SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 617

Predviđanje ishoda odbojkaških utakmica korištenjem metoda strojnog učenja

Domagoj Matošević

Zagreb, lipanj 2022.

Umjesto ove stranice umetnite izvornik Vašeg rada.
Da bi ste uklonili ovu stranicu obrišite naredbu `\izvornik`.

SADRŽAJ

1. Uvod

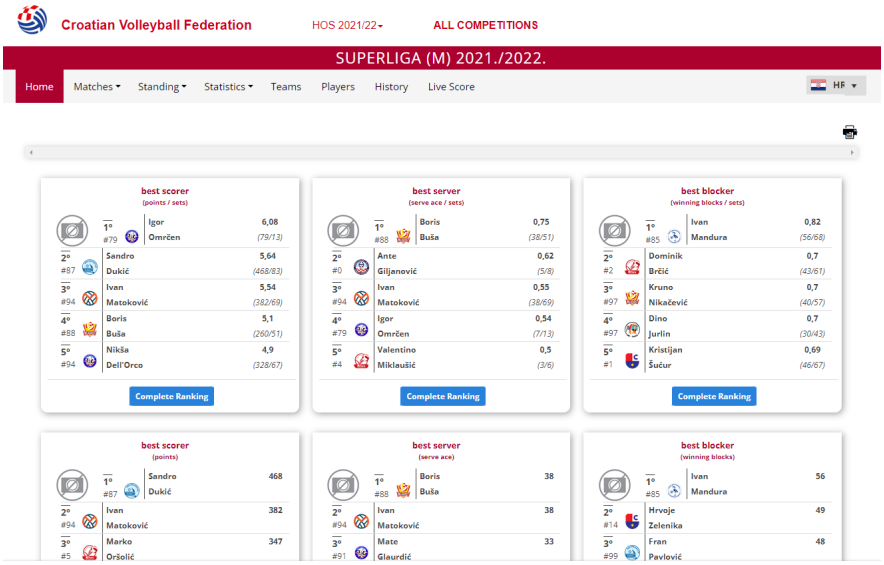
Odbojka je jedan od onih sportova u Hrvatskoj koji ne dobiva previše pažnje u usporedbi sa npr. košarkom ili nogometom. Muška odbojka još i manje. Zbog tog razloga malo ljudi uopće zanima raditi predviđanja ishoda odbojkaških utakmicama. Ipak, postoji veća potražnja za saznanjem o ishodu rezultata utakmice. Puno odbojkaša i odbojkaša rekreativaca prati rezultate hrvatske odbojke te bi im bilo zanimljivije da uz najavu slijedeće utakmice imaju i neko predviđanje koja će ekipa najvjerojatnije pobijediti.

Cilj ovog rada je predviđanje pobjednika odbojkaške utakmice radi davanja novih informacija o nadolazećim utakmicama za koje postoji potražnja od strane ljudi koji prate hrvatsku mušku odbojku.

Rad je podijeljen u osam cjelina. U 2. cjelini opisan je problem, tj. ono što se namjerava riješiti ovim radom. 3. cjelina opisuje tehnologije koje su korištene pri izradi ovog rada. U 4. cjelini opisano je sve što se tiče dohvata podataka, konkretno na kojim internet stranicama se ti podaci nalaze te načini na koju su se oni dohvatili. 5. cjelina sadrži opis čišćenja podataka, načini na koji su oni pripremljeni za korištenje u algoritmima i klasifikacija podataka. U 6. cjelini nalazi se izračun novih podataka iz postojećih poput učestalost pobjede određenog kluba. 7. cjelina se odnosi na algoritme koji su korišteni i rezultati koji su dobiveni njihovim korištenjem. Algoritmi poput Naivan Bayesov klasifikator opisani su u toj cjelini. 8. i zadnja cjelina je zaključak.

2. Opis problema

Problem koji ovaj rad želi riješiti je taj da nema predviđanja rezultata utakmica hrvatske odbojkaške lige. Ljudi mogu vidjeti prijašnje rezultate ili najave budućih utakmica na stranicama poput https://hos-web.dataproject.com/[1] ili https://natjecanja.hos-cvf.hr/[4]. Moguće je čak vidjeti analize utakmica i koliko je koji igrač osvojio bodova, ali nigdje nekakva predviđanja ishoda utakmica. Odbojka nije toliko popularan sport u Hrvat-



Slika 2.1: naslovna stranica HOS Data Project

skoj. Broj gledatelja je manji nego za nogomet ili košarku te je zato manje resursa uloženo u predviđanje rezultata utakmica. Unatoč tome publika za taj sport postoji. Također, postoji potražnja za nečim poput predikcije rezultata utakmice koja će se dogoditi uskoro. Najčešće se to odnosi na ekipe koje su podjednake na ljestvici, jer ljudi mogu predvidjeti tko će pobjediti ako je razlika mjesta na poziciji velika. Problem je sada sljedeći. Kako dohvatiti podatke o utakmicama i spremiti ih u obliku koji se može koristiti pri metodama strojnog učenja? Kako iz podataka o utakmicama procjeniti tko će biti pobjednik sljedeće utakmice? U nastavku će biti objašnjen cijeli proces dohvata, spremanja i korištenja podataka.

Slika 2.2: naslovna stranica Hrvatskih Odbojkaških Natjecanja

3. Korištene tehnologije

3.1. Web Scraper

Web Scraper[6] je plugin alat za Google Chrome koji omogućava struganje podataka s internetskih stranica. Njegovo korištenje je moguće bez ikakvog programerskog znanja. Ovaj alat koristi objekte zvane Selectors koji označavaju objekte na stranici iz kojih se izvlače tekstni podaci. Omogućuje skupljanje podatka s internet stranica u obliku csv i xlsx datoteka.

3.2. pandas

Pandas[10] je biblioteka, najčešće korištena u Pythonovim programima, koja sadrži definicije funkcija i objekata pogodnih za analizu i obradu podatka. Prva verzija pandasa je objavljena 11. siječnja 2008 kao slobodni softver pod licencom BSD. Ime je dobiveno iz izraza "panel data". Korištenjem DataFramea, tablične strukture podatka, pandas omogućava korištenje podataka različitih formata poput tablica iz Micorsoft Excela ili tekstualnih podataka razdvojenih zarezom.

