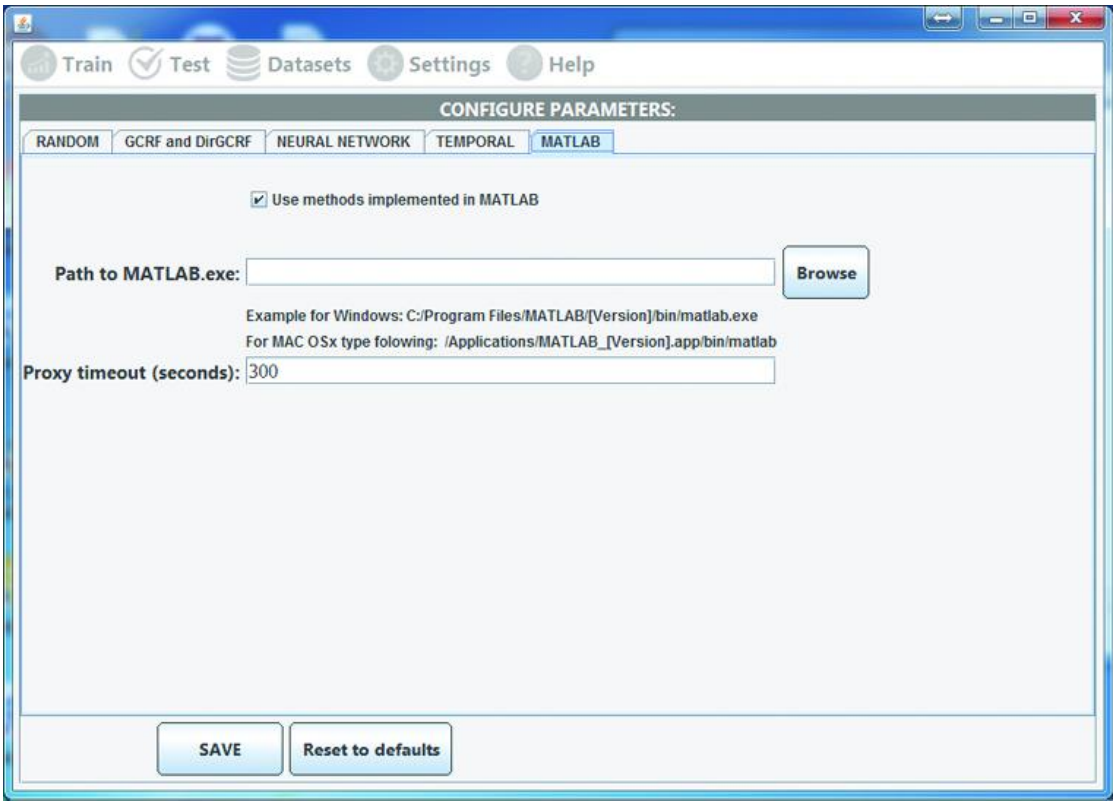
Slika 14. Struktura GCRF GUI TOOL foldera

4.2.2. Konfiguracija

Kada se softver prvi put pokrene, pojaviće se prozor za konfiguraciju (*Configuration*) i glavni meni će biti onemogućen (slika 15). Za većinu parametara definisane su podrazumijevane vrijednosti, ali one mogu biti promijenjene. Prozor za konfiguraciju se sastoji od 5 tabova:

1. Random – podrazumijevane vrijednosti za α i β parametre koji se koriste za računanje niza izlaznih vrijednosti (y) prilikom generisanja sintetičkih setova podataka
2. GCRF i DirGCRF - podrazumijevane vrijednosti za početne α i β parametre, stopu učenja (engl. learning rate) i maksimalan broj iteracija za algoritam penjanja u pravcu gradijenata
3. Neural Network – podrazumijevani broj skrivenih neurona i maksimalan broj iteracija za treniranje neuronske mreže
4. Temporal - podrazumijevani broj iteracija i parametri za m-GCRF (podrazumijevane vrijednosti za α i β parametre regularizacije) i RLSR (podrazumijevani λ set, broj iteracija za neuronsku mrežu i maksimalan broj iteracija za strukturno učenje - SSE i za backtracking linijsku pretragu u strukturnom učenju - SSE LS)
5. Matlab – putanja do matlab.exe fajla i vrijednost za Matlab proxy timeout. Ukoliko korisnik ne želi da koristi Matlab, ili nema Matlab instaliran na svom kompjuteru, može da opozove opciju „Use methods implemented in MATLAB“. U tom slučaju softver će prikazivati samo metode koje su implementirane u Javi.

Nakon klika na „Save“ dugme prozor za konfiguraciju će nestati i glavni meni će biti omogućen. Ukoliko korisnik želi naknadno da promijeni podrazumijevane vrijednosti parametara može ponovo pristupiti prozoru za konfiguraciju preko stavke menija „Settings/Configuration“.



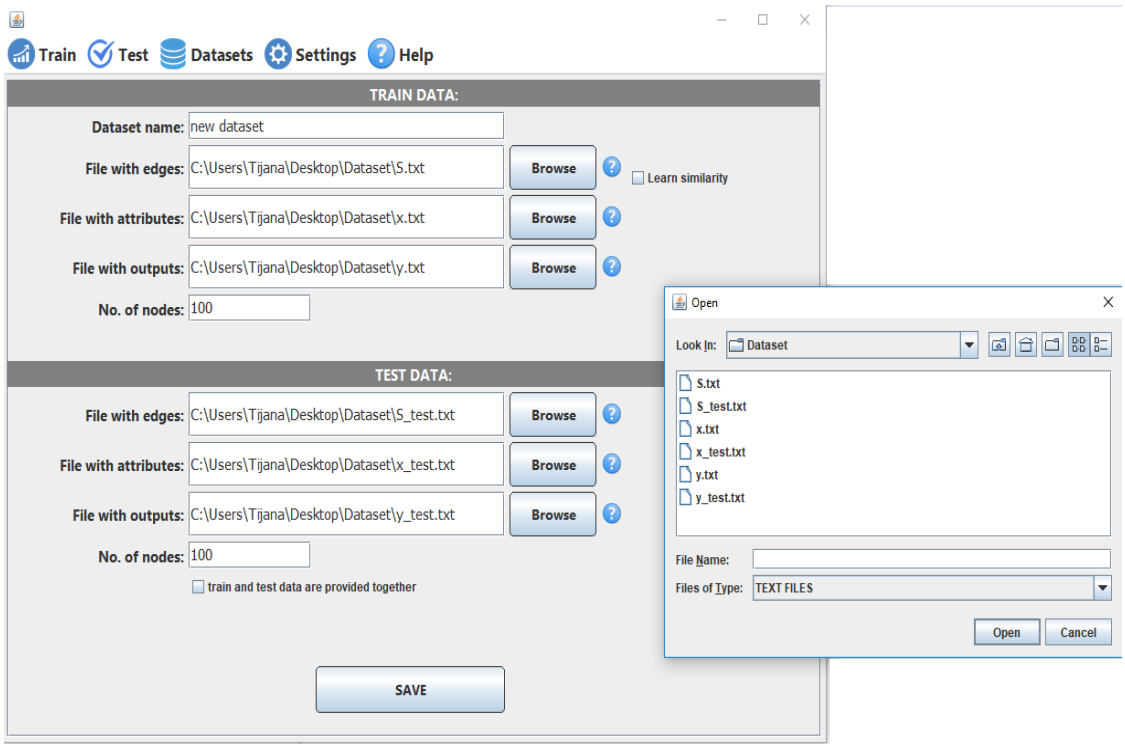
Slika 15. Prozor za konfiguraciju

4.2.3. Setovi podataka

Podaci za metode bazirane na GCRF-u se modeluju preko grafova, u kojima se objekti predstavljaju kao čvorovi, dok se odnosi među objektima modeluju preko veza između čvorova. Matrica sličnosti koja kvantifikuje ove veze označava se sa S. Svaki čvor karakteriše se preko jednog ili više atributa (x) i ima jednu izlaznu promjenljivu (y). Softver sadrži 7 predefinisanih setova podataka, ali i korisnik može da doda svoje setove podataka preko opcije glavnog menija „Datasets/Add dataset“. Za svaki set podataka korisnik treba da obezbijedi tekstualne fajlove sa vezama, atributima i očekivanim izlazima (slika 16). Formati ovih fajlova detaljnije su objašnjeni u tabeli 4, dok primjer na slici 17 predstavlja fajlove za trening, za set podatka u kom se nalaze informacije o 25 čvorova, za svaki čvor ima 6 atributa (x), jedna izlazna promjenjiva

(y), graf je usmjeren i težine veza su od 0 do 1 (ima ukupno 528 veza, na slici se vidi samo prvih 25). Nakon što korisnik doda novi set podataka on će biti sačuvan u „Datasets“ folderu koji se nalazi unutar glavnog „GCRF_GUI“ foldera. Za sve setove podataka fajlovi će imati sledeća imena:

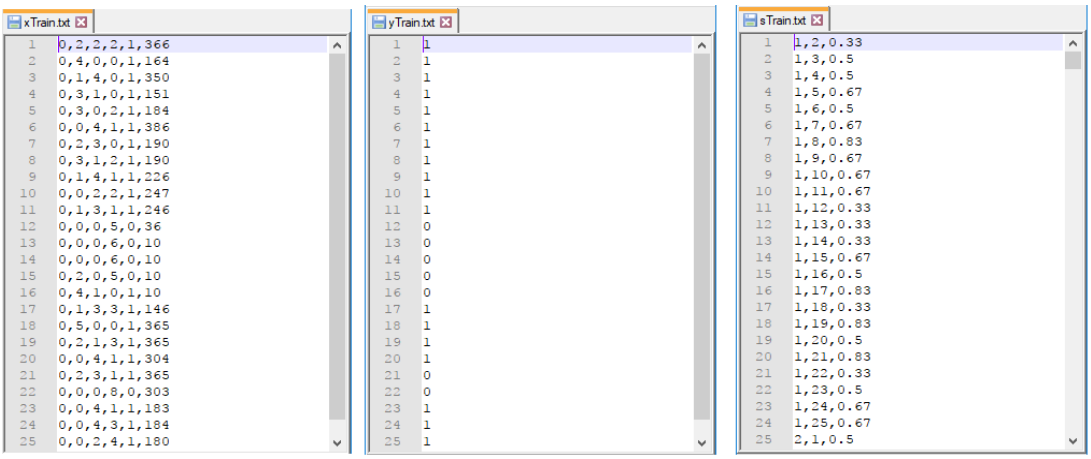
- x.txt - atributi
- y.txt - očekivani izlazi
- s.txt – graf koji predstavlja veze između objekata



Slika 16. Dodavanje novog seta podataka

Tabela 4. Formati fajlova za setove podataka

Fajl	Opis	Format
Veze (s.txt)	Veze među čvorovima grafa (objektima)	- Format: od čvora, do čvora, težina (ukoliko je u pitanju neusmjereni graf potrebno je navesti oba smjera). - Svaka veza treba da bude u novom redu. - Čvorovi su predstavljeni rednim brojevima (od 1 do broja čvorova).
Atributi (x.txt)	Vrijednost svakog atributa za svaki čvor	- Atributi za svaki čvor treba da budu u posebnom redu. - Atributi treba da budu odvojeni zarezima. - Svi atributi moraju biti brojevi. - Redosled treba da bude u skladu sa rednim brojevima čvorova koji se koriste u fajlu sa vezama. - Ukoliko ima više tačaka uzorkovanja svi atributi za jedan čvor treba da budu u istom redu, odvojeni zarezima.
Očekivani izlazi (y.txt)	Vrijednost izlazne promjenjive za svaki čvor	- Izlaz za svaki čvor treba da bude u posebnom redu. - Vrijednost izlazne promjenjive mora biti broj. - Redosled treba da bude u skladu sa rednim brojevima čvorova koji se koriste u fajlu sa vezama. - Ukoliko ima više tačaka uzorkovanja svi izlazi za jedan čvor treba da budu u istom redu, odvojeni zarezima.

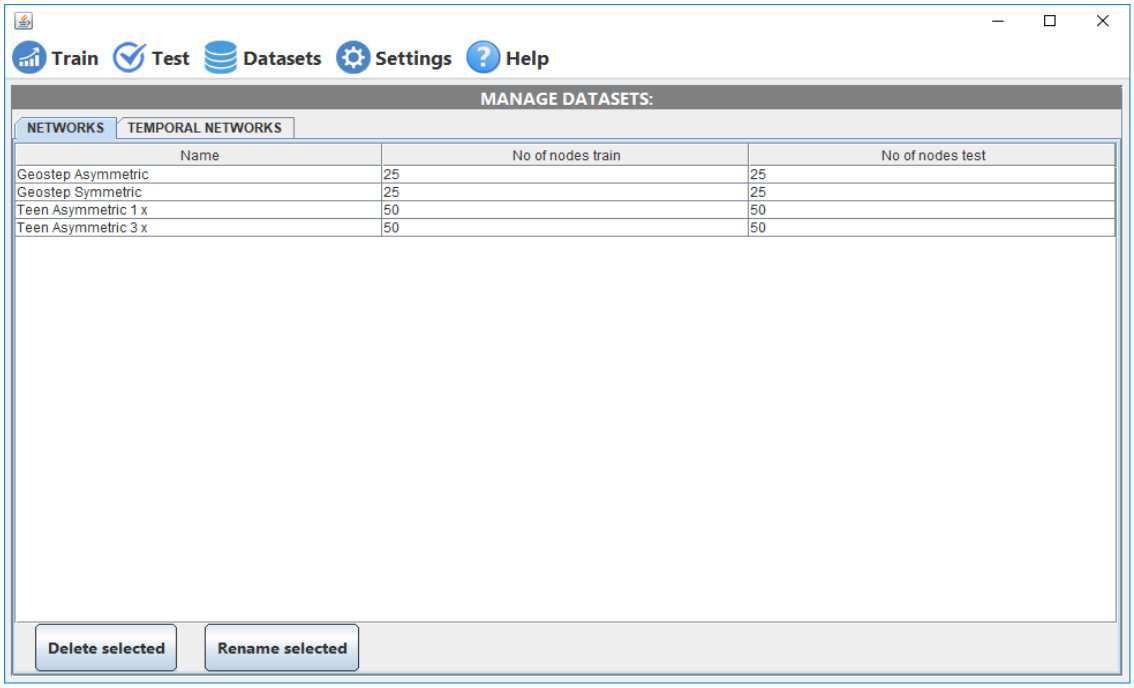


Slika 17. Primjer fajlova

Softver može da koristi dvije vrste setova podataka:

1. Setovi u kojima su podaci za treniranje i testiranje odvojeni – nad ovim setovima podataka mogu se primijeniti modeli koji se nalaze u okviru stavke menija „Train on networks“.
2. Setovi u kojima se podaci za treniranje i testiranje nalaze zajedno - nad ovim setovima podataka mogu se primijeniti modeli koji se nalaze u okviru stavke menija „Train on temporal networks“. Prije primjenjivanja određenog modela od korisnika će se tražiti da definiše kako podaci treba da budu podijeljeni (koliko tačaka uzorkovanja se koristi za treniranje, koliko za testiranje, a koliko za validaciju). Za ove setove podataka fajl sa vezama nije obavezan, jer korisnik može da odabere da želi da metod „nauči“ sličnost među objektima (selektovanjem „Learn similarity“ opcije).

Svi setovi podataka mogu se vidjeti klikom na „Manage datasets“ opciju (koja se nalazi pod stavkom menija „Datasets“) gdje se setovi mogu obrisati ili preimenovati (slika 18).



Slika 18. Upravljanje setovima podataka

Kako bi se detaljnije objasnile funkcionalnosti softvera predstavljene su dvije studije slučaja:

- **Studija slučaja 1:** U prvoj studiji slučaja koristi se Geostep set podatka koji sadrži podatke o 50 igara lova na blago i detaljno je opisan u odjeljku 3.4.2. Dvije verzije ovog seta podataka se nalaze među predefinisanim setovima: „Geostep Asymmetric“ (sa asimetričnom matricom sličnosti) i „Geostep Symmetric“ (sa simetričnom matricom sličnosti). Cilj je da se predvidi da li se igra može koristi u turističke svrhe i da se koriste izvorni podaci (asimetrična sličnost).
- **Studija slučaja 2:** U drugoj studiji slučaja koristi se Energy[7] set podataka koji sadrži informacije o proizvodnji solarne energije za države iz Oklahoma Mesonet mreže. Originalni set podataka sadrži 15 atributa za 98 lokacija koji su izmjereni u 1600 vremenskih tačaka. Kako bi se kreirao manji set podataka koji će biti uključen u predefinisane setove za GCRF GUI TOOL, iz ovih podataka je metodom slučajnog izbora izdvojeno 10 lokacija, za svaku lokaciju preuzeta je

[7] https://www.kaggle.com/c/ams-2014-solar-energy-prediction-contest

vrijednost samo jednog atributa za svih 1600 vremenskih tačaka. Fajl sa vezama nije dat, što znači da metod koji se koristi mora biti u stanju da nauči sličnost između objekata. Cilj je da se predvidi dnevni prihod od solarne energije na ovim lokacijama.

4.2.4. Treniranje i testiranje modela

Treniranje i testiranje nad mrežama

Cilj „Train on networks“ funkcije je da se istreniraju parametri za izabrani metod. Ova funkcija uključuje i treniranje parametara za nestrukturne prediktore (neuronske mreže ili linearnu regresiju). Vrijednosti parametara za izabrani metod i nestrukturni prediktor, kao i vrijednosti koje je nestrukturni prediktor predvidio, čuvaju se na lokalnom fajl sistemu i kasnije se koriste za testiranje metoda. Opcija „Train on networks“ koristi se za treniranje sledećih metoda: standardni GCRF, Directed GCRF (DirGCRF) i Unimodal GCRF (UmGCRF). UmGCRF je implemetiran u Matlab-u i biće vidljiv samo ukoliko je opcija „Use methods implemented in MATLAB“ odabrana prilikom konfiguracije softvera.

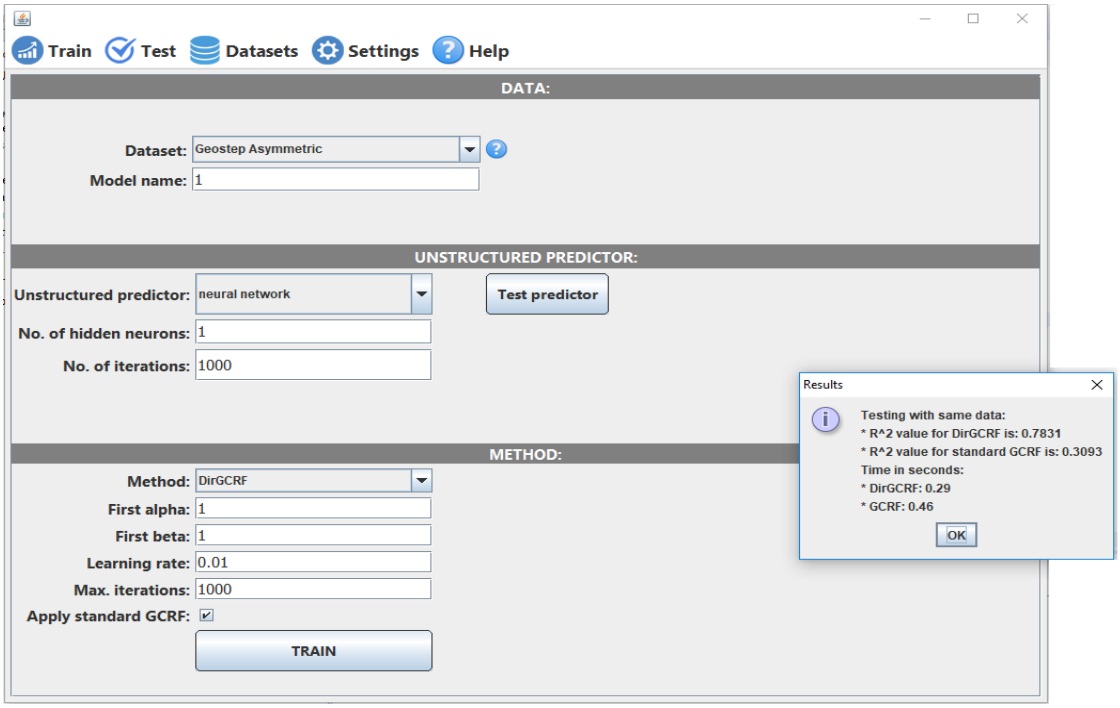
U prvom dijelu prozora od korisnika se traži da izabere set podataka iz padajuće liste i da unese ime za novi model. Drugi dio prozora sadrži informacije za nestrukturni prediktor i mogu se izabrati neuronska mreža ili linerana regresija. Ukoliko izabere neuronsku mrežu, korisnik mora unijeti broj skrivenih neurona i broj iteracija, a podaci će automatski biti normalizovani (ako u izvornom obliku nijesu normalizovani). Ukoliko korisnik izabere lineranu regresiju, nad podacima će se primijeniti standardna ili višestruka linearna regresija, u zavisnosti od broja atributa u razmatranom setu podataka. U trećem dijelu prozora nalazi se padajuća lista za izbor željnog metoda. Nakon što se odabere neki metod, ispod padajuće liste će se pojaviti polja sa neophodnim parametrima. Klikom na dugme „Train“ započinje proces treniranja. Nakon što se završi treniranje prikazaće se vrijeme izvršenja i R^2 koeficijent za podatke za trening.

Svi modeli koji koriste određeni metod će biti sačuvani u folderu za taj metod unutar „GCRF_GUI“ foldera. Za svaki od istreniranih modela kreira se novi folder koji nakon treninga sadrži sledeće podfoldere:

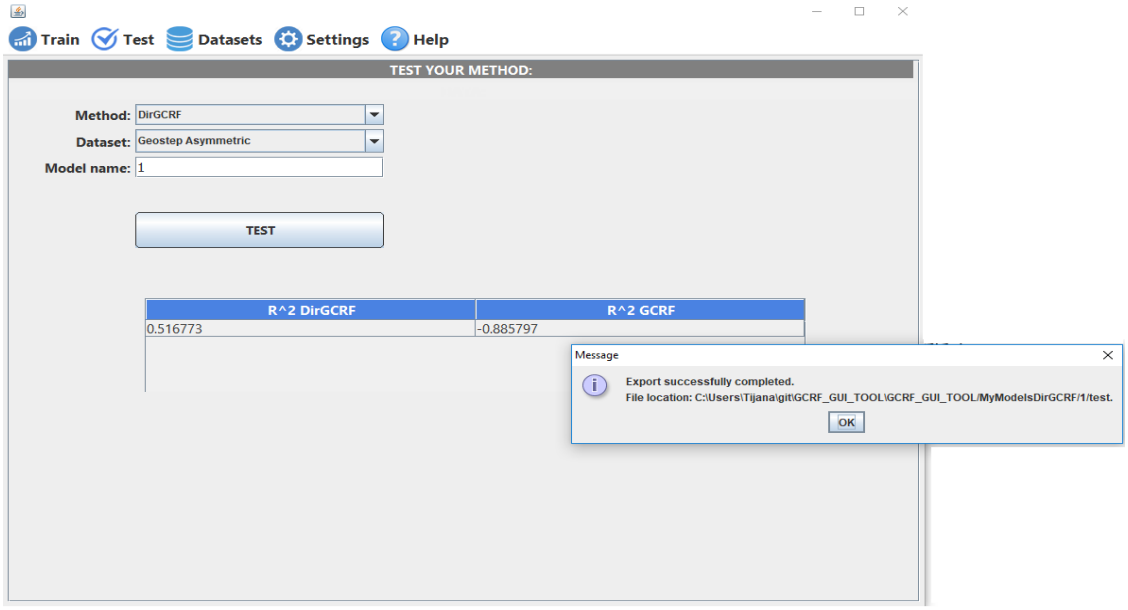
- data – folder koji sadrži fajlove sa podacima (tekstualne fajlove s, x i y za treniranje i testiranje) i fajl sa vrijednostima koje je predvidio nestrukturni prediktor (r.txt fajl)
- nn, lr ili mlr – folder koji sadrži fajlove sa parametrima za nestrukturni prediktor
- parameters – folder koji sadrži fajlove sa parametrima za metod

Nakon što se završi treniranje modela, on se može testirati preko „Test on networks“ opcije koja se nalazi u stavci menija „Test“. Da bi se model testirao neophodno je izabrati metod i set podataka za testiranje i unijeti ime modela. Nakon što se završi testiranje, prikazaće se vrijednost R^2 koeficijenta i predviđene vrijednosti će biti eksportovane u fajl (imenovan results+[ime metoda].txt) koje će biti sačuvan unutar foldera za taj model, u podfolderu „test“.

