Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Luka Mavrič

Modeliranje pretoka igralnega časa med zaporednima dogodkoma na košarkarski tekmi

diplomsko delo

univerzitetni študijski program prve stopnje
računalništvo in informatika

Mentor: prof. dr. Igor Kononenko

Ljubljana, 2017

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

Kronološko zaporedje ključnih dogodkov na tekmi (t.i. podatki play-by-play), ki se dandanes rutinsko zbirajo v vseh večjih ligah najpopularnejših športov, predstavljajo osnovo za simulacijo poteka hipotetične tekme poljubnih nasprotnikov. Verjetnost naslednjega dogodka na tekmi je odvisna od tega, kaj se je pred tem zgodilo in tudi, kdaj se je zgodilo. V okviru diplomske naloge uporabite metode strojnega učenja na podatkih play-by-play iz lige NBA in zgradite ter ovrednotite modele za napovedovanje pretoka igralnega časa med dvema zaporednima dogodkoma.

Rad bi se zahvalil družini za podporo ter profesorju Kononenku in asistentu Vračarju za potrpljenje.

Kazalo

Povzetek

Naslov: Modeliranje pretoka igralnega časa med zaporednima dogodkoma na košarkarski tekmi

Avtor: Luka Mavrič

V diplomskem delu smo reševali problem napovedovanja pretoka časa med dvema dogodkoma na košarkarski tekmi. Najprej smo preučili vhodne podatke, kjer so bile kombinacije dogodkov zapisane v kronološkem zaporedju. Podatke smo preoblikovali v obliko, primerno za uporabo v strojnem učenju. Pretok časa smo napovedovali s pomočjo linearne regresije, regresijskih dreves in nevronskih mrež. Vsak algoritem smo na kratko predstavili, poiskali najboljšo kombinacijo neodvisnih spremenljivk in ostalih parametrov ter na koncu predstavili najboljši model. V zaključku smo primerjali najboljše modele uporabljenih metod strojnega učenja. Najboljše rezultate je dosegla nevronska mreža, najslabše pa pričakovano linearna regresija. Za konec smo strnili naše ugotovitve ter predlagali še nekaj predlogov za izboljšave.

Ključne besede: košarka, napovedovanje pretoka časa, linearna regresija, regresijska drevesa, nevronske mreže.

Abstract

Title: Modelling the Elapsed Time between Two Events in a Basketball Game

Author: Luka Mavrič

In this thesis, the problem of predicting the elapsed time between two events in a basketball game is researched. First, our input data consisting of combinations of events in chronological order is studied, and then, the data is prepared and transformed so that it becomes better suited for machine learning. The elapsed time is predicted with the help of linear regression, regression trees and neural networks. For each algorithm, a short description is created; further, the best combination of independent variables and other parameters is found and finally, the best model is presented. To conclude, the best models of the utilized machine learning methods are compared. The best results were achieved by neural networks, while linear regression, as expected, proved to be the worst. Finally, the findings are presented and a few suggestions for improvements are added.

Keywords: basketball, prediction of the elapsed time, linear regression, regression trees, neural networks.

Poglavje 1

Uvod

V današnjem svetu ekipni športi veljajo za najpopularnejše. Veliko ljudi se z njimi ukvarja rekreativno, najbolj talentirani in resni pa se z njimi ukvarjajo profesionalno. V tej diplomi se osredotočamo na profesionalni vidik ekipnega športa.

Ekipni šport lahko razdelimo na dve komponenti. Prva je fizična pripravljenost in tehnično znanje posameznika, druga pa je ekipna taktika. Treniranje fizičnih sposobnosti in posameznih elementov določenega športa je ponavadi podobna za igralce vseh ekip, medtem ko se taktična raven lahko od ekipe do ekipe precej razlikuje. Taktiko določajo razni strokovnjaki, ki se ponavadi imenujejo trenerji, selektorji, managerji ali kaj podobnega. Glavne naloge teh strokovnjakov so taktična priprava ekipe, motivacija igralcev in sprejemanje pravilnih odločitev ter prilagoditev na nasprotnika med potekom tekme. Ko se igralci sredi tekme znajdejo v težki situaciji, jim pretekle izkušnje v podobnih situacijah lahko pomagajo do boljše odločitve. V težkih situacijah ima trener pomembno vlogo, saj se ponavadi on na podlagi preteklih izkušenj in pa seveda tudi glede na situacijo na igrišču odloči, katera taktika je najprimernejša v tistem trenutku. Toda kako dobro se trener lahko odloči? Koliko preteklih situacij je trener lahko preučil? Koliko različnih spremenljivk je pri tem upošteval? Tu se pojavi vprašanje, ali bi mu pri tem lahko pomagal računalnik?

Najbolj znan primer uporabe analitike v športu prihaja iz sveta baseballa, ko je ekipa Oakland Athletics izumila koncept, imenovan Moneyball. Ker niso imeli dovolj finančnih sredstev, da bi bogatejšim ekipam konkurirali v kvaliteti igralcev, so se osredotočili na statistično analizo in ugotovili, kateri elementi statistike so pravzaprav zaslužni za uspešen napad posamