PRIMJENA STROJNOG UČENJA NA PREDVIĐANJE PREŽIVLJAVANJA TITANIC-A

Vedran Ciganović, Eduard Kalčićek,
Mihael Končić & Valentino Novak

Lipanj 2018.

SADRŽAJ

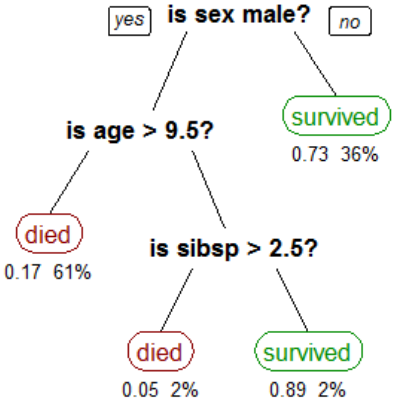
UVOD

Potonuće Titanica najpoznatija je morska tragedija u povijesti čovječanstva, ali također i jedan od najpoznatijih primjera korištenih u strojnom učenju, tj. u *data science*. Titanic je iz britanske luke Southampton na svoje prvo i jedino prekooceansko putovanje krenuo 10. travnja 1912. godine. U nastojanju da osvoji Plavu vrpcu Atlantika na svom prvom putovanju u New York, kapetan Edward John Smith namjerno je krenuo kraćim kursom, sjevernijim od propisanoga te je time zanemario opasnost od sudara s ledenim santama. Samo četiri dana kasnije, 14. travnja 1912. godine u 23:40 sati, Titanic se pri brzini od 22 čvora sudario sa santom leda, koja mu je probila oplatu, nakon čega je u 2:40 sati brod potonuo. Od ukupno 2224 putnika i članova posade koji su se u to vrijeme nalazili na brodu, život je izgubilo njih 1502. Jedan od najvećih razloga za gubitak tolikog broja života bio je nedostatak čamaca za spašavanje. Iako je ulogu u preživljavanju tragedije jednim dijelom igrala sreća, neki su ljudi poput žena, djece i bogate klase imali veće izglede za preživljavanje od drugih.

U ovom seminaru krenut ćemo od podataka o putnicima kao što su spol / dob / klasa te ćemo na temelju tih podataka alatima koji se koriste u strojnom učenju pokušati predvidjeti koji su putnici preživjeli. Dani podaci nisu potpuni - „praznine“ u podacima za učenje ručno smo popunili. Primjerice, za putnike čija starost nije poznata uzeli smo medijan godina svih poznatih putnika (28), dok smo za broj kabine uzeli vrijednost `Z0`. „Predviđanje“ smo vršili na 3 načina. U prvom i najjednostavnijem rekli smo da će žene i djeca preživjeti. U drugom smo koristili *random forest* algoritam s dvije ulazne varijable (dob i spol), a u trećem s njih 7. U nastavku seminara nešto ćemo više reći o korištenim metodama i dobivenim rezultatima.

STABLO ODLUKE

Stablo odlučivanja (engl. *decision tree*) je metoda često korištena u *data mining*–u i strojnom učenju. Cilj metode je napraviti model koji predviđa vrijednost tražene (*output*) varijable Y obzirom na nekoliko *input* varijabli $X = (x_1, x_2, x_3, \ldots x_k,)$.



Slika 1: Primjer stabla odlučivanja

Unutarnji čvorovi stabla predstavljaju *input* varijable, dok listovi predstavljaju vrijednost *output* varijable. Iz stabla sa slike 1 možemo zaključiti primjerice kolike su šanse za preživljavanje osoba koje su a) žene; b) dječaci ispod 9.5 godina s manjom obitelji.

Stabla odlučivanja koriste se za probleme klasifikacije i regresije. Ovdje pretpostavljamo da su *input* varijable dane na konačnoj diskretnoj domeni. Svaki element domene zovemo klasa. Stablo možemo „učiti“ tako da dane informacije - *input* varijable - rekurzivno dijelimo na podskupove dok ne dođemo do određenog kriterija zaustavljanja. Tada dani čvor pretvaramo u list, te svaku novu osobu koje ima dane atribute klasificiramo kao preživjelu.

Glavna poteškoća konstrukcije stabla odlučivanja je odlučiti koji atribut postaviti kao korijen stabla te kada čvor pretvoriti u list. Postoji više različitih metoda grananja stabla. Ovdje ćemo objasniti jedan matematički jednostavan te efektivan pristup.

