Univerzitet u Sarajevu
Elektrotehnički fakultet u Sarajevu
Odsjek za računarstvo i informatiku

Metode i tehnike predviđanja sportskih rezultata

Završni rad
I ciklusa studija

Mentor:
prof. dr. Haris Šupić

Kandidat:
Rasim Šabanović

Univerzitet u Sarajevu
Elektrotehnički fakultet u Sarajevu
Odsjek za računarstvo i informatiku

Izjava o autentičnosti rada

Student: Rasim Šabanović
Naslov rada: *Metode i tehnike predviđanja sportskih rezultata*
Vrsta rada: Završni rad 1. ciklusa studija
Broj stranica:

Potvrđujem:

a) da sam pročitao/pročitala odredbe koji se odnose na plagijarizam, kako je to definirano Statutom Univerziteta u Sarajevu, Etičkim kodeksom Univerziteta u Sarajevu i Pravilima studiranja za drugi ciklus studija na Fakultetu islamskih nauka Univerziteta u Sarajevu, kao i upute o plagijarizmu navedenim na web stranici Univerziteta u Sarajevu;

b) da sam svjestan/na univerzitetskih i fakultetskih disciplinskih pravila koja se tiču plagijarizma;

c) da je rad koji predajem potpuno moj, samostalni rad, osim u dijelovima gdje je to naznačeno;

d) da rad nije predat, u cjelini ili djelimično, za stjecanje zvanja na Univerzitetu u Sarajevu ili nekoj drugoj visokoškolskoj ustanovi;

e) da sam dosljedno naveo/la korištene i citirane izvore ili bibliografiju po nekom od propisanih stilova citiranja, sa navođenjem potpune reference koja obuhvata potpuni bibliografski opis korištenog i citiranog izvora;

f) da sam odgovarajuće naznačio/la svaku pomoć koju sam dobio/la pored pomoći mentora.

Mjesto i datum: Sarajevo, septembar 2018. god.

Potpis studenta/studentice: _____

Sažetak

U ovom radu će biti objašnjeni osnovni koncepti mašinskog učenja kroz problem predviđanja ishoda sportskih utakmica. Bit će objašnjeni odabrani algoritmi klasifikacije koji se najbolje uklapaju u kalup rješavanja ovog problema. Pored teoretske analize ovih algoritama, kroz praktičnu implementaciju, koja je jedan od važnijih dijelova rada, bit će objašnjen čitav proces mašinskog učenja, od prikupljanja do pripremanja podataka, pa do samog treniranja algoritama klasifikacije nad pripremljenim podacima.

Nakon uspješne implementacije i treniranja svih odabranih algoritama, bit će izvršena analiza postignutih rezultata, kao i komparacija performansi odabranih algoritama, a na osnovu ovih performansi će biti iznesen zaključak rada. Osnovni cilj rada je proći kroz osnove obrade podataka, mašinskog učenja i klasifikacije, ali i provjeriti da li je moguće napraviti sistem koji sa jako visokim nivoom precinznošću predviđa ishode sportskih mečeva. Mašinsko učenje kao samo jedna od oblasti vještačke inteligencije je jedan od stubova na kojim se gradi budućnost ljudske rase.

Abstract

In this paper, the basic concepts of machine learning shall be explained through the problem of predicting outcomes of sports matches. Algorithms of classification that best fit the mould of sports outcome prediction will be explained. Besides the theoretic analysis of these algorithms, the whole process of data collection, data preprocessing and ultimately the training of classification algorithms on the already processed data shall be explained through the practical implementation of the problem.

After the successful implementation and the training of all the chosen algorithms, an analysis of the results shall be conducted, as will a comparation of the performances of each algorithm, and a conclusion will be made based on these results. The main goal of the paper is to go through the basics of data processing, machine learning and classification, but also to check if it is possible to make a system that predicts outcomes of sports matches with a very high level of accuracy. Machine learning, as just a field of artificial intelligence is one of the foundations on which the future of the human race is being built.

Sadržaj

Popis tabela

Popis slika

1 Uvod

U današnjem svijetu, sport je neizostavan dio mnogih života na ovoj planeti. Negdje je to fudbal, negdje američki fudbal, negdje tenis, negdje karling. Svugdje na planeti ljudi prate neki sport. Dovoljno je samo reći da se procjenjuje da je finale Svjetskog prvenstva u Rusiji 2018 gledalo oko 700 miliona ljudi. Dovoljno je reći da je Super Bowl 2018 gledalo 111 miliona ljudi. Imajući u vidu da je velika većina tih ljudi iz Sjedinjenih Američkih Država, to je više nego trećina svih ljudi u državi.

Nije strano da mnogi ljudi žele prestiž, da li to bilo u vidu nekog priznanja od mase, da li to bilo u vidu velike količine novca ili u nekom drugom vidu. Isto tako, nije strano da sistemi za predviđanje sportskih rezultata mogu mnogo pomoći u dostizanju tog prestiža, da li dobitkom na kladionici ili nekim drugim putem.

Želja za radom na ovoj temi proizilazi najviše iz radoznalosti. Ljudi su iz radoznalosti, uz dodatnu dozu truda doznali i otkrili mnoge stvari koje se i dan danas koriste. Mada, kao što je već i spomenuto, motivacija za rad na ovakvoj temi može proizaći i iz nekih drugih razloga, kao što su sticanje materijalnog bogatstva na način da korišteni algoritmi sa potpunom tačnošću predviđaju ishode sportskih događaja. S druge strane, ljudi zaduženi za koeficijente u kladionicama također mnogo profitiraju od ovakvih sistema jer im oni daju do znanja kakve koeficijente treba da postavljaju.

Glavni cilj ovog rada je kreirati sisteme koji će do maksimalne moguće preciznosti predviđati ishode sportskih događaja. Da bi se maksimizirala preciznost, fokusirat će se na jedan sport umjesto da se pažnja usmjerava na više sportova. Sekundarni cilj je čisto testirati do koje granice računarski algoritmi mogu precizno predviđati ishode sportskim događaja.

1.1 Struktura završnog rada

U drugom poglavlju se ulazi u osnove nauke iza predviđanja sportskih rezultata. Definiše se mašinsko učenje i njegova klasifikacija, zatim se kratko osvrne na dosadašnji rad na ovu temu, a na kraju se opisuju metode i tehnike koje će biti korištene za implementaciju sistema za predviđanje sportskih rezultata.

U trećem poglavlju se priča o ulaznim podacima u sisteme za predviđanje rezultata. Opisuju se svi problemi koji mogu nastati tokom prikupljanja i sređivanja podataka, kao i problemi koji mogu nastati zbog prevelikih dimenzija ili premalog broja podataka, a onda se objašnjava kako se ti problemi rješavaju. Zatim se prolazi kroz proces pronalaska seta podataka adekvatnog za ovaj rad i opisuju se svi potencijalni setovi.

Četvrto poglavlje se bavi odabirom svojstava. Set podataka nudi mnogo podataka, od kojih se bira nekolicina i formatira na način koji je pogodan za implementaciju algoritama. Također se naglašava važnost ulaznih i izlaznih podataka kod nadgledanog učenja, a zatim se modeliraju ishodi sportskih događaja.

Peto poglavlje obuhvata praktični dio ovog završnog rada, tj. implementaciju svih ovih već navedenih algoritama. Vrše se procesi treniranja i predviđanja nad setom podataka i odabranim i proračunatim svojstvima. Glavni cilj ovog poglavlja je da se na praktičnom primjeru pokaže proces

predprocesiranja podataka i izvršenja algoritama mašinskog učenja. Ovo poglavlje se također bavi analizom rezultata implementacije algoritama, poređenjem rezultata pojedinih metoda. Također obuhvata diskusiju o tome zašto su neke metode pružile bolju tačnost od drugih, kao i zašto su neke metode efikasnije od drugih.

U šestom i posljednjem poglavlju se autor rada osvrće na postignute i nepostignute ciljeve, kao i naučeno tokom istraživanja i izrade završnog rada.

2 Mašinsko učenje

Mašinsko učenje je polje računarskih nauka koje daje računarima sposobnost da "uče" iz određenih skupova podataka, bez da ih ljudi programiraju da rade eksplicitno to. To je skup metoda koji automatski prepoznaju šablone u podacima, a zatim koriste te šablone da previđaju buduće podatke ili donose neke druge odluke bez prethodnog znanja o ishodima. [1]

2.1 Podjela učenja

Mašinsko učenje se većinom dijeli na dva tipa: **nadgledano i nenadgledano učenje.** [1]

2.1.1 Nadgledano učenje

Nadgledano učenje ima za cilj predvidjeti izlaz y, ako se na ulazu nađe neka vrijednost x, s tim da je algoritmu učenja poznat skup parova ulaza i izlaza $D = \{(x_i, y_i)\}_{i=1}^{N}$. Ovaj skup parova ulaza i izlaza koji algoritam učenja koristi se naziva **trening set (eng. training set)**, a N je broj **trening parova (eng. training example).** [1]

Nadgledano učenje se opet dijeli na dva tipa, a to su: **klasifikacija** i **regresija.** [1] Razlike između ova dva tipa učenja su objašnjene u nastavku.

2.1.1.1 Klasifikacija

Cilj ovog tipa nadgledanog učenja je da algoritam nauči da predvidi izlaz y na osnovu ulaza x, ako $y \in \{1, \dots, C\}$, ako je C broj klasa u koje treba rasporediti izlaze. U zavisnosti od broja klasa, klasifikacija može biti **binarna** (2 klase), **ternarna** (3 klase) ili **multiklasna** (eng. multiclass) klasifikacija. [1] Ukoliko pripadnost klasama nije međusobno isključiva, odnosno ako jedan izlaz može pripadati više izlaznih klasa u isto vrijeme, onda se to naziva **višeizlazna klasifikacija** (eng. multilabel classification). [1]

Klasifikacija se najbolje može objasniti na jednom jednostavnom primjeru. Recimo da imamo skup nekih lopti i da algoritam treba da na osnovu nekih osnovnih ulaza zaključi da li se radi o teniskoj lopti ili o fudbalskoj lopti.