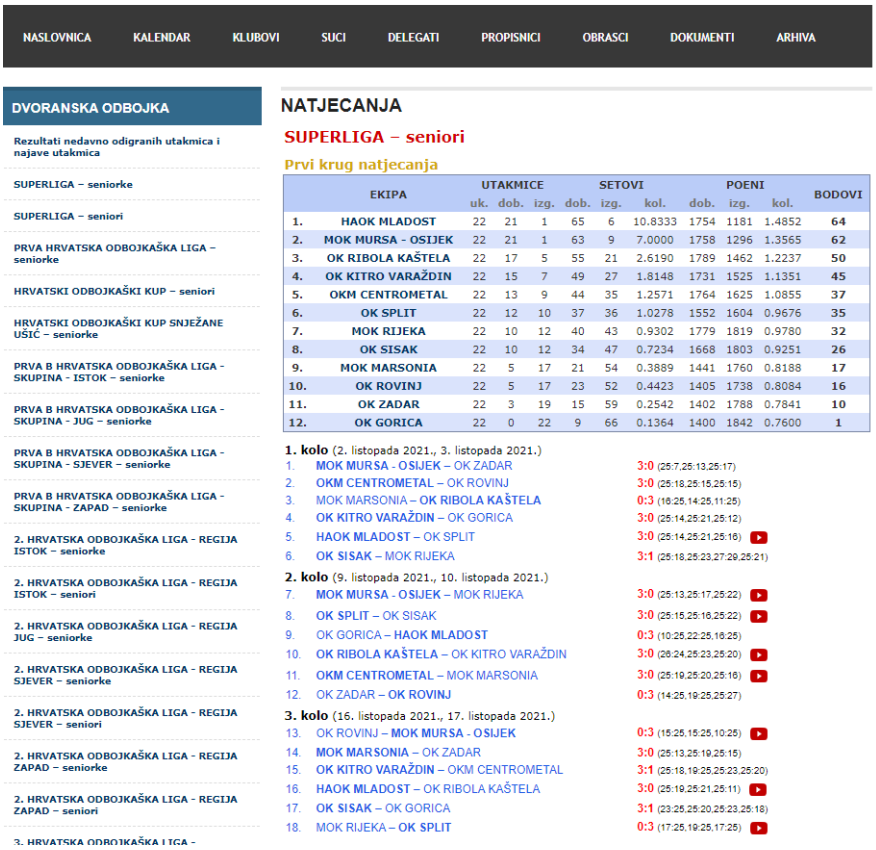
3.3. scikit-learn

Scikit-learn[11], isto poznat kao sklearn je Pythonova biblioteka korištena za primjenu algoritama strojnog učenja. Prva verzija je objavljena 2007. godine od strane francuskog podatkovnog znastvenika David Cournapeaua. Podupire algoritme strojnog učenja poput Naivnog Bayesovog klasifikatora i nasumičnih šuma koji su korisni u analizi podataka.

4. Dohvat podataka

4.1. Stranice dohvata podataka

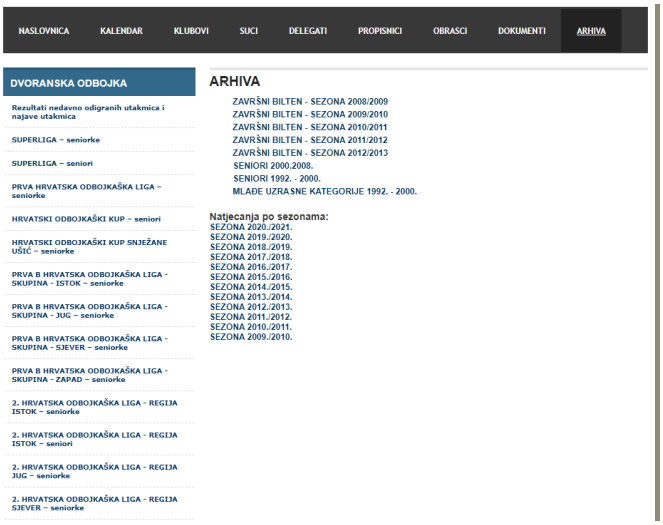
Podaci o svim utakmicama mogu se naći na internetskoj stranici https://natjecanja.hos-cvf.hr/. Na lijevoj strani stranice nalazi se lista linkova koji vode do podataka o trenutnim odbojkaškim ligama. Klikom na link na kojem piše "SUPERLIGA – seniori"[5] ili "HRVATSKI ODBOJKAŠKI KUP – seniori"[3] dolazi se na listu podataka o trenutnoj sezoni muške Superlige ili muškog Hrvatskog Odbojkaškog kupa. U gornjem desnom



Slika 4.1: stranica rezultata i najava utakmica Superlige

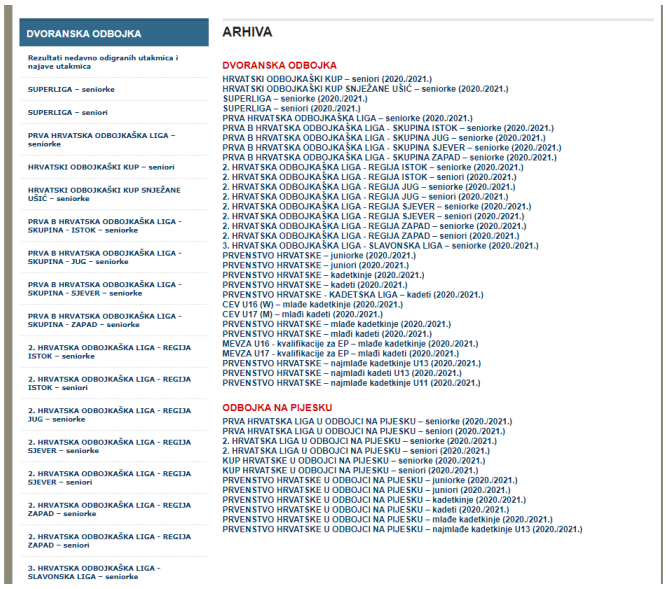
kutu nalazi se gumb na kojem piše ARHIVA[2]. Klikom na njega dolazi se na stranicu

na kojoj se nalazi lista prošlih sezona svih odbojkaških natjecanja. Klikom na jednu