GINI IMPURITY

Čvorovi obično neće klasificirati podatke sa 100% vjerojatnošću. U našem primjeru vjerojatnost preživljavanja ženske osobe znosi 36%. *Gini impurity* je mjera koliko često će nasumično izabran element iz skupa biti krivo klasificiran kada bismo ga nasumično klasificirali prema distribuciji u podskupu. *Gini impurity* računamo tako da sumiramo vjerojatnost P_i da smo izabrali element i iz klase, pomnoženo s vjerojatnosti $\sum_{k \neq i} P_k = 1 - P_i$ da smo element pogrešno klasificirali. Za skup s J klasa $\{i = 1, 2, \dots J\}$, formula se svodi na

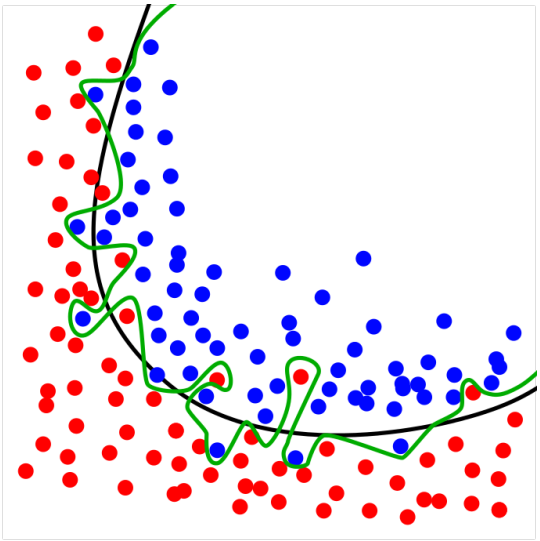
$$I_G(p) = 1 - \sum_{i=1}^{J} {P_i}^2.$$

Gini impurity poprima minimum (vrijednost nula) kada svi slučajevi čvora padaju u jednu kategoriju. Primjerice, ako bismo imali podatak da su sve žene preživjele, dok su svi muškarci umrli, tada bi čvor „*is sex male?*“ imao *gini impurity* nula, jer bismo sa 100% vjerojatnošću novu osobu mogli klasificirati obzirom na spol. Kao korijen stabla uzimamo atribut koji ima najmanji *gini impurity* te rekurzivno nastavljamo proces dokle god novo grananje smanjuje *gini impurity*, u suprotnom čvor pretvaramo u list.

BOOTSTRAP AGGREGATING

Bootstrap aggregating ili *bagging* je popularna metoda u strojnom učenju kojom pokušavamo popraviti stabilnost i točnost algoritma koji se koriste za problem klasifikacije ili regresije.

Neka je D skup podataka za učenje, takav da $\mathtt{card}(D) = n$. *Bootstrap aggregating* generima m skupova D_i kardinalnog broja n', dobivenih uzimanjem elemenata uniformno s ponavljanjem iz skupa D. Za velike n, ako je $n = n'$ skup D_i će imati otprilike 63.2% jedinstvenih elemenata iz skupa D. Može se pokazati da ovim algoritmom smanjujemo varijancu, te *overfitting* (izglađujemo krivulju koja opisuje training set) podataka. Na slici 2 možemo vidjeti problem *overfitting*-a. Primjećujemo da zelena krivulja bolje reprezentira podatke iz skupa učenja, no greška kod klasifikacije novih podataka će biti velika, za razliku od crne regularizirane krivulje.



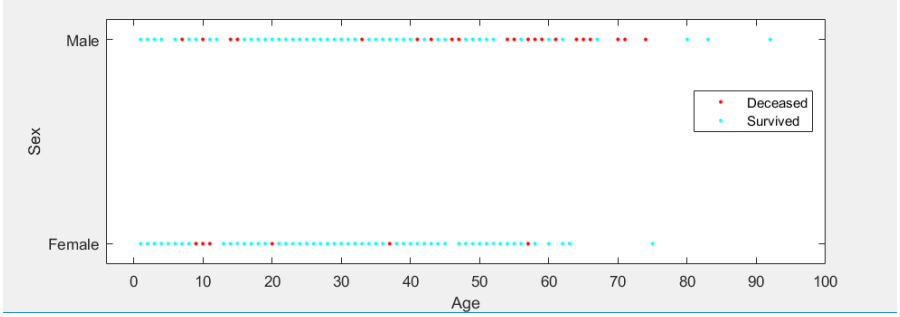
Slika 2: Primjer *overfitting*-a

RANDOM FOREST

Duboka stabla odlučivanja s puno čvorova često daju nepouzdane modele (*overfitting*). Iako dobro opisuju podatke iz skupa učenja, loše klasificiraju nove podatke (niska pristranost, velika varijanca). Ideja algoritma je osrednjavanje više dubokih stabla učenja, učenih na različitim dijelovima istog skupa s ciljem smanjenja varijance. *Random forest* koristi općeniti *bootstrap aggregating* algoritam na skupu podataka za učenje. Na danom skupu za učenje $X = x_1, x_2, \ldots x_n$ s ishodima $Y = y_1, y_2, \ldots y_n$, konstruiramo B podskupova s ponavljanjem. Nakon učenja, novi element x' klasificiramo tako da pogledamo klasifikaciju svakog od B stabala posebno, te klasificiramo s obzirom na glas većine.