Slika 2.1 Fudbalske lopte

Slika 2.2 Teniske lopte

Kao što vidimo na slikama 2.1 i 2.2, imamo dva skupa lopti: fudbalske lopte i teniske lopte. Ove lopte su, osim svojim izgledom, opisane još nekim atributima, datim u sljedećoj tabeli:

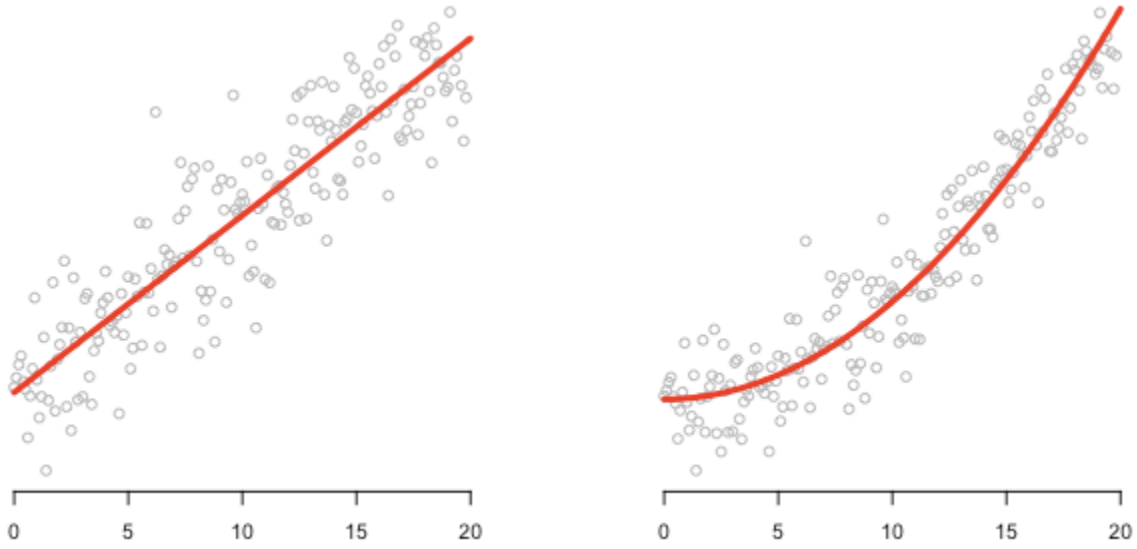
Poluprečnik	Dominantna boja	Odluka
22cm	bijela	fudbalska
6.8cm	zelena	teniska

Tabela 2.1 Primjer klasifikacije

Vidimo da na osnovu nekih osnovnih atributa svake lopte, algoritam odlučuje o kojoj se lopti ustvari radi, tj. on svaku loptu pridružuje jednoj od ponuđenih klasa, što i jeste ideja klasifikacije.

2.1.1.2 Regresija

Regresija je jako slična klasifikaciji, samo što kod regresije algoritam predviđa kontinualnu vrijednost izlaza za kontinualnu vrijednost ulaza, umjesto da izlaz svrstava u jednu od predodređenih klasa. Formalno opisano, to u biti znači da na ulazu imamo neku jedinstvenu vrijednost $x_i \in R$, a na izlazu vrijednost $y_i \in R$. [1]



Slika 2.3 Primjer regresije

Na slici 2.3, grafički je prikazana sama ideja regresije kao metode predviđanja izlaza. Za svaki kontinualni ulaz, ona nam daje kontinualni izlaz, pa samim tim imamo i tipove regresija čija imena ovise od dimenzionalnosti funkcija kojima predstavljamo zavisnost ulaza od izlaza. Imajući to u vidu, zaključujemo da postoje linearna i polinomialna regresija.

Regresija se jako često koristi za svakodnevne probleme u raznim sferama života, pa se recimo može koristiti za predviđanje temperature na nekom mjestu na osnovu raznih vremenskih prognoza, historijskog stanja, očitavanja senzora itd. To je, naravno, samo jedan od mnogobrojnih primjera u kojima je regresija izuzetno korisna u svakodnevnoj upotrebi.

2.1.2 Nenadgledano učenje

Kratko ćemo se dotaći ideje **nenadgledanog učenja** [1], jer se njome nećemo baviti u ovom radu. U biti, razlika u odnosu na nadgledano učenje je što su nam dati samo izlazni podaci, bez ikakvog znanja o ulaznim podacima. Zadatak nenadgledanog učenja je da prepozna neke šablone ili strukture u izlaznim podacima, što se nekada zove **otkriće znanja** (eng. knowledge discovery).[1]

2.2 Prethodni rad na ovu temu

Mnogi su se ljudi do sada bavili ovom ili sličnim temama. Postoje mnoge web stranice koje pokušavaju što bolje da predviđaju rezultate, kao što su npr. www.scorepredictor.com, www.predictz.com itd.

Neki od radova koji se bave ovom temom, a koji mogu biti itekako korisni svakome koga interesuje konkretno ova tema su:

Niek Tax, Yme Joustra [6]

Ovaj rad se bavi predviđanjem rezultata u holandskoj Eredivisie ligi. Skup podataka koji je korišten pri izradi ovog rada je ručno napravljen od strane čovjeka koji je pisao rad i sastojao se od podataka u proteklih 13 sezona Eredivisie lige. Mnogi modeli su implementirani uz veliki broj svojstava (eng. feature), a najbolji rezultat je postignut kombinacijom PCA (Principal component analysis) sa Naïve Bayes klasifikatorom.

Igiri, Chinwe Peace; Nwachukwu, Enoch Okechukwu [7]

U ovom radu, sistem za predviđanje fudbalskih rezultata je implementiran koristeći vještačke neuronske mreže (eng. Artificial neural network) i logističku regresiju (eng. logistic regression), uz Rapid Miner kao alat za data mining. Postignuti su zavidni rezultati sa obe implementacije, 83% tačnost za neuronsku mrežu, a 93% tačnost za logističku regresiju.

Michal Sipko [8]

Iako se ovaj rad ne bavi konkretno predviđanjem fudbalskih rezultata, nego teniskih, možda je i najkorisniji sa kojim sam se ja susreo. U implementaciju su modelirana 22 svojstva, uključujući neka i nediskretna i abstraktna, kao što su npr. umor ili povrede. Koristeći odgovarajući set podataka i ova 22 svojstva u kombinaciji sa implementiranim modelima, vještačkim neuronskim mrežama i logističkom regresijom, dobijeni su rezultati koji bi nam garantirali profit od 4.35% pri konstantnom ulaganju na kladionicu.

Ville Sillanpää, Olli Heino [9]

U ovom radu je napravljen model koji je u stanju nadmašiti sve već postojeće modele koji su slobodni za upotrebu, ali nije u stanju garantovati profit pri konstantnom ulaganju na kladionicu.

Glavni model koji se koristi je logistička regresija, uz posebnu modifikaciju svojstava seta podataka zvanu Elo ocjena (eng. Elo rating).

2.3 Metode

U ovom dijelu će biti teoretski objašnjene tri metode koje su odabrane za implementaciju predviđanja sportskih rezultata, s tim da će biti objašnjene samo konceptualno, bez trenutne konkretne veze sa temom rada.

2.3.1 k-najbližih susjeda

Algoritam **k-najbližih susjeda** (eng. k-nearest neighbors) [2] je jedan od najjednostavnijih algoritama klasifikacije. Metode najbližih susjeda se također nazivaju i **metodama baziranim na sjećanju** (eng. memory-based methods). [2]

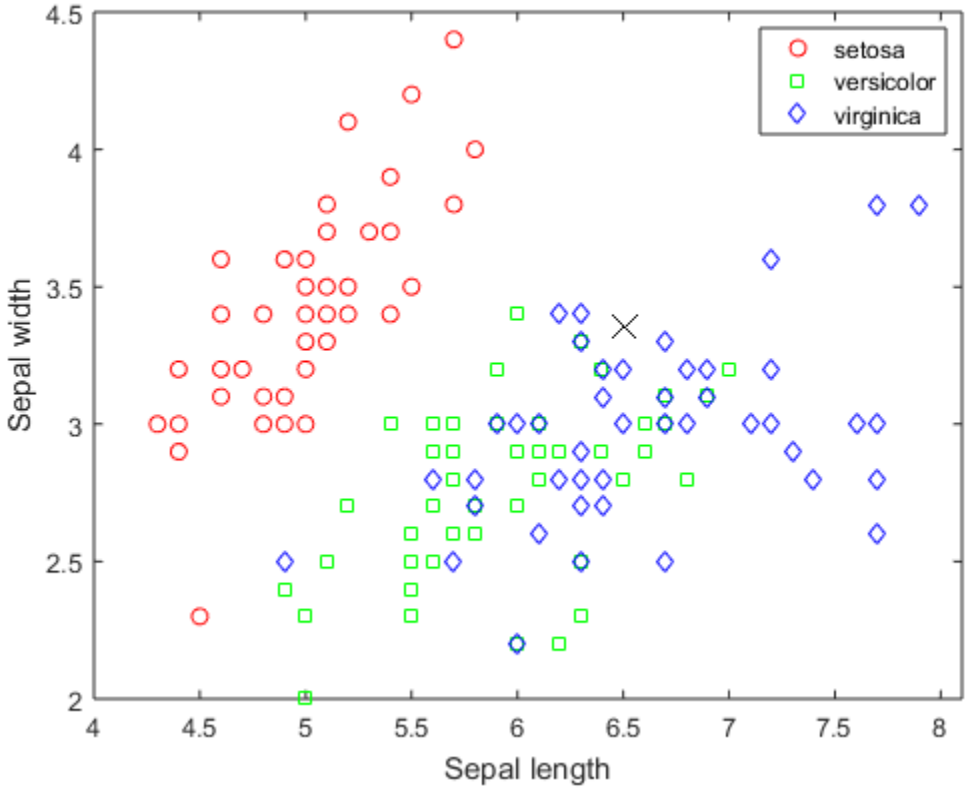
Ovaj algoritam, najjednostavnije rečeno, provjeri kojim klasama pripada k najbližih susjeda nekog ulaza X, i zaključuje klasu našeg ulaza na osnovu frekventnosti određenih klasa u tih k elemenata. Koristeći velike vrijednosti k, smanjujemo vjerovatnoću da će frekventnost neke nebitne skupine ulaza koja je jako bliska našem ulazu mnogo utjecati na konačnu klasifikaciju našeg ulaza, ali velike vrijednosti također smanjuju preciznost ove metode. Ovaj algoritam se u biti svodi na određivanje vjerovatnoće kojoj klasi od ponuđenih pripada naš ulaz. Što je veća gustina tačaka oko našeg ulaza, a i veće k, to algoritam daje tačniji izlaz. [2]

Udaljenost našeg ulaza od obližnjih tačaka da bi se odredilo kojih je to k tačaka najbliže se može računati na mnogo načina, a najjednostavniji način jeste u biti **Euklidska udaljenost** (eng. Euclidean distance). [2] Ako naš ulaz ima n svojstava, onda se Euklidska udaljenost za 2 tačke računa kao:

$$d = \sqrt{\sum_{i=1}^{n} (x_{1i} - x_{2i})^2}$$

Osim Euklidske udaljenosti, koriste se još neke vrste proračuna za ove udaljenosti, kao što su npr. **Mahnattan udaljenost** (eng. Manhattan distance), **Minkowski udaljenost** (eng. Minkowski distance) itd. [3]

Koristeći MATLABov *fisheriris* data set, možemo na primjeru pokazati kako to radi ovaj algoritam.