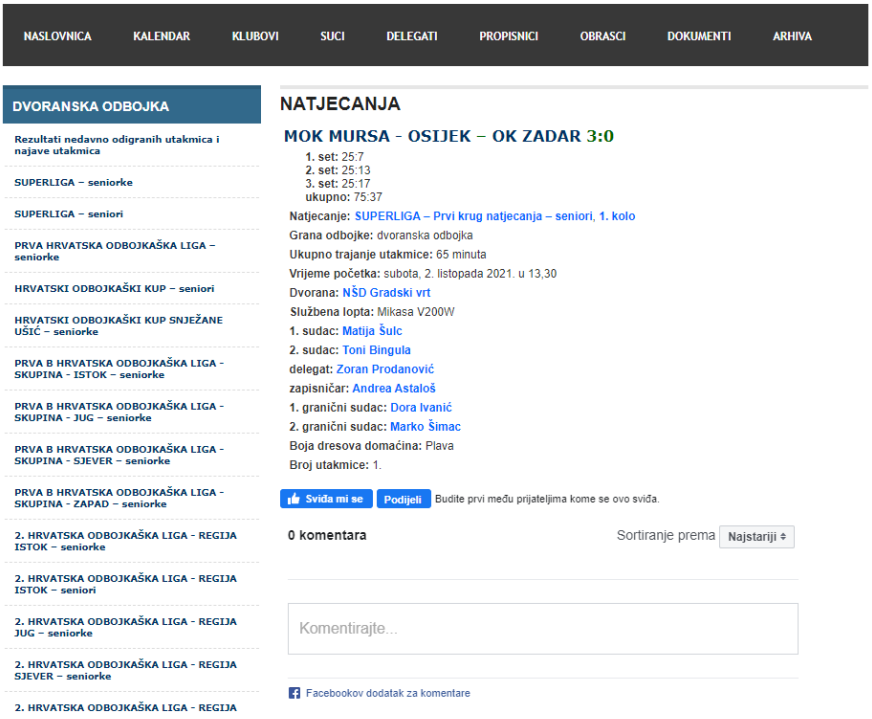
Slika 4.2: stranica ARHIVA

od sezona dobivamo listu natjecanja koja su se odvijala te sezone. Na listi se nalaze dva linka koja su ovdje važna, a to su "SUPERLIGA – seniori (sezona)" i "HRVATSKI ODBOJKAŠKI KUP – seniori (sezona)". Ta dva linka vode na stranice gdje se nalaze podaci o tim natjecanjima za tu sezonu. Taj proces odabira natjecanja u arhivi napravljen je za sezone 2019./2020. i 2020./2021., a za sezonu 2021./2022. korišten je proces odabira trenutne sezone. Na stranici svake sezone Superlige nalazi se tablica poretka



Slika 4.3: stranica natjecanja u sezoni 2020./2021.

ekipa za svaki dio natjecanja i sve utakmice koje su se odigrale te sezone. Klikom na



Slika 4.4: stranica podatka o utakmici

utakmicu dolazi se na stranicu gdje su podaci o toj utakmici. Na stranici svake sezone Hrvatskog Odbojkaškog Kupa nalaze se sve utakmice koje su se odigrale te sezone. Klikom na utakmicu dolazi se na stranicu gdje su podaci o toj utakmici.

4.2. Struganje podataka

Za dohvat podataka korišten je plugin za Chrome alat naziva Web Scraper. Za svaku stranicu gdje se prikupljaju podaci napravljen je Sitemap s nazivom 'hos_natjecanja_NAZIV NATJECANJA_SEZONA'. U svakom Sitemapu nalazi se Selector tipa SelectorLink

ID	Domain
hos_natjecanja_kup_20192020	natjecanja.hos-cvf.hr
hos_natjecanja_kup_20202021	natjecanja.hos-cvf.hr
hos_natjecanja_kup_20212022	natjecanja.hos-cvf.hr
hos_natjecanja_superliga_20192020	natjecanja.hos-cvf.hr
hos_natjecanja_superliga_20202021	natjecanja.hos-cvf.hr
hos_natjecanja_superliga_20212022	natjecanja.hos-cvf.hr

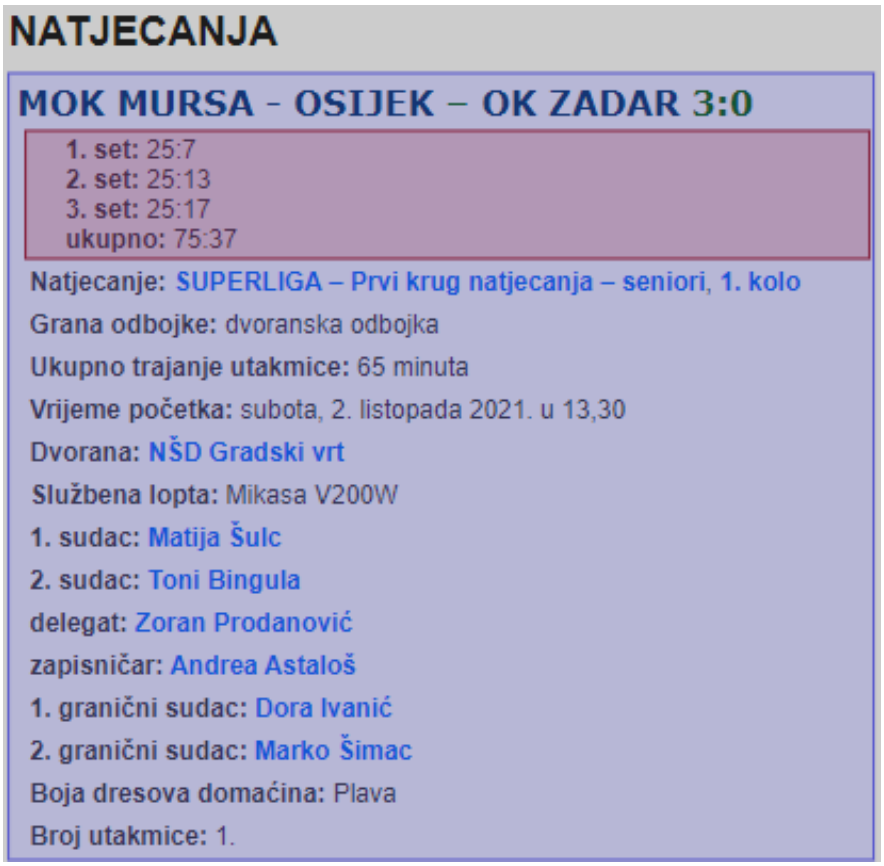
Slika 4.5: Sitemapeovi za svako natjecanje