REZULTATI

Kao što je rečeno u uvodu, pomoću tri različite metode u našem predviđanju dobili smo i tri različita postotka točnosti. U prvom pokušaju u kojem smo pretpostavili da će samo žena i djeca preživjeti dobili smo postotak točnosti 76.55%. Ovaj jednostavni pristup opravdavaju podaci sa slike 3. Prilikom korištenja *random forest* metode sa samo dvije varijable - dob i spol - dobivena točnost iznosi 76.07%, dok s uporabom 7 različitih varijabli dobivamo točnost od 75.59%. Navedena točnost dobivena je testiranjem predviđenih podataka na web stranici `kaggle.com` odakle smo i dohvatili korištene *train*, tj. *test* podatke.



Slika 3: Prikaz preživjelih osoba, grupirano po varijablama dob i spol

PRILOG: KÔD U MATLAB-U

```
function [Train, Test] = drvece

   % Procitaj datoteke
   [~,~,raw] = xlsread('train.csv');
   Train = cell2table(raw(2:end,:),'VariableNames',raw(1,:));

   [~,~,raw] = xlsread('test.csv');
   Test = cell2table(raw(2:end,:),'VariableNames',raw(1,:));

   % Definiranje varijabli
   Train.Sex = nominal(Train.Sex);

   Test.Sex = nominal(Test.Sex);

   % Podijeli cabin number u dva dijela
   Train.CabinDeck = cellfun(@(x) x(1),Train.Cabin);
   Train.CabinNum = cellfun(@(x)
       str2double(strtok(x(2:end))),Train.Cabin);

   Test.CabinDeck = cellfun(@(x) x(1),Test.Cabin);
   Test.CabinNum = cellfun(@(x)
       str2double(strtok(x(2:end))),Test.Cabin);

   % Seed
   rng(123);
   savedRng = rng;

   % Naivni pristup
   figure
      gscatter(Train.Age,Train.Sex,Train.Survived)
      set(gca,'YTick',[1 2])
      set(gca,'YTickLabel',{'Female','Male'})
      xlabel('Age')
      ylabel('Sex')
      ylim([0.9 2.1])
      legend({'Deceased','Survived'})

   % Baseline pristup
   Test.Survived = Test.Age < 16 | Test.Sex == 'female';

   baselinePredikcija =
       table(Test.PassengerId,Test.Survived,'VariableNames',{'PassengerId','Survived'});
   writetable(baselinePredikcija,'Predikcija\BaselinePredikcija.csv')

   % Konstrukcija obitelji
   Train.velicinaObitelji = Train.Parch + Train.SibSp + 1;
   Test.velicinaObitelji = Test.Parch + Test.SibSp + 1;

   % Leaf broj, random broj
   leaf = 15;
```

```
    nTrees = 40;
    rng(savedRng);


    %-------------------------------------------------------------


    % Prvi tree bagger
    X = [Train.Sex=='female' Train.Age];
    Xcat = logical([1 0]);
    Y = Train.Survived;

    Xtest = [Test.Sex=='female' Test.Age];
    treebagModel = TreeBagger(nTrees,X,Y);

    % Predikcija
    Ytest = predict(treebagModel,Xtest);
    Ytest = strcmpi(Ytest,'1');

    % Rjesenje prve predikcije
    predikcija =
        table(Test.PassengerId,Ytest,'VariableNames',{'PassengerId','Survived'});
    writetable(predikcija,'Predikcija\PrviRandomForest.csv')


    %-------------------------------------------------------------


    % Drugi tree bagger
    X = [Train.Sex=='female' Train.Age Train.Pclass Train.Fare
        Train.velicinaObitelji double(Train.CabinDeck)
        Train.CabinNum];
    Y = Train.Survived;

    Xtest = [Test.Sex=='female' Test.Age Test.Pclass Test.Fare
        Test.velicinaObitelji double(Test.CabinDeck)
        Test.CabinNum];

    rng(savedRng);
    treebagModel = TreeBagger(nTrees,X,Y);

    % Predikcija
    Ytest = predict(treebagModel,Xtest);
    Ytest = strcmpi(Ytest,'1');

    % Zapis u datoteku
    predikcija =
        table(Test.PassengerId,Ytest,'VariableNames',{'PassengerId','Survived'});
    writetable(predikcija,'Predikcija\DrugiRandomForest.csv')
end
```

LITERATURA

[1] Kaggle, https://www.kaggle.com.

[2] Nastavni materijali kolegija "strojno učenje", pmf-mo, https://web.math.pmf.unizg.hr/nastava/su/materijali/.

[3] Lior Rokach i Oded Maimon. Decision trees, http://www.ise.bgu.ac.il/faculty/liorr/hbchap9.pdf.

[4] Random forest, https://en.wikipedia.org/wiki/random$_f$orest, pribavljeno16.06.2018.