Slika 2.4 k-najbližih susjeda

Na slici xy su podaci podijeljeni u tri izlazne klase (setosa, versicolor, virginica). Ako uzmemo da je $k = 3$, i da se za određivanje udaljenosti koristi standardna Euklidska udaljenost, možemo i intuitivno bez potrebe za računanjem zaključiti da se sva 3 najbliža susjeda našem ulazu (označen sa X) nalaze u klasi označenoj plavim dijamantom. Iz svega ovoga slijedi da će ovaj ulaz algoritam k-najbližih susjeda klasificirati u istu klasu kao i sva tri njegova najbliža susjeda, a to je upravo klasa označena plavim dijamantima.

2.3.2 Naivni Bayes klasifikator

Prije nego što počnemo sa objašnjenjem kako to radi **Naivni Bayes klasifikator**, moramo proći kroz osnove **Bayesove teoreme**. [4] Bayesova teorema glasi:

Ako postoje dva događaja A i B, i ako nam je poznata uvjetna vjerovatnoća A u odnosu na B, označena kao $P(A|B)$, možemo izračunati $P(B|A)$ kao:

$$P(B|A) = \frac{P(A|B)P(B)}{P(A)}$$

Performanse Naivnog Bayes klasifikatora su se u mnogim slučajevima pokazale kao uporedive sa performansama nekih naprednijih algoritama klasifikacije. Na ulaz dovodimo set svojstava a, a zadatak algoritma je da na osnovu ovog seta svojstava klasificira naš ulaz u neku od ponuđenih klasa. Način na koji to radi ovaj klasifikator je da ulaz dodijeli onoj klasi za koju se pokazalo da je najveća vjerovatnoća da ulaz njoj pripada.

$$y = argmax\, P(y_i, a_1, a_2 \ldots, a_n)$$

Koristeći Bayesovu teoremu iz jednačine xy, ovo možemo napisati kao:

$$y = argmax\, \frac{P(a_1, a_2 \ldots, a_n|y_i)P(y_i)}{P(a_1, a_2 \ldots, a_n)}$$

$$= argmax\, P(a_1, a_2 \ldots, a_n|y_i)P(y_i)$$

Ovu formulu možemo koristiti za klasificiranje ulaza, ali postoji jedan problem. Vjerovatnoće $P(y_i)$ možemo lako računati prebrojavanjem koliko puta se koja klasa nalazi na izlazu. Međutim, vjerovatnoće $P(a_1, a_2 \ldots, a_n|y_i)$ ne možemo ovako računati osim ako nemamo jako veliki set podataka.

Naivni Bayes klasifikator je baziran na jednoj pretpostavci koja čitav ovaj proces mnogo pojednostavljuje, a ta pretpostavka je da su vrijednosti svih svojstava ulaza uvjetno nezavisne jedne od drugih, uz poznatu ciljnu vrijednost. Drugim riječima, u ovom slučaju vjerovatnoća $P(a_1, a_2 \ldots, a_n)$ postaje proizvod vjerovatnoća $P(n)$, tj. $P(a_1, a_2 \ldots, a_n) = \prod_j P(a_j|y_i)$.

Tako dobijamo formulu koja predstavlja formalni zapis Naivnog Bayes klasifikatora:

$$y_{NB} = argmax \prod_j P(a_j|y_i)$$

gdje je y_{NB} izlaz klasifikatora pri ulazu a. [4]

2.3.2.1 Primjer

Recimo da imamo skup podataka o simptomima gripe i finalne zaključke da li neki pacijenti imaju gripu ili ne, i da taj skup podataka izgleda ovako:

drhtanje	kašalj	glavobolja	temperatura	gripa?
DA	NE	blaga	DA	NE
DA	DA	NE	NE	DA
DA	NE	jaka	DA	DA
NE	DA	blaga	DA	DA
NE	NE	NE	NE	NE
NE	DA	jaka	DA	DA
NE	DA	jaka	NE	NE
DA	DA	blaga	DA	DA

Tabela 2.2 Naivni Bayes - skup podataka

Naš zadatak je da koristeći Naivni Bayes klasifikator odredimo da li sljedeći pacijent ima gripu ili nema:

drhtanje	kašalj	glavobolja	temperatura	gripa?
DA	NE	blaga	DA	NE

Tabela 2.3 Naivni Bayes - novi ulaz

Kada izračunamo sve potrebne vjerovatnoće za korištenje Naivnog Bayes klasifikatora, dobijamo tabelu u kojoj se nalaze sve te vrijednosti:

$P(gripa = DA) = 0.625$	$P(gripa = NE) = 0.375$
$P(drhtanje = DA \mid gripa = DA) = 0.6$	$P(drhtanje = DA \mid gripa = NE) = 0.333$
$P(drhtanje = NE \mid gripa = DA) = 0.4$	$P(drhtanje = NE \mid gripa = NE) = 0.666$
$P(kašljanje = DA \mid gripa = DA) = 0.8$	$P(kašljanje = DA \mid gripa = NE) = 0.333$
$P(kašljanje = NE \mid gripa = DA) = 0.2$	$P(kašljanje = NE \mid gripa = NE) = 0.666$
$P(glavobolja = blaga \mid gripa = DA) = 0.4$	$P(glavobolja = blaga \mid gripa = NE) = 0.333$
$P(glavobolja = NE \mid gripa = DA) = 0.2$	$P(glavobolja = NE \mid gripa = NE) = 0.333$
$P(glavobolja = jaka \mid gripa = DA) = 0.4$	$P(glavobolja = jaka \mid gripa = NE) = 0.333$
$P(temperatura = DA \mid gripa = DA) = 0.8$	$P(temperatura = DA \mid gripa = NE) = 0.333$
$P(temperatura = NE \mid gripa = DA) = 0.2$	$P(temperatura = NE \mid gripa = NE) = 0.666$

Tabela 2.4 Naivni Bayes – proračunate vjerovatnoće

Kada imamo izračunate sve ove vrijednosti, onda po formuli Naivnog Bayes klasifikatora možemo izračunati obe vrijednosti za $P(gripa = DA)$ i $P(gripa = NE)$. Računanjem obe vrijednosti zaključujemo da pacijent sa ovim svojstvima nema gripu.

2.3.3 Vještačke neuronske mreže

Vještačke neuronske mreže [5] su nastale sa namjerom da repliciraju prirodne ljudske neuronske mreže. To znači da se i vještačke, kao i prirodne neuronske mreže, sastoje od **neurona** i **veza između njih**. [5] Svaka neuronska mreža ima dva osnovna sloja, a to su **ulazni i izlazni sloj**. [5] Svi slojevi koji se nalaze između ta dva sloja se nazivaju **skriveni slojevi**. [5] Duboke neuronske mreže mogu imati i stotine skrivenih slojeva.

Neuronska mreža sa barem jednim skrivenim slojem sa konačnim brojem neurona u tom sloju može aproksimirati bilo koju neprekidnu funkciju. Ovo opravdava mišljenje da se vještačke neuronske mreže mogu koristiti u svim sferama vještačke inteligencije, jer mogućnost aproksimiranja funkcija automatski povlači i mogućnost učenja određenog načina ponašanja.

2.3.3.1 Struktura

Ulazni sloj se sastoji od n neurona, koji odgovaraju n svojstava ulaza $(X_1 \dots, X_n)$. Broj neurona u skrivenim slojevima bira projektant neuronske mreže, a taj broj se dobija empiričkim postupcima. U izlaznom sloju ima onoliko neurona koliko ima finalnih klasa u koje treba klasificirati ulaze u neuronsku mrežu. Svaka veza između dva nerona ima vezanu težinu (eng. weight), koja na početku ima neku fiksnu zadatu vrijednost, a mijenja se procesom treniranja mreže.

Stanje ulaznog neurona ovisi samo od ulaznih varijabli, a stanje svih ostalih neurona (skrivenih i izlaznih) ovisi od prethodnih slojeva, i to po relaciji:

$$a_j = \sum_{i=1}^{I} X_i W_{ji}$$

gdje je a_j ulaz u neuron j, X_i je izlaz iz neurona i iz prethodnog sloja, a W_{ji} je težinski faktor veze između neurona i i j. [5]

2.3.3.2 Treniranje mreže

Najbolja tehnika za treniranje mreže u nadgledanom učenju se zove **propagacija greške unazad (eng. error backpropagation).** [5] Svakom iteracijom treniranja mreže se težinski faktori veza između neurona modificiraju na osnovu izlaza, i to od izlaza prema ulazu.

Na trajanje treniranja mreže utiču tri glavna faktora, a to su **struktura mreže**, **broj ulaznih podataka u setu podataka** i **broj iteracija treniranja mreže**. [5]

2.3.3.3 Testiranje mreže

Nakon treniranja jedne neuronske mreže, ona bi trebala biti testirana. Postoje mnogi načini testiranja neuronske mreže, ali jedna od najjednostavnijih, ali i najboljih indikacija o tačnosti neuronske mreže je jednostavno procenat tačnih izlaza. Ako se radi o nadgledanom učenju, dostupni su nam izlazi za sve ulaze koje možemo dovesti na ulaz neuronske mreže, tako da lako možemo izračunati sa kolikom tačnošću neuronska mreža daje odgovore na nove ulaze.

3 Ulazni podaci

Za svaki problem koji rješavamo metodama mašinskog učenja nam je potreban neki set ulaznih podataka. Onog trenutka kada imamo pripremljene podatke, možemo implementirati model mašinskog učenja koji hoćemo da bismo napravili sistem koji predviđa ono što želimo. Međutim, podaci ne moraju uvijek biti u idealnoj formi onog momenta kada ih preuzmemo. Naprotiv, u većini slučajeva se nad podacima moraju vršiti neke modifikacije da bi se ti isti podaci mogli efikasno iskoristiti u modelu koji želimo da implementiramo. Čak i kada su podaci u toj idealnoj formi, može često doći do nekih problema kao što je npr. **preprilagođavanje** (eng. overfitting). [1]

3.1 Redukcija dimenzionalnosti

Kada u setu podataka na ulazu imamo jako puno svojstava, često je dobra ideja smanjiti dimenzionalnost problema. To se radi tako što više svojstava pokušavamo preslikati u jedno svojstvo koje jednom vrijednošću obuhvata sva ta prijašnja svojstva. Redukcija dimenzionalnosti se radi da bi se izbjegla **kletva dimenzionalnosti (eng. curse of dimensionality)** [1] naročito kod algoritama najbližeg susjeda.