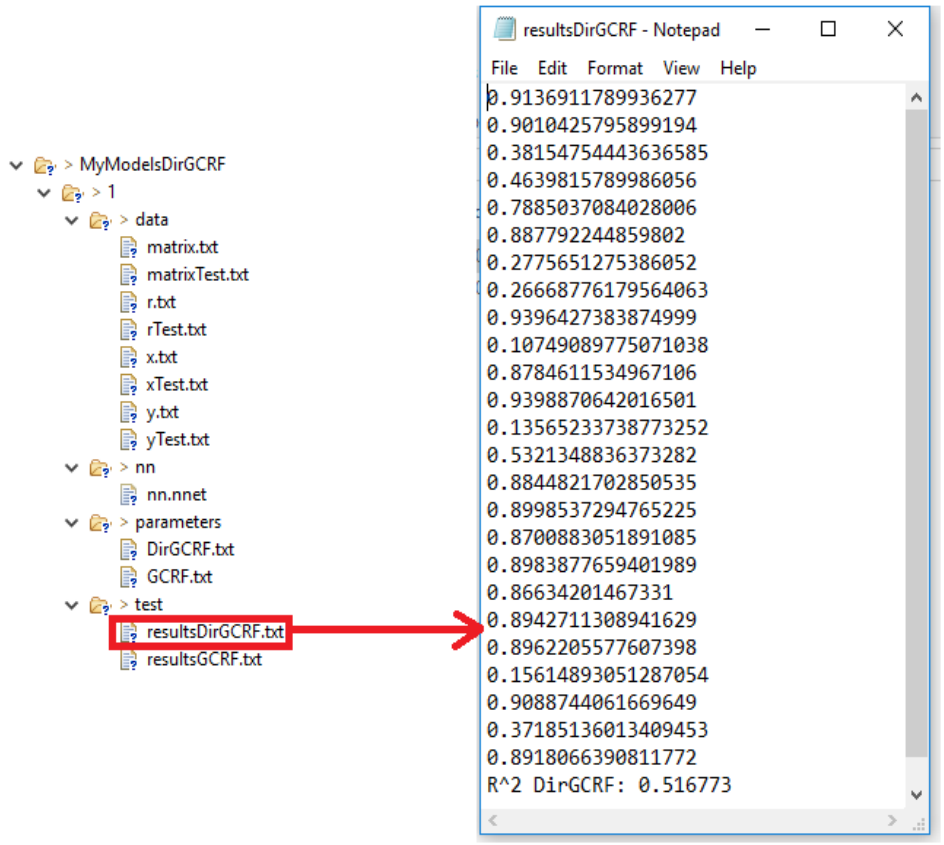
Sudija slučaja 1: U ovoj studiji slučaja potrebno je iskoristi „Train on networks“ funkcionalnost kako bi se izvršilo predviđanje za set podataka „Geostep Asymmetric“. S obzirom da ovaj set podataka ima asimetričnu matricu sličnosti, jedini metod koji se može primijeniti je DirGCRF. Neuronska mreža se koristi kao nestrukturni prediktor. Ukoliko se selektuje opcija „Apply standard GCRF“, asimetrična matrica će automatski biti konvertovana u simetričnu, kako bi se mogao primijeniti standardni GCRF i kako bi se dobila njegova preciznost za isti set podataka (u cilju poređenja). Primjer podešavanja i rezultata treniranja može se vidjeti na slici 19. Ime modela je „1“ i to ime se više ne može koristiti za model koji koristi DirGCRF metod. Testiranje se vrši preko opcije „Test on networks“ (slika 20). Struktura foldera za model „1“ prikazana je na slici 21.



Slika 19. Primjer „Train on networks“ prozora sa podešavanjima i rezultatima za Studiju slučaja 1



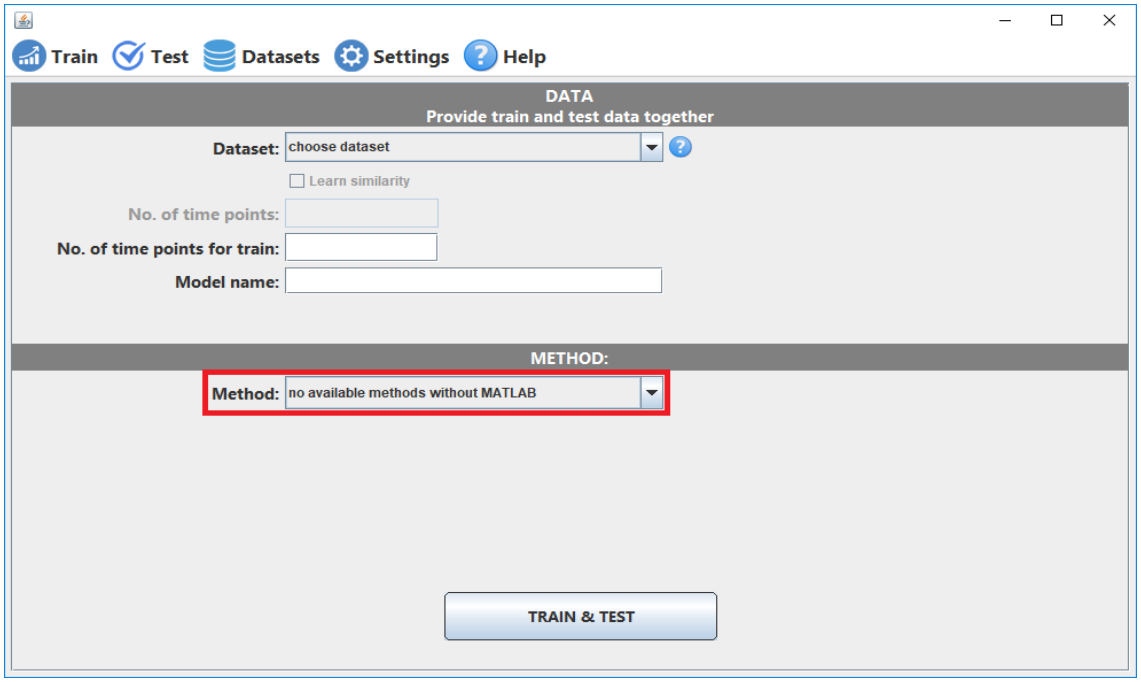
Slika 20. Primjer „Test on networks“ prozora sa podešavanjima i rezultatima za Studiju slučaja 1



Slika 21. Folder za model „1“ i fajl sa rezultatima

Treniranje i testiranje nad vremenskim mrežama

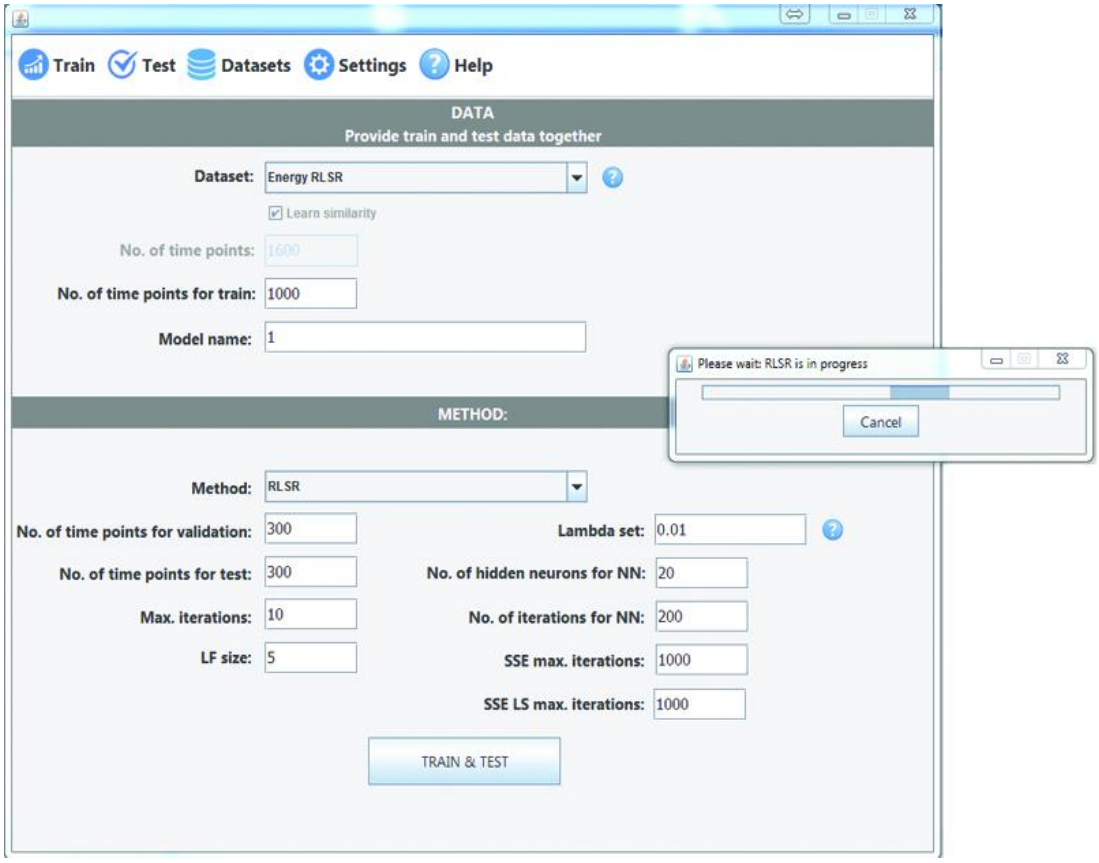
Funkcija „Train on temporal networks“ se koristi za treniranje i testiranje metoda koje rade sa vremenskim grafovima. Glavna razlika u odnosu na funkciju koja je prethodno predstavljena je to što će u ovom slučaju proces testiranja automatski da se izvrši nakon procesa treniranja. Opcija „Train on temporal networks“ koristi se za treniranje sledećih metoda: Representation Learning based Structured Regression (RLSR), Uncertainty propagation GCRF (up-GCRF) i Marginalized Gaussian Conditional Random Fields (m-GCRF). Svi metodi su implementirani u Matlab-u i biće vidljivi samo ukoliko je opcija „Use methods implemented in MATLAB“ odabrana prilikom konfiguracije softvera.To znači da korisnik koji nema Matlab u ovom prozoru neće imati nijedan dostupan metod (slika 22).



Slika 22. Izgled „Train on temporal networks“ prozora ukoliko opcija „Use methods implemented in MATLAB“ nije odabrana prilikom konfiguracije softvera

Podaci za treniranje i testiranje u setovima za ove modele dati su zajedno. Korisnik bira kako želi da podijeli set podataka, tj. koliko vremenskih tačka želi da koristi za treniranje, validaciju i testiranje. Nakon klika na dugme „Train & Test“ započinje proces treniranja modela, a odmah nakon toga i testiranja sa odgovarajućim podacima. RLSR i up-GCRF metodi ne zahtijevaju podatke o sličnosti, jer imaju mogućnost „učenja“ sličnosti objekata. Nakon što se iz padajuće liste odabere neki metod ispod nje će se pojaviti polja sa neophodnim parametrima. Struktura foldera za novi model je ista kao što je opisano u prethodnom primjeru.

Sudija slučaja 2: U ovoj studiji slučaja potrebno je iskoristiti „Train on temporal networks“ funkcionalnost, kako bi se izvršilo predviđanje za set podataka „Energy“ upotrebom metoda RLSR. Za treniranje se koristi 1000 tačaka uzorkovanja, dok se 300 tačaka uzorkovanja koristi za validaciju i 300 za testiranje. Za sve ostale parametre koje softver traži koriste se podrazumijevane vrijednosti. S obzirom da ovaj set podataka ne sadrži podatke o sličnosti, opcija „Learn similarity“ je automatski selektovana. Nakon klika na „Train & Test“ dugme automatski se poziva Matlab koji završava sve proračune i nakon toga softver prikazuje rezultate. Primjer podešavanja za ovu studiju slučaja prikazan je na slici 23.

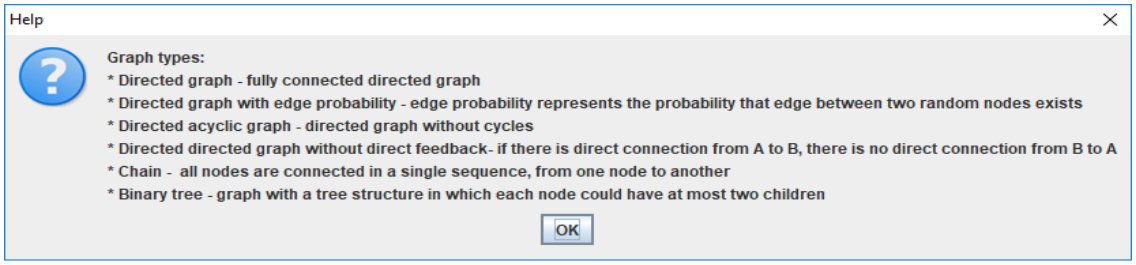


Slika 23. Primjer „Train on temporal networks“ prozora sa podešavanjima i rezultatima za Studiju slučaja 2

Treniranje i testiranje nad sintetičkim mrežama

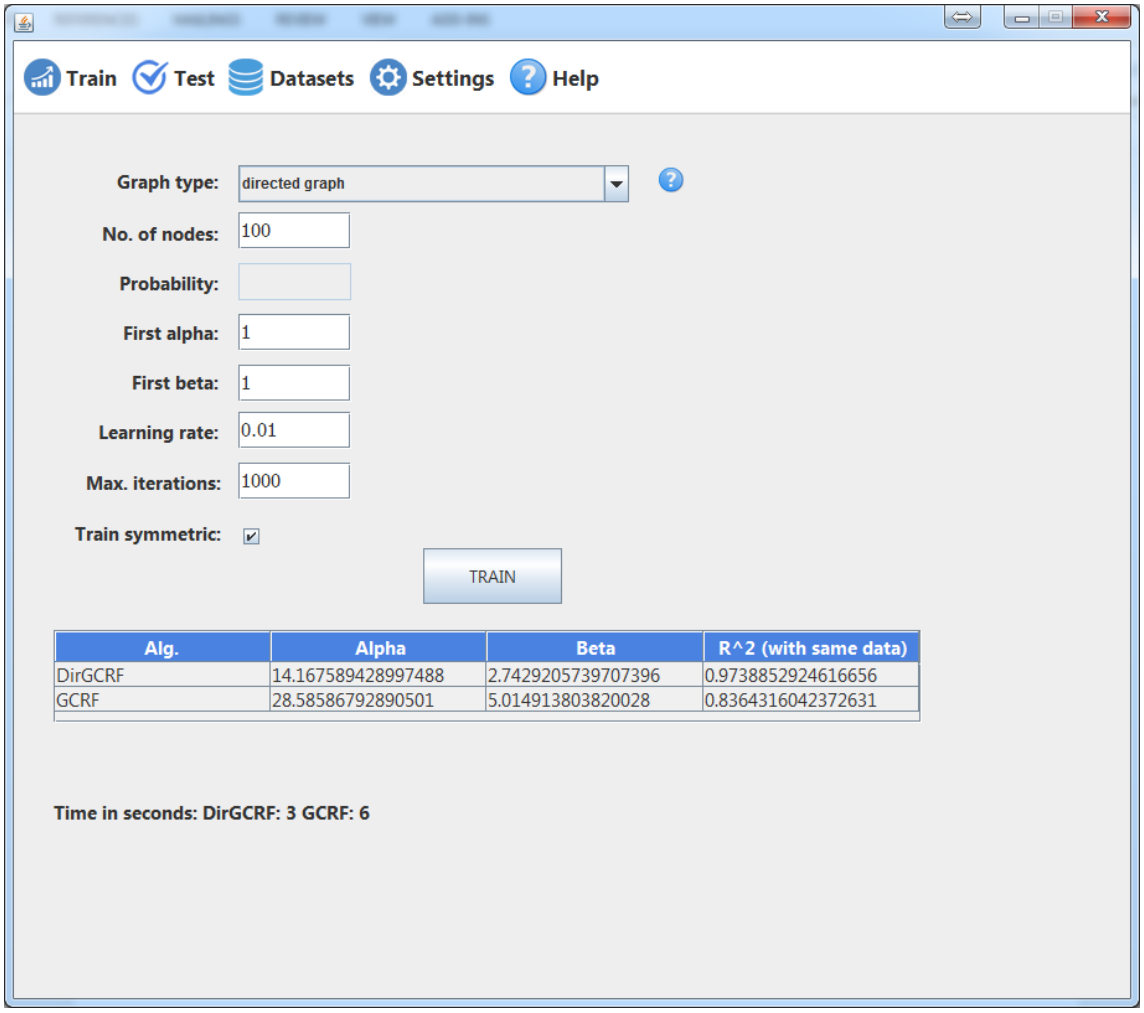
Cilj opcija „Train on random networks“ i „Test on random networks“ je da se DirGCRF metod testira na različitim vrstama sintetički generisanih grafova i da se njegova preciznost uporedi sa standardnim GCRF-om. Nakon što korisnik izabere vrstu usmjerenog grafa (slika 24 - ponuđeno je 6 opcija: potpuno povezan usmjereni graf, usmjereni graf sa p vjerovatnoćom veze, usmjereni graf bez povratnih veza, usmjereni aciklični graf, lanac i binarno stablo), softver generiše set podataka koji obuhvata:

- graf koji ima strukturu u skladu sa pravilima odabrane vrste grafa
- težine veza (matrica S)
- vrijednosti koje predviđa nestrukturni prediktor (R)
- vrijednosti za izlaznu varijablu (y) - generisani S i R se koriste za izračunavanje izlazne varijable za svaki čvor, a vrijednosti za α i β parametre definisani su prilikom konfiguracije softvera

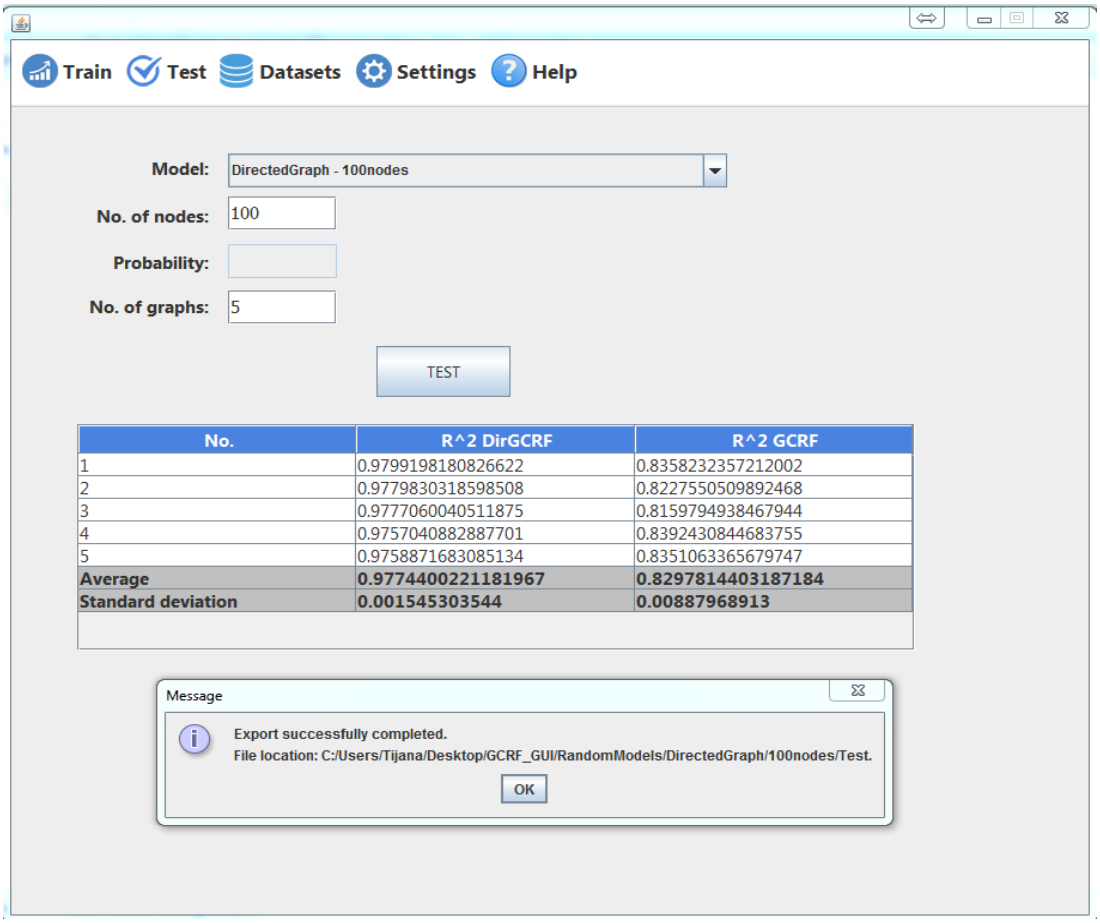


Slika 24. Objašnjenje iz GCRF GUI TOOL-a za vrste grafova koje se mogu generisati

Ukoliko se selektuje „Train symmetric“ opcija, matrica S će automatski biti konvertovana u simetričnu kako bi se mogao primijeniti standardni GCRF. Klikom na dugme „Train“ započinje proces treniranja. Nakon što se završi treniranje, prikazaće se vrijeme izvršenja i R^2 koeficijent za podatke za trening za oba metoda (ukoliko je odabrana opcija „Train symmetric“). Modeli koji su trenirani na sintetičkim mrežama testiraju se preko opcije „Test on random networks“ koja se nalazi u stavci menija „Test“. Grafovi za testiranje se takođe generišu. Korisnik treba da unese na koliko grafova želi da testira model, koliko svaki od grafova ima čvorova i koji model da se primijeni. Ime modela sastoji se od vrste grafa i broja čvorova u grafu za trening (na primjer, ukoliko je model treniran na usmjerenom grafu od 100 čvorova njegovo ime će biti „DirectedGraph - 100 nodes“). Nakon što se završi proces testiranja prikazaće se tabela koja sadrži R^2 vrijednosti za svaki graf, kao i prosječni R^2 i standardnu devijaciju. Primjeri za treniranje i testiranje metoda na sintetičkim podacima prikazani su na slikama 25 i 26.

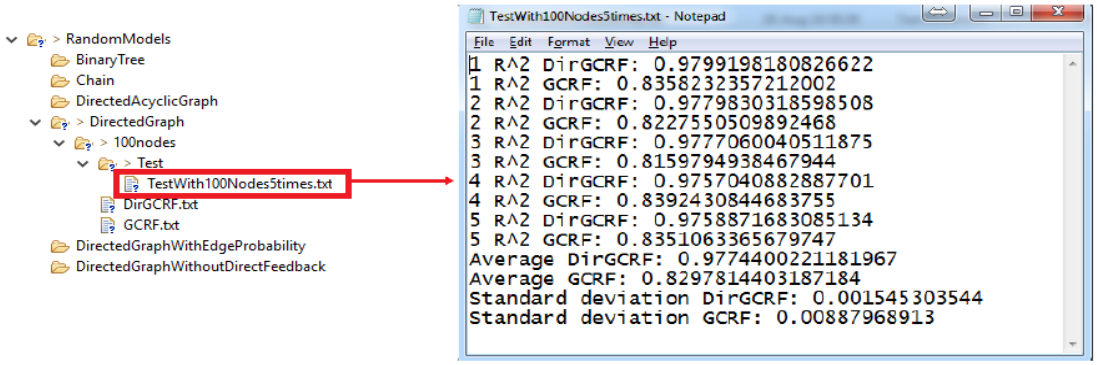


Slika 25. Primjer „Train on random networks“ prozora sa podešavanjima za treniranje modela na sintetički generisanom usmjerenom grafu od 100 čvorova



Slika 26. Primjer „Test on random networks“ prozora sa podešavanjima za testiranje prethodno istreniranih modela na 5 sintetički generisanih usmjerenih grafova od 100 čvorova

Ovi modeli čuvaju se u folderu „RandomModels“ koji se nalazi u glavnom „GCRF_GUI“ folderu. Modeli su grupisani po vrstama grafova i za jednu vrstu grafa može da se generiše samo jedan model sa određenim brojem čvorova (slika 27).

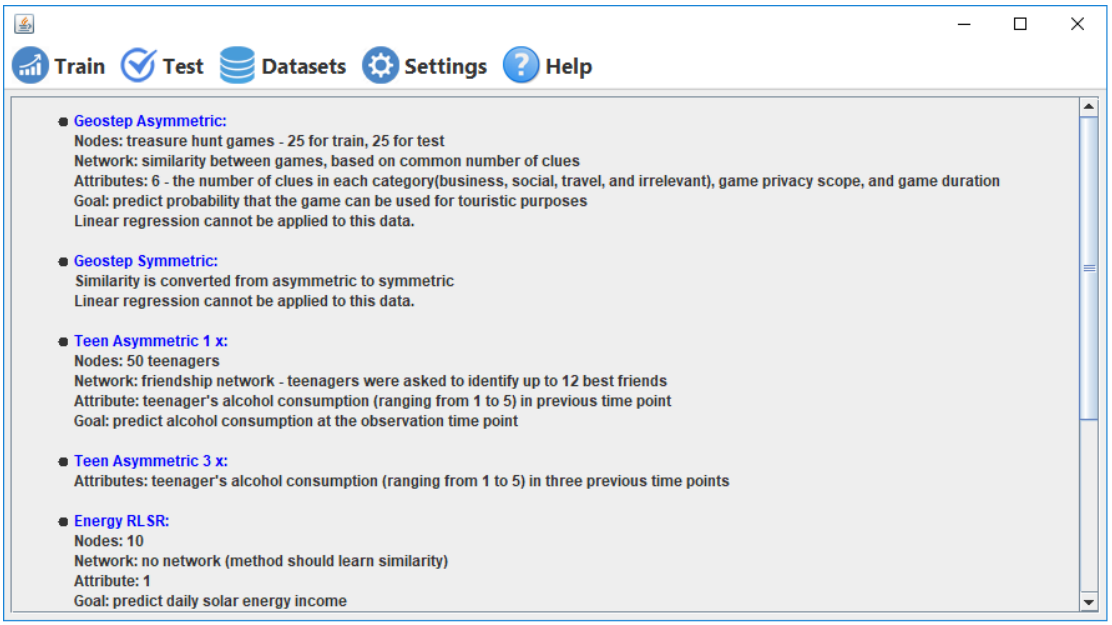


Slika 27. Primjer foldera za model za sintetički generisan usmjereni graf od 100 čvorova i fajl sa rezultatima

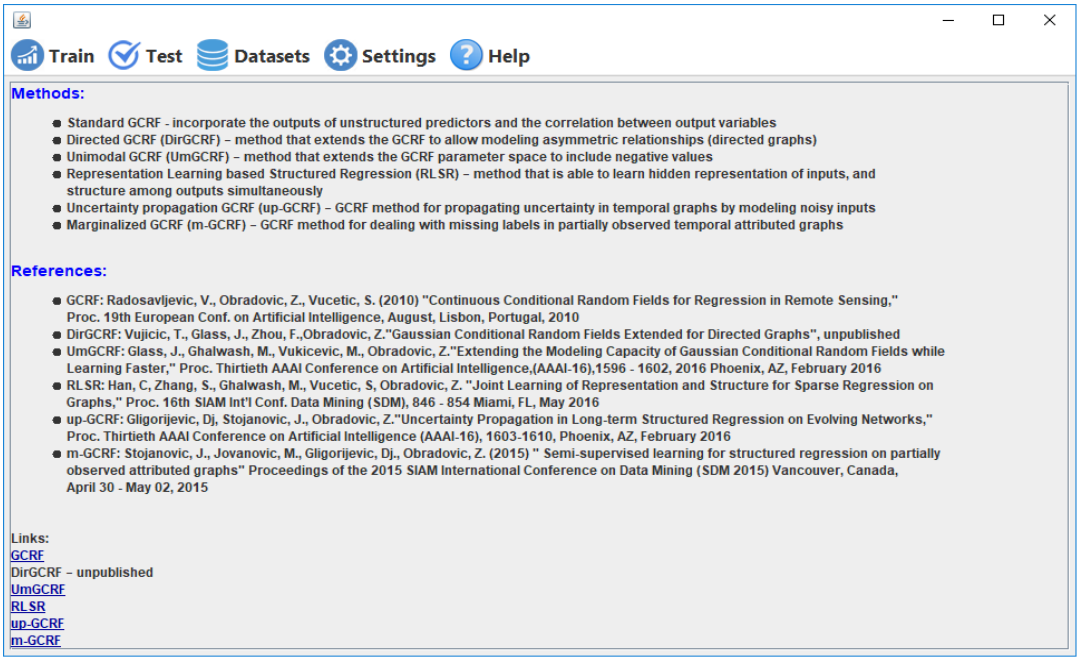
4.2.5. Dokumentacija

U stavci menija „Help“ korisnik može da pronađe opšte informacije o softveru, predefinisanim setovima podataka i metodama. „Help“ obuhvata sledeće opcije:

- About – definicije osnovnih pojmova i opšte informacije o softveru.
- Datasets (slika 28) – za sve setove podataka objašnjeno je šta čvor predstavlja, koje su veze između čvorova, koliko svaki čvor ima atributa i šta oni predstavljaju, šta se predviđa itd.
- Methods (slika 29) – definicije svakog od metoda i reference i link na Web stranicu koja sadrži rad u kom je objavljen dati metod (pdf fajl).



Slika 28. „Help/ Datasets“ stavka menija



Slika 29. „Help/Methods“ stavka menija

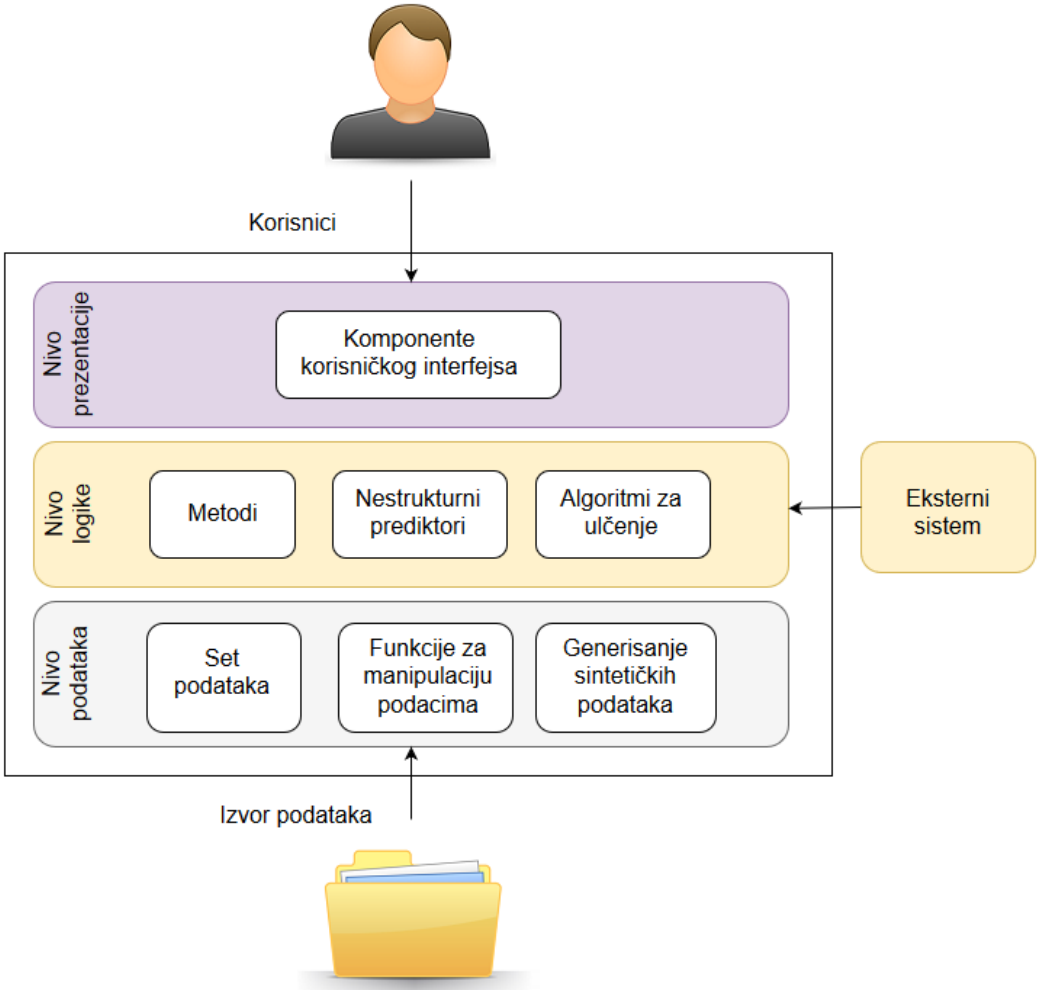
4.3. Dizajn i implementacija

4.3.1. Arhitektura rješenja

Glavni cilj softvera je da omogući korisnicima da upotrebom korisničkog interfejsa primijene različite GCRF metode na različitim setovima podataka koji se nalaze na lokalnom fajl sistemu, u obliku tekstualnih fajlova. Strukturu aplikacije čine tri nivoa (slika 30):

1. Nivo podataka – funkcije koje čitaju podatke sa fajl sistema, provjeravaju ih i po potrebi mijenjaju (npr. normalizacija) i od njih kreiraju set koji se sastoji od Java struktura podataka (objekata, nizova, matrica itd.), kao i funkcije za generisanje sintetičkih setova podataka
2. Nivo logike – algoritmi za učenje i funkcije koje omogućavaju treniranje i testiranje GCRF metoda i nestrukturnih prediktora (uključujući sve matematičke proračune i pomoćne metode, kao što su metoda za računanje R^2 koeficijenta, eksport rezultata itd.) kao i funkcije za komunikaciju sa eksternim sistemima (Matlab).

3. Nivo prezentacije – komponente korisničkog interfejsa koje upravljaju interakcijama korisnika (prikaz podataka korisniku, preuzimanje podataka od korisnika, informisanje korisnika o statusu aplikacije itd.)



Slika 30. Struktura aplikacije

Tabela 5 sadrži spisak funkcionalnosti koje je potrebno implemetirati, koje su grupisane po nivoima.

Tabela 5. Spisak funkcionalnosti

NIVO PODATAKA	
Setovi podatka	· Čitanje podataka iz tekstualnih fajlova · Kreiranje Java struktura
Manipulacija podacima	· Provjera podataka · Normalizacija podataka
Generisanje sintetičkih podataka	· Generisanje grafova · Generisanje nizova
NIVO LOGIKE	
Nestrukturni prediktori	· Neuronske mreže · Linerana regresija · Višestruka linearna regresija
Algoritmi za učenje	· algoritam penjanja u pravcu gradijenata (gradient ascent)
Metodi	· GCRF · DirGCRF · UmGCRF · m-GCRF · up-GCRF · RLSR
Pomoćne funkcionalnosti	· Računanje preciznosti (R^2 koeficijenta) · Upis podataka u tekstualni fajl · Povezivanje sa Matlab-om

NIVO PREZENTACIJE	
Komponente korisničkog interfejsa	· Meniji · Forme za unos podataka · Forme za prikaz podataka · Prozori za dijalog · Prozori za informisanje o napretku

GCRF GUI TOOL je implemetiran u Javi upotrebom razvojnog okruženja Eclipse[8]. U pitanju je projekat otvorenog koda i sav kod je javno dostupan na GitHub-u[9]. Korisnički interfejs je implementiran upotrebom klasa za GUI iz *Swing* paketa. Sledeće funkcionalnosti implemetirane su u Javi:

- Nestrukturni prediktori
- Algoritam za učenje
- Manipulacija podacima
- Generisanje sintetičkih setova podataka
- Dva metoda (GCRF i DirGCRF)
- Korisnički interfejs

Ostala četiri metoda (UmGCRF, m-GCRF, RLSR, up-GCRF) su implementirana u Matlab-u i pozivaju se iz Jave.

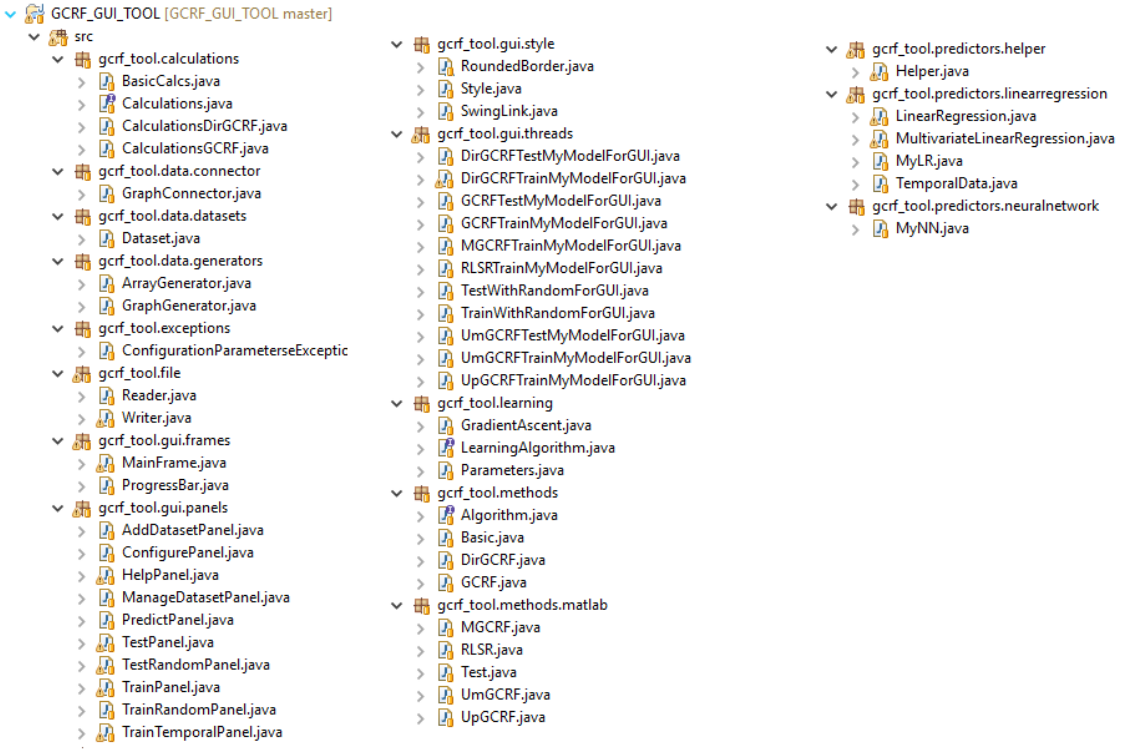
Korišćene su sledeće Java biblioteke:

- operacije sa matricama - OjAlgo (oj! Algorithms) [10]
- implementacija neuronskih mreža - Neuroph [11]
- pozivanje Matlab-a iz Jave - matlabcontrol [12]

Lista svih paketa i klasa prikazana je na slici 31, dok je u tabeli 6 data logička i fizička organizacija alata po paketima. Većina paketa (osim paketa *gcrf_tool.gui*) sadrže klase

[8] https://www.eclipse.org/
[9] https://github.com/vujicictijana/GCRF_GUI_TOOL
[10] http://ojalgo.org/
[11] http://neuroph.sourceforge.net/
[12] https://code.google.com/archive/p/matlabcontrol/

koje implementiraju funkcionalnosti za nivo podataka i nivo logike i zasnivaju se na osnovnim klasama koje su predstavljene u odjeljku 3.3 i koje obezbjeđuju opštu strukturu, logiku i komponente koje su zajedničke za sve modele bazirane na GCRF-u. U narednim odjeljcima su predstavljeni detalji implementacije.



Slika 31. Prikaz svih paketa i klasa u GCRF GUI TOOL projektu

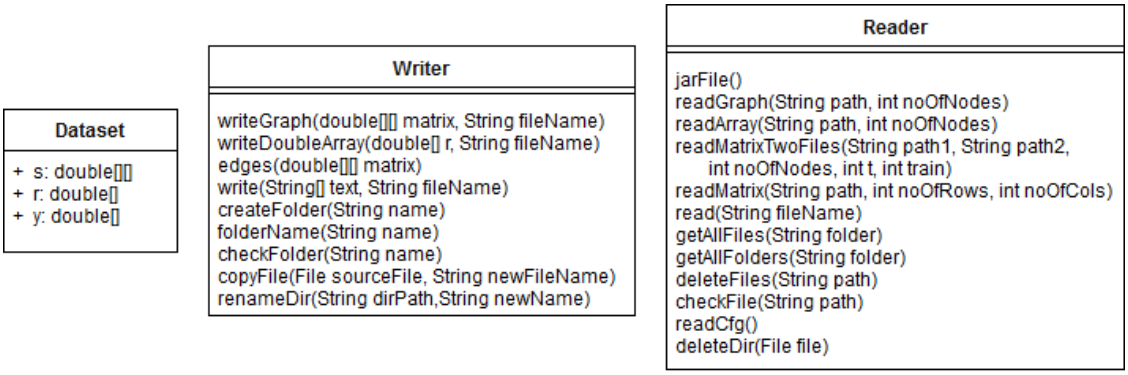
Tabela 6. Organizacija GCRF GUI TOOL projekta po paketima

Naziv paketa	Opis
gcrf_tool.calculations	Sadrži klase za matematičke proračune.
gcrf_tool.data	Sadrži klase za rad sa podacima.
gcrf_tool.exceptions	Sadrži klase koje predstavljaju izuzetke.
gcrf_tool.file	Sadrži klase za rad sa fajlovima.
gcrf_tool.gui	Sadrži klase za korisnički interfejs.
gcrf_tool.learning	Sadrži klase za algoritme za učenje.

gcrf_tool.methods	Sadrži klase za GCRF metode.
gcrf_tool.predictors	Sadrži klase za nestrukturne prediktore.

4.3.2. Implementacija setova podataka

Setovi podatka predstavljeni su klasom *Dataset* koja sadrži dvodimenzionalni niz *s* (koji predstavlja matricu povezanosti za graf sličnosti) i nizove *r* (koji predstavlja vrijednosti koje je predvidio nestrukturni prediktor) i *y* (koji predstavlja očekivane vrijednosti izlazne promjenjive). Osnovu podršku za rad sa podacima pružaju klase za upis i čitanje podatka iz fajla (klase *Writer* i *Reader* iz paketa *gcrf_tool.file*). Dijagram klasa za rad sa setovima podataka prikazan je na slici 32.



Slika 32. Dijagram klasa za rad sa setovima podataka

4.3.3. Generisanje sintetičkih podataka

GCRF GUI TOOL omogućava korisnicima da testiraju neke od metoda na sintetički generisanim setovima podataka. Zbog toga je bilo neophodno da se kreiraju klase za generisanje setova podataka na osnovu predefinisanih karakteristika. Kreirane su klase za generisanje nizova (*ArrayGenerator*) i grafova (*GraphGenerator*) i ove klase se nalaze u paketu *gcrf_tool.data.generators*.

Klasa *ArrayGenerator* sadrži metodu *generateArray* koja vraća niz od n slučajno generisanih decimalnih vrijednosti (double). Ova metoda se koristi za generisanje niza *r* (vrijednosti koje je predvidio nestrukturni prediktor). Niz R može da sadrži decimalne brojeve između 0 i neke zadate vrijednosti.

```
public static double[] generateArray
                              (int noOfElements, int maximum) {
      double[] r = new double[noOfElements];
      Random rand = new Random();
      for (int i = 0; i < r.length; i++) {
            r[i] = rand.nextInt(maximum) + Math.random();
      }
      return r;
}
```

Klasa *GraphGenerator* sadrži metode za generisanje matrice povezanosti za različite vrste grafova. Moguće je generisati 6 vrsta usmjerenih grafova (potpuno povezan usmjereni graf, usmjereni graf sa p vjerovatnoćom veze, usmjereni graf bez povratnih veza, usmjereni aciklični graf, lanac i binarno stablo). Težine veza u grafu (vrijednosti u matrici *S*) su u opsegu od 0 do 1. Primjer metode koja slučajno generiše matricu povezanosti za usmjereni graf sa *p* vjerovatnoćom veze:

```
public static double[][]
           generateDirectedGraphWithEdgeProbability
                      (int noOfNodes, double probability) {
      double p = 1 - probability;
      double[][] graph = new double[noOfNodes][noOfNodes];
      double tempP = 0;
      for (int i = 0; i < graph.length; i++) {
      for (int j = 0; j < graph.length; j++) {
                      if (i != j) {
                                  tempP = Math.random();
                                  if (tempP >= p) {
      graph[i][j] = Math.random();
                                  }
                      }
                }
      }
      return graph;
}
```

Svaki od generisanih grafova može se konvertovati u neusmjereni upotrebom metode *converteGraphToUndirected.* U novoj (simetričnoj) matrici, svaki par čvorova povezan je jednom neusmjerenom vezom čija je težina jednaka prosjeku težina iz odgovarajuće asimetrične matrice.

```
public static double[][] converteGraphToUndirected
                                        (double[][] matrix) {
      double[][] graphUndirected = new
                      double[matrix.length][matrix.length];
      double first = 0;
      double second = 0;
      for (int i = 0; i < matrix.length; i++) {
            for (int j = 0; j < matrix.length; j++) {
                  if (graphUndirected[i][j] == 0) {
                  if (i != j) {
                     first = matrix[i][j];
                     second = matrix[j][i];
                     graphUndirected[i][j] = (first + second) / 2;
                     graphUndirected[j][i] = (first + second) / 2;
                  }
                  }
            }
      }
      return graphUndirected;
}
```

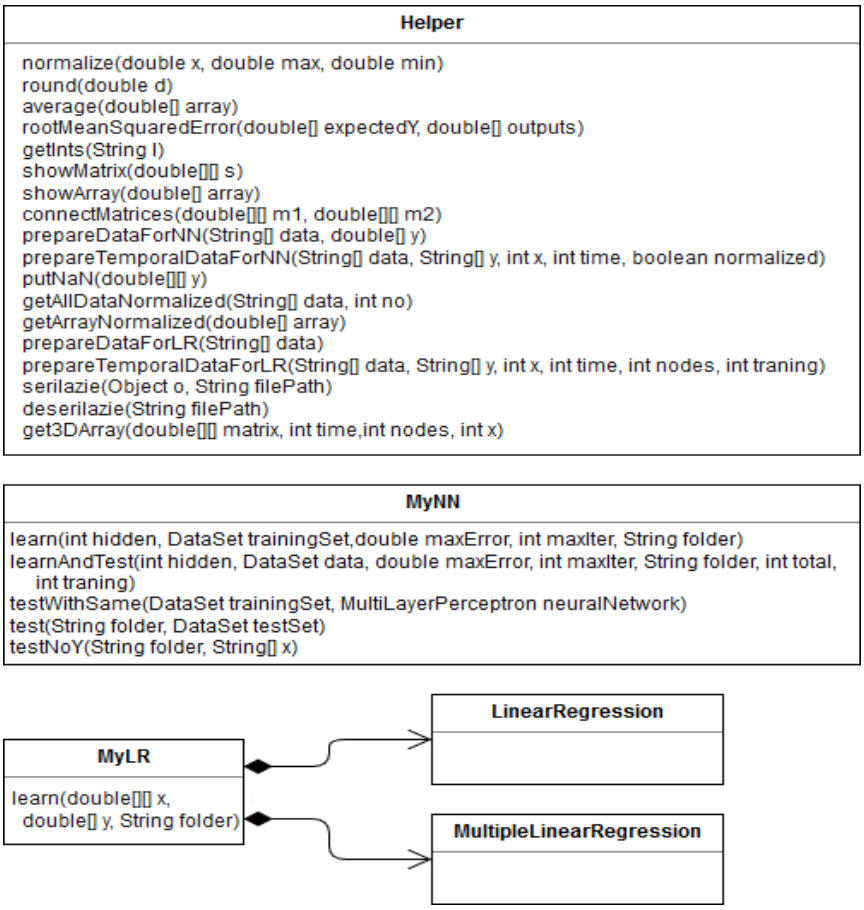
Ove klase koriste se za generisanje matrice *s* i niza *r*, ali da bi set podatka bio potpun neophodno je i da se definiše niz očekivanih izlaza (*y*). Ovaj niz se ne može slučajno generisati, već ga je potrebno izračunati na osnovu generisanih *s* i *r*, u skladu sa pravilima metode koja se testira, uz dodavanje slučajno generisanog odstupanja. Kako bi se izračunao *y* niz, neophodno je kreirati objekat klase koja sadrži pravila za izračunavanje za određeni metod, pozvati metodu *y* i kao ulazne argumente joj proslijediti vrijednosti α i β parametara i fiksan parametar sa kojim će se množiti

slučajno generisano odstupanje. Nakon generisanja niza *y*, moguće je kreirati novi set podatka.

```
double[][] s = GraphGenerator.generateDirectedGraph(200);
double[] r = ArrayGenerator.generateArray(200, 5);
CalculationsDirGCRF c = new CalculationsDirGCRF(s, r);
double[] y = c.y(1, 2, 0.05);
Dataset dataset = new Dataset(s, r, y);
```

4.3.4. Implementacija prediktora

U paketu *gcrf_tool.predictors* nalaze se klase koje implementiraju nestrukturne prediktore: neuronske mreže (paket *gcrf_tool.predictors.neuralnetwork*) i lineranu regresiju (paket *gcrf_tool.predictors. linearregression*). Pomoćne metode za nestrukturne prediktore nalaze se u klasi *Helper*. Dijagram klasa koje realizuju podršku za nestrukturne prediktore prikazan je na slici 33.



Slika 33. Dijagram klasa koje realizuju podršku za nestrukturne prediktore

Neuronske mreže implementirane su pomoću biblioteke Neuroph. Neuroph (Ševarac, 2008) je besplatni framework za neuronske mreže koji se koristi za kreiranje i upotrebu neuronskih mreža u Java programima. Neuroph obezbjeđuje dobro dizajniranu Java biblioteku otvorenog koda koja sadrži klase koje odgovaraju osnovnim konceptima neuronskih mreža. Klasa *MyNN* koristi Neuroph klase *MultiLayerPerceptron* (jedna od vrsta neuronskih mreža, nasleđuje klasu *NeuralNetwork*) i *BackPropagation* (jedan od dostupnih algoritama za učenje, nasleđuje klasu *LearningRule*). Istrenirana mreža čuva se u fajlu, pomoću metode *save* (ekstenzija fajla je *nnet*), a iz njega se čita pomoću metode *createFromFile* (obje metode su iz klase *NeuralNetwork)*. Vrijednosti koje je neuronska mreža predvidjela čuvaju se u tekstualnom fajlu sa imenom „r.txt“.

```
BackPropagation b = new BackPropagation();
b.setMaxError(maxError);
b.setMaxIterations(maxIter);

MultiLayerPerceptron neuralNetwork = new MultiLayerPerceptron(
                                TransferFunctionType.TANH,
                    trainingSet.getRowAt(0).getInput().length,
                    hidden, 1);
double[] outputs = new double[trainingSet.getRows().size()];
String[] rArray = new String[outputs.length];
int i = 0;
for (DataSetRow row : trainingSet.getRows()) {
neuralNetwork.setInput(row.getInput());
      neuralNetwork.calculate();
      outputs[i] = Helper.round(neuralNetwork.getOutput()[0]);
      rArray[i] = outputs[i] + ""; i++;
}
if (folder != null) {
      Writer.createFolder(folder + "/nn");
      neuralNetwork.save(folder + "/nn/nn.nnet");
      Writer.write(rArray, folder + "/data/r.txt");
}
```

Prije nego što se podaci proslijede neuronskoj mreži, neophodno je provjeriti da li su normalizovani. Ukoliko nijesu, neophodno je izvršiti normalizaciju podatka upotrebom metode *prepareDataForNN* iz klase *Helper*.

```
public static DataSet prepareDataForNN
                        (String[] data, double[] y) {
      int no = data[0].split(",").length;
      DataSet d = new DataSet(no, 1);
      double[][] x = getAllDataNormalized(data, no);
      double[] yNormalized = getArrayNormalized(y);
      if (x == null || yNormalized == null) {
            return null;
      }
      for (int i = 0; i < data.length; i++) {
                        d.addRow(new DataSetRow(x[i], new double[]
                                          { yNormalized[i] }));
      }
      return d;
}
```

Linerana regresija je implementirana pomoću klasa *LinearRegression* i *MultipleLinearRegression* koje su preuzete iz knjige „Introduction to Programming in Java: An Interdisciplinary Approach“ (Sedgewick & Wayne, 2017) [13]. Klasa MyLR sadrži statičku metodu *learn* koja u zavisnosti od broja atributa u setu podataka poziva jednostavnu ili višestruku linearnu regresiju.

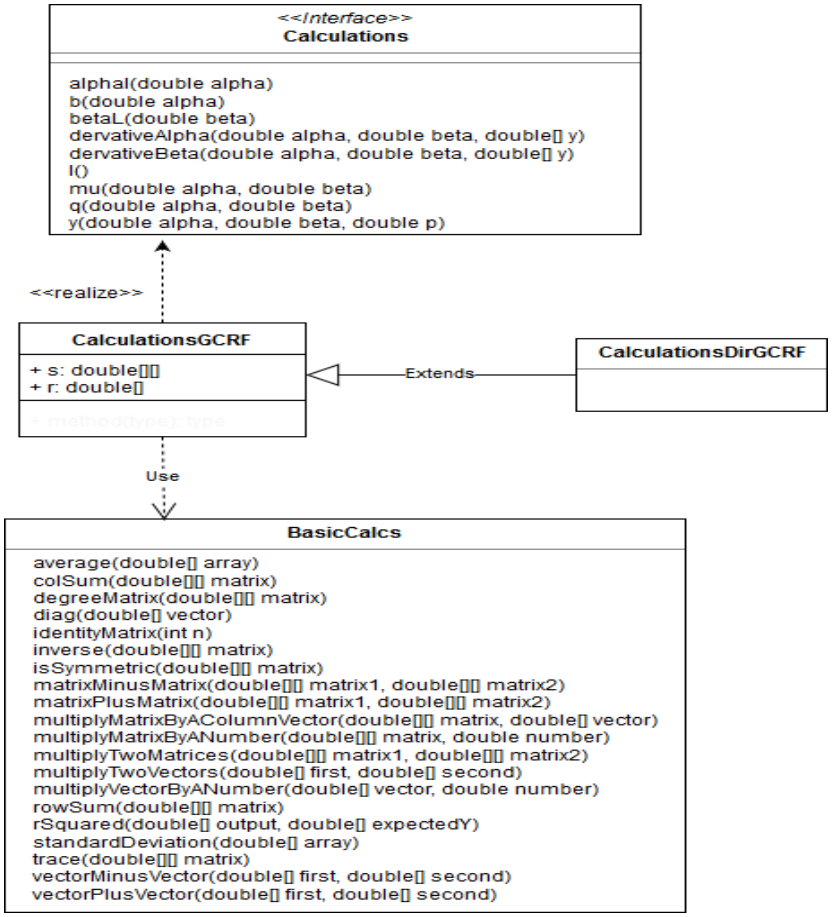
```
public static double learn
                  (double[][] x, double[] y, String folder) {
      if (x[0].length == 1) {
            double[] xOne = new double[x.length];
            for (int i = 0; i < xOne.length; i++) {
                  xOne[i] = x[i][0];
```

[13] https://introcs.cs.princeton.edu/java/home/

```
            }
            LinearRegression lr = new LinearRegression(xOne, y);
            return LinearRegression.test
                                    (y, xOne, folder, lr,false);
      } else {
            try {
                  MultipleLinearRegression m =
                  new MultipleLinearRegression(x, y);
                  return m.test(y, x, folder, false);
            } catch (Exception e) {
                  return -3000;
            }
      }
}
```

4.3.5. Implementacija GCRF metoda

Za implementaciju metoda najvažnije je da se implemetiraju pravila izračunavanja. Spisak metoda neophodnih za implementaciju bilo kog GCRF modela definisan je u interfejsu *Calculations*, dok klasa *BasicCalcs* sadrži pomoćne metode koje olakšavaju implementaciju formula (množenje vektora, množenje matrica, računanje inverzne matrice itd.). Dijagram klasa koje realizuju podršku za matematičke proračune prikazan je na slici 34.



Slika 34. Dijagram klasa koje realizuju podršku za matematičke proračune

Prilikom implementacije matematičkih proračuna bilo je veoma važno da se izabere Java biblioteka za operacije sa matricama. Razmatrane su tri biblioteke: Jama (Java Matrix Package) [14], OjAlgo (oj! Algorithms) [15] i UJMP (Universal Java Matrix Package) [16]. Prije odabira biblioteke izvršeno je testiranje performansi svake od biblioteka za dvije najzahtjevnije i najčešće korišćene operacije: računanje inverzne matrice i množenje matrica. Testiranje je izvršeno na potpuno povezanom grafu od 5000 čvorova i vrijeme izvršenja je dato u tabeli 7. Na osnovu ovih podatka izabrana je biblioteka OjALgo i ona je korišćena za implementaciju svih operacija sa matricama.

[14] http://math.nist.gov/javanumerics/jama/
[15] http://ojalgo.org/
[16] https://ujmp.org/

Tabela 7. Analiza performansi različitih Java biblioteka za operacije sa matricama

Biblioteka	Računanje inverzne matrice	Množenje matrica
JAMA	341,1 s	162,6 s
OjALgo	96,9 s	5,7 s
UJMP	208,4 s	9 s

Sledeći kod predstavlja primjer metoda iz klase *BasicCalcs* koje koriste klase i metode iz biblioteke OjAlgo. U pitanju su dvije metode: metoda *inverse* (koja računa inverznu matricu) i metoda *multiplyTwoMatrices* (koja množi dvije matrice).

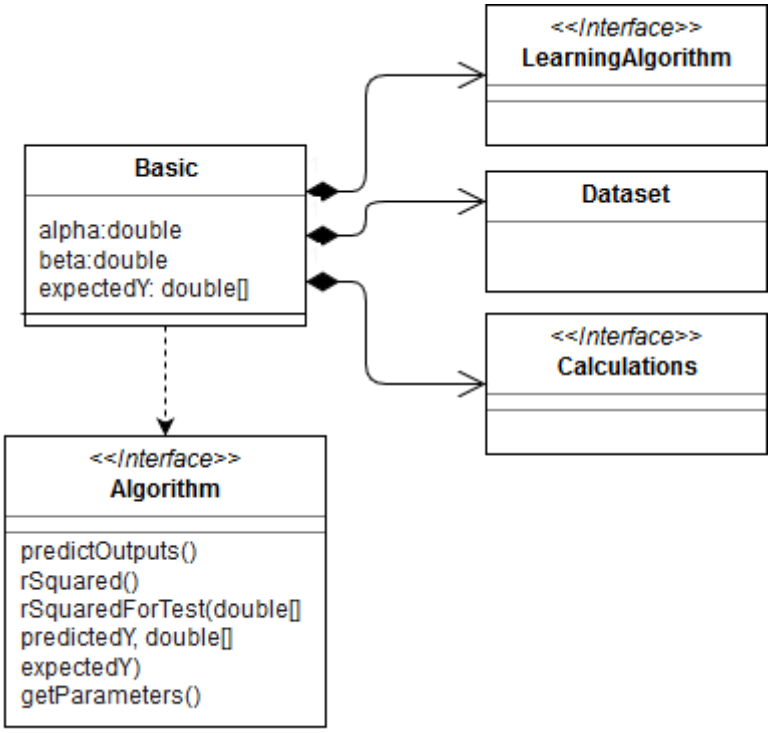
```
public static double[][] inverse(double[][] matrix) {
      BasicMatrix.Factory<PrimitiveMatrix> mtrxFactory =
                                          PrimitiveMatrix.FACTORY;
      PrimitiveMatrix mtrxA = mtrxFactory.rows(matrix);
      PrimitiveMatrix mtrxI = mtrxA.invert();
      return mtrxI.toRawCopy2D();
}
public static double[][] multiplyTwoMatrices
                        (double[][] matrix1, double[][] matrix2) {
      BasicMatrix.Factory<PrimitiveMatrix> mtrxFactory =
                                          PrimitiveMatrix.FACTORY;
      PrimitiveMatrix mtrxA = mtrxFactory.rows(matrix1);
      PrimitiveMatrix mtrxB = mtrxFactory.rows(matrix2);
      PrimitiveMatrix res = mtrxA.multiply(mtrxB);
      return res.toRawCopy2D();
}
```

Sva pravila izračunavanja i formule koje su ukratko opisane u poglavlju 2.3 specificirane su u interfejsu *Calculations* i implementirane u klasama *CalculationsGCRF i CalculationsDirGCRF*. Na primjer, metoda *q* u klasi

CalculationsGCRF izračunava matricu tačnosti na osnovu pravila za standardni GCRF algoritam.

```
 public double[][] q(double alpha, double beta) {
      // Q = 2*Alpha*I + 2*Beta*L
      double[][] alphaI = alphaI(alpha);
      double[][] betaL = betaL(beta);
      return BasicCalcs.matrixPlusMatrix(
                  BasicCalcs.multiplyMatrixByANumber(alphaI, 2),
                  BasicCalcs.multiplyMatrixByANumber(betaL, 2));
}
public double [][] alphaI(double alpha) {
      // Alpha * I (I - identity matrix)
      double[][] identity = BasicCalcs.identityMatrix(s.length);
      return BasicCalcs.multiplyMatrixByANumber
                                              (identity, alpha);
}
public double [][] betaL(double beta) {
      // Beta * L (L- Laplacian matrix)
      // L = degreeMatrix - adjacencyMatrix
      return BasicCalcs.multiplyMatrixByANumber(l(), beta);
}
```

Klase koje predstavljaju konkretan metod nalaze se u paketu *gcrf_tool.methods)*. Dijagram klasa koje realizuju podršku za GCRF metode prikazan je na slici 35. Ove klase koriste se da treniraju i testiraju metod. Neophodno je da se definišu set podataka, pravila za izračunavanje i algoritam za učenje. Osnovu za implementaciju metoda predstavlja klasa *Basic*. Svaki GCRF metod treba da naslijedi ovu klasu i definiše svoja pravila za izračunavanje (klasa koja implementira interfejs *Calculations*) i algoritam za učenje (klasa koja implementira interfejs *LearningAlgorithm*).



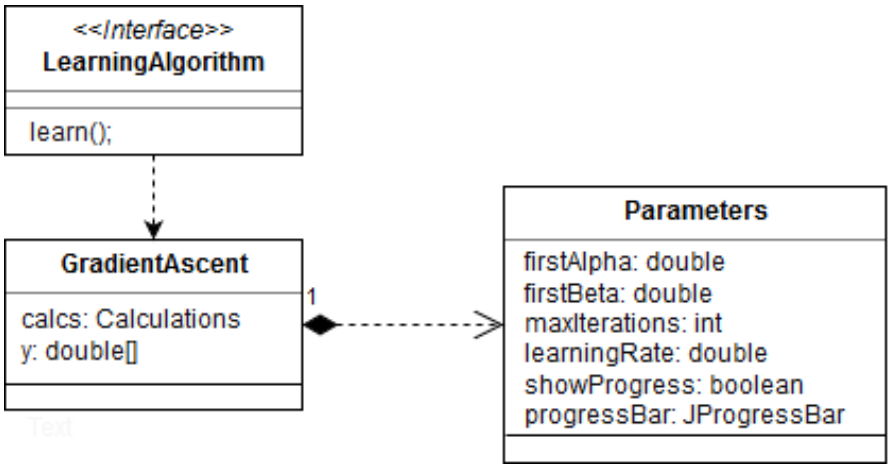
Slika 35. Dijagram klasa koje realizuju podršku za GCRF metode

Za klasu *Basic* pravila za izračunavanje i algoritam za učenje prosleđuju se kao ulazni argumenti konstruktoru klase, dok se za podklase oni unaprijed definišu u konstruktoru. Na primjer, u konstruktoru *GCRF* klase je definisano da se za pravila za izračunavanje koristi klasa CalculationsGCRF, a da se kao algoritam za učenje koristi *GradientAscent*.

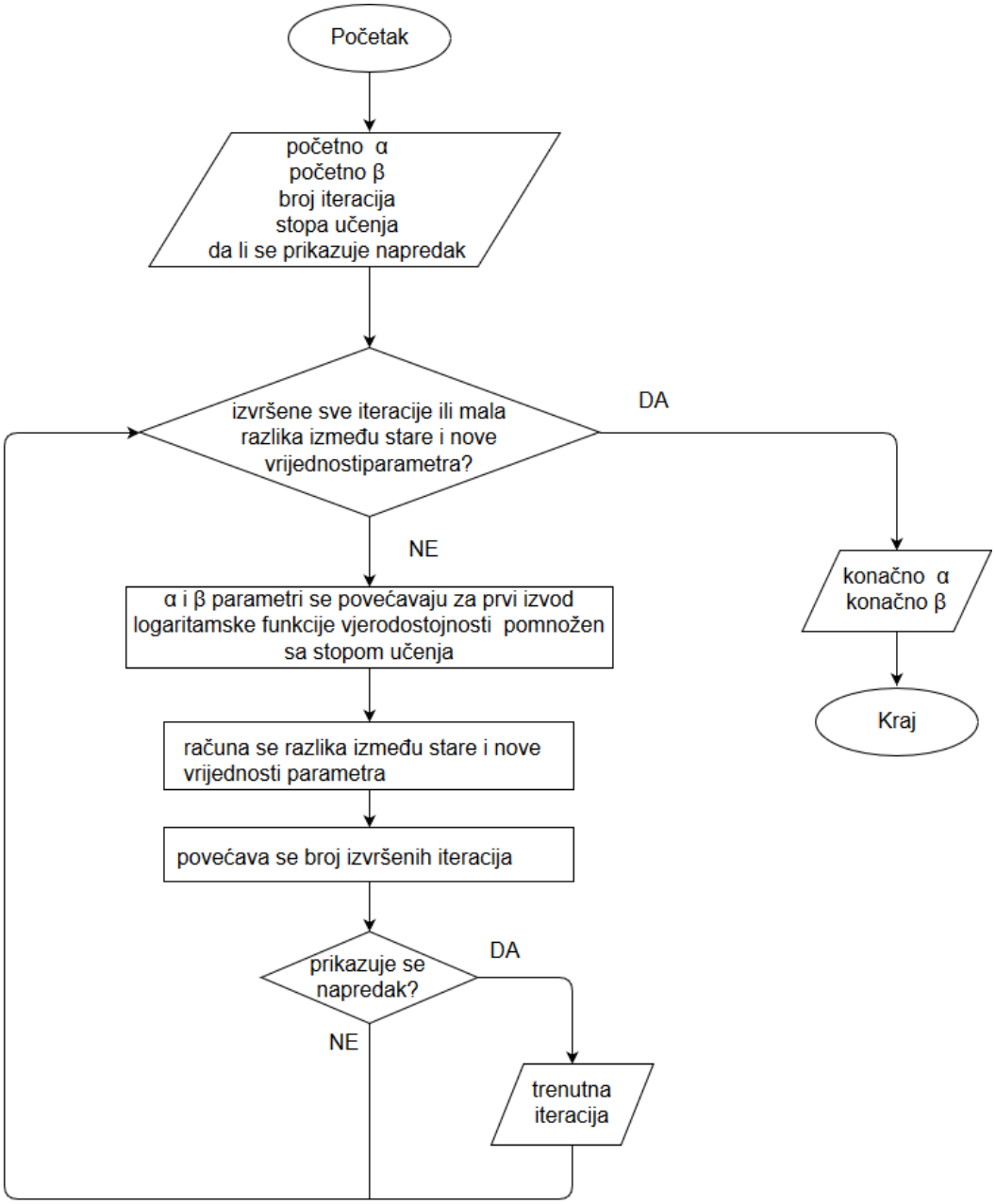
```
public GCRF(Parameters parameters, Dataset data) {
      this.expectedY = data.getY();
      this.calcs = new CalculationsGCRF
                           (data.getS(), data.getR());
      this.learning = new GradientAscent
                 (parameters, calcs, expectedY, false, null);
      double[] params = learning.learn();
      this.alpha = params[0];
      this.beta = params[1];
}
```

4.3.6. Implementacija algoritama za učenje

Klase koje predstavljaju algoritme za učenje nalaze se u paketu *gcrf_tool.learning*. Interfejs *LearningAlgorithm* definiše da svaki algoritam za učenje mora imati metodu *learn* koja kao povratnu vrijednost daje niz parametara koji su naučeni. Klasa *GradientAscent* implementira interfejs *LearningAlgorithm* upotrebom pravila algoritma garadient ascent, koji se koristi za treniranje metoda baziranih na GCRF-u kako bi se dobile vrijednosti za parametre α i β. Klasa *Parameters* definiše sve parametre koje je neophodno definisati za ovaj algoritam: početne vrijednosti za parametre α i β, maksimalan broj iteracija, stopa učenja, da li da se napredak algoritama prikazuje u konzoli i da li da se napredak algoritama prikazuje u nekom od prozora korisničkog interfejsa (JProgressBar). Na slici 36 dat je dijagram klasa koje realizuju podršku za algoritme za učenje, dok je dijagram toka procesa učenja preko algoritma penjanja u pravcu gradijenata prikazan na slici 37.



Slika 36. Dijagram klasa koje realizuju podršku za algoritme za učenje



Slika 37. Dijagram toka procesa učenja preko algoritma penjanja u pravcu gradijenata (gradient ascent)

Proces učenja se vrši tako što se α i β parametri povećavaju za prvi izvod logaritamske funkcije vjerodostojnosti pomnožen sa stopom učenja:

```
tempAlpha = alpha + lr * calcs.dervativeAlpha(alpha, beta, y);
tempBeta = beta + lr * calcs.dervativeBeta(alpha, beta, y);
```

Metoda koja računa izvod poziva se iz interfejsa *Calculations*, što znači da je prilikom kreiranja objekta *GradientAscent* klase neophodno proslijediti objekat klase koja implementira interfejs *Calculations* za neki konkretan metod (npr. *CalculationsGCRF* za standardni GCRF).

4.3.7. Povezivanje sa MatLab-om

Kao što je već navedeno, dva metoda implementirana su u Javi, dok su ostala četiri implementirana u Matlab-u i iz Jave se samo pozivaju. Izvorni kod za sve metode koje su implemetirane u Matlab-u nalazi se folderu „matlab“ unutar projekta. U cilju povezivanja, za svaki od ovih metoda kreirana je posebna klasa u *gcrf_tool.methods.matlab* paketu. Svaka klasa sadrži metodu *train* koja služi za pozivanje Matlab funkcije. S obzirom da Matlab implementacija za svaki od ovih metoda sadrži jednu funkciju koja obavlja proces treniranja i testiranja, metoda *train* će završiti sve neophodne korake. Metoda otvara novu sesiju i pokreće Matlab konzolu, poziva odgovarajuću funkciju iz izvornog koda, prosleđuje joj ulazne argumente i čeka povratnu vrijednost. Nakon što se funkcija završi, povratne vrijednosti se preuzimaju, sesija se zatvara i Matlab konzola se gasi bez intervencije korisnika.

Povezivanje sa Matlab-om implementirano je pomoću biblioteke *matlabcontrol*. Da bi se Java povezala sa Matlab-om upotrebom ove biblioteke, neophodno je da se prvo kreira objekat klase *MatlabProxyFactory,* a nakon toga i objekat klase *MatlabProxy*, koja se koristi za komunikaciju sa Matlab-om. Neophodno je da se definiše putanja do fajla *matlab.exe (matlabPath)* i vrijeme koje će Java čekati da se *MatlabProxy* kreira (*proxyTime*).

```
MatlabProxyFactoryOptions options = new
MatlabProxyFactoryOptions.Builder()
      .setHidden(true).setProxyTimeout(proxyTime)
      .setMatlabLocation(matlabPath).build();
MatlabProxyFactory factory = new MatlabProxyFactory(options);
MatlabProxy proxy = factory.getProxy();
```

Nakon kreiranja proksija, potrebno je da se doda putanja do fajlova sa izvornim kodom i da se pripreme i proslijede ulazni podaci. Klasa *MatlabTypeConverter* koristi se za

konverziju Java nizova u Matlab nizove (koji se prosleđuju preko metode *setNumericArray*), dok se promjenjive koje imaju primitivan tip podatka mogu proslijediti direktno (upotrebom metode *setVariable).*

```
String path =  Reader.jarFile() + "/matlab/upGCRF";
proxy.eval("addpath('" + path + "')");
processor.setNumericArray("S", new MatlabNumericArray(s, null));
proxy.setVariable("lag", lag);
```

Matlab funkcije se pozivaju preko metode *eval*, kojoj se u formi Stringa prosleđuju Matlab komande. Povratne vrijednosti preuzimaju se upotrebom metoda *getNumericArray* ili *getVariable*, a konverzija u Java niz vrši se pomoću metode *getRealArray2D.*

```
proxy.eval("[Data,muNoisyGCRF] =
          upGCRF(lag,trainTs,predictTs,maxiter,select_features,
                    N, X, y, similarities);");
MatlabNumericArray array =
          processor.getNumericArray("muNoisyGCRF");
double[][] outputs = array.getRealArray2D();
```

Nakon što se preuzmu povratne vrijednosti, potrebno je da se proksi otkači od Matlab-a pozivom metode *disconnect.* Međutim, ova metoda neće ugasiti Matlab konzolu i korisnik će u svom status baru vidjeti da je Matlab pokrenut. Stoga je neophodno da se Matlab proces ugasi programskim putem.

```
proxy.disconnect();
Runtime rt = Runtime.getRuntime();
rt.exec("taskkill /F /IM MATLAB.exe");
```

4.3.8. Implementacija korisničkog interfejsa

Korisnički interfejs je implementiran upotrebom Swing komponenti. Klase iz paketa *javax.swing* definišu fleksibilne GUI elemente koji su u cjelosti implementirani u Javi. Veoma je bitno da ove komponente nijesu ograničene karakteristikama platforme na kojoj se izvršavaju, što znači da sve komponente jednako izgledaju i rade na svim

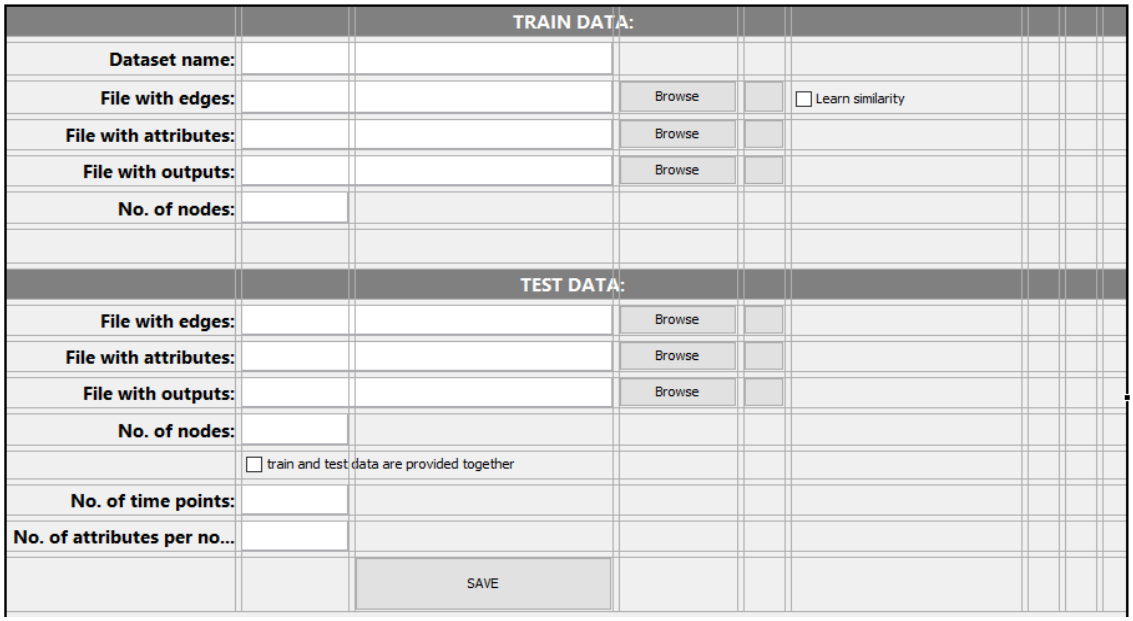
operativnim sistemima. Glavni grafički elementi iz Swing paketa koji se koriste za kreiranje korisničkog interfejsa su:

- Prozori - trajni prozori tj. okviri (*JFrame*) i privremeni prozori tj. prozori za dijalog (*JOptionPane)*
- Komponente - labele (*JLabel*), dugmad (*JButton*), tekstualna polja (*JTextField*), padajuće liste (*JComboBox*), check box (*JCheckBox*), radio dugmad (*JRadioButton*), tabele (*JTable*) itd.
- Kontejneri (služe za grupisanje komponenti) - klasa *JPanel*

Klase za rad sa korisničkim interfejsom u GCRF GUI TOOL projektu nalaze se u paketu *gcrf_tool.gui*. U okviru ovog paketa nalaze se sledeći paketi:

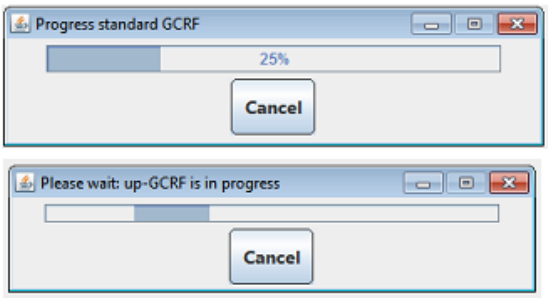
- *gcrf_tool.gui.frames* – sadrži okvire (klase koje nasleđuju Swing klasu *JFrame*)
- *gcrf_tool.gui.panels* - sadrži panele (klase koje nasleđuju Swing klasu *JPanel*)
- *gcrf_tool.gui.style* – sadrži klase koje imaju statičke metode za stilizovanje određenijih komponenti interfejsa (podešavanje vrste fonta, boje, veličine, okvira, ikonica itd.)
- *gcrf_tool.gui.threads* – sadrži niti (klase koje nasleđuju Java klasu *Thread*) za izvršavanje pozadinskih procesa

Interfejs je koncipiran tako da postoji jedan glavni okvir (klasa *MainFrame)*, koji sadrži samo glavni navigacioni meni. Centralnom dijelu prozora dodjeljuju se različiti paneli u zavisnosti od odabrane stavke menija. Postoji 10 klasa koje predstavljaju različite panele (kontejnere). Ove klase nasleđuju klasu *JPanel* i definišu komponente interfejsa za različite funkcionalnosti. Za svaki kontejner neophodno je definisati kako će se razmještati komponente unutar njega (layout). Korišćen je *GridBagLayout* koji je jedan od najfleksibilnijih layout-a u Java platformi. Komponente se smještaju u mrežu redova i kolona i dozvoljeno je da jedna komponenta koristi više redova ili kolona ukoliko je to potrebno (slika 38). Upotreba ovog layout-a omogućava da se veličina okvira i svih njegovih komponenti prilagođava veličini ekrana, ili veličini prozora.



Slika 38. Prikaz *GridBag* layout-a na primjeru panela *AddDataset*

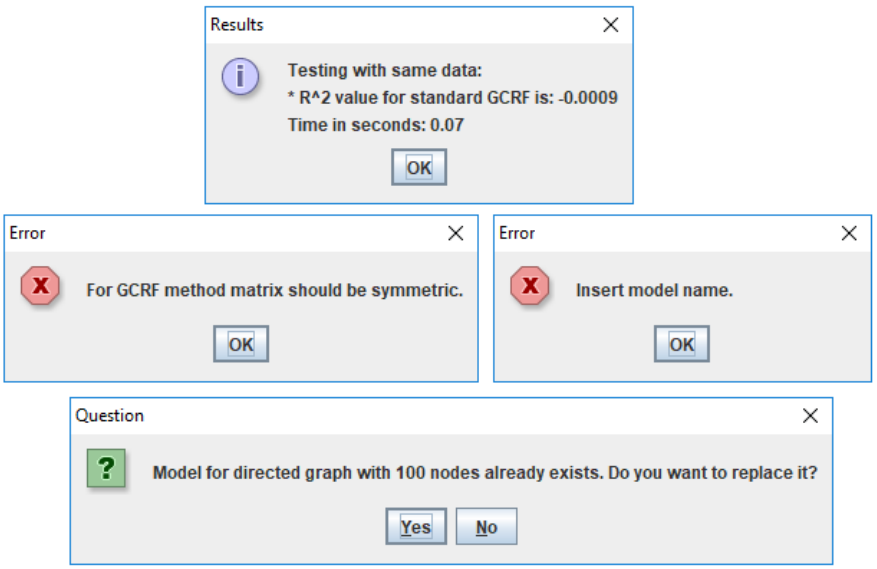
Kako bi se korisniku prikazao prozor sa informacijama o napretku, neophodno je da se treniranje metoda pokrene u zasebnoj niti koja će se odvijati u pozadini. Zbog toga je kreirana posebna podklasa klase Thread za treniranje svakog od metoda, kao i klasa ProgressBar koja implementira okvir za prikazivanje napretka. Kada su u pitanju metode u Javi, u ovom prozoru se prikazuje procenat završenog posla, dok za metode implementirane u Matlab-u te informacije nije moguće preuzeti, tako da se korisnicima prikazuje poruka da je neophodno sačekati jer je izvršenje metode u toku (slika 39).



Slika 39. Primjeri prozora za informisanje o napretku

Za informisanje korisnika o uspješno završenim koracima, o greškama ili za traženje potvrde od korisnika (da/ne opcije) koriste se prozori za dijalog (slika 40). Sistem je dizajniran tako da provjeri sve unesene vrijednosti i tako minimizuje mogućnost korisničke greške. Ukoliko neka vrijednost nije unesena kako treba, prikazuju se poruke

o grešci i funkcija se ne pokreće, čime se sprječava pojava neželjenih posledica. Svaka greška propraćena je razumljivom porukom koja sadrži informaciju o vrsti greške i postupcima za njeno ispravljanje.



Slika 40. Primjeri prozora za dijalog

4.4. Efikasnost

Efikasnost i skalabilnost alata testirane su na sintetički generisanim grafovima sa različitim brojem čvorova: 100, 500, 1.000 i 5.000. Vrijeme koje je potrebno softveru da izvrši treniranje metoda zavisi od metoda koji je izabran, njegove kompleksnosti i načina implementacije. Treniranje je imalo 50 iteracija i eksperimenti su izvršeni na računaru sa Windows operativnim sistemom koji ima 16GB memorije i 3,4 GHz CPU (tabela 8). Iz rezultata se može zaključiti da metodima koji su implementirani u Javi treba više vremena da se izvrše, prije svega zbog objektno-orjentisane prirode Java jezika koja zahtjeva više memorije i više vremena za računske operacije sa velikim matricama. S druge strane, Matlab ima mnogo bolju podršku za matematičke operacije višeg nivoa i brže izvršava ugrađene operacije sa matricama. Prednost upotrebe Jave je to što je besplatna i većina korisnika već ima instaliranu Javu na svojim računarima, dok je Matlab softver koji se plaća i za korisnike je skup za instalaciju.

Tabela 8. Vrijeme potrebno softveru da istrenira različite metode sa različitim brojem čvorova

Broj čvorova	**100**	**500**	**1000**	**5000**
Broj veza	5.094	127.540	509.376	12.749.518
GCRF	0,27 s	16,98 s	129,4 s	4h 45 min
DirGCRF	0,17 s	9,49 s	69,57 s	2h 12 min
UmGCRF	6,62 s	7,45 s	8 s	34,48 s
m-GCRF	8,49 s	17 s	53,15 s	65 min
up-GCRF	26,84 s	6,58 min	27,6 min	N/A
RLSR	69,25 s	8,25 min	1h 18 min	N/A

4.5. Evaluacija upotrebljivosti

4.5.1. Metodologija

Upotrebljivost (Nielsen, 2012) je atribut kvaliteta softvera koji opisuje koliko je korisnički interfejs lak za korišćenje i predstavlja vrlo važan aspekt svakog softvera. Korisnici će odustati od upotrebe softvera ukoliko je on težak za korišćenje, ukoliko nije jasan ili ne zadovoljava njihova očekivanja. Glavni atributi upotrebljivosti su (Rubin & Chisnell, 2008):

- Korisnost – da li softver omogućava korisnicima da ispune željeni cilj.
- Efikasnost – vrijeme koje je potrebno da se ispuni cilj koji korisnik želi da postigne upotrebnom softvera.
- Efektivnost – lakoća sa kojom korisnik može koristiti softver za ispunjenje određenog cilja.
- Lakoća učenja – da li je korisnik u mogućnosti da ispuni osnove zadatke upotrebom softvera, čak i ako ga nikad ranije nije koristio.
- Zadovoljstvo - percepcija, osjećanja i mišljenja korisnika o softveru.

Cilj testiranja upotrebljivosti je posmatranje korisnika dok koriste softver, kako bi se prikupili empirijski podaci koji mogu pomoći unapređenju softvera, i doprinijeti njegovoj korisnosti i upotrebljivosti. Postoje različite metodologije za testiranje upotrebljivosti i ovo poglavlje ukratko opisuje metodologiju koja je korišćenja za testiranje upotrebljivosti softvera GCRF GUI TOOL.

U prvoj fazi softver je testirala interna grupa eksperata koji su pokušali da identifikuju propuste u dizajnu, greške ili bilo koje druge probleme koji se mogu desiti prilikom upotrebe softvera. Neke greške su identifikovane i softver je ažuriran i optimizovan. U drugoj fazi je izvršena evaluacija sa korisnicima, kojoj su prethodili sledeći koraci:

1. Definisanje različitih vrsta korisnika koji bi trebalo da izvrše testiranje upotrebljivosti i odabir reprezentativnih primjera za svaku od grupa.
2. Definisanje zadataka koje korisnici treba da ispune upotrebom softvera.
3. Dizajniranje upitnika koji će korisnici popuniti nakon testiranja.

S obzirom da GCRF GUI TOOL treba da bude intuitivan i lak za korišćenje, kako za eksperte iz oblasti mašinskog učenja tako i za početnike, odlučeno je da softver testiraju dvije grupe korisnika: eksperti i studenti osnovnih i postdiplomskih studija.

Od svih korisnika se tražilo da upotrebom softvera ispune 4 zadatka. Detaljan opis zadataka dat je u odjeljku 4.5.2. Prije nego što su dobili zadatke, korisnicima je dat kratak opis funkcionalnosti sistema, bez ikakvih instrukcija vezanih za konkretne zadatke. Svi ispitanici morali su da imaju računar sa instaliranom Javom 8, dok Matlab nije bio obavezan.

Kako bi se detaljnije sagledala iskustva i mišljenja korisnika, kreiran je upitnik za evaluaciju koji su korisnici popunjavali nakon testiranja softvera. Glavni cilj upitnika je bio da se prikupe informacije od korisnika, kako bi se razjasnili i bolje razumjeli nedostaci i prednosti proizvoda (Rubin & Chisnell, 2008). Detaljan opis upitnika dat je u odjeljku 4.5.3.

U toku evaluacije upotrebljivosti, GCRF GUI TOOL je testiralo 34 korisnika, koji su bili podijeljeni po grupama:

- 12 eksperata iz oblasti mašinskog učenja sa Univerziteta Temple (Computer and Information Sciences Department, Univerzitet Temple, Filadelfija, SAD)
- 22 studenta osnovnih i postdiplomskih studija (15 sa Univerziteta Temple iz Filadelfije i 7 sa Univerziteta Mediteran iz Podgorice)

Rezultati evaluacije predstavljeni su u odjeljcima od 4.5.4. do 4.5.8.

4.5.2. Zadaci

Svi učesnici u evaluaciji sistema dobili su dokument koji detaljno opisuje šta se od njih očekuje. Dokument sadrži:

- kratak opis softvera GCRF GUI TOOL, link i uputstvo za instalaciju
- spisak neophodnog softvera (Java 8 i opciono Matlab) i linkove za instalaciju
- zadatke koje korisnik treba da ispuni
- link za upitnik

Od korisnika se tražilo da ispune sledeće zadatke:

- **Zadatak 1:** Primijeniti DirGCRF algoritam na „Teen Asymmetric 3x“ (Teenagers) setu podataka. Koristiti neuronsku mrežu kao nestrukturni prediktor. Nad istim setom podataka primijeniti i standardni GCRF. Nazvati model „Problem1“. Nakon treniranja testirati kreirani model.
- **Zadatak 2:** Primijeniti m-GCRF algoritam na „Random m-GCRF“ setu podataka. Koristiti linearnu regresiju kao nestrukturni prediktor. Nazvati model „Problem2“.
- **Zadatak 3:** Trenirati DirGCRF model na sintetički generisanom usmjerenom grafu i acikličnom usmjerenom grafu sa proizvoljnim brojem čvorova. Trenirati i simetričan model. Nakon treniranja testirati kreirane modele.
- **Zadatak 4:** Dodati novi set podataka (dat je link sa kojeg se mogu preuzeti txt fajlovi za novi set podataka). Broj čvorova za treniranje je 25, a za testiranje 25. Nazvati set podataka „Problem4“. Nakon dodavanja seta podataka, promijeniti mu ime u „Problem4New“.

Kako bi se utvrdilo da li su ispitanici uspješno izvršili zadatke od njih se tražilo da dostave sliku ekrana sa porukom nakon uspješno završenog koraka (treniranje, testiranje, dodavanje seta podataka itd.), kao i foldere sa modelima za zadatke 1, 2 i 3, i folder sa setom podataka za zadatak 4. Ispitanici koji nijesu imali Matlab nijesu mogli da urade zadatak 2.

4.5.3. Upitnik

Upitnik je definisan tako da olakša rad i autorima (lakša analiza odgovora) i ispitanicima (minimizovano vrijeme potrebno za popunjavanje upitnika). Od ispitanika

se traži da zaokruže neki od ponuđenih odgovora, ili da ocijene koliko se slažu sa određenim iskazima na skali od 1 do 5 (Likertova skala). Upitnik ima samo jedno pitanje otvorenog tipa, koje nije obavezno. Upitnik se sastoji od 5 djelova:

1. Profil korisnika – nivo znanja/iskustva korisnika.
2. Lakoća zadataka – lakoća sa kojom su korisnici izvršili tražene zadatke.
3. Terminologija i informacije koje sistem pruža – zadovoljstvo korisnika korisničkim interfejsom.
4. Upotrebljivost sistema – zadovoljstvo korisnika softverom.
5. Komentari/predlozi – otvoreno za predloge i sugestije korisnika.

Ova pitanja omogućavaju prikupljanje mišljenja i utisaka korisnika prilikom korišćenja softvera, kada je u pitanju jednostavnost ovladavanja i korišćenja, kao i da se sazna cjelokupni utisak i zadovoljstvo korisnika. Upitnik testira sve glavne atribute upotrebljivosti osim efikasnosti, koja je testirana posebno i rezultati su predstavljani u odjeljku 4.4.

Upitnik za evaluaciju je na engleskom jeziku i sadrži sledeća pitanja[17]:

1) Godine
2) Pol
3) Nivo obrazovanja (mora se izabrati jedan odgovor):
    a. Istraživač
    b. Profesor
    c. Student osnovih studija
    d. Student master studija
    e. Student doktorskih studija
    f. Ostalo (uz mogućnost unosa)
4) Operativni sistem (mora se izabrati jedan odgovor):
    a. Windows
    b. Linux
    c. Mac
5) Da li je Matlab instaliran na vašem računaru? (mora se izabrati jedan odgovor)

[17] https://goo.gl/forms/zlnVWgQtT3FCvA2S2

a. Da
b. Ne

6) Znanje iz oblasti mašinskog učenja (može se izabrati više odgovora):
    a. Imam dobro teorijsko znanje iz oblasti mašinskog učenja.
    b. Imam praktično iskustvo iz oblasti mašinskog učenja.
    c. Koristio/la sam framework-e za mašinsko učenje.
    d. Koristio/la sam GUI alate za mašinsko učenje.
    e. Upoznat/a sam sa mašinskim učenjem, ali nemam praktičnog iskustva.
    f. Nijesam upoznat/a sa mašinskim učenjem.
7) Znanje iz oblasti strukturne regresije (može se izabrati više odgovora):
    a. Imam dobro teorijsko znanje iz oblasti strukturne regresije.
    b. Imam praktično iskustvo iz oblasti strukturne regresije.
    c. Upoznat/a sam sa strukturnom regresijom, ali nemam praktičnog iskustva.
    d. Nijesam upoznat/a sa strukturnom regresijom.
8) Koliko su laki određeni zadaci na skali od 1 do 5 (5-veoma lako, 4-lako, 3-neutralno, 2-teško, 1-veoma teško):
    a. Treniranje i testiranje nad mrežama
    b. Treniranje i testiranje nad vremenskim mrežama
    c. Treniranje i testiranje nad sintetičkim mrežama
    d. Dodavanje seta podataka
    e. Upravljanje setovima podataka
    f. Konfiguracija
9) Terminologija i informacije koje sistem pruža: Ocijeniti iskaze od 1 do 5 (5-u potpunosti se slažem, 4-slažem se, 3-neutralan/a sam, 2- ne slažem se, 1-u potpunosti se ne slažem):
    a. Upotreba termina je konzistentna kroz cijeli softver.
    b. Terminologija je vezana za zadatke.
    c. Pozicija poruka na ekranu je konzistentna.
    d. Zahtjevi za unos podataka su jasni.
    e. Aplikacija izvještava o napretku.
    f. Poruke o greškama su od pomoći.
    g. Aplikacija sadrži koristan meni za pomoć (Help).

10) Upotrebljivost sistema: Ocijeniti iskaze od 1 do 5 (5-u potpunosti se slažem, 4-slažem se, 3-neutralan/a sam, 2- ne slažem se, 1-u potpunosti se ne slažem):
    a. Želio/la bih da koristim ovu aplikaciju.
    b. Smatram da je aplikacija kompleksnija nego što bi trebalo.
    c. Mislim da je aplikacija laka za korišćenje.
    d. Potrebna mi je pomoć iskusnije osobe da bih mogao/la da koristim ovaj softver.
    e. Mislim da u aplikaciji ima previše nekonzistentnosti.
    f. Mislim da većina korisnika može brzo naučiti da koristi ovaj softver.
    g. Aplikacija dobro integriše različite funkcionalnosti. Aplikacija je veoma komplikovana za korišćenje.
11) Komentari i sugestije

Dodatni upitnik je kreiran samo za eksperte, kako bi se dobilo njihovo mišljenje o korisnosti softvera. Ovaj upitnik je takođe na engleskom jeziku i sadrži 10 pitanja [18]:

1) Da li biste koristili ovaj softver u svom radu? (Ponuđeni odgovori: da, ne i možda)
2) Da li biste preporučili ovaj softver svojim kolegama? (Ponuđeni odgovori: da, ne i možda)
3) Da li biste preporučili ovaj softver ekspertima u oblasti mašinskog učenja? (Ponuđeni odgovori: da, ne i možda)
4) Da li biste preporučili ovaj softver studentima (početnicima u oblasti mašinskog učenja)? (Ponuđeni odgovori: da, ne i možda)
5) Da li mislite da bi se ovaj softver trebao predstaviti istraživačima u oblasti mašinskog učenja na nekoj konferenciji ili workshop-u? (Ponuđeni odgovori: da, ne i možda)
6) Da li mislite da će ovaj softver olakšati implementaciju metoda baziranih na GCRF-u?
    a. da, za istraživače u oblasti mašinskog učenja
    b. da, za istraživače u ostalim oblastima
    c. da, za studente
    d. da, za sve navedeno

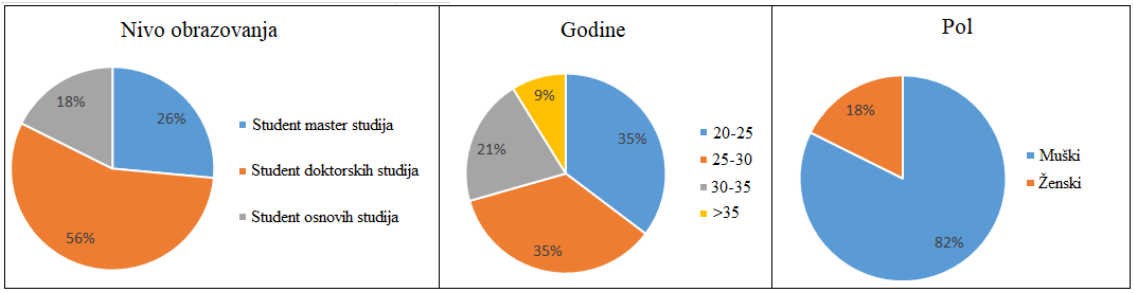
[18] https://goo.gl/forms/xDdyDKF3Lrg1vfC13

e. ne

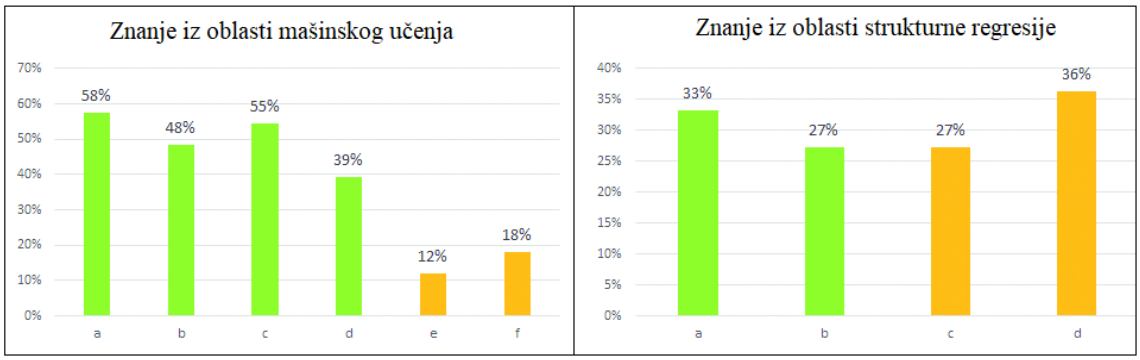
7) Da li mislite da će ovaj softver povećati primjenu metoda baziranih na GCRF-u? (Ponuđeni odgovori: da, ne i možda)
8) Zašto mislite da ovaj softver jeste/nije koristan? (Pitanje otvorenog tipa)
9) Šta mislite da je najveća prednost/mana ovog softvera, sa ekspertske tačke gledišta? (Pitanje otvorenog tipa)
10) Šta mislite da bi eksperti iz oblasti mašinskog učenja željeli da vide u sledećoj verziji ovog softvera? (Pitanje otvorenog tipa)

4.5.4. Osnovne informacije o korisnicima

U prvom dijelu upitnika od korisnika se tražilo da daju osnovne informacije o sebi (godine, pol, nivo obrazovanja), kao i da ocjene svoje znanje iz oblasti mašinskog učenja i strukturne regresije. Dijagrami koji prikazuju osnovne informacije o korisnicima dati su na slici 41. Većina ispitanika su bili muškarci (82%), od 20 do 30 godina (71%). Kada je u pitanju obrazovanje, većina ispitanika bili su studenti doktorskih studija (56%), ali je bilo i studenta master (26%) i osnovnih studija (18%), što znači da su u evaluaciji učestvovali korisnici različitih profila. Sa slike 42 se vidi da 30% ispitanika nije imalo iskustva u oblasti mašinskog učenja, kao i da 64% nije upoznato sa strukturnom regresijom.

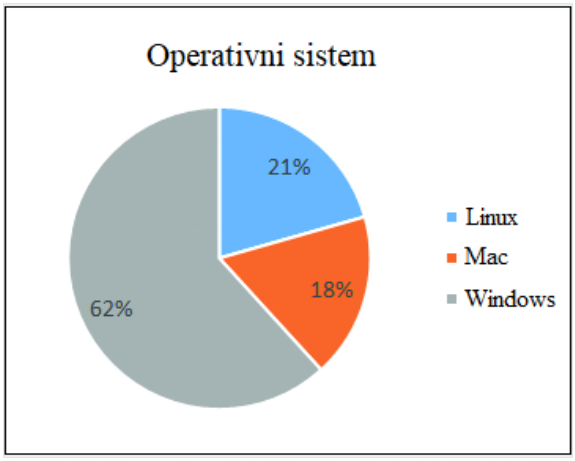


Slika 41. Osnovne informacije o korisnicima



Slika 42. Nivo znanja korisnika

Ovaj dio upitnika obuhvatao je i pitanja vezana za sistem koji su ispitanici koristili da testiraju softver: koji operativni sistem imaju i da li imaju instaliran Matlab. Informacija koji operativni sistem je korišćen je veoma važna, jer se neke funkcionalnosti drugačije ponašaju na različitim operativnim sistemima i to može uticati na mišljenje korisnika. Ispitanici su koristili različite operativne sisteme (slika 43), a preovladavao je Windows (62%). Većina ispitanika je imala instaliran Matlab (76%).



Slika 43. Operativni sistemi koje su ispitanici koristili da testiraju softver

4.5.5. Jednostavnost korišćenja

Glavni cilj ovog dijela upitnika je bio da se zaključi da li su specifični zadaci teški ili laki za različite tipove korisnika. Odgovori ispitanika predstavljeni su u tabeli 9. Iz rezultata se može vidjeti da je većina zadataka imala prosječnu ocjenu 4.4 ili veću. Takođe, 80-94% ispitanika je ocijenilo zadatke kao „veoma lake“ ili „lake“, dok su za 3-8% ispitanika zadaci bili „veoma teški“ ili „teški“. Niže ocjene pojedinih zadataka

prouzrokovane su činjenicom da su neki korisnici imali problem da konfigurišu konekciju sa Matalab-om na Linux ili Mac operativnom sistemu.

Tabela 9. Rezultati za lakoću zadataka

Zadatak	**1**	**2**	**3**	**4**	**5**	**Prosjek**	**Standardna devijacija**
a) Treniranje i testiranje nad mrežama	0	1	1	7	25	4,7	0,7
b) Treniranje i testiranje nad vremenskim mrežama	0	3	0	8	21	4,5	0,9
c) Treniranje i testiranje nad sintetičkim mrežama	0	1	4	7	20	4,4	0,8
d)Dodavanje seta podataka	1	0	2	7	24	4,6	0,8
e)Upravljanje setovima podataka	1	1	0	4	27	4,7	0,9
f) Konfiguracija	0	2	3	9	19	4,4	0,9

* 5-veoma lako, 4-lako, 3-neutralno, 2-teško, 1-veoma teško

4.5.6. Terminologija

Pitanja u ovom dijelu upitnika služe da se uvidi mišljenje ispitanika o grafičkom korisničkom interfejsu i njegovim komponentama. Iz rezultata u tabeli 10 se može vidjeti da su svi iskazi imali prosječnu ocjenu veću od 4, što znači da su korisnici generalno zadovoljni sa jasnoćom korišćenje terminologije. Ipak, neki od njih bi željeli da postoji bolji „Help“, kao i da softver pruža detaljnije informacije o greškama i jasnije poruke za unos podataka.

Tabela 10. Rezultati za terminologiju i informacije koje sistem pruža

Iskaz	**1**	**2**	**3**	**4**	**5**	**Prosjek**	**Standardna devijacija**
a) Upotreba termina je konzistentna kroz cijeli softver	0	0	0	9	25	4,7	0,4
b) Terminologija je vezana za zadatke	0	0	1	11	22	4,6	0,5
c) Pozicija poruka na ekranu je konzistentna	0	1	1	8	24	4,6	0,7
d) Zahtjevi za unos podataka su jasni	1	1	3	11	18	4,3	1
e) Aplikacija izvještava o napretku	0	2	2	7	23	4,5	0,8
f) Poruke o greškama su od pomoći	1	4	5	8	16	4	1,2
g) Aplikacija sadrži koristan meni za pomoć (Help)	3	1	3	14	13	4	1,2

* 5-u potpunosti se slažem, 4-slažem se, 3-neutralan/a sam, 2- ne slažem se, 1-u potpunosti se ne slažem

4.5.7. Upotrebljivost sistema

Skala upotrebljivosti sistema (System Usability Scale – SUS) (Brooke, 1996) je korišćena za globalnu procjenu upotrebljivosti sistema. SUS se pokazao kao dobar alat i pouzdana mjera upotrebljivosti sistema. Pitanja koja SUS obuhvata mogu pomoći u otkrivanju nivoa zadovoljstva korisnika sistemom, kao i da se uvidi koliko im je bilo teško da savladaju korišćenje sistema, što je veoma važno za GCRF GUI TOOL, koji je namijenjen korisnicima sa različitim nivoom znanja.

Skala upotrebljivosti sistema obuhvata 10 iskaza i od korisnika se traži da ih ocijene upotrebom Likertove skale. U ovom upitniku koristi se 8 iskaza, jer se od ispitanika u prethodnim djelovima upitnika već tražilo da procijene svoje znanje i lakoću zadataka. Rezultati su prikazani u tabeli 11. Za ovaj dio upitnika veća ocjena je poželjna za iskaze a, c, f i g, i svaki od ovih iskaza je imao prosječnu ocjenu 4 ili veću. Sa druge strane, poželjno je da iskazi b, d, e i h imaju što manju ocjenu, i svaki od ovih iskaza je imao prosječnu ocjenu 2,3 ili nižu. Iz ovih rezultata se može vidjeti da bi 76% ispitanika željelo da koristi aplikaciju i da 82% ispitanika misli da je aplikacija laka za korišćenje i da dobro integriše različite funkcionalnosti. Takođe, može se vidjeti da 12% ispitanika misli da je aplikacija previše kompleksna, a 6% da je previše komplikovana za korišćenje. 18% ispitanika smatra da im je potrebna pomoć iskusnije osobe da bih mogli da koriste softver, dok 9% ne smatra da bi većina korisnika mogla brzo naučiti da koristi softver.

Tabela 11. Rezultati za upotrebljivost sistema

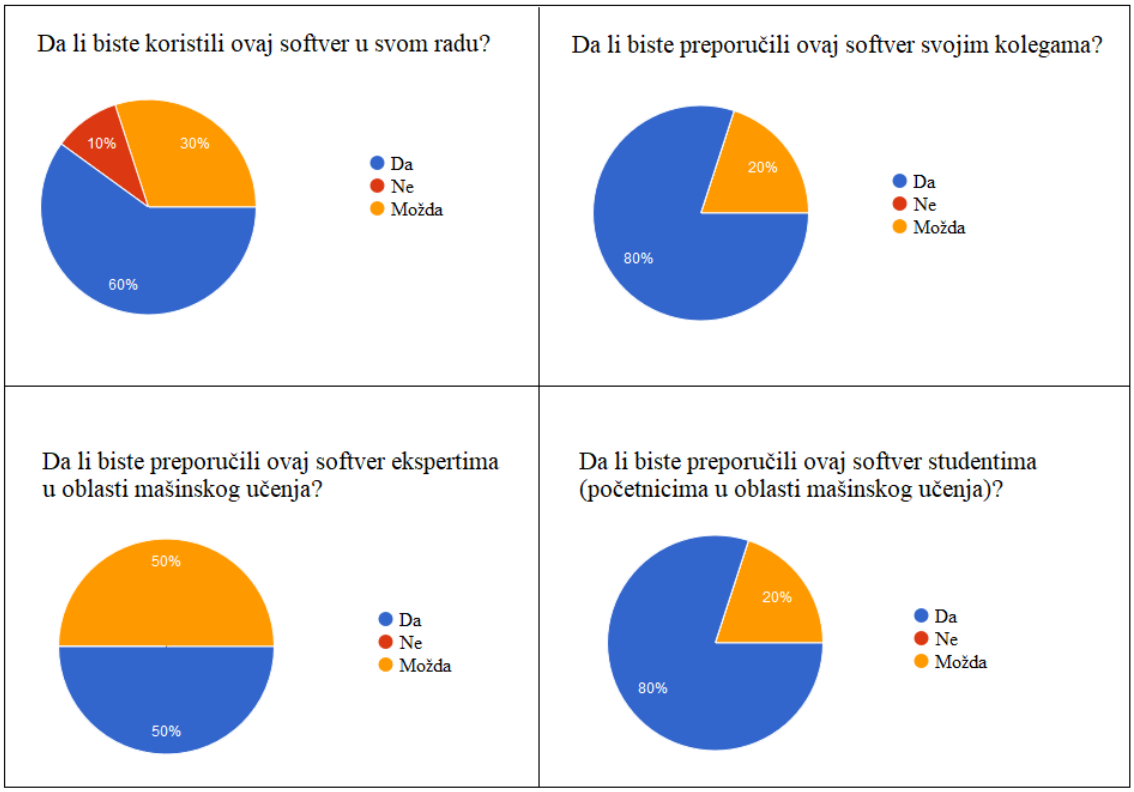
Iskaz	**1**	**2**	**3**	**4**	**5**	**Prosjek**	**Standardna devijacija**
a) Želio/la bih da koristim ovu aplikaciju	1	2	5	13	13	4	1
b) Smatram da je aplikacija kompleksnija nego što bi trebalo	15	13	2	3	1	1,9	1
c) Mislim da je aplikacija laka za korišćenje	1	2	3	12	16	4,2	1
d) Potrebna mi je pomoć iskusnije osobe da bih mogao/la da koristim ovaj softver	10	14	4	3	3	2,3	1,2
e) Mislim da u aplikaciji ima previse nekonzistentnosti	20	11	1	1	1	1,6	0,9
f) Mislim da većina korisnika može brzo naučiti da koristi ovaj softver	1	2	7	9	15	4	1
g) Aplikacija dobro integriše različite funkcionalnosti	1	2	3	12	16	4,2	1
h) Aplikacija je veoma komplikovana za korišćenje	17	12	2	3	0	1.7	0,9

* 5-u potpunosti se slažem, 4-slažem se, 3-neutralan/a sam, 2- ne slažem se, 1-u potpunosti se ne slažem

4.5.8. Korisnost sistema

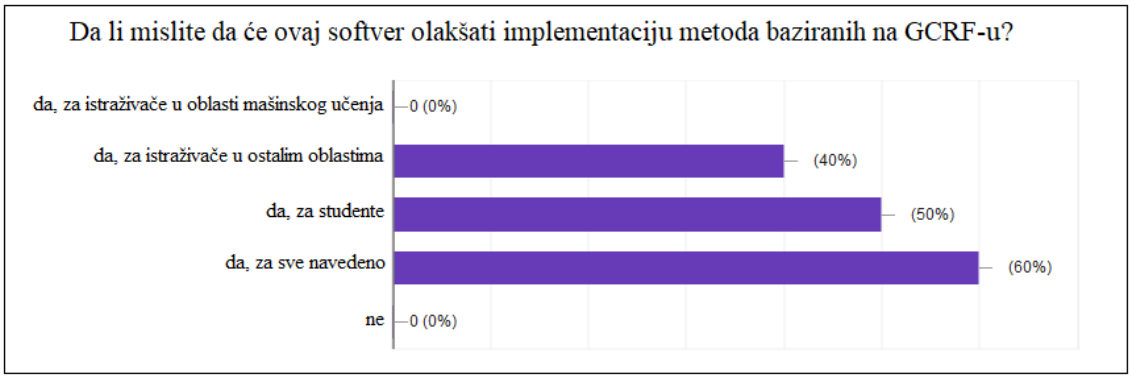
Dodatni upitnik kojim se analizira korisnost sistema popunjavali su samo eksperti. Sa grafika koji su prikazani na slici 44 može se vidjeti da eksperti imaju pozitivno mišljenje o sistemu. U svom radu sistem bi koristilo od 60% do 90% ispitanika. Ispitanici bi sistem preporučili svojim kolegama (80%) i ekspertima u oblasti mašinskog

učenja (50%), kao i početnicima (80%). Takođe se može vidjeti da ispitanici nijesu imali negativan stav (odgovor „ne“), kada je u pitanju preporuka alata nekoj od potencijalnih grupa korisnika. 70% ispitanika smatra da bi se ovaj softver trebao predstaviti na nekoj konferenciji ili workshop-u iz oblasti mašinskog učenja.



Slika 44. Stav eksperata da li bi koristili sistem i da li bi ga preporučili

Na slici 45 se mogu vidjeti odgovori eksperata na pitanje da li softver olakšava implementaciju metoda baziranih na GCRF-u. Na ovo pitanje ispitanici su mogli da izaberu više ponuđenih dogovora. 60% ispitanika misli da softver može olakšati implementaciju metoda za sve navedene grupe (istraživače u oblasti mašinskog učenja, istraživače u ostalim oblastima i studente), dok smatraju da je softver najkorisniji za studente (50%) i istraživače u ostalim oblastima (40%). Na pitanje šta misle da li će ovaj softver povećati primjenu metoda baziranih na GCRF-u 70% je odgovorilo „da“, 30% „možda“, dok odgovor „ne“ nije odabrao nijedan ispitanik.



Slika 45. Mišljenje eksperata da li softver olakšava implementaciju metoda baziranih na GCRF-u

Preostala tri pitanja su bila otvorenog tipa i u daljem tekstu su dati odgovori ispitanika. Odgovori su prevedeni sa engleskog jezika i nijesu navedeni slični odgovori.

Zašto mislite da ovaj softver jeste/nije koristan?

- Koristan je jer olakšava implementaciju modela koji se mogu koristiti za poređenje.
- Koristan je jer omogućava korisnicima da pokrenu, testiraju i evaluiraju GCRF metode preko jednostavnog i intuitivnog interfejsa, koji čini metode dostupnim čak i za ljude van oblasti mašinskog učenja.
- Veoma je koristan za ljude koji koriste GCRF metode (ne toliko za one koji ih razvijaju). Lak je za korišćenje.
- Ne može se koristiti za specijalne slučajeve koji još uvijek nijesu implementirani u softveru (na primjer više informacija o sličnosti u istom grafu).
- Koristan je za istraživače iz drugih oblasti, pogotovo zbog korisničkog interfejsa.
- Koristan je jer omogućava upotrebu metoda bez poznavanja matematičke pozadine.
- Aplikacija je korisna jer je laka za upotrebu i omogućava jednostavnu promjenu parametra i setova podataka.
- Aplikacija je veoma korisna za početnike.

Šta mislite da je najveća prednost/mana ovog softvera, sa ekspertske tačke gledišta?

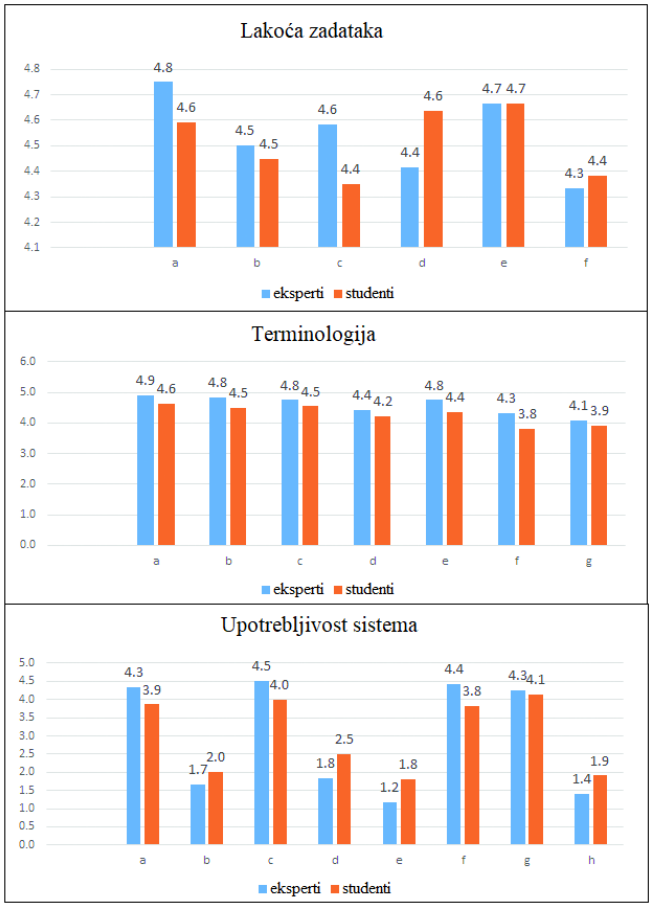
- Kao istraživač u oblasti mašinskog učenja smatram da je najveća prednost to što se, kada se razvije novi GCRF metod (ili bilo koji drugi metod strukturne regresije), performanse postojećih GCRF metoda mogu lako evaluirati i novi metod se može brzo i jednostavno uporediti sa njima.
- Prednost je korisnički interfejs koji je lak za korišćenje.
- Prednost je to što omogućava brz i lak način za eksperimentisanje sa GCRF metodima, bez programiranja.
- Prednost je to što olakšava međusobno poređenje performansi različitih GCRF metoda.

Šta mislite da bi eksperti iz oblasti mašinskog učenja željeli da vide u sledećoj verziji ovog softvera?

- Automatsko formatiranje setova podataka.
- Vizuelizaciju procesa treniranja (kako bi korisnik imao uvid u dešavanja u toku faze treniranja modela).
- Podršku za različite formate podataka (grafova).
- Evaluaciju postavke eksperimenta (da li je potrebno više podatka za treniranje ili testiranje, da li je graf koristan itd.).
- Podršku za više jezika i vizuelizaciju rezultata.
- Statistički rezime setova podataka (preko grafika ili tabela).
- Nove GCRF modele.
- Ubrzanje softvera i napredne funkcionalnosti (vizuelizacija i automatsko poređene različitih metoda).

4.5.9. Poređenje rezultata različitih vrsta korisnika

Cilj ovog poglavlja je da se odvojeno posmatraju i analiziraju odgovori različitih vrsta korisnika, eksperata i početnika u oblasti mašinskog učenja. Prosječne ocjene za sva pitanja predstavljene su na graficima na slici 46. Sa ovih grafika se može zaključiti da su obje vrste korisnika imale slično iskustvo sa softverom i da su ga ocijenile na sličan način. Ovo dokazuje hipotezu da GCRF GUI TOOL mogu koristiti kako i eksperti u oblasti mašinskog učenja, tako i početnici kojima GCRF model može pomoći da dođu do željenih informacija (H4).



Slika 46. Prosječne ocjene različitih vrsta korisnika (eksperti i početnici)

4.5.10. Dalji razvoj i unapređenja

Iako se GCRF GUI TOOL nalazi u prvoj fazi razvoja, iz komentara ispitanika se može zaključiti da je prvo iskustvo korisnika sa softverom bilo ohrabrujuće i da većina njih smatra da je alat veoma jednostavan i lak za korišćenje. Rezultati su potvrdili hipotezu da će alat pojednostaviti treniranje i testiranje GCRF modela i njegovih proširenja na različitim setovima podataka (H3).

Dalji razvoj alata se pažljivo planira i cilj evaluacije je bio da pomogne da se odrede prioriteti za budući razvoj. Glavne zamjerke korisnika odnosile su se na „Help“ i objašnjenja u softveru, tako da je prvi cilj za sledeću verziju softvera da se:

- poboljša „Help“ meni
- poboljšaju objašnjenja i vođenje korisnika kroz aplikaciju
- pruže detaljnija objašnjenja svih parametra i istakne dozvoljeni opseg vrijednosti
- pored padajuće liste sa modelima doda kratak opis svih dostupnih modela

- označe i naglase polja koja moraju biti promijenjena kako bi se riješile određene greške

Takođe, plan je da se kreiraju ilustrativni primjeri mreža/grafova (problemi iz realnog života kao što su pušenje, konzumiranje alkohola itd.) i studije slučaja koje će olakšati korisnicima izbor odgovarajuće metode za svoj problem.

Poslednji dio upitnika bio je otvoren za komentare i sugestije korisnika, koji su bili veoma korisni i značajni. Na osnovu komentara korisnika izdvojili su se sledeći zaključci:

- Nakon što se završi proces treniranja metoda, prikazati i R^2 nestrukturnog prediktora, jer on prilično utiče na preciznost korišćenog metoda.
- Potrebno je sačuvati rezultate faze treniranja, jer se u trenutnoj verziji čuvaju samo rezultati faze testiranja.
- Integrisati testiranje u „Train on networks“ funkcionalnost, kao što je to urađeno sa funkcionalnosti „Train on temporal networks“.
- Dodati opciju da se model sa određenim imenom obriše ili zamjeni novim modelom, jer u trenutnoj verziji korisnici mogu samo ručno da obrišu model sa fajl sistema.
- Prikazati informacije o sintetički generisanim grafovima ili omogućiti grafički prikaz.

5. Java biblioteka GCRFs

5.1. Implementacija

Osnovne klase iz GCRF GUI TOOL projekta su izdvojene i kreirana je Java biblioteka pod imenom „GCRFs“. Softverski alat koji je predstavljen u prethodnom poglavlju pojednostavljuje treniranje i testiranje postojećih GCRF metoda preko korisničkog interfejsa, dok biblioteka GCRFs obezbjeđuje Java klase i metode koje istraživači sa iskustvom u Java programiranju mogu koristiti da ugrade postojeće GCRF metode u svoj kod ili da kreiraju proširenja postojećih metoda. Ova biblioteka sadrži osnovne koncepte za implementaciju metoda baziranih na GCRF, kao i implementaciju dvije konkretne metode (standardnog i usmjerenog GCRF-a). GCRFs biblioteka ima intuitivan i jednostavan programski interfejs (API) [19], fleksibilna je i lako proširiva. Biblioteka sadrži implementaciju svih klasa koje su predstavljene na klasnom dijagramu u odjeljku 3.3 i koje su detaljno opisane u odjeljku 4.3.

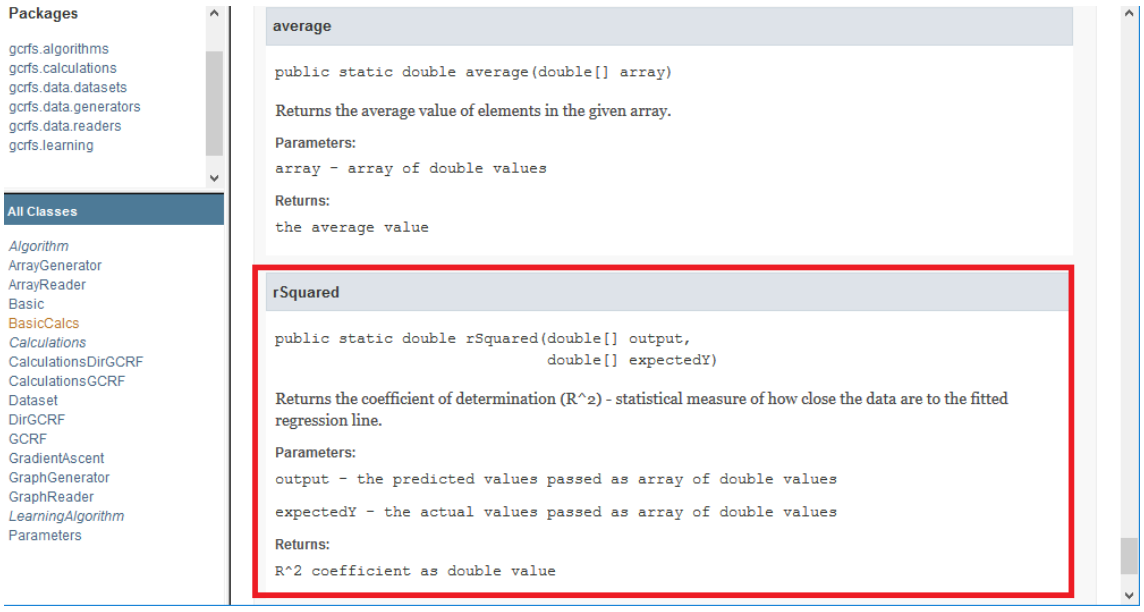
API je generisan upotrebom Javadoc [20] alata koji omogućava generisanje dokumentacije u HTML formatu, na osnovu komentara koji se nalaze u izvornom kodu. Komentari moraju biti napisani prije definicije klase, atributa, konstruktora ili metode. Svaki komentar ima dio za opis nakon kog slijede različiti tagovi (na primjer *@param* i *@return*). Primjer koda i komentara za metodu *rSquared* koja ima dva ulazna argumenta i vraća koeficijent odlučnosti dat je na kodu ispod, dok je prikaz API-a za datu metodu prikazan na slici 47.

```
/**
* Returns the coefficient of determination (R^2) - statistical
measure of
* how close the data are to the fitted regression line.
* @param output
*          the predicted values passed as array of double values
```

[19] http://htmlpreview.github.io/?https://github.com/vujicictijana/Library/blob/master/Library/api/index.html

[20] http://www.oracle.com/technetwork/articles/java/index-jsp-135444.html

```
* @param expectedY
*          the actual values passed as array of double values
* @return R^2 coefficient as double value
*/
public static double rSquared
                    (double[] output, double[] expectedY) {
      double avg = average(output);
      double firstSum = 0;
      double secondSum = 0;
      for (int i = 0; i < output.length; i++) {
            firstSum += Math.pow(expectedY[i] - output[i], 2);
            secondSum += Math.pow(expectedY[i] - avg, 2);
      }
      return 1 - (firstSum / secondSum);
}
```



Slika 47. Primjer generisanog API-a za metodu *rSquared* iz klase *BasicCalcs*

Projekat je otvorenog koda i dostupan je na GitHub-u [21]. U opisu projekta nalazi kratko uputstvo za upotrebu biblioteke, link ka kompletnom API-u, kao i link sa kog se može preuzeti *jar* fajl.

5.2. Način upotrebe

Ukoliko Java projekat zahtjeva neke Java biblioteke da bi mogao da funkcioniše, onda je neophodno konfigurisati projekat tako da se te biblioteke uključe u njega. Prvi korak je da se sa Interneta preuzme fajl *gcrfs.jar* i da se u projektu kreira referenca ka tom fajlu. S obzirom da biblioteka GCRFs koristi biblioteku OjAlgo, neophodno je dodati referencu za još jedan jar fajl (*ojalgo-40.0.0.jar*). Svaka klasa koja se koristi mora se importovati kako bi mogli da se kreiraju njeni objekti i koriste njene metode. Na primjer:

```
import gcrfs.algorithms.GCRF;
```

Lista svih paketa, klasa i metoda može se vidjeti u API-u.

5.2.1. Kreiranje setova podataka

Setovi podatka mogu se pročitati iz tekstualnih fajlova ili generisati sintetički. Ukoliko se setovi čitaju iz fajlova, koriste se metode iz klasa *ArrayReader* i *GraphReader* (iz paketa *gcrfs.data.readers*).

```
double[] r = ArrayReader.readArray("data/r.txt");
double[] y = ArrayReader.readArray("data/y.txt");
double[][] s = GraphReader.readGraph("data/s.txt", y.length);
Dataset dataset = new Dataset(s, r, y);
```

Ukoliko se setovi generišu sintetički, koriste se metode iz klasa *ArrayGenerator* i *GraphGenerator* (iz paketa *gcrfs.data.generators*). Primjer za kreiranje sintetičkog seta podataka za usmjereni aciklični graf od 50 čvorova nad kojim će biti primijenjen usmjereni GCRF:

```
double[][] s = GraphGenerator.generateDirectedAcyclicGraph(50);
double[] r = ArrayGenerator.generateArray(50, 5);
```

[21] https://github.com/vujicictijana/GCRFs_Library

```
CalculationsDirGCRF c = new CalculationsDirGCRF(s, r);
double[] y = c.y(1, 2, 0.05);
Dataset dataset = new Dataset(s, r, y);
```

5.2.2. Primjena metoda

Kako bi se primijenio bilo koji od postojećih metoda, neophodno je da se definišu set podataka i parametri za algoritam penjanja u pravcu gradijenata. Niz vrijednosti koje je metod predvidio preuzima se preko metode *predictOutputs*. Primjer za metod DirGCRF:

```
double alpha = 1;
double beta = 1;
double lr = 0.0001;
int maxIter = 100;
Parameters p = new Parameters(alpha, beta, maxIter, lr);

DirGCRF method = new DirGCRF(p, dataset);

double[] predictedOutputs = method.predictOutputs();
for (int i = 0; i < predictedOutputs.length; i++) {
      System.out.println(predictedOutputs[i]);
}
System.out.println("R^2: " + method.rSquared());
```

Prethodni kod je izvršio kreiranje i testiranje novog modela, koji je sada neophodno testirati. Predviđanje izlaza za testni set podataka vrši se preko metode *predictOutputsForTest* (kojoj se prosleđuju matrica *s* i niz *r* iz testnog seta), a koeficijent odlučnosti se računa preko metode *rSquaredForTest* (kojoj se prosleđuju izlazi koje je metod predvidio i očekivani izlazi).

```
double[] yTest = ArrayReader.readArray("data/y.txt");
double[] rTest = ArrayReader.readArray("data/r.txt");
double[][] sTest = GraphReader.readGraph
                                    ("data/s.txt", y.length);
```

```
double[] predictedOutputsTest = g1.predictOutputsForTest
                                                    (sTest, rTest);
for (int i = 0; i < predictedOutputsTest.length; i++) {
      System.out.println(predictedOutputsTest[i]);
}
System.out.println("R^2 Test: " +
                  g1.rSquaredForTest(predictedOutputsTest,yTest));
```

Takođe, postoji klasa *Basic* koja se može koristiti ukoliko korisnik želi ručno da specificira algoritam za učenje i pravila za izračunavanje, umjesto da koristi one koji su definisani u klasama GCRF i DirGCRF. Na primjer, sledeći kod će dati isti rezultat kao i prethodno dati kod:

```
// calculation rules
CalculationsDirGCRF c = new CalculationsDirGCRF(s, r);
// learning algorithm
GradientAscent g = new GradientAscent(p, c, y, false, null);

Basic method = new Basic(g, c, dataset);
double[] predictedOutputs = method.predictOutputs();
for (int i = 0; i < predictedOutputs.length; i++) {
      System.out.println(predictedOutputs[i]);
}
System.out.println("R^2: " + method.rSquared());
```

5.2.3. Mogućnost proširenja

Novi GCRF metod se može lako dodati nasleđivanjem postojećih klasa i redefinisanjem određenih metoda. Glavna razlika između GCRF metoda ogleda se u pravilima za izračunavanje, tako da je prvi korak da se naslijedi klasa *CalculationsGCRF* i da se redefinišu metode koje implementiraju proračune koji su izmjenjeni. Nakon toga se objekat nove klase za proračune može proslijediti klasi *Basic*, ili se može kreirati nova klasa koja nasleđuje klasu *Basic* i direktno u konstruktoru specificirati nova klasa za proračune. Takođe je moguće dodati novi algoritam za učenje, implementacijom

interfejsa *LearningRule*. U biblioteci se već nalazi jedno od proširenja GCRF-a (usmjereni GCRF - DirGCRF) koji može poslužiti kao dobar primjer za dodavanje novih GCRF metoda.

5.3. Poređenje sa postojećim rješenjima

Conditional Random Fields (CRF) se često koristi za predviđanje izlaznih varijabli među kojima postoji neka struktura. Originalno je dizajniran za klasifikaciju sekvencijalnih podataka, ali je našao primjenu u različitim oblastima, na primjer u oblasti:

- obrade prirodnog jezika (Lafferty, McCallum, & F., 2001).
- računarske vizije za klasifikaciju i označavanje regiona na statičkim slikama (Kumar & Hebert, 2006).
- računske biologije za predviđanje sekundarne strukture proteina (Liu, Carbonell, Klein-Seetharaman, & Gopalakrishnan, 2004).
- robotike za analizu pokreta i procjenu kretanja robotske ruke (Kim & Pavlović, 2009).

Postoji nekoliko softverskih biblioteka i alata koji nude implementaciju CRF-a u različitim programskim jezicima:

- **CRF package** [22] **–** implementacija CRF-a za označavanje sekvencijalnih podataka napisana u Javi.
- **Wapiti** [23] **-** alat za segmentiranje i označavanje sekvencijalnih podataka implementiran u C-u.
- **CRFsuite** [24]**, FlexCRFs** [25] **i CRF++** [26] **-** implementacija CRF-a za segmentiranje i označavanje sekvencijalnih podataka napisana u jeziku C++.

[22] http://crf.sourceforge.net/
[23] https://wapiti.limsi.fr/#features
[24] http://www.chokkan.org/software/crfsuite/
[25] http://flexcrfs.sourceforge.net/
[26] https://taku910.github.io/crfpp/

- **R CRF package** [27] **-** R paket koji sadrži alate za modelovanje i izračunavanje za CRF model i druge probabilističke modele za neusmjerene grafove sa diskretnim podacima.

- **UGM toolbox** [28] **-** kolekcija Matlab funkcija koje implementiraju CRF model i druge probabilističke modele za neusmjerene grafove sa diskretnim podacima.
- **PyStruct** [29] **-** Python biblioteka za strukturno učenje koja sadrži implementaciju CRF-a, ali ne podržava učenje preko funkcije maksimalne vjerodostojnosti

Za sve navedene projekte dostupna je dokumentacija koja objašnjava način primjene i za većinu je dostupan API i izvorni kod. U zavisnosti od programskog jezika i vrste problema koji rješavaju, istraživači mogu koristiti neku od ovih biblioteka kako bi implementirali CRF. Ne postoji biblioteka koja nudi implementaciju standardnog GCRF-a ili njegovih proširenja. Postoji samo biblioteka koja nudi Sparse GCRF model implementiran u Python-u (SGCRFpy [30]). U nekim radovima (Stojanović, J., Gligorijević, & Obradović, 2015), (Gligorijević, Stojanović, & Obradović, 2016) (u kojima su objavljene metode bazirane na GCRF-u) mogu se naći linkovi za repozitorijume na kojima su dostupni kodovi napisani u R-u i Matlab-u. Ovi kodovi su obično vezani za specifičnu primjenu i moraju biti modifikovani kako bi se mogli primijeniti na druge setove podataka.

U ovom odjeljku je predstavljena komparativna analiza biblioteke GCRFs i ostalih trenutno dostupnih biblioteka i softverskih paketa koji mogu biti korišćeni kao osnova za implementaciju GCRF-a. Prilikom analize biblioteka razmatrani su sledeći aspekti (tabela 12):

- **programski jezik –** u kom programskom jeziku je implementirano rješenje
- **dostupnost** – da li je programski kod dostupan (da li je u pitanju rješenje otvorenog koda)
- **CRF** - da li rješenje nudi implementaciju CRF-a

[27] https://github.com/wulingyun/CRF
[28] http://www.cs.ubc.ca/~schmidtm/Software/UGM.html
[29] https://pystruct.github.io/index.html
[30] https://github.com/dswah/sgcrfpy

- **GCRF** - da li rješenje nudi implementaciju GCRF-a
- **fleksibilnost** – da li se rješenje može koristiti za primjenu različitih modela na različitim setovima podataka
- **proširivost** - da li rješenje dozvoljava izmjene i nadogradnju

Tabela 12. Poređenje različitih biblioteka/alata za CRF i/ili GCRF

Rješenje	**Programski jezik**	**Otvoren kod**	**CRF**	**GCRF**	**Fleksibilnost**		**Proširivost**
					Setovi podataka	modeli	
CRF package	Java	da	da	ne	da	ne	da
Wapiti	C	da	da	ne	da	da	da
CRFsuite, FlexCRFs, CRF++	C++	da	da	ne	da	ne	da
R CRF package	R	da	da	ne	da	da	da
UGM toolbox	Matlab	da	da	ne	da	da	da
PyStruct	Python	ne	djelimično	ne	da	ne	No
SGCRFpy	Python	da	da	djelimično	da	ne	da
Kod iz radova	R/Matlab	djelimično	ne	da	ne	ne	da
GCRFs library	Java	da	ne	da	da	da	da

Iz tabele 12 se vidi da postojeće biblioteke podržavaju samo CRF i da se mogu koristiti za ograničen broj modela za različite setove podataka. Skoro sva analizirana rješenja su besplatna i otvorenog koda, što znači da ih je moguće mijenjati i proširivati. Jedini izuzetak je biblioteka PyStruct koja sadrži djelimičnu implementaciju CRF-a i njen kod nije javno dostupan. Ove biblioteke se mogu iskoristiti za implementaciju GCRF-a, ali u

tom slučaju se on mora prevesti u Gausovu raspodjelu vjerovatnoće, što zahtjeva promjenu metoda izračunavanja, za šta su neophodne napredne matematičke i programerske vještine. Takođe, GCRF ne koriti atribute čvorova direktno, već formira svoje atribute upotrebom nestrukturnih prediktora. Kodovi koji su dati u naučnim radovima odnose se na specifične modele i setove podataka, obično nijesu potpuni i da bi se upotrijebili neophodno je da se izvrši adaptacija koda. GCRFs biblioteka implemetira GCRF model i njegova proširenja i lako se može primijeniti na bilo koji set podataka, bez pisanja dodatnog koda. Novi GCRF modeli se mogu lako dodati nasleđivanjem postojećih klasa ili implementiranjem određenih metoda.

Vremenska kompleksnost standardnog GCRF-a je $O(TN^3)$ (Radosavljević, Vučetić, & Obradović, 2010), gdje je N broj čvorova, a T broj iteracija za treniranje. Vremenska kompleksnost CRF-a je O(TNQ2n) (Nikitinsky, 2016), gdje je N broj sekvenci, T broj iteracija za treniranje, Q broj klasnih oznaka, a n broj CRF atributa. Analiza performansi izvršena je za biblioteke koje su implementirane u Javi, R-u i Python-u. Testiranje je izvršeno na sintetički generisanim setovima podatka sa 100, 500 i 1000 čvorova, a maksimalan broj iteracija je bio 50. Iz rezultata koji su predstavljeni u tabeli 13 se vidi da je GCRFs biblioteka najsporija, čak i za graf od 500 čvorova, što je izazvano objektno-orjentisanom prirodom Java jezika.

Tabela 13. Poređenje vremena izvršavanja za Java, R i Python biblioteke

Rješenje	**Vrijeme u sekundama**		
	100 čvorova	**500 čvorova**	**1000 čvorova**
CRF package	0,131	0,321	0,644
R CRF package	2,75	14,23	20,81
PyStruct	4,05	16,92	31,76
SGCRFpy	0,067	0,098	0,391
GCRFs library	0,468	19,41	152,80

6. Primjeri primjene

6.1. Primjena DirGCRF

U realnim primjerima mnogi objekti su povezani usmjerenim vezama. Nad takvim grafovima se GCRF ne može direktno primijeniti, a transformacija grafa iz usmjerenog u neusmjereni dovodi do gubitka preciznosti. Stoga je proširenje GCRF-a za usmjerene grafove (DirGCRF) veoma značajan napredak za sve oblasti u kojima je potrebno primijeniti metod strukturne regresije na graf u kome je važan smjer veza. Primjer ovakvih grafova su društvene ili komunikacione mreže, u kojima često postoje jednosmjerne relacije.

Kao konkretan primjer dat je set podataka pod nazivom „Glasgow“:

- **Cilj:** Predvidjeti da li će đaci konzumirati cigarete.
- **Opis problema:** Dostupni su podaci o đacima iz jedne srednje škole u Glazgovu, koji su prikupljeni kroz tri periodična ispitivanja u toku dvije godine. Za svakog đaka postoje informacije o njegovim životnim navikama i poznati su njegovi najbolji prijatelji. Čvor u grafu predstavlja jednog đaka, a đaci međusobno povezani na osnovu informacija o prijateljstvu.
- **Podaci:**

  Broj vremenskih tačaka: 3

  Broj čvorova: 129

  Veza u grafu: prijateljstvo (đaci su birali do 6 prijatelja i trebali su da ih ocjene ocjenom od 0 do 2, na sledeći način: 1 - najbolji prijatelj, 2 - samo prijatelj, 0 - nije prijatelj)

  Za svaki čvor dati su sledeći podaci:

  - Konzumiranje alkohola (od 1 do 5)
  - Konzumiranje kanabisa (od 1 do 4)
  - Ljubavna veza (označava da li je osoba u vezi ili ne)
  - Iznos mjesečnog džeparca
- **Rezultati:** S obzirom da je u pitanju usmjereni graf na ovaj problem se može primijeniti samo DirGCRF (DirectedGCRF), jer je to jedini GCRF model koji podržava asimetričnu sličnost između objekata. DirGCRF je postigao preciznost

od 34% i povećao preciznost standardnog GCRF-a za 5%, neuronske mreže za 3%. Detaljni rezultati su predstavljeni u odjeljku 4.4.2 , na slici 9.

6.2. Primjena GCRF GUI TOOL-a

Kako bi se na nekom problemu mogle primijeniti metode strukturne regresije potrebno je da se problem može predstaviti u formi grafa, na primjer:

- mreže prijateljstva
- društvene mreže (Facebook, Twitter, Instagram, GitHub itd.)
- prostorne/geografske mreže
- transportne mreže (putevi, željezničke veze, avio veze itd.)
- komunikacione mreže (email, poruke, pozivi itd.)

Primjeri problema koji su riješeni primjenom metoda baziranih na GCRF-u:

- Predviđanje dnevnih troškova bolničkog liječenja u nekoj bolnici na osnovu podatka o pacijentima i grafa u kom su veze između bolnica bazirane na sličnosti njihovih specijalizacija (Polychronopoulou & Obradović, 2014).
- Predviđanje broja citata za naučni rad u narednom periodu na osnovu informacije o trenutnom broju citata i grafa u kom su veze između radova kreirane na osnovu sličnosti radova koji ih citiraju (Radosavljević, Vučetić, & Obradović, 2014).
- Predviđanje da li će tinejdžer konzumirati cigarete, na osnovu informacija o njegovim životnim navikama i grafa u kom su tinejdžeri povezani sa svojim najbližim prijateljima (Vujičić, Glass, Zhou, & Obradović, 2017).
- Predviđanje padavina na osnovu klimatskih varijabli i grafa u kom je prostorna udaljenost iskorišćena da se izračuna sličnost između lokacija (Stojanović, J., Gligorijević, & Obradović, 2015).

Kao konkretan primjer dat je set podataka pod nazivom „Precipitation“:

- **Cilj:** Predviđanje padavina na određenoj lokaciji.
- **Opis problema:** Dostupni su podaci za različite meteorološke stanice u kontinentalnom dijelu SAD-a. Čvor u grafu predstavlja meteorološku stanicu, a one su međusobno povezane na osnovu prostorne udaljenosti.

- **Podaci:**

  Broj vremenskih tačaka: svaki mjesec od januara 1948 do danas

  Broj čvorova: 1218

  Veza u grafu: Prostorna udaljenost

  Za svaki čvor dati su sledeći podaci:

    - padavine
    - omega (Lagranžova tendencija vazdušnog pritiska)
    - vodena para
    - relativna vlažnost
    - temperatura
    - komponente vjetra (u,v)
- **Rezultati:** S obzirom da je u pitanju djelimično posmatrani vremenski graf i da ima podataka koji nedostaju na ovaj set podataka je najbolje primijeniti m-GCRF (Marginalized GCRF) koji rješava problem nedostatka podataka u vremenskim grafovima. m-GCRF je postigao preciznost od 61% i povećao preciznost neuronske mreže za 5%. Detaljni rezultati dati su u (Stojanović, J., Gligorijević, & Obradović, 2015).

6.3. Primjena GCRFs biblioteke

GCRF GUI TOOL se može koristiti za eksperimentisanje i poređenje performansi postojećih GCRF metoda, dok GCRFs biblioteka omogućava laku implementaciju postojećih, ali i jednostavno kreiranje novih GCRF metoda. Može se primijeniti na svim navedenim primjerima, samo što se proces treniranja i testiranje ne odvija kroz alat (preko korisničkog interfejsa) već je potrebno pisati kod.

Kao primjer upotrebe biblioteke za primjenu modela na realnim podacima i poređenje performansi različitih GCRF metoda može se uzeti „Glasgow“ set podataka koji je

opisan u odjeljku 6.1. Kako bi se upotrebom GCRFs biblioteke DirGCRF metod primijenio na ovaj set podataka potrebno je:

1. Kreirati objekat klase *DataSet* i učitati podatke za trening iz unaprijed pripremljenih tekstualnih fajlova. Da bi se uspješno kreirao *DataSet* objekat neophodno je da se definiše niz koji sadrži vrijednosti koje je predvidio nestrukturni prediktor (r). S obzirom da biblioteka ne obezbjeđuje podršku za rad sa nestrukturnim prediktorima, potrebno je koristiti neku drugu biblioteku. U ovom primjeru korišćena je biblioteka Neuroph, pomoću koje je kreirana neuronska mreža koja se koristi da se dobije predviđanje. Ova neuronska mreža ima četiri ulazna i jedan izlazni neuron.
2. Kreirati objekat klase *Parameters* i u konstruktoru proslijediti sve neophodne parametre.
3. Kreirati objekat klase *DirGCRF* i u konstruktoru proslijediti objekte koji su kreirani u prethodna dva koraka (objekte klasa *DataSet i Parameters*).
4. Učitati podatke za trening iz unaprijed pripremljenih tekstualnih fajlova i preko prethodno kreirane neuronske mreže izračunati r niz za testne podatke.
5. Metodi *predictOutputs* proslijediti matricu sličnosti (s) i niz koji je predvidjela neuronska mreža (r) iz testnih podatka.
6. Metodi *rSquaredForTest* proslijediti očekivane i predviđene izlaze, kako bi se dobila preciznost modela za testne podatke. Izračunata preciznost se može prikazati u konzoli:

```
double[] yPredicted  = d.predictOutputs(sTest, rTest);
double[] yTest = ArrayReader.readArray("data/yTest.txt");
System.out.println("R^2:" +
                d.rSquaredForTest(yPredicted, yTest));
```

Rezultat je:

```
R^2: 0.34253288143247153
```

Naravno, preciznost zavisi od parametara i promjena njihovih vrijednosti dovešće do različitih rezultata.

Na isti način se, u cilju poređenja performansi može primijeniti i standardni GCRF, upotrebom klase *GCRF*. Da bi ovo bilo moguće neophodno je prebaciti matricu sličnosti u simetričnu (upotrebom statičke metode *converteGraphToUndirected iz klase GraphReader*).

Implementacija metoda DirGCRF (koja se već nalazi u GCRFs biblioteci) predstavlja dobar primjer za proširenje biblioteke novim GCRF modelom. Kreirane su nove klase *CalculationsDirGCRF* i *DirGCRF* koje nasleđuju osnovne klase i redefinišu određene metode i tako kreiraju novi model, koji se može primijeniti na usmjerene grafove.

6.4. Primjena u nastavi

GCRF GUI TOOL-a i GCRFs biblioteka mogu se koristi u nastavi na fakultetu, na predmetima iz oblasti mašinskog učenja. Evaluacija alata sa studentima je pokazala da ga smatraju jednostavnim za korišćenje, što otvara mogućnosti za primjenu u nastavi, jer jednostavan grafički interfejs omogućava studentima da primjene metode strukturne regresije za rješavanje konkretnih problema, prije nego što u potpunosti savladaju sve teorijske aspekte. Nakon što savladaju osnove, mogu pokušati da integrišu metode u svoj kod upotrebom Java biblioteke. Takođe, raspoloživost programskog koda omogućava studentima da analiziraju kako su implementirani kompleksni proračuni i algoritmi, i tako uče na praktičnom primjeru.

Na primjer, kako bi se problem strukturne regresije približio studentima, moguće je preuzeti njihove podatke iz studentske službe (pol, godinu rođenja, prosjek iz srednje škole, prosjek sa prethodnih godina studija itd.). Nakon toga studenti treba da označe svoje najbolje prijatelje, kako bi se kreirao graf. Cilj je da se predvidi prosjek za svakog studenta. Prvo je potrebno predvidjeti prosjek za prethodnu godinu studija, kako bi se provjerila preciznost modela, a nakon toga i prosjek za tekuću godinu.

Studenti treba da sprovedu sledeće korake:

1. Pretprocesiranje podataka softverskim putem – učitavanje podataka, obrada i kreiranje tekstualnih fajlova u traženom formatu.
2. Primjena neuronske mreže na podatke dobijene iz studentske službe, kako bi se vidjela preciznost predikcije kada se ne uzima u obzir graf prijateljstva.

3. Primjena različitih GCRF modela upotrebom GCRF GUI TOOL-a i poređenje rezultata.
4. Primjena različitih GCRF modela upotrebom GCRFs biblioteke i poređenje rezultata.
5. Integracija modela koji je dao najbolje rezultate u neko softversko rješenje (web ili mobilnu aplikaciju).

7. Diskusija

7.1. Analiza prednosti i nedostataka

Istraživanje koje je predstavljeno u ovom radu obuhvata tri glavna doprinosa:

- DirGCRF - GCRF metod koji je primjenjiv na usmjerene grafove.
- GCRF GUI TOOL - softver koji integriše različite vrste GCRF metoda
- GCRFs biblioteka - Java biblioteka za metode bazirane na GCRF-u.

Za svaki od navedenih doprinosa postoje određene prednosti i nedostaci, koji su predstavljeni u tabeli 14.

Tabela 14. Prednosti i nedostaci doprinosa koji su predstavljeni u radu

Doprinos	Prednosti	Nedostaci
DirGCRF	● GCRF metod koji se može primijeniti na usmjerene grafove ● preciznije predviđanje za usmjerene grafove od standardnog GCRF-a ● preciznije predviđanje za usmjerene grafove od tradicionalnih nestrukturnih prediktora	● konveksnost modela je moguće pokazati samo empirijski ● sporiji je od dosadašnjih implementacija GCRF metoda (zbog objektno-orjentisane prirode Java jezika)

GCRF GUI TOOL	● softver otvorenog koda koji je svima dostupan ● jednostavna instalacija i konfiguracija ● jednostavno treniranje i testiranje različitih GCRF metoda ● korisnički interfejs koji je lak za korišćenje ● korisnici različitih profila su pozitivno ocijenili softver ● nije neophodno da se poznaje način rada određenih metoda da bi se one praktično primijenile	● za 4 od 6 metoda korisnici moraju imati instaliran Matlab kako bi mogli da ih primjene ● za veće setove podataka potrebno je imati računar sa više RAM memorije ● potrebno je pripremiti podatke (pretprocesiranje) ● korisnička uputstva nijesu dovoljno detaljna za početnike
GCRFs biblioteka	● biblioteka otvorenog koda koji je svima dostupan ● sadrži osnovne koncepte za implementaciju metoda baziranih na GCRF ● ima intuitivan i jednostavan programski interfejs (API) ● omogućava laku implementaciju GCRF metoda u Java kodu ● fleksibilna je i omogućava laku implementaciju proširenja GCRF-a	● ne sadrži implementaciju nestrukturnih prediktora

7.2. Pravci daljeg razvoja

Ideje za dalji razvoj i unaprjeđenje istraživanja koje je predstavljeno u ovom radu:

- Implementacija DirGCRF modela u funkcionalnom ili proceduralnom jeziku kako bi se ubrzao i postao efikasniji za veće setove podataka. Za početak je planirano je da se DirGCRF model implementira u funkcionalnom jeziku Clojure i da se vidi kako će to uticati na performanse modela. Ukoliko dovede do značajnog poboljšanja svi modeli se mogu prevesti u Clojure, jer je on hostovan na JVM-u (Java Virtual Machine) i može se ugraditi u Java programe.
- Primjena DirGCRF-a na što više različitih realnih setova podataka, kao i setova podataka koji sadrže više grafova (više β parametara) i na koje se može primijeniti više nestrukturnih prediktora (više α parametara).
- Kreiranje video tutorijala i studija slučaja koji će pomoći korisnicima da bolje razumiju koncept GCRF metoda. Tutorijal bi trebao da pokriva cijeli proces, od prezentacije problema (koji realni problem se rješava, koji podaci su dostupni i šta je cilj), preko odabira metoda (koji od dostupnih GCRF metoda je najbolji za konkretni problem i zašto) do finalnog rješenja problema upotrebom GCRF GUI TOOL-a (ili GCRFs biblioteke, za napredne korisnike).
- Unaprjeđenje softvera GCRF GUI TOOL u skladu sa komentarima korisnika. Na primjer, potrebno je poboljšati uputstva za korisnike i pojednostaviti pojedine opcije.
- Integracija novih modela baziranih na GCRF-u u softver GCRF GUI TOOL. S obzirom da se često pojavljuju nova proširenja GCRF modela potrebno ih je integrisati u GCRF GUI TOOL i objavljivati nove verzije alata, kako bi korisnici u svakom trenutku imali dostupne najnovije modele.
- Integracija komponenti za vizuelizaciju podataka, procesa treniranja i rezultata metoda u softver GCRF GUI TOOL. Setovi podataka za GCRF modele su grafovi i za njihovu vizuelizaciju se može koristiti Java biblioteka GraphStream [31]. Rezultati procesa treniranja i testiranja mogu se predstaviti preko dijagrama i

[31] http://graphstream-project.org/

grafikona, za čije je generisanje planirana upotreba biblioteke JFreeChart [32]. Takođe, potrebno je omogućiti eksport ovih grafikona u različite formate (jpg, png, eps itd.).

- Ubrzanje procesa treniranja metoda u okviru GCRF GUI TOOL-a. To se može postići prebacivanjem matematičkih proračuna u funkcionalni programski jezik, čime bi se i smanjila količina potrebne memorije (pogotovo za veće setove podataka). Funkcije bi se i dalje pozivale iz Jave (kao što je prethodno rečeno Clojure kod se lako može ugraditi u Java programe), tako da ne bi bila potrebna izmjena korisničkog interfejsa.
- Integracija novih modela baziranih na GCRF-u u Java biblioteku GCRFs. Trenutno biblioteka sadrži dva GCRF modela (standardni i usmjereni), koji su izvorno implemetirani u Javi. Plan je da se ostali modeli koji su obuhvaćeni GCRF GUI TOOL-om (trenutno implementirani u Matlab-u) prevedu u Javu, kako bi se mogli integrisati u GCRFs biblioteku.
- Razvoj alata za rad sa setovima podataka koji obezbjeđuje unos podataka različitih formata i njihovo konvertovanje u format koji je odgovarajući za primjenu GCRF metoda. Alat bi trebao da omogući dostavljanje podataka u xls, csv, xml ili json formatu, kao i da pruži podršku za čitanje zapisa iz baze podataka. Potrebno je omogućiti da alat može da se koristi nezavisno (da se nakon konverzije preuzmu podaci u txt formatu, koji se kasnije mogu proslijediti GCRF GUI TOOL softveru ili metodama iz GCRFs biblioteke), kao i da se može povezati sa GCRF GUI TOOL-om (da se nakon procesa prilagođavanja set podataka automatski doda u listu dostupnih setova GCRF GUI TOOL-a).

[32] http://www.jfree.org/jfreechart/

8. Zaključak

Istraživanje izloženo u ovom radu imalo je za cilj proširenje GCRF modela za usmjerene grafove (DirGCRF) i razvoj softverskog alata za ispitivanje algoritama strukturne regresije bazirane na GCRF modelu.

U uvodom dijelu rada izvršena je analiza oblasti mašinskog učenja i strukturne regresije, standardnog GCRF modela i njegovih proširenja.

Nakon toga je detaljno predstavljen predloženi DirGCRF model, prezentovan je matematički model i njegova implementacija, kao i performanse modela na sintetičkim i realnim podacima. Rezultati testiranja dokazali su da DirGCRF daje tačnija predviđanja od standardnog GCRF modela: od 5% do 19% za realne setove podataka, i u prosjeku 30% za sintetičke setove podataka. Ukoliko je u nekom setu podataka uticaj strukture (grafa) jači od uticaja nestrukturnih prediktora (npr. neuronskih mreža), dolazi čak i do dupliranja preciznosti u odnosu na standardni GCRF.

Rad obuhvata i prezentaciju alata GCRF GUI TOOL, softvera otvorenog koda koji integriše različite GCRF modele i omogućava njihovu primjenu na setove podataka iz različnih oblasti. Softver je razvijen u Java programskom jeziku, upotrebom savremenih metoda i tehnologija, a dati su i detalji implementacije. Glavne funkcionalnosti ovog softvera su detaljno predstavljene i demonstrirane kroz dvije studije slučaja. S obzirom da je veoma važno da GCRF GUI TOOL bude lak i jednostavan za korišćenje za eksperte i početnike, izvršena je evaluacija softvera sa različitim grupama korisnika. Rezultati evaluacije su pokazali da su obje grupe korisnika veoma zadovoljne softverom, a njihovi predlozi i sugestije su veoma pomogli prilikom planiranja budućeg razvoja.

Takođe je predstavljena i Java biblioteka GCRFs, koja sadrži osnovne koncepte za implementaciju metoda baziranih na GCRF, kao i implementaciju dvije konkretne metode (standardnog i usmjerenog GCRF-a). Dati su detalji implementacije ove biblioteke, predstavljen je način upotrebe biblioteke i objašnjeno je kako može biti proširena.

U uvodnom dijelu rada predstavljene su četiri polazne hipoteze i svaka od njih je potvrđena:

- **H1 - Postojeća proširenja GCRF modela se ne mogu primijeniti na usmjerene grafove**, što je potvrđeno analizom dostupne literature i objavljenih radova koji prikazuju GCRF modele i njihovu primjenu u praksi.
- **H2 - Predloženo proširenje GCRF modela za usmjerene grafove omogućava preciznije predviđanje od standardnog GCRF modela i tradicionalnih nestrukturnih prediktora**, što je potvrđeno testiranjem predloženog metoda na sintetičkim i realnim podacima.
- **H3 - Softverski alat pojednostavljuje treniranje i testiranje GCRF modela i njegovih proširenja na različitim setovima podataka**, što potvrđuju rezultati evaluacije i mišljenja korisnika-eksperata.
- **H4 - Softverski alat koji nudi implementaciju standardnog GCRF modela i njegovih proširenja mogu koristiti kako i eksperti u oblasti mašinskog učenja**, tako i početnici kojima GCRF model može pomoći da dođu do željenih informacija, što potvrđuju rezultati evaluacije iz kojih se vidi da su obje vrste korisnika imale slično iskustvo sa softverom i da su ga ocijenile na sličan način.

Literatura

Anselin, L. (1988). *Spatial Econometrics: Methods and Models.* Kluwer Academic.

Baklr, G. (2007). *Predicting Structured Data.* MIT Press.

Beguerisse-Díaz, M., Garduno-Hernández, G., Vangelov, B., Yaliraki, S. N., & Barahona, M. (2014). Interest communities and flow roles in directed networks: The twitter network of the UK riots. *Journal of the Royal Society Interface, 11 (101)*, 20140940. doi:10.1098/rsif.2014.0940

Bo, L., & Sminchisescu, C. (2009). Twin Gaussian Processes for Structured Prediction. *International Journal of Computer Vision, 87 (1-2)*, 28-52. doi:10.1007/s11263-008-0204-y

Brooke, J. (1996). SUS-A quick and dirty usability scale. *189 (194)*, 4-7.

Bush, H., West, P., & Michell, L. (1997). *The role of friendship groups in the uptake and maintenance of smoking amongst pre-adolescent and adolescent children: Distribution of frequencies.* MRC Medical Sociology Unit Glasgow.

Đurić, N., Radosavljević, V., Obradović, Z., & Vučetić, S. (2015). Gaussian conditional random fields for aggregation of operational aerosol retrievals. *IEEE Geoscience and Remote Sensing Letters, 12 (4),* 761–765.

Glass, J., & Obradović, Z. (2017). Structured Regression on Multi-Scale Networks. *IEEE Intelligent Systems, 32 (2)*(2), 23-30. doi:10.1109/MIS.2017.37

Glass, J., Ghalwash, M., Vukićević, M., & Obradović, Z. (2016). Extending the modelling capacity of gaussian conditional random fields while learning faster. *Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)*, 1596-1602.

Gligorijević, D., Stojanović, J., & Obradović, Z. (2015). Improving confidence while predicting trends in temporal disease networks. *4th Workshop on Data Mining for Medicine and Healthcare, 2015 SIAM International Conference on Data Mining*, 1-10.

Gligorijević, D., Stojanović, J., & Obradović, Z. (2016). Uncertainty propagation in long-term structured regression on evolving networks. *Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)*, 1603-1610.

Guo, H. (2013). Modeling short-term energy load with continuous conditional random fields. *European conference on machine learning and principles and practice of knowledge discovery in databases (ECML/PKDD)*, 433-448.

Han, C., Zhang, S., Ghalwash, M., Vučetić, S., & Obradović, Z. (2016). Joint learning of representation and structure for sparse regression on graphs. *SIAM International Conference on Data Mining*, 846-854.

Haykin, S. (2009). *Neural networks and learning machines (Vol. 3).* Upper Saddle River Pearson.

HCUP. (2005-2009). Healthcare Cost and Utilization Project, HCUP State Inpatient Databases. Agency for Healthcare Research and Quality, Rockville.

Khorram, S., Bahmaninezhad, F., & Sameti, H. (2014). Speech synthesis based on Gaussian conditional random fields. *International Symposium on Artificial Intelligence and Signal Processing, 427*, 183-193. doi:10.1007/978-3-319-10849-0_19

Kim, M., & Pavlović, V. (2009). Discriminative learning for dynamic state prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 31 (10)*, 1847-1861. doi:10.1109/TPAMI.2009.37

Kumar, S., & Hebert, M. (2006). Discriminative Random Fields. *International Journal of Computer Vision, vol. 68, no. 2, pp. 179-201, 68 (2)*, 179-201. doi:10.1007/s11263-006-7007-9

Lafferty, J., McCallum, A., & F., P. (2001). Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. *18th International Conference on Machine Learning, 18*, 282-289.

Lafferty, J., McCallum, A., & Pereira, F. (2001). Conditional Random Fields:Probabilistic Models for Segmenting and Labeling Sequence Data. *18th International Conference on Machine Learning*, 282-289.

Liu, C., Adelson, E. H., & Freeman, W. T. (2007). Learning Gaussian conditional random fields for low-level vision. *IEEE Conference on Computer Vision and Pattern Recognition*, 79-86.

Liu, Y., Carbonell, J., Klein-Seetharaman, J., & Gopalakrishnan, V. (2004). Comparison of probabilistic combination methods for protein secondary structure prediction. *Bioinformatics, 20 (17),* 3099-3107. doi:10.1093/bioinformatics/bth370

Menne, M. J., Williams, C. N., & Vose, R. S. (2009). The US historical climatology network monthly temperature data. *Bulletin of the American Meteorological Society, 90 (7),* 993-1007. doi:10.1175/2008BAMS2613.1

Michell, L., & Amos, A. (1997). Girls, pecking order and smoking. *Social Science & Medicine, 44 (12),* 1861–1869. doi:10.1016/S0277-9536(96)00295-X

Mohri, M., Rostamizadeh, A., & Talwalkar, A. (2012). *Foundations of Machine Learning.* The MIT Press.

Nielsen, J. (2012). Usability 101: Introduction to usability. Preuzeto Januar 20, 2018 sa https://www.nngroup.com/articles/usability-101-introduction-to-usability/

Nikitinsky, N. (2016). Conditional Random Fields (CRF): Short Survey. Preuzeto Januar 20, 2018 sa http://nlpx.net/archives/464

Polychronopoulou, A., & Obradović, Z. (2014). Hospital pricing estimation by gaussian conditional random fields based regression on graphs. *IEEE international conference on bioinformatics and biomedicine (BIBM),* 564–567.

Qian, X., Ukkusuri, S. V., Yang, C., & Yan, F. (2017). Forecasting short-term taxi demand using boosting-GCRF. *The 6th International Workshop on Urban Computing (UrbComp 2017).*

Radosavljević, V., Vučetić, S., & Obradović, Z. (2010). Continuous conditional random fields for regression in remote sensing. *19th European Conference on Artificial Intelligence* , 809–814.

Radosavljević, V., Vučetić, S., & Obradović, Z. (2014). Neural Gaussian conditional random fields. *Joint European conference on machine learning and knowledge discovery in databases*, 614-629.

Rubin, J., & Chisnell, D. (2008). *Handbook of usability testing: howto plan, design, and conduct effective tests.* John Wiley & Sons.

Samuel, A. (1959). Some Studies in Machine Learning Using the Game of Checkers. *IBM Journal of Research and Development, 3 (3),* 210-229. doi:10.1147/rd.33.0210

Sedgewick, R., & Wayne, K. (2017). *Introduction to Programming in Java: An Interdisciplinary Approach.* Addison-Wesley Professional.

Shalev-Shwartz, S., & S., B.-D. (2014). *Understanding Machine Learning: From Theory to Algorithms.* Cambridge University Press.

Snijders, T. A., Van de Bunt, G. G., & Steglich, C. E. (2010). Introduction to stochastic actor-based models for network dynamics. *Social Networks, 32 (1),* 44–60. doi:10.1016/j.socnet.2009.02.004

Solberg, A. H., Taxt, T., & Jain, A. K. (1996.). A Markov random field model for classification of multisource satellite imagery. *IEEE Transactions on Geoscience and Remote Sensing, 34 (1),* 100-113. doi:10.1109/36.481897

Stojanović, J., J. M., Gligorijević, D., & Obradović, Z. (2015). Semi-supervised learning for structured regression on partially observed attributed graphs. *SIAM International Conference on Data Mining*, 217-225.

Stojković, I., Jelisavčić, V., Milutinović, V., & Obradović, Z. (2017). Fast Sparse Gaussian Markov Random Fields Learning Based on Cholesky Factorization. *26th International Joint Conference on Artificial Intelligence (IJCAI)*, 2758 - 2764.

Šćepanovic, S., Vujičić, T., Matijević, T., & Radunović, P. (2015). Game based mobile learning—Application development and evaluation. *6th conference on e-learning*, 142–147.

Ševarac, Z. (2008). *Neuroph Java framework za neuronske mreže*. Preuzeto Januar 20, 2018 sa Neuroph: http://neuroph.sourceforge.net

Vujičić, T., Glass, J., Zhou, F., & Obradović, Z. (2017). Gaussian Conditional Random Fields Extended for Directed Graphs. *Machine Learning, 106 (9-10)*, 1271–1288. doi:10.1007/s10994-016-5611-7

Weisberg, S. (2005). *Applied linear regression (Vol. 528).* John Wiley & Sons.

Wytock, M., & Kolter, J. Z. (2013). Sparse Gaussian conditional random fields: Algorithms, theory, and application to energy forecasting. *International conference on machine learning*, 1265-1273.

Zhou, X. Z., Menche, J., Barabasi, A.-L., & Sharma, A. (2014). Human symptoms-disease network. *Nature Communications, 5*, 4212. doi:10.1038/ncomms5212

Изјава о ауторству

Име и презиме аутора Тијана Вујичић

Број индекса 5023/2015

Изјављујем

да је докторска дисертација под насловом

Софтверски алат за испитивање алгоритама структурне регресије базиране на GCRF моделу

- резултат сопственог истраживачког рада;
- да дисертација у целини ни у деловима није била предложена за стицање друге дипломе према студијским програмима других високошколских установа;
- да су резултати коректно наведени и
- да нисам кршио/ла ауторска права и користио/ла интелектуалну својину других лица.

Потпис аутора

У Београду, _____

_____

Изјава о истоветности штампане и електронске верзије докторског рада

Име и презиме аутора Тијана Вујичић

Број индекса 5023/2015

Студијски програм Информациони системи и квантитативни менаџмент

Наслов рада Софтверски алат за испитивање алгоритама структурне регресије базиране на GCRF моделу

Ментор проф. др Владан Девеџић, редовни професор Факултета организационих наука

Изјављујем да је штампана верзија мог докторског рада истоветна електронској верзији коју сам предао/ла ради похрањена у **Дигиталном репозиторијуму Универзитета у Београду.**

Дозвољавам да се објаве моји лични подаци везани за добијање академског назива доктора наука, као што су име и презиме, година и место рођења и датум одбране рада.

Ови лични подаци могу се објавити на мрежним страницама дигиталне библиотеке, у електронском каталогу и у публикацијама Универзитета у Београду.

Потпис аутора

У Београду, _____

_____

Изјава о коришћењу

Овлашћујем Универзитетску библиотеку „Светозар Марковић“ да у Дигитални репозиторијум Универзитета у Београду унесе моју докторску дисертацију под насловом:

Софтверски алат за испитивање алгоритама структурне регресије базиране на GCRF моделу

која је моје ауторско дело.

Дисертацију са свим прилозима предао/ла сам у електронском формату погодном за трајно архивирање.

Моју докторску дисертацију похрањену у Дигиталном репозиторијуму Универзитета у Београду и доступну у отвореном приступу могу да користе сви који поштују одредбе садржане у одабраном типу лиценце Креативне заједнице (Creative Commons) за коју сам се одлучио/ла.

1. Ауторство (CC BY)

2. Ауторство – некомерцијално (CC BY-NC)

3. Ауторство – некомерцијално – без прерада (CC BY-NC-ND)

4. Ауторство – некомерцијално – делити под истим условима (CC BY-NC-SA)

5. Ауторство – без прерада (CC BY-ND)

6. Ауторство – делити под истим условима (CC BY-SA)

(Молимо да заокружите само једну од шест понуђених лиценци.
Кратак опис лиценци је саставни део ове изјаве).

Потпис аутора

У Београду, _____

_____

1. **Ауторство**. Дозвољавате умножавање, дистрибуцију и јавно саопштавање дела, и прераде, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце, чак и у комерцијалне сврхе. Ово је најслободнија од свих лиценци.

2. **Ауторство – некомерцијално**. Дозвољавате умножавање, дистрибуцију и јавно саопштавање дела, и прераде, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце. Ова лиценца не дозвољава комерцијалну употребу дела.

3. **Ауторство – некомерцијално – без прерада**. Дозвољавате умножавање, дистрибуцију и јавно саопштавање дела, без промена, преобликовања или употребе дела у свом делу, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце. Ова лиценца не дозвољава комерцијалну употребу дела. У односу на све остале лиценце, овом лиценцом се ограничава највећи обим права коришћења дела.

4. **Ауторство – некомерцијално – делити под истим условима**. Дозвољавате умножавање, дистрибуцију и јавно саопштавање дела, и прераде, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце и ако се прерада дистрибуира под истом или сличном лиценцом. Ова лиценца не дозвољава комерцијалну употребу дела и прерада.

5. **Ауторство – без прерада**. Дозвољавате умножавање, дистрибуцију и јавно саопштавање дела, без промена, преобликовања или употребе дела у свом делу, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце. Ова лиценца дозвољава комерцијалну употребу дела.

6. **Ауторство – делити под истим условима**. Дозвољавате умножавање, дистрибуцију и јавно саопштавање дела, и прераде, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце и ако се прерада дистрибуира под истом или сличном лиценцом. Ова лиценца дозвољава комерцијалну употребу дела и прерада. Слична је софтверским лиценцама, односно лиценцама отвореног кода.