koji odabire svaku utakmicu na stranici i time podatke o nazivima ekipa. Unutar tog Selectora nalazi se Selector tipa SelectorElement naziva koji sadrži dva Selectora tipa SelectorText koji sadrže podatke o datumu i osvojenim bodovima svake ekipe u svakom setu. Podatke onda dohvatimo klikom opcije 'Scrape'. Zatim nakon nekog vre-

mena Web Scraper dohvati sve te podatke i spremimo ih u obliku CSV datoteke.

Slika 4.6: linkovi na stranice utakmica

Slika 4.7: podaci o datumu označeni crvenim kvadratom



Slika 4.8: podaci o bodovima u setu označeni crvenim kvadratom

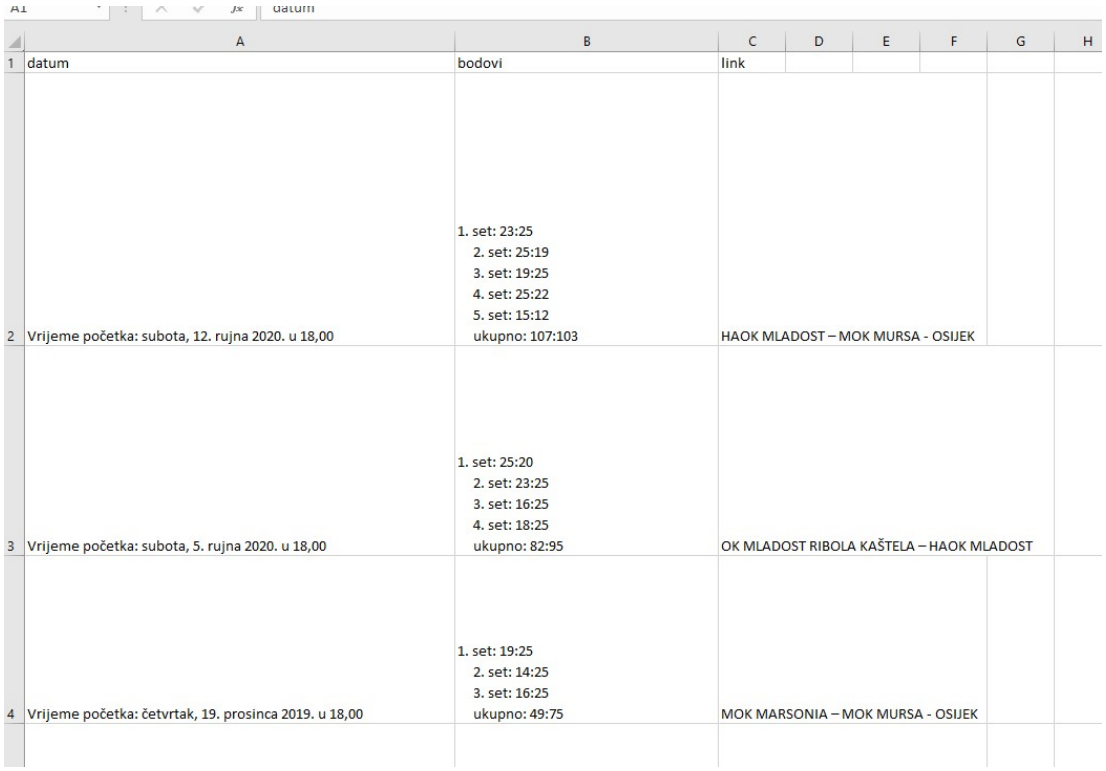
5. Čišćenje podataka

5.1. Priprema podataka za klasificiranje

Podaci koji su dohvaćeni nisu dobri za korištenje u algoritmima strojnog učenja. Prije toga potrebno je 'očistiti' podatke tj. riješiti se onoga što je višak, izvući iz podataka ono što nam je stvarno potrebno i spremiti podatke u obliku koji nam odgovara za potrebe korištenja. Dohvaćeni podaci imaju nekoliko stupaca viška koji uopće nisu potrebni te je njih potrebno obrisati. Time dobivamo csv datoteku koja sadrži tri stupca koji se zovu: datum, bodovi, i link. Datum sadrži datum i vrijeme početka utakmice. Bodovi sadrži rezultat svakog seta i ukupni broj osvojenih bodova svake ekipe u toj utakmici. Link je sadržaj linka koji je vodio do podataka o utakmici i on sadrži imena ekipa. Ovakve podatke unose se u Pythonov program koji će te podatke pretvoriti u oblik pogodan za korištenje u algoritmima strojnog učenja.