Glavni razlog zašto se koristi redukcija dimenzionalosti upravo i jeste taj što se možda čini da je ulaz jako visoke dimenzionalnosti, ali da ustvari postoji mnogo manje nekih globalnih dimenzija koje na kraju krajeva obuhvataju sve ove ostale dimenzije. Ove globalne dimenzije se nazivaju još i **latentni faktori (eng. latent factors)** [1] i oni ne moraju biti u potpunosti mjerljivi kao i dimenzije koje sami obuhvataju. Za primjer, latentni faktor može biti zdravlje. Zdravlje i nije baš mjerljivo, nego se sastoji od nekog broja mjerljivih faktora.

Najčešći pristup redukciji dimenzionalnosti se zove **metoda osnovnih komponenti (eng. principal component analysis ili PCA)**. [1]

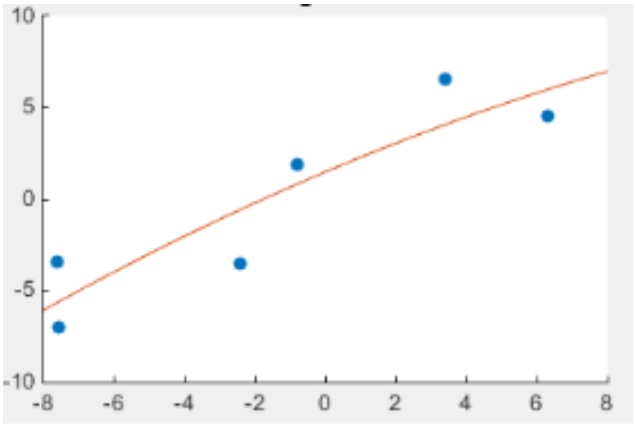
Redukcija dimenzionalnosti se koristi gdje god se koriste i metode mašinskog učenja, a neke od sfera u kojima se mnogo koristi su biologija, procesiranje jezika, računarska grafika…

3.2 Preprilagođavanje (eng. overfitting)

Pri implementaciji bilo kojeg modela ili algoritma mašinskog učenja, jako je bitno podijeliti naš ulazni set podataka na one podatke koji će se koristiti za treniranje modela i one podatke koji će se koristiti za testiranje modela. Ova dva seta podatak se zovu **trening set** i **testni set** respektivno. Ne postoji tačna mjera kakva treba biti ova podjela, ali u većini slučajeva se uzima od 70% do 80% podataka za trening set, a ostalo za testni set podataka. [1]

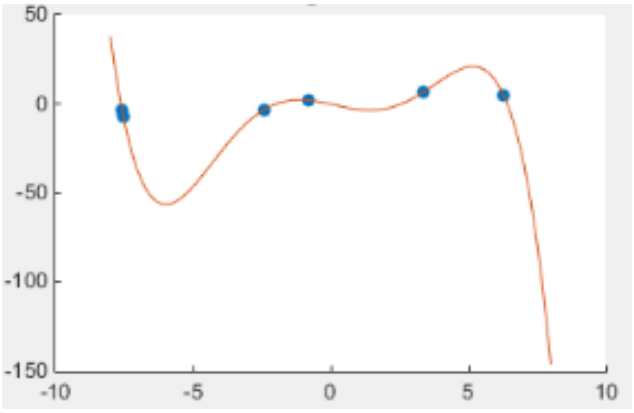
Ako se nekada desi da pri treniranju naš algoritam daje 100% tačnih izlaza, a pri testiranju sa testnim setom daje 50% tačnih izlaza, ovo znači da je došlo do preprilagođavanja. Preprilagođavanje se događa kada pokušavamo modelirati bilo kakvu malu izmjenu ulaza, jer je veća vjerovatnoća da je ova promjena ustvari samo neki šum nego da je pravi ulaz.

U slučaju preprilagođavanja, krive koje aproksimiraju izlaze iz algoritma imaju potpuno neprirodan oblik upravo zato što su algoritmi naučeni tako da samo prepoznaju i tačno predviđaju ulaze koji se već nalaze u trening setu.



Slika 3.1 Regresija bez preprilagođavanja

Na slici xy iznad možemo vidjeti aproksimaciju bez preprilagođavanja. Ova aproksimacija vjerovatno neće dati u potpunosti tačan rezultat za bilo koji izlaz, ali daje rezultate sa visokom tačnošću čak i za izlaze koji se nalaze van trening seta, tj. u testnom setu.



Slika 3.2 Regresija sa preprilagođavanjem

Međutim, na ovoj slici imamo prisutan problem preprilagođavanja. Ova aproksimacija daje u potpunosti tačne rezultate za sve ulaze iz trening seta, međutim pravi nerealne i potpuno netačne aproksimacije za neke nove ulaze. Vidimo kako drastično opada funkcija nakon zadnje tačke, ovo je klasičan znak preprilagođavanja.

3.3 Odabir seta podataka

Mnogo toga u mašinskom učenju zavisi od odabira seta podataka koji će se koristiti. Jako je važno odabrati set koji pruža mnogo ključnih informacija, a u isto vrijeme nema previsoku dimenzionalnost i ne povećava masovno vrijeme treniranja algoritma i ostale usporavajuće faktore. Naravno, nijedan set podataka nije baš spreman za korištenje onakav kakav bude pronađen, tako

da je ključno i naći set kojem su potrebne minimalne izmjene da bi bio optimalan za korištenje za implementaciju modela mašinskog učenja.

Što se tiče fudbala, postoji mnogo dostupnih setova podataka koji su dostupni za slobodnu upotrebu na internetu, pa i nije bilo toliko teško naći neki koji odgovara.

openfootball (football.db)

Ovaj set podataka je besplatan i open source. Ovo je historijski set podataka, tako da ne podržava ažuriranje rezultata uživo, ali to ovdje nije ni potrebno. Podaci su spremljeni u brojnim GitHub repozitorijima, a postoji i open source API koji se može koristiti za potrebe demonstracija.

soccerstats.us

Ovi podaci se također nalaze u brojnim GitHub repozitorijima i sadržavaju rezultate iz mnogih različitih takmičenja, što klupskih, to i internacionalnih, ali ne sadržavaju podatke iz evropskih fudbalskih liga, konkretno engleske lige.

football-data.org

Ovo je besplatni API koji sadrži redovno ažurirane podatke. Pruža mnoge mogućnosti kroz brojnje endpointe, kao što su /soccerseasons, /fixtures, /teams itd. Kada npr. pokušamo dobaviti podatke o nekom timu, u JSON odgovoru dobijamo podatke kao što su ime tima, vrijednost ekipe, logo, link API poziva za igrače tima…

Sports Open Data API

Ovo je također besplatan API koji je nastao 2016. godine. Limitiran je na 10,000 API poziva mjesečno. Svi endpointi koji su dostupni su vezani za lige, a onda kada specificiramo ligu, možemo dobiti mnogo više informacija vezanih za tu ligu, kao što su npr. menadžeri, sudije, ekipe, rezultati itd.

Betlines Ninja API

Betlines Ninja API je ustvari besplatni API koji prvenstveno služi za podatke o klađenju, međutim sadrži i mnogo korisnih informacija o predstojećim utakmicama.

API-Football

API-Football pokriva i velike ali i neke male fudbalske lige. Podržava mnogo mogućnosti, kao što su naprimjer rezultati uživo, koeficijenti, svi događaji, stanje na tabeli i još mnogo toga. Besplatna varijanta nam pruža 50 zahtjeva dnevno.

football-api

football-api.com je API servis koji se plaća. Prati vašu IP adresu i na osnovu IP adrese i paketa koji je vezan za tu IP adresu stavlja restrikcije na broj zahtjeva koji možete poslati. Sadrži mnogo

podataka, vjerovatno najviše od ovih svih setova. Možete naći sva takmičenja, sve timove, stanja na tabelama, rezultate uživo, predstojeće utakmcice, komentare i još mnogo toga.

opta

Opta je jedan od najvećih i najsnažnijih servisa vezanih za fudbalsku statistiku na svijetu. Opta sadrži podatke ne samo o utakmicama i timovima, nego i o svim igračima posebno.

football-data.co.uk

Ova web stranica ustvari najviše služi za kladionice i koeficijente u kladionicama, ali ima mnogo historijskih podataka o fudbalu u formatu .csv fajlova. Sadrži podatke iz mnogo europskih liga, kao što su npr. 5 nivoa engleskog fudbala, 4 nivoa škotskog fudbala, 2 nivoa njemačkog i talijanskog itd.

Osim ovih setova podataka, bilo je još mnogo i besplatnih i plaćenih. Najbolja opcija i krajnji odabir seta podataka je **football-data.co.uk** set podataka iz više razloga. Jedan od razloga jeste taj što ovaj set podataka sadrži punih 10 sezona engleske Premier lige, odnosno sve utakmice iz svake sezone i mnoge detalje iz tih utakmica.

Date	HomeTea	AwayTea	FTHG	FTAG	FTR	HTHG	HTAG	HTR	Referee	HS	AS	HST	AST	HF	AF	HC	AC	HY	AY	HR	AR
#######	Aston Vil	Wigan	0	2	A	0	1	A	M Clatter	11	14	5	7	15	14	4	6	2	2	0	0
#######	Blackbur	Man City	0	2	A	0	1	A	M Dean	17	8	9	5	12	9	5	4	2	1	0	0
#######	Bolton	Sunderla	0	1	A	0	1	A	A Marrin	11	20	3	13	16	10	4	7	2	1	0	0
#######	Chelsea	Hull	2	1	H	1	1	D	A Wiley	26	7	12	3	13	15	12	4	1	2	0	0
#######	Everton	Arsenal	1	6	A	0	3	A	M Halse	8	15	5	9	11	13	4	9	0	0	0	0
#######	Portsmou	Fulham	0	1	A	0	1	A	M Atkins	16	9	4	3	11	18	6	4	3	2	0	0
#######	Stoke	Burnley	2	0	H	2	0	H	S Bennet	12	9	5	5	15	10	3	6	2	2	0	0
#######	Wolves	West Ha	0	2	A	0	1	A	C Foy	19	16	11	13	9	5	8	6	0	0	0	0
#######	Man Unit	Birmingh	1	0	H	1	0	H	L Mason	26	6	17	4	13	7	13	2	1	1	0	0
#######	Tottenha	Liverpool	2	1	H	1	0	H	P Dowd	17	6	11	3	14	16	6	5	3	3	0	0
#######	Sunderla	Chelsea	1	3	A	1	0	H	S Bennet	4	20	3	9	14	10	1	14	2	2	0	0
#######	Wigan	Wolves	0	1	A	0	1	A	M Jones	18	9	5	3	8	21	4	4	0	3	0	0
#######	Birmingh	Portsmou	1	0	H	0	0	D	L Probert	9	9	5	4	11	20	4	5	0	5	0	0
#######	Burnley	Man Unit	1	0	H	1	0	H	A Wiley	8	18	2	9	8	12	1	12	2	1	0	0
#######	Hull	Tottenha	1	5	A	1	3	A	C Foy	9	18	7	12	23	13	6	5	3	2	0	0
#######	Liverpool	Stoke	4	0	H	2	0	H	P Walton	18	6	13	3	10	9	11	5	0	1	0	0
#######	Arsenal	Portsmou	4	1	H	2	1	H	S Bennet	19	9	16	4	9	10	8	4	1	0	0	0
#######	Birmingh	Stoke	0	0	D	0	0	D	C Foy	8	13	6	3	8	13	3	3	0	3	0	0
#######	Hull	Bolton	1	0	H	0	0	D	M Jones	12	20	5	10	19	19	4	12	3	2	0	0
#######	Man City	Wolves	1	0	H	1	0	H	L Mason	16	10	11	2	7	11	5	6	0	1	0	0
#######	Sunderla	Blackbur	2	1	H	1	1	D	A Wiley	8	18	5	5	12	14	0	10	2	1	0	0
#######	Wigan	Man Unit	0	5	A	0	0	D	H Webb	16	16	7	13	11	8	3	5	2	2	0	0
#######	Burnley	Everton	1	0	H	1	0	H	P Dowd	8	17	4	11	11	13	3	7	1	0	0	0
#######	Fulham	Chelsea	0	2	A	0	1	A	A Marrin	4	12	1	4	11	10	4	7	0	0	0	0
#######	West Ha	Tottenha	1	2	A	0	0	D	M Clatter	17	17	8	10	13	11	4	8	2	1	0	0
#######	Liverpool	Aston Vil	1	3	A	0	2	A	M Atkins	21	7	12	4	15	11	10	4	3	2	0	0
#######	Blackbur	West Ha	0	0	D	0	0	D	P Dowd	19	11	9	5	13	18	5	2	1	3	0	0

Slika 3.3 Izgled seta podataka

Kao što vidimo na slici xy, ovaj set podataka nam pruža datum odigravanja utakmice, imena domaće i gostujuće ekipe, broj golova domaće i gostujuće ekipe na kraju utakmice, broj golova domaće i gostujuće ekipe na poluvremenu, glavnog sudiju utakmice, broj šuteva domaćina i gosta, broj šuteva u okvir gola domaćina i gosta, broj žutih kartona i broj crvenih kartona.

U ovom setu podataka nije prevelika dimenzionalnost, ali ima dovoljan broj ključnih podataka i faktora da bi implementirani algoritmi davali dobre rezultate.

4 Odabir svojstava

Svako algoritam nadgledanog mašinskog učenja zahtijeva neki set podataka na osnovu kojeg on uči da predviđa neke buduće ulaze. U pogledu fubdalskih mečeva, ovaj set podataka na ulazu se sastoji od dva elementa:

1. Vektor ulaznih svojstava (X) – predstavlja karakteristike fudbalskog meča
2. Ciljana vrijednost (Y) – predstavlja ishod fudbalskog meča

Problem predviđanja ishoda fudbalskog meča je ustvari problem klasifikacije nekog seta ulaza u jednu od tri klase, a to su pobjeda domaćina, neriješen rezultat i pobjeda gosta. Da bi algoritam mogao klasificirati naše ulaze, moramo ove klase nekako modelirati:

$$Y = \begin{cases} 1, & ako\ je\ pobijedio\ domaćin \\ 0, & ako\ je\ neriješen\ rezultat \\ -1, & ako\ je\ pobijedio\ gost \end{cases}$$

U ovu klasifikaciju ne ubrajamo mečeve koji nisu održani ili koji su prekinuti, ali takvih mečeva i nema u ovom setu podataka, pa nije bilo potrebe da se modeliraju.

4.1 Forma

Svi mi koji pratimo bilo koji sport, a pogotovo fudbal, znamo da forma igra veliku ulogu u odlučivanju ishoda utakmica. Svaka pobjeda ekipi diže samopouzdanje, a samim tim i povećava šansu za boljim rezultatima u budućnosti. Ali, koliko god da skorija forma utiče na sam ishod utakmice, toliko može biti i varljiv faktor. Ako neka ekipa ima pet poraza u nizu, sigurno je to jednim dijelom loša forma, ali se isto tako može desiti i da ta ekipa igra pet utakmica u nizu protiv mnogo jačih ekipa, pa čak i ako jesu u dobroj formi, nizaju pet poraza zaredom.

Upravo zbog ovog faktora i činjenice da u engleskoj Premier ligi uvijek ima 6 ekipa koje se smatraju nivoom iznad ostalih, forma će se računati kao koeficijent koji zavisi od prethodnih 10 utakmica jedne ekipe. Ako uzimamo 10 utakmica, ni u jednom slučaju nije moguće da je svih 10 utakmica ta ekipa odigrala protiv izrazitog favorita, čak je i mala vjerovatnoća da je igrala protiv svih 6, tako da ovaj koeficijent daje relativno realnu procjenu u kakvoj je to formi ova ekipa.

Za svaku utakmicu će se forma ekipe računati po formuli:

$$X_{1n} = \begin{cases} 0, & u\ slučaju\ poraza \\ 0.5, & u\ slučaju\ remija \\ 1, & u\ slučaju\ pobjede \end{cases}$$

Imajući u vidu ovu formulu, finalni koeficijent forme jedne ekipe će se računati kao:

$$X_1 = \frac{1}{10}\sum_{i=1}^{10} X_{1i}$$

Izračunavanjem ove formule ćemo dobiti broj $X_1 \in [0, 1]$, koji predstavlja formu ekipe.

U slučaju da ekipa nije do trenutna izračunavanja odigrala nijednu utakmicu u ligi, forma se automatski postavlja na 0, a svakim idućim mečem se ažurira ta vrijednost u ovisnosti od ishoda meča. Onog momenta kada ekipa odigra 10 mečeva u ligi, automatska vrijednost koja je vrijedila na početku više ne vrijedi.

domaćin	gost	ishod
Chelsea F.C.	Manchester United F.C.	1
Burnley F.C.	Chelsea F.C.	0
Chelsea F.C.	Watford F.C.	1
Chelsea F.C.	Everton F.C.	1
Manchester City F.C.	Chelsea F.C.	1
Arsenal F.C.	Chelsea F.C.	0
Chelsea F.C.	Sunderland F.C.	-1
Southampton F.C.	Chelsea F.C.	-1
Leicester F.C.	Chelsea F.C.	0
Chelsea F.C.	Brighton & Hove Albion F.C.	1

Tabela 4.1 Primjer – računanje forme

U slučaju u tabeli xy, imamo predstavljene posljednjih 10 utakmica ekipe Chelsea F.C. Vidimo da, čak iako ima utakmica u kojima su protivničke ekipe favoriti, ima i dovoljno utakmica u kojima je i Chelsea favorit, a u kojima i nema favorita, pa zaključujemo da je veličina uzorka dovoljno velika. Kada bismo iskoristili formulu koju smo odredili za računanje forme ekipe, dobili bismo da je forma ove ekipe pred sljedeću utakmicu 0.65.

4.2 Međusobni omjer

Jedan od presudnih faktora, uz formu i još neke druge, za ishod jednog fudbalskog meča jeste svakako mešusobni omjer ekipa koje igraju. Znamo da u nekim velikim derbijima forma ne igra nikakvu ulogu, upravo zato što ta utakmica ima mnogo više motiva od samo obične pobjede protiv neke druge ekipe. Zbog toga je jako važno uzeti u obzir međusobni omjer ekipa koje igraju.

Međutim, ni ovdje ne možemo posmatrati sve utakmice sa istom težinom. Ne može istu važnost imati utakmica koja se odigrala prije 10 godina i utakmica koja se odigrala prije 10 dana, utakmice koje su skorije moraju imati veću važnost.

Međusobni omjer će biti koeficijent $X_2 \in [-1, 1]$ koji će se računati po formuli:

$$X_{2n} = \begin{cases} 1, n \leq 6 \wedge p = 1 \\ -1, n \leq 6 \wedge p = -1 \\ 0.8, n > 6 \wedge p = 1 \\ -0.8, n > 6 \wedge p = -1 \\ 0, ostalo \end{cases}$$

gdje je n redni broj utakmice (gledajući od sadašnjeg trenutka na unazad), p je pobjednik meča (ako je $p = 1$, pobjednik je ekipa1 itd.).

Finalna vrijednost koeficijenta X_2 se računa po formuli:

$$X_2 = \frac{1}{n}\sum_{i=1}^{n} X_{2i}$$

U slučaju da dvije ekipe nemaju nijedan međusobni meč u setu podataka, prednost se ne daje nikom, tj. koeficijent međusobnog omjera je jednak nuli.

n	domaćin	gost	ishod
1	Chelsea F.C.	Burnley F.C.	0
2	Burnley F.C.	Chelsea F.C.	-1
3	Chelsea F.C.	Burnley F.C.	1
4	Chelsea F.C.	Burnley F.C.	1
5	Burnley F.C.	Chelsea F.C.	0
6	Burnley F.C.	Chelsea F.C.	1
7	Chelsea F.C.	Burnley F.C.	1
8	Burnley F.C.	Chelsea F.C.	-1
9	Burnley F.C.	Chelsea F.C.	-1
10	Chelsea F.C.	Burnley F.C.	0

Tabela 4.2 Primjer – računanje međusobnog omjera

U slučaju u tabeli xy, imamo 10 posljednjih susreta ekipa Chelsea F.C. i Burnley F.C. Ukoliko bismo primijenili formule koje smo definisali za računanje koeficijenta međusobnog omjera, dobili bismo da je koeficijent međusobnog omjera na ovoj utakmici jednak 0.5 na stranu ekipe Chelsea F.C. Predznak koeficijenta će zavisiti od toga koja je ekipa u predstojećoj utakmici domaćin, tj. ako je Chelsea F.C. u predstojećoj utakmici domaćin, ovaj koeficijent će biti jednak 0.5, a ako je gost, onda će koeficijent biti jednak -0.5.đ

4.3 Prednost domaćeg terena

Mnoge ekipe u svjetskom fudbalu statistički bolje rezultate postižu na kućnom terenu. To naravno ne mora biti pravilo, tj. ne mora svaki tim svake sezone postizati bolje rezultate kod kuće, ali u većini slučajeva to jeste tako. Uzmimo za primjer Chelsea F.C. u prethodne dvije sezone (sezona 2016/17 i 2017/18). U sezoni 2016/17 je ekipa Chelsea F.C. osvojila 51 bod na domaćem terenu, a 42 boda na gostujućem. U sezoni 2017/18 su osvojili 37 bodova kod kuće, a 33 boda u gostima. Sve ovo ukazuje na to da ekipe zaista postižu bolje performanse na domaćem terenu, što znači da moramo modelirati svojstvo koje će se pobrinuti da prednost domaćeg terena bude uračunata u finalnu predikciju ishoda meča.