5.2. Klasificiranje podataka

Podatke klasificiramo po tome je li pobijedio domaćin ili gost. Vrijednost za pobjedu domaćina je 0, a vrijednost za pobjedu gosta je 1. U podacima imamo koliko je bodova svaka ekipa osvojila u svakom setu te sveukupno u utakmici. Da bi se pobjednik odredio treba odrediti koliko je svaka ekipa pobijedila setova tj. koliko svaka ekipa ima setova s većim brojem bodova od druge. Uz to još treba pretvoriti ostale podatke u oblike koje želimo npr. datum u yyyy-mm-dd. Datum je napisan u obliku 'DAN_U_TJEDNU, BROJ_U_MJESECU. NAZIV_MJESECA GODINA. u VRIJEME_POČETKA'(npr. subota, 19. rujna 2020. u 17,00). Bodovi su zapisani u obliku '1. set: REZULTAT_SETA NOVI RED 2. set: REZULTAT_SETA NOVI RED 3. set: REZULTAT_SETA NOVI RED (može ići do 5.seta) ukupno: UKUPNO_OSVOJENIH_BODOVA_SVAKE_EKIPE'. Link je zapisan u obliku 'DOMACIN - GOST'



	A	B	C	D	E	F	G	H
1	datum	bodovi	link					
2	Vrijeme početka: subota, 12. rujna 2020. u 18,00	1. set: 23:25 2. set: 25:19 3. set: 19:25 4. set: 25:22 5. set: 15:12 ukupno: 107:103	HAOK MLADOST – MOK MURSA - OSIJEK					
3	Vrijeme početka: subota, 5. rujna 2020. u 18,00	1. set: 25:20 2. set: 23:25 3. set: 16:25 4. set: 18:25 ukupno: 82:95	OK MLADOST RIBOLA KAŠTELA – HAOK MLADOST					
4	Vrijeme početka: četvrtak, 19. prosinca 2019. u 18,00	1. set: 19:25 2. set: 14:25 3. set: 16:25 ukupno: 49:75	MOK MARSONIA – MOK MURSA - OSIJEK					

Slika 5.1: ostrugani podaci utakmica prikazani u Excelu

Podaci koje ćemo spremiti mogu se vidjeti na slici 5.2 u obliku liste koja će biti indeksi objekta pandas DataFrame. Zadnji indeks, 'pobjednikDomacinIliGost' može

```
HEADER = ['datum', 'domacin', 'gost', 'rezultat', 'brOsSetDomacin', 'brOsSetGost',
          'osBodDomacin', 'osBodGost', 'proRazBodUSet', 'pobjednik', 'pobjednikDomacinIliGost']
```

Slika 5.2: podaci o utakmicama nakon klasifikacija podataka

imati vrijednosti 0, pobjeda domaćina, ili 1, pobjeda gosta. Pobjednik je ona ekipa koja ima osvojena 3 seta. Gubitnička ekipa može imati između 0 i 2 osvojena seta. Time je utakmica klasificirana. Postupak dobivanja takvog podataka je sljedeći. Podaci se prvo učitavaju u objekt DataFrame. Zatim se iterativno prolazi kroz svaku utakmicu i zapisuju se podaci u obliku koji je poželjan te se izračunavaju novi podaci o utakmici (npr. prosječna razlika sveukupnih bodova po setu). Za izračun broja osvojenih setova svake ekipe korišten je kod u Pythonu koji se može vidjeti na slici 5.3.

Nakon što je klasifikacija podataka gotova podaci imaju oblik kao na slici 5.4.

```
# izracun broja osvojenih setova svake ekipe
setD = 0
setG = 0
for n in bodovi[i].split('\n'):
    if 'set' in n:
        d = n.split('set: ')[1].split(':')[0]
        g = n.split('set: ')[1].split(':')[1]
        if d > g:
            setD = setD + 1
        else:
            setG = setG + 1
```

Slika 5.3: kod za izračun broja osvojenih setova svake ekipe

```
['2020-9-12', 'HAOK MLADOST', 'MOK MURSA - OSIJEK', ' 3-2', 3, 2, '107', '103', 0.8, 'HAOK MLADOST', 0]
```

Slika 5.4: podaci nakon klasifikacije

6. Izračun podataka

6.1. Poredak podataka

Utakmice su klasificirane, ali nemaju dovoljno podataka za uspješno korištenje u algoritmima strojnog učenja. Nema smisla koristiti broj osvojenih setova jer onda je pobjednik već poznat. Moraju se koristiti podaci prošlih utakmica kako bi se dobili podaci relevantni za trenutnu utakmicu. Utakmice se prvo poredaju po datumu. Od starije prema novoj silazno. To se postiže pomoću koda prikazanog na slici 6.1. Time dobivamo podatke kojima, kada se čitaju, uvijek prvo budu pročitane starije utakmice.

```
import pandas as pd
import csv

HEADER = ['datum', 'domacin', 'gost', 'rezultat', 'brOsSetDomacin', 'brOsSetGost',
          'osBodDomacin', 'osBodGost', 'proRazBodUSet', 'pobjednik', 'pobjednikDomacinIliGost']

if __name__ == "__main__":

    f = open(r"utakmice/ordered_classified_data/sveUtakmice_ordered_classified.csv", 'w', newline='',
             encoding='windows-1252')
    df = pd.read_csv(r"utakmice\unordered_classified_data\sveUtakmice_unordered_classified.csv", encoding='windows-1252')
    df["datum"] = pd.to_datetime(df["datum"]).dt.date
    df = df.sort_values(by="datum")

    writer = csv.writer(f)
    writer.writerow(HEADER)

    for i in range(len(df)):
        writer.writerow(df.iloc[i])

    f.close()
```

Slika 6.1: kod za poredak podataka po datumu

6.2. Dodavanje izračunatih podataka

Podaci koji će se zapisivati mogu se vidjeti na slici 6.2. Na slici 6.2. je vidljivo da su podaci isti kao i na slici 5.2, ali imaju još dodane izračunate vrijednosti koje su

```
HEADER = ['datum', 'domacin', 'gost', 'rezultat', 'brOsSetDomacin', 'brOsSetGost',
          'osBodDomacin', 'osBodGost', 'proRazBodUSet',
          'pobjednik', 'pobjednikDomacinIliGost',
          'omjProPobj',
          'proPobjDomacinNadGost',
          'brDaOdZadDomacin',
          'brDaOdZadGost',
          'omjSvukOsSet',
          'omjSvukOsSetDomacinProtGost',
          'omjSvukIzgSet',
          'omjSvukIzgSetDomacinProtGost',
          'omjSvukOsBod',
          'omjSvukOsBodDomacinProtGost',
          'omjSvukIzgBod',
          'omjSvukIzgBodDomacinProtGost']
```