Koeficijent prednosti domaćeg terena će biti broj $X_3 \in [0, 1]$, a koji će se računati kao procenat pobjeda na domaćem terenu u odnosu na ukupni broj odigranih utakmica.

$$X_{3n} = \begin{cases} 1, & p = 1 \\ 0, & ostalo \end{cases}$$

gdje p označava pobjednika utakmice, u ovom slučaju domaćina.

Iz ove formule slijedi da će se koeficijent prednosti domaćeg terena računati kao:

$$X_3 = \frac{1}{n}\sum_{i=1}^{n} X_{3n}$$

gdje je n broj utakmica koje je jedna ekipa odigrala u Premier ligi.

Ako je neka ekipa odigrala 4 sezone u Premier ligi, tj. 152 utakmice, a pobijedila 74 od tih 152 utakmice, njen koeficijent prednosti domaćeg terena će biti $\frac{74}{152} \approx 0.487$.

4.4 Prosječne količine golova i kartona

U setu podataka koji se koristi su nam, pored konačnih ishoda mečeva i informacija o tome koja je ekipa domaćin, a koja gost, dostupni i podaci o tome koliko je šuteva imala koja ekipa, a koliko je od tih šuteva bilo u okvir gola i također nam nudi informaciju o tome koliko je žutih, a koliko crvenih kartona imala koja ekipa.

Sve ove minijaturne informacije mnogo utiču na krajnji ishod utakmice. U većini slučajeva je za očekivati da ekipa koja ima više šuteva, a pogotovo šuteva u okvir gola, ima veću šansu da pobijedi utakmicu. Također, ako jedna ekipa dobije crveni karton, onda ta ekipa ima igrača manje i automatski je vjerovatnoća za pobjedu njihovog protivnika veća.

Dakle, pored četiri svojstva koja su zasnovana na prethodnom izračunavanju, postojat će još 8 svojstava koja će se odnositi na šuteve, šuteve u okvir gola, žute kartone i crvene kartone respektivno. Ova svojstva će biti računata na najjednostavniji mogući način, pronalaskom aritmetičke sredine svih ovih vrijednosti za sve utakmice odigrane od strane jedne ekipe.

5 Implementacija, rezultati i diskusija

U ovom poglavlju će biti demonstriran rad svih prethodno opisanih algoritama na također prethodno opisanim setovima podataka. Ovo poglavlje, osim implementacije samih algoritama, uključuje također i proces učitavanja, kao i pripreme podataka za te algoritme. Cilj implementacije algoritama je uporediti performanse navedenih metoda pri predviđanju sportskih rezultata, kao i postizanje respektabilne tačnosti u odnosu na stvarne ishode sportskih događaja.

5.1 Okruženje

Praktični dio završnog rada je rađen u MATLAB R2016a okruženju. MATLAB je idealan izbor za egzekuciju praktičnog dijela ovog rada iz više razloga. Kao prvo, pruža podršku u vidu već implementiranih funkcija za sve metode korištene za previđanje rezultata u ovom radu. MATLAB je također jedan od najboljih alata za matematičke proračune, što je itekako korisno u procesu pripreme podataka.

5.2 Priprema podataka

Prije nego što počne priprema i sređivanje podataka, podaci se moraju učitati. U svrhu učitavanja podataka, korištena je skripta **readData.m**. Podaci su raspoređeni u devet .csv fajlova, za svaku od sezona Premier lige respektivno. Skripta **readData.m** svaki od ovih .csv fajlova učita u svoju posebnu matricu, a zatim sve te matrice spoji u jednu, koja se zove **fullData**. Pri tom, izbacuje prve redove iz svih ovih matrica jer su prvi redovi ustvari labele kolona svake od tih tabela.

Cijeli kod skripte je prikazan na sljedećoj slici:

```
[num,txt,prva] = xlsread('season-0910_csv.csv');
[num,txt,druga] = xlsread('season-1011_csv.csv');
[num,txt,treca] = xlsread('season-1112_csv.csv');
[num,txt,cetvrta] = xlsread('season-1213_csv.csv');
[num,txt,peta] = xlsread('season-1314_csv.csv');
[num,txt,sesta] = xlsread('season-1415_csv.csv');
[num,txt,sedma] = xlsread('season-1516_csv.csv');
[num,txt,osma] = xlsread('season-1617_csv.csv');
[num,txt,deveta] = xlsread('season-1718_csv.csv');

prva(1, :) = [];
druga(1, :) = [];
treca(1, :) = [];
cetvrta(1, :) = [];
peta(1, :) = [];
sesta(1, :) = [];
sedma(1, :) = [];
osma(1, :) = [];
deveta(1, :) = [];
```

```
fullData = [prva; druga; treca; cetvrta; peta; sesta; sedma; osma; deveta];
```

Slika 5.1 readData.m

Podaci da bi se koristili u MATLABovim numeričkim operacijama ili implementacijama algoritama mašinskog učenja, moraju biti strogo numerički. U ovom setu podataka postoje kolone koje nisu numeričkog tipa (string), kao što su npr. imena klubova, pa se na neki način te kolone moraju izbaciti. Da bi se znalo koje dvije ekipe igraju utakmicu bez njihovih imena, imena se moraju vezati za jedinstvene identifikatore. Za ovu svrhu, služe dvije skripte **teams.m** i **swapNamesIDs.m**.

Skripta **teams.m** prolazi redom kroz sve utakmice i za svaki novi tim dodaje red u tabelu **teamsNames** i svakom timu dodjeljuje jedinstveni identifikator. Cijeli kod ove skripte je na sljedećoj slici:

```
counter = 1;
teamsNames = ['team';];
for i = 1:size(fullData, 1)
    if ~any(ismember(teamsNames(:, 1), fullData(i, 2)))
        teamsNames = [teamsNames; fullData(i, 2)];
        counter = counter + 1;
    end
end
```

Slika 5.2 teams.m

Skripta **swapNamesIDs.m** prolazi kroz set podataka i na mjestima gdje su zapisana imena klubova upisuje njihove jedinstvene identifikatore, da bi se ovaj set podataka mogao efektivno koristiti. Cijeli kod skripte se nalazi na slici ispod:

```
for i = 1:size(fullData, 1)
    fullData(i, 2) = {find(ismember(teamsNames, fullData(i, 2)))};
    fullData(i, 3) = {find(ismember(teamsNames, fullData(i, 3)))};
    if any(ismember(fullData(i, 6), {'A'}))
        fullData(i, 6) = {-1};
    elseif any(ismember(fullData(i, 6), {'D'}))
        fullData(i, 6) = {0};
    else
        fullData(i, 6) = {1};
    end
end
```

Slika 5.3 swapNamesIDs.m

Osim imena klubova, postoje još neki suvišni podaci neodgovarajućeg tipa (string), kao što su npr. datum igranja utakmice ili ime sudije. Za čišćenje ostalih kolona se pobrinjava skripta **excessData.m**, čiji se kod nalazi na slici ispod:

```
fullData(:, 1) = [];
fullData(:, 8) = [];
fullData(:, 8) = [];
```

Slika 5.4 excessData.m

Kada se uspješno izbace svi suvišni i neodgovarajući podaci, početna fullData matrica se pretvara u običnu numeričku matricu, pogodnu za razne numeričke manipulacije uz pomoć skripte **convertCellToMatrix.m**, čiji se kod nalazi na slici ispod:

```
fullData = cell2mat(fullData);
```

Slika 5.5 convertCellToMatrix.m

5.3 Izračunavanje svojstava

Kao što je opisano u poglavlju 4, sva svojstva koja će biti korištena za algoritme učenja za predviđanje ishoda fudbalskih mečeva zahtijevaju prethodnu računicu.

Jedno od svojstava je prosječni broj šuteva jedne ekipe po utakmici. Ovo svojstvo se računa uz pomoć skripte **calculateShots.m.** Ova skripta prolazi redom kroz sve mečeve u setu podataka, a uporedo u matricu **matrixShots** pohranjuje podatke o svim ekipama respektivno, tj. podatke o ukupnom broju šuteva i ukupnom broju odigranih utakmica jedne ekipe. Ovi podaci su kasnije korišteni pri popunjavanju finalnog seta podataka. Cijeli kod ove skripte se nalazi na slici ispod:

```
matrixShots = [1, 0, 0; 2, 0, 0; 3, 0, 0; 4, 0, 0; 5, 0, 0;
               6, 0, 0; 7, 0, 0; 8, 0, 0; 9, 0, 0; 10, 0, 0;
               11, 0, 0; 12, 0, 0; 13, 0, 0; 14, 0, 0; 15, 0, 0;
               16, 0, 0; 17, 0, 0; 18, 0, 0; 19, 0, 0; 20, 0, 0;
               21, 0, 0; 22, 0, 0; 23, 0, 0; 24, 0, 0; 25, 0, 0;
               26, 0, 0; 27, 0, 0; 28, 0, 0; 29, 0, 0; 30, 0, 0;
               31, 0, 0; 32, 0, 0; 33, 0, 0; 34, 0, 0; 35, 0, 0;
               36, 0, 0];

for i = 1:size(fullData, 1)
    matrixShots(fullData(i, 1)-1, 2) = matrixShots(fullData(i, 1)-1, 2) +
fullData(i, 8);
    matrixShots(fullData(i, 2)-1, 2) = matrixShots(fullData(i, 2)-1, 2) +
fullData(i, 9);
    matrixShots(fullData(i, 1)-1, 3) = matrixShots(fullData(i, 1)-1, 3) + 1;
    matrixShots(fullData(i, 2)-1, 3) = matrixShots(fullData(i, 2)-1, 3) + 1;
end
```

Slika 5.6 calculateShots.m

Što se tiče šuteva u okvir gola, žutih kartona i crvenih kartona, ta svojstva se računaju po istom principu kao i šutevi. Razlike su u tome što se u tim slučajevima popunjavanju matrice

matrixShotsOnTarget, matrixYellows i **matrixReds** respektivno, a podaci se izvlače iz različitih kolona matrice fullData.

Nakon proračunavanja jednostavnih svojstava, računaju se malo kompleksnija svojstva. Prvo od tih svojstava je forma. Forma se računa na način već opisan u poglavlju 4. Dakle, skripta **calculateForm.m** prolazi redom kroz sve utakmice i po odgovarajućoj formuli računa forme domaće i gostujuće ekipe za svaku utakmicu. Cijeli kod ove skripte se nalazi na sljedećoj slici:

```
formColumn = zeros(3420, 1);
fullData = [fullData formColumn formColumn];

for i = 1:size(fullData, 1)
    homeTeam = fullData(i, 1);
    awayTeam = fullData(i, 2);
    homeTeamForm = 0;
    awayTeamForm = 0;
    counterHomeTeam = 0;
    counterAwayTeam = 0;
    for j = i-1:-1:1
        if fullData(j, 1) == homeTeam || fullData(j, 2) == homeTeam
            if counterHomeTeam >= 10
                break;
            end
            counterHomeTeam = counterHomeTeam + 1;
            if fullData(j,1) == homeTeam
                if fullData(j, 3) == 1
                    homeTeamForm = homeTeamForm + 1;
                elseif fullData(j, 3) == 0
                    homeTeamForm = homeTeamForm + 0.5;
                else
                    homeTeamForm = homeTeamForm;
                end
            else
                if fullData(j, 3) == -1
                    homeTeamForm = homeTeamForm + 1;
                elseif fullData(j, 3) == 0
                    homeTeamForm = homeTeamForm + 0.5;
                else
                    homeTeamForm = homeTeamForm;
                end
            end
        end
    end
    for j = i-1:-1:1
        if fullData(j, 1) == awayTeam || fullData(j, 2) == awayTeam
            if counterAwayTeam >= 10
                break;
            end
            counterAwayTeam = counterAwayTeam + 1;
            if fullData(j,1) == awayTeam
                if fullData(j, 3) == 1
```

```
                    awayTeamForm = awayTeamForm + 1;
                elseif fullData(j, 3) == 0
                    awayTeamForm = awayTeamForm + 0.5;
                else
                    awayTeamForm = awayTeamForm;
                end
            else
                if fullData(j, 3) == -1
                    awayTeamForm = awayTeamForm + 1;
                elseif fullData(j, 3) == 0
                    awayTeamForm = awayTeamForm + 0.5;
                else
                    awayTeamForm = awayTeamForm;
                end
            end
        end
    end
    if counterHomeTeam > 0
        fullData(i, 4) = homeTeamForm / counterHomeTeam;
    end
    if counterAwayTeam > 0
        fullData(i, 5) = awayTeamForm / counterAwayTeam;
    end

end
```

Slika 5.7 calculateForm.m

Nakon forme, računa se koeficijent međusobnog omjera dvije ekipe. I on se, kao i koeficijent forme računa po formuli definisanoj u poglavlju 4, tako što skripta **calculateH2H.m** prolazi redom kroz sve utakmice i na osnovu čitave historije računa taj koeficijent. Cijeli kod ove skripte se nalazi na slici ispod:

```
formColumn = zeros(3420, 1);
fullData = [fullData formColumn];

for i = 1:size(fullData, 1)
    homeTeam = fullData(i, 1);
    awayTeam = fullData(i, 2);
    headToHead = 0;
    counter = 0;

    for j = i-1:-1:1
        if (fullData(j, 1) == homeTeam && fullData(j, 2) == awayTeam) ||
(fullData(j, 1) == awayTeam && fullData(j, 2) == homeTeam)
            counter = counter +  1;
            if fullData(j, 1) == homeTeam
                if fullData(j, 3) == 1
                    if counter <= 6
                        headToHead = headToHead + 1;
                    else
```

```
                        headToHead = headToHead + 0.8;
                    end
                elseif fullData(j, 3) == 0

                else
                    if counter <= 6
                        headToHead = headToHead - 1;
                    else
                        headToHead = headToHead - 0.8;
                    end
                end
            else
                if fullData(j, 3) == 1
                    if counter <= 6
                        headToHead = headToHead - 1;
                    else
                        headToHead = headToHead - 0.8;
                    end
                elseif fullData(j, 3) == 0

                else
                    if counter <= 6
                        headToHead = headToHead + 1;
                    else
                        headToHead = headToHead + 0.8;
                    end
                end
            end
        end
    end

    if counter > 0
        fullData(i, 6) = headToHead / counter;
    end

end
```

Slika 5.8 calculateH2H.m

Posljednje svojstvo koje se računa jeste koeficijent prednosti domaćeg terena. Kao i ostala svojstva, i on se računa po formuli definisanoj u poglavlju 4. Za računanje ovog koeficijenta se koristi skripta **calculateHomeAdvantage.m**, čiji se kod nalazi na sljedećoj slici:

```
homeAdvantageColumn = zeros(3420, 1);
fullData = [fullData homeAdvantageColumn];

for i = 1:size(fullData, 1)
    homeTeam = fullData(i, 1);
    counter = 0;
    homeAdvantage = 0;
```

```
    for j = i-1:-1:1
        if fullData(j, 1) == homeTeam
            counter = counter + 1;
            if fullData(j, 3) == 1
                homeAdvantage = homeAdvantage + 1;
            end
        end
    end

    if counter > 0
        fullData(i, 7) = homeAdvantage / counter;
    end


end
```

Slika 5.9 calculateHomeAdvantage.m

Nakon izračunavanja svih potrebnih svojstava i pohranjivanja koeficijenata forme, međusobnog omjera i prednosti domaćeg terena u finalni set podataka, potrebno je pohraniti i ostale već izračunate podatke u set podataka (broj šuteva, broj žutih kartona…). Za ovu svrhu se koristi skripta **fillData.m**, koja se pobrinjava da se svi potrebni podaci upišu u finalni set podataka. Kompletan kod ove skripte se nalazi na slici ispod:

```
dataColumn = zeros(3420, 1);
fullData = [fullData dataColumn dataColumn dataColumn dataColumn dataColumn
dataColumn dataColumn dataColumn];

for i = 1:size(fullData, 1)
    homeShots = double(matrixShots(fullData(i, 1)-1, 2)) /
double(matrixShots(fullData(i, 1)-1, 3));
    awayShots = double(matrixShots(fullData(i, 2)-1, 2)) /
double(matrixShots(fullData(i, 2)-1, 3));
    homeShotsOnTarget = double(matrixShotsOnTarget(fullData(i, 1)-1, 2)) /
double(matrixShotsOnTarget(fullData(i, 1)-1, 3));
    awayShotsOnTarget = double(matrixShotsOnTarget(fullData(i, 2)-1, 2)) /
double(matrixShotsOnTarget(fullData(i, 2)-1, 3));
    homeYellows = double(matrixYellows(fullData(i, 1)-1, 2)) /
double(matrixYellows(fullData(i, 1)-1, 3));
    awayYellows = double(matrixYellows(fullData(i, 2)-1, 2)) /
double(matrixYellows(fullData(i, 2)-1, 3));
    homeReds = double(matrixReds(fullData(i, 1)-1, 2)) /
double(matrixReds(fullData(i, 1)-1, 3));
    awayReds = double(matrixReds(fullData(i, 2)-1, 2)) /
double(matrixReds(fullData(i, 2)-1, 3));

    fullData(i, 8) = homeShots;
    fullData(i, 9) = awayShots;
    fullData(i, 10) = homeShotsOnTarget;
    fullData(i, 11) = awayShotsOnTarget;
```

```
    fullData(i, 12) = homeYellows;
    fullData(i, 13) = awayYellows;
    fullData(i, 14) = homeReds;
    fullData(i, 15) = awayReds;
end
```

Slika 5.10 fillData.m

5.4 Egzekucija odabranih metoda

Nakon učitavanja i pripreme podataka, ti podaci su spremni da se koriste za treniranje algoritama mašinskog učenja. Međutim, prije nego što se počnu trenirati algoritmi mašinskog učenja, podaci trebaju biti podijeljeni na dva dijela, trening dio i testni dio.

Podjelu podataka na trening i testni dio vrši skripta **splitData.m**, čiji je kod prikazan na sljedećoj slici:

```
trainingData = fullData(1:2394, :);
testData = fullData(2395:end, :);
```

Slika 5.11 splitData.m

Dakle, skripta splitData.m vrši podjelu ulaznih podataka na ova dva dijela, ali da bi se mogle koristiti MATLABove funkcije treniranja algoritama, moraju se i izlazni podaci podijeliti na ova dva dijela, što je posao skripte **outputDataSplit.m**, čiji je kod prikazan na sljedećoj slici:

```
ModelNB = fitcnb(trainingData, classificationOutputTraining);
```

Slika 5.12 outputDataSplit.m

Nakon što su podaci podijeljeni na ova dva dijela, može se pristupiti treniranju algoritama mašinskog učenja na ovom setu podataka.

Prva metoda je metoda k-najbližih susjeda. Nakon detaljnog testiranja odabran je broj od 20 susjeda jer je tada algoritam davao najpreciznije rezultate. Za treniranje algoritma k-najbližih susjeda se koristi skripta **trainKNN.m**, čiji je kod prikazan na sljedećoj slici:

```
ModelKNN = fitcknn(trainingData, classificationOutputTraining,
'NumNeighbors', 20);
```

Slika 5.13 trainKNN.m

Nakon treniranja metode, izlazna vrijednost je *ModelKNN*, a to je varijabla koja se može koristiti za predviđanje ishoda bilo kojeg meča koji se nađe na ulazu u budućnosti.

Sljedeća metoda je Naivni Bayes klasifikator. Za treniranje ove metode je korištena skripta **trainNB.m**, čiji se kod nalazi na slici ispod:

```
ModelNB = fitcnb(trainingData, classificationOutputTraining);
```

Slika 5.14 trainNB.m

Kao i kod metode k-najbližih susjeda, izlazna vrijednost *ModelNB*, koji se može koristiti za predviđanje ishoda bilo kojeg meča koji se nađe na ulazu.