Slika 6.2: popis indeksa za krajnju verziju podataka

dobivene koristeći podatke o prijašnjim utakmicama ekipa u trenutnoj utakmici. Indeksi su sami po sebi jasni, ali valja napomenuti da kratica 'Pro' označava riječ prosjek i 'SvUk' označava riječ sveukupno. Za općenite podatke o svakoj ekipi korišten je rječnik gdje je za svaku ekipu spremljen: datum zadnje utakmice, broj sveukupno odigranih utakmica, broj pobjeda, broj sveukupno osvojenih setova, broj sveukupno izgubljenih setova, broj sveukupno osvojenih bodova i broj sveukupno izgubljenih bodova. Ovi podaci su potrebni za izračun općih podataka o ekipi za trenutnu utakmicu

```
ekipe[ekipa] = {
                'datumZadUtak':'2019-07-01',
                'brUtak':brUtakmicepretSez,
                'brPobj':pobjedeEkipe(ekipa),
                'brOsSet':brOsSet,
                'brIzgSet':brIzgSet,
                'osBod':brOsBod,
                'izgBod':brIzgBod
            }
```

Slika 6.3: rječnik za podatke ekipe

poput prosjeka pobjeda ili broja dana od zadnje igrane utakmice. Uz opće podatke potrebni su još podaci između svake ekipe tj. podaci o utakmicama kada su dvije ekipe igrale jedna protiv druge. Za taj problem korišten je rječnik koji ima sličan zapis kao i prijašnji rječnik, ali uz dodani podatak o broju poraza jedne ekipe. Budući da su podaci inverzni (broj poraza jedne ekipe je broj pobjeda druge) Iz podatka je očito da

```
parovi[ekipa1 + '#' + ekipa2] = {
                        'datumZadUtak': '2019-07-01',
                        'brUtak':brUtak,
                        'brPobjEk1':brPobjEk1,
                        'brGubEk1':brUtak - brPobjEk1,
                        'brOsSetEk1':brOsSetEk1,
                        'brIzgSetEk1':brIzgSetEk1,
                        'osBodEk1':osBodEk1,
                        'izgBodEk1':izgBodEk1
                       }
```

Slika 6.4: rječnik za podatke o parovima ekipa

postoji problem ako nemamo već postojeće vrijednosti za prijašnje utakmice. Ako bi pokušali računati prosjek pobjeda, a ekipa ima odigrano 0 utakmica dobili bi grešku (nešto/0 = problem). Jedan način je da se postavi poseban slučaj i to se riješi na taj način. Ali onda podaci postaju jednoliki za utakmice na početku kada podataka nema. Bolji način je postaviti početne podatke koji se mogu koristiti za prve utakmice. To se postiže korištenjem podataka prijašnje sezone, u ovom slučaju sezone 2018./2019. Koristeći podatke iz te sezone možemo ih samo ubaciti kao početne podatke. Tu sada dolazi problem da neke ekipe nisu bile u istom rangu natjecanja (npr. OK Zadar je tada igrao u Prvoj Ligi, a ne u Superligi). Za ekipe koje sezone 2018./2019. se nisu natjecale u Superligi, podaci su procjenjeni uzimajući u obzir njihov plasman u sezonama: 2019./2020., 2020./2021. i 2021./2022. Superlige. Time dobivamo broj osvojenih utakmica za svaku ekipu koji se može vidjeti na slici 6.5. Za broj osvojenih setova i bodova korištene su formule kako bi se dobili brojevi koji su približni stvarnom stanju ekipa. Kada su početni uvjeti postavljeni potrebno je samo izračunati sve potrebne podatke koristeći početne podatke.

```
EKIPE= ['OK MEDICINAR TRNJE', 'OK ROVINJ', 'MOK MARSONIA', 'OK ZADAR', 'OK SPLIT', 'MOK RIJEKA', 'OKM CENTROMETAL',
        'OK KITRO VARAŽDIN', 'MOK MURSA - OSIJEK', 'OK SISAK', 'OK RIBOLA KAŠTELA', 'HAOK MLADOST', 'OK GORICA']
brUtakmicepretSez = 14

def pobjedeEkipe(i):
    switcher ={
        'OK MEDICINAR TRNJE':4,
        'OK ROVINJ':3,
        'MOK MARSONIA':1,
        'OK ZADAR':2,
        'OK SPLIT':6,
        'MOK RIJEKA':11,
        'OKM CENTROMETAL':4,
        'OK KITRO VARAŽDIN':6,
        'MOK MURSA - OSIJEK':7,
        'OK SISAK':4,
        'OK RIBOLA KAŠTELA':11,
        'HAOK MLADOST':14,
        'OK GORICA':0
       }
    return switcher.get(i, "Invalid team name")
```

Slika 6.5: podaci korišteni za izračun počentih podataka

```
def podaciOEkipama():
    ekipe = {}
    for ekipa in EKIPE:
        brOsSet = 3*pobjedeEkipe(ekipa) + brUtakmicepretSez - pobjedeEkipe(ekipa)
        brIzgSet = (brUtakmicepretSez - pobjedeEkipe(ekipa))*3 + pobjedeEkipe(ekipa)
        brOsBod = brOsSet*25 + brIzgSet*random.randint(12, 23)
        brIzgBod = brOsSet*random.randint(12, 23) + brIzgSet*25
        ekipe[ekipa] = {
                        'datumZadUtak':'2019-07-01',
                        'brUtak':brUtakmicepretSez,
                        'brPobj':pobjedeEkipe(ekipa),
                        'brOsSet':brOsSet,
                        'brIzgSet':brIzgSet,
                        'osBod':brOsBod,
                        'izgBod':brIzgBod
                    }
    return ekipe
```

Slika 6.6: izračun početnih podataka za svaku ekipu

```python
def Parovi():
    listaEkipa = EKIPE.copy()
    parovi = {}
    for ekipa1 in EKIPE:
        listaEkipa.remove(ekipa1)
        for ekipa2 in listaEkipa:
            brUtak = 2
            brPobjEk1 = 1
            if pobjedeEkipe(ekipa1) > pobjedeEkipe(ekipa2):
                brPobjEk1 = 2
            elif pobjedeEkipe(ekipa1) < pobjedeEkipe(ekipa2):
                brPobjEk1 = 0
            brOsSetEk1 = brPobjEk1 * 3 + brUtak - brPobjEk1
            brIzgSetEk1 = (brUtak - brPobjEk1) * 3 + brPobjEk1
            osBodEk1 = brOsSetEk1 * 25 + brIzgSetEk1 * random.randint(12, 23)
            izgBodEk1 = brOsSetEk1 * random.randint(12, 23) + brIzgSetEk1 * 25
            parovi[ekipa1 + '#' + ekipa2] = {
                                    'datumZadUtak': '2019-07-01',
                                    'brUtak':brUtak,
                                    'brPobjEk1':brPobjEk1,
                                    'brGubEk1':brUtak - brPobjEk1,
                                    'brOsSetEk1':brOsSetEk1,
                                    'brIzgSetEk1':brIzgSetEk1,
                                    'osBodEk1':osBodEk1,
                                    'izgBodEk1':izgBodEk1
                                   }
```

Slika 6.7: izračun početnih podataka za svaki mogući par ekipa

7. Algoritmi strojnog učenja

U ovom poglavlju su opisani algoritmi strojnog učenja koji su korišteni u radu. Podatke smo raspodijelili u grupu za učenje i grupu za testiranje koristeći funkciju sklearn.model_selection.train_test_split. Vrijednosti koje su gledane za određivanje kvalitete predviđanja su F1-mjera i točnost predviđanja. Za izračun F1-mjere korištena je funkcija sklearn.metrics.f1_score. Za izračun točnosti predviđanja korištena je funkcija sklearn.metrics.accuracy_score.

7.1. Naivan Bayesov klasifikator

7.1.1. Općenito

Naivan Bayesov klasifikator[9] je algoritam nadziranog strojnog učenja, koji je baziran na Bayesovom teoremu. On je jedan od jednostavnijih klasifikacijskih algoritama koji se koristi u izgradnji modela za predviđanja. Naziva se naivnim jer koristi pretpostavku da su svi atributi nezavisni.

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

Slika 7.1: Bayesov teorem

7.1.2. Implementacija

Za implementaciju algoritma[8] naivnog Bayesovog klasifikatora koristi se Pythonova biblioteka sklearn. Algoritam je naučen koristeći podjelu 70:30, odnosno, 70 posto podataka se koristi za učenje algoritma, a ostalih 30 posto se koristi za testiranje. Korišteni atributi za učenje su omjSvukOsSet i omjSvukIzgSet.

```python
import pandas as pd
from sklearn.metrics import accuracy_score, f1_score
from sklearn.naive_bayes import GaussianNB
from sklearn.model_selection import train_test_split


if __name__ == "__main__":
    df = pd.read_csv(r"utakmice/done_data/sveUtakmice.csv", encoding='windows-1252')
    y = df["pobjednikDomacinIliGost"]
    df = df[['omjSvukOsSet', 'omjSvukIzgSet']]

    x_train, x_test, y_train, y_test = train_test_split(df, y, test_size=0.3, train_size=0.7, random_state=42)

    model = GaussianNB()

    model.fit(x_train, y_train)

    y_pred = model.predict(x_test)

    print("Training data accuracy is ", str(accuracy_score(y_pred, y_test)), "%")
    print("F1 score is ", f1_score(y_test, y_pred, average='binary'))
```

Slika 7.2: implementacija Naivnog Bayesovog klasifikatora

Dobivena je točnost predikcije 71.8%.
F1-mjera ima vrijednost 0.67.

```
Training data accuracy is  71.7948717948718 %
F1 score is  0.6666666666666667
```

Slika 7.3: ispis vrijednosti predviđanja

7.2. Algoritam K-najbližih susjeda

7.2.1. Općenito

Algoritam K-najbližih susjeda ili KNN je algoritam nadziranog strojnog učenja. On uzima nove podatke i klasificira ih u klasu najsličniju obilježjima tih podataka. Algoritam mapira podatke kao točke u prostoru. Zatim se mapira testne podatke te gleda K najbližih susjeda i odlučuje u koju klasu sprema novi podatak. Problem leži u tome da se nađe K vrijednost pomoću koje se dobiva najbolje predviđanje.

7.2.2. Implementacija

Za implementaciju algoritma[7] KNN koristi se Pythonova biblioteka sklearn. Algoritam je koristi podjelu 60:40, odnosno, 60 posto podataka se koristi za učenje, a ostalih

40 posto se koristi za testiranje. Konačna K vrijednost je dobivena tako da se naučio model za svaku k vrijednost od 1 do 20 te je izabran model s najboljom F1-mjerom i najboljom točnošću predviđanja.

```python
import pandas as pd
import numpy as np
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, f1_score

if __name__ == "__main__":
    df = pd.read_csv(r"utakmice/done_data/sveUtakmice.csv", encoding='windows-1252')
    y = df["pobjednikDomacinIliGost"]
    df = df[['omjSvukOsSet', 'omjSvukIzgSet']]

    x_train, x_test, y_train, y_test = train_test_split(df, y, test_size=0.4, train_size=0.6, random_state=42)

    f1 = 0
    kvalue = 1
    model = KNeighborsClassifier(n_neighbors=kvalue)
    model.fit(x_train, y_train)
    modelAccu= model.score(x_test, y_test) * 100

    for k in range(1, 21):
        knn = KNeighborsClassifier(n_neighbors=k)
        knn.fit(x_train, y_train)

        # Compute test data accuracy
        test_accuracy = knn.score(x_test, y_test) * 100
        temp = f1_score(y_test, knn.predict(x_test), average='binary')
        if (temp > f1) & (test_accuracy > modelAccu):...
```

Slika 7.4: implementacija KNN algoritma

K vrijednost je 13.
Dobivena je točnost predikcije 78.8%.
F1-mjera ima vrijednost 0.76.

```
k-value is  13
Training data accuracy is  78.84615384615384 %
F1 score is  0.759124087591241
```

Slika 7.5: postotak točnosti predikcije algoritma

8. Zaključak

Radeći na ovom radu, shvatio sam koliko je zahtjevno iz podataka izvući korisne informacije. U današnjem vremenu, gdje se generira veća količina podatka nego ikada u povijesti, oni ništa ne znače ako iz njih ne možemo izvući informacije koje su nam potrebne za rješavanje problema.

Shvatio sam važnost označavanja i spremanja podataka u obliku u kojem se oni mogu analizirati. U počecima izrade rada mislio sam da postoje već pripremljeni podaci o utakmicama, ali nakon istraživanja shvatio sam da to nije slučaj. Prije ovog rada koristio sam samo već unaprijed pripremljene skupove podataka. Ovdje sam morao sam prikupiti, spremiti podatke i klasificirati ih. Shvatio sam koliko su zapravo važni već unaprijed pripremljeni podaci za analizu.

Zahvalan sam na tome da me je mentor potaknuo da sam odaberem temu rada. Iz tog razloga odabrao sam temu koja je bliska mojim interesima. Kao osoba koja se odbojkom se bavim od svoje 11. godine, imam puno poznanika koji prati hrvatsku odbojku i mislim da bih ovaj rad zanimao. Time sam zapravo napravio nešto što nije samo zanimljivo meni nego i ostalim ljudima.

LITERATURA

[1] HOS Data Project. Hrvatska odbojkaška natjecanja. https://hos-web.dataproject.com/CompetitionHome.aspx?ID=63, svibanj 2022. pristupljeno 15. svibnja 2022.

[2] Hrvatski Odbojkaški Savez. Hrvatska odbojkaška natjecanja arhiva. https://hos-web.dataproject.com/CompetitionHome.aspx?ID=63, svibanj 2022. pristupljeno 15. svibnja 2022.

[3] Hrvatski Odbojkaški Savez. Hrvatski odbojkaŠki kup – seniori. https://natjecanja.hos-cvf.hr/index.php?rubrika=utakmica&natjecanje=1959, svibanj 2022. pristupljeno 15. svibnja 2022.

[4] Hrvatski Odbojkaški Savez. Hrvatska odbojkaška natjecanja. https://natjecanja.hos-cvf.hr/, svibanj 2022. pristupljeno 15. svibnja 2022.

[5] Hrvatski Odbojkaški Savez. Superliga-seniori. https://natjecanja.hos-cvf.hr/index.php?rubrika=utakmica&natjecanje=1819, svibanj 2022. pristupljeno 15. svibnja 2022.

[6] Web Graph SIA. Webscraper plugin. https://chrome.google.com/webstore/detail/web-scraper-free-web-scra/jnhgnonknehpejjnehehllkliplmbmhn?hl=en, listopad 2021. pristupljeno 25. svibnja 2022.

[7] Tavish Srivastava. Introduction to k-nearest neighbors: A powerful machine learning algorithm (with implementation in python r). https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/, ožujak 2018. pristupljeno 2. lipnja 2022.

[8] Simon Tettmar. Machine learning: using naive bayes for sports results classification. https://medium.com/@simon.tettmar/data-science-ca87a98d5637, studeni 2019. pristupljeno 2. lipnja 2022.

[9] Wikipedia. Naive bayes classifier. https://en.wikipedia.org/wiki/Naive_Bayes_classifier, lipanj 2022. pristupljeno 1. lipnja 2022.

[10] Wikipedia. pandas (software). https://en.wikipedia.org/wiki/Pandas_(software), svibanj 2022. pristupljeno 25. svibnja 2022.

[11] Wikipedia. scikit-learn. https://en.wikipedia.org/wiki/Scikit-learn, svibanj 2022. pristupljeno 1. lipnja 2022.

Predviđanje ishoda odbojkaških utakmica korištenjem metoda strojnog učenja

Sažetak

U radu je objašnjen postupak analize i obrade podataka utakmica muških hrvatskih odbojkaških natjecanja. Motivacija za izradu ovog rada bila je potreba za nekakvi predviđanjem ishoda budućih utakmica. Napisane su stranice gdje se nalaze podaci i načini pomoću kojih su podaci dohvaćeni s tih stranica. Dohvat podataka je napravljen pomoću Chrome plugina Web Scraper. Dohvaćeni podaci o utakmicama su očišćeni koristeći programe napisane u Pythonu. Dodatni podaci su također izračunati korištenjem podataka dobivenih s internet stranica. Pripremljeni podaci su potom korišteni za algoritme strojnog učenja. Algoritam naivnog Bayesovog klasifikatora daje predikciju s točnošću od 71.8%. Algoritam KNN daje predikciju s točnošću od 78.8%.

Ključne riječi: Odbojka, strojno učenje, obrada podatka, predikcija ishoda utakmica, analize podataka

Title

Abstract

In this paper, the procedure of analysing and processing the data of matches of men's Croatian volleyball competitions is explained. Motivation for the creation of this paper was the need for some kind of prediction for the outcome of future matches. In this paper there are mentioned web pages where the data was found and the ways that data was collected from this pages. Data collection was done by using a Chrom plugin called Web Scraper. The collected data was then cleaned using programs written in Python. The additional data was also calculated using the data that was collected from the web pages. The prepared data was the used for machine learning algorithms. Naive Bayes classifier algorithm gives prediction with an accuracy of 71.8%. KNN algorithm gives prediction with accuracy of 78.8%.

Keywords: Volleyball, machine learning, data processing, match outcome prediction, data analysis