Posljednja odabrana metoda klasifikacije je vještačka neuronska mreža. Vještačke neuronske mreže u MATLABu zahtijevaju nešto drugačiji format podataka od ostalik algoritama mašinskog učenja, pa su podaci iz našeg seta podataka prilagođeni za vještačke neuronske mreže u skripti **prepareDataForNN.m**. Cijeli kod ove skripte se nalazi na slici ispod:

```
nnData = transpose(fullData);
nnOutputData = zeros(3, 3420);
for i=1:3420
    if classificationOutput(i) == 1
        nnOutputData(1, i) = 1;
    elseif classificationOutput(i) == 0
        nnOutputData(2, i) = 1;
    else
        nnOutputData(3, i) = 1;
    end
end
```

Slika 5.15 prepareDataForNN.m

Nakon što se podaci formatiraju na način kako to zahtijevaju vještačke neuronske mreže, onda se iste treniraju uz pomoć skripte **trainNN.m.** Kod ove skripte se nalazi na sljedećoj slici:

```
net = patternnet(15);

[net, tr] = train(net, nnData, nnOutputData);
nntraintool
```

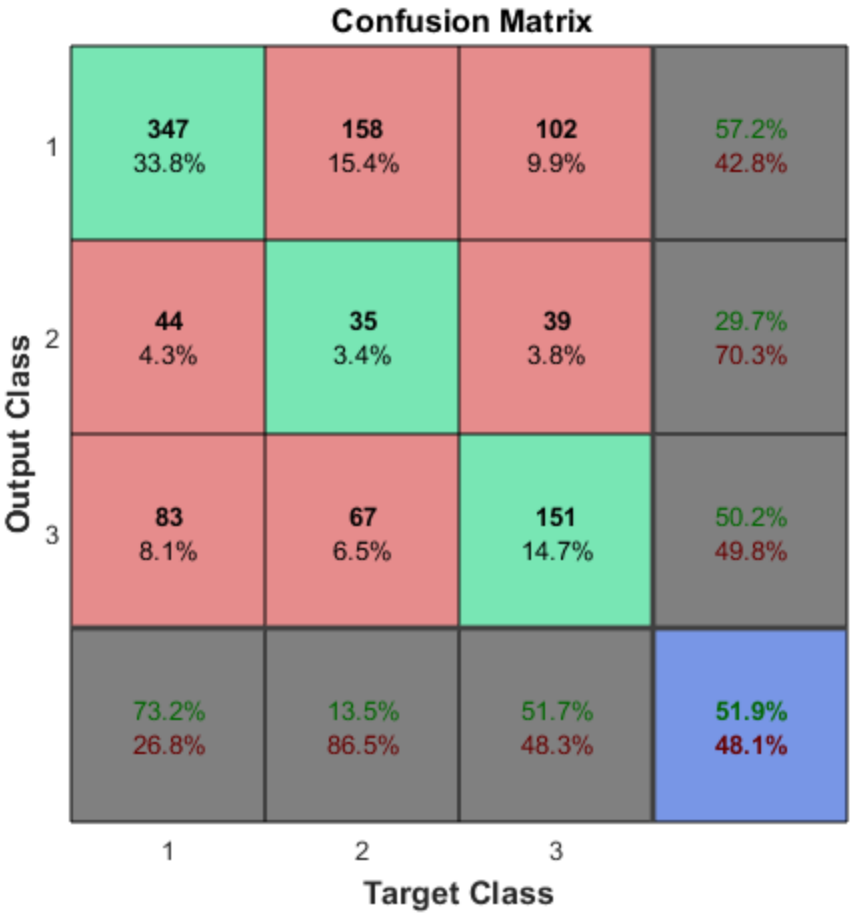
Slika 5.16 trainNN.m

Nakon implementacije svih ovih algoritama, dobijeni su respektabilni rezultati, koji će biti obrađeni i prodiskutovani u sljedećem poglavlju.

5.5 Rezultati i diskusija

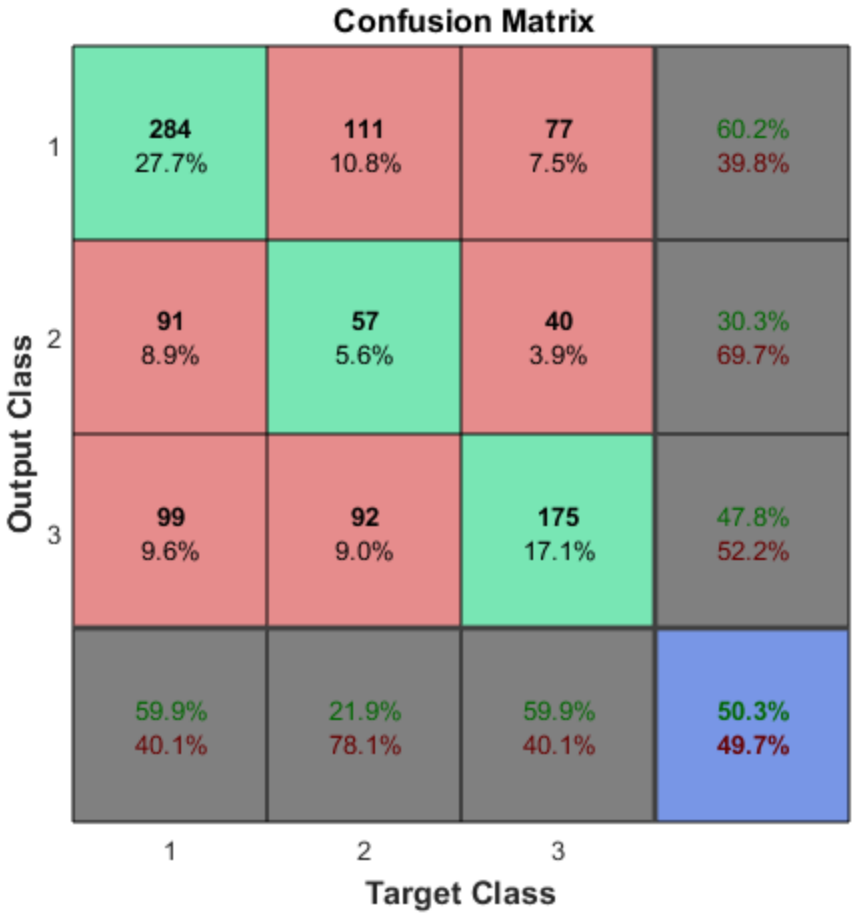
Budući da je ovaj problem problem klasifikacije na tri klase, svaka tačnost ispod 33.3% ne bi bila prihvatljiva. Imajući u vidu da set podataka nije preveliki, a da ni dimenzije ulaza nisu prevelike, očekivana preciznost ovih algoritama i nije baš visoka. Preciznost metoda će biti predstavljena brojčano, a zatim i sa matricama konfuzije (eng. confusion matrix).

Metoda k-najbližih susjeda je nakon treniranja nad testnim setom imala 51.9% tačnosti izlaza. Matrica konfuzije za metodu k-najbližih susjeda izgleda ovako:



Slika 5.17 Matrica konfuzije – k-najbližih susjeda

Zatim, metoda Naivni Bayes ima finalnu tačnost izlaza od 50.3%, što je neznatno lošije od prve metode. Njena matrica konfuzije izgleda ovako:



Slika 5.18 Matrica konfuzije – Naivni Bayes

I na kraju, sveukupna tačnost izlaza trenirane vještačke neuronske mreže iznosi 53.3%, dok je tačnost samo nad testnim dijelom seta podataka 52.2%. Matrica konfuzije vještačke neuronske mreže izgleda ovako:

Slika 5.19 Matrice konfuzije – vještačka neuronska mreža

Iz priloženog vidimo da se nijedna od izabranih metoda ne ističe po tačnosti, tj. svaka od metoda ima približno istu tačnost. No, ipak, vještačka neuronska mreža na kraju ipak najpreciznije predviđa ishode mečeva, dok je Naivni Bayes klasifikator metoda sa najmanjom tačnošću.

6 Zaključak

Prvobitni cilj ovog rada je bio u biti provjeriti da li je uopće moguće napraviti sistem koji predviđa sportske rezultate sa jako visokim nivoom tačnosti. Odabrane su metode koje su se nakon dugog razmišljanja pokazale kao konceptualno najpogodnije rješenje ovog problema. Nakon odabira metoda, iste su detaljno teoretski obrađene i izučene, kao i sve ono što te metode povlače za sobom.

Od odabira skupa ulaznih podataka, pa do odabira svojstava, svaki korak u implementaciji ovih algoritama je na kraju doprinio rezultatima. Čitav proces učitavanja, pripreme i proračunavanja podataka je na kraju kulminirao implementacijom tri metode klasifikacije koje su dale respektabilne rezultate predviđanja rezultata, pogotovo uzimajući u obzir mali broj ulaznih podataka i malu dimenzionalnost problema.

Od svih implementiranih metoda, najbolje rezultate je dala vještačka neuronska mreža, iako ni preostale dvije metode nisu postigle mnogo lošiju preciznost od iste. Performanse svih algoritama su detaljno predstavljene matricama konfuzije, sa kojih se jasno vidi da je vještačka neuronska mreža dala najbolje rezultate.

Vrlo vjerovatno bi veći broj ulaznih podataka, a i veći broj svojstava na ulazu dao mnogo bolje rezultate. Također je vrlo moguće da bi neki drugi algoritam sa istim ovim ulaznim podacima dao mnogo bolje rezultate od odabranih algoritama, ali možda i ne bi.

Finalni zaključak ovog rada je da se mašinsko učenje samo po sebi poprilično dobro nosi sa predviđanjem ishoda sportskih mečeva, a sa dovoljno velikim i detaljnim skupom podataka možda nekada u budućnosti bude u mogućnosti da predviđa ishode sa apsolutnom sigurnošću.

7 Literatura

[1]	Kevin P. Murphy - *Machine Learning: A Probabilistic Perspective*, Massachusetts Institute of Technology, 2012.
[2]	Nils J. Nilsson: *Introduction to Machine Learning*, Stanford University, 2005.
[3]	Izabela Moise, Evangelos Pournaras, Dirk Helbing: *K-Nearest Neighbour Classifier.* Dostupno na: https://www.ethz.ch/content/dam/ethz/special-interest/gess/computational-social-science-dam/documents/education/Spring2015/datascience/K-Nearest-Neighbour-Classifier.pdf
[4]	Tom M. Mitchell: *Machine Learning*, 1997.
[5]	David Reby, Sovan Lek, Ioannis Dimopoulos, Jean Joachim, Jacques Lauga, Stéphane Aulagnier: *Artificial neural networks as a classification method in the behavioural sciences,* 1996. Dostupno na: http://www.lifesci.sussex.ac.uk/cmvcr/Publications_files/rebyproc.pdf
[6]	Niek Tax, Yme Joustra: *Predicting The Dutch Football Competition Using Public Data: A Machine Learning Approach*, 2015.
[7]	Igiri, Chinwe Peace; Nwachukwu, Enoch Okechukwu: *An Improved Prediction System for Football a Match Result*, 2014.
[8]	Michal Sipko: *Machine Learning for the Prediction of Professional Tennis Matches, 2015.*
[9]	Ville Sillanpää, Olli Heino: *Forecasting football match results – A study on modeling principles and efficiency of fixed-odds betting markets in football*, 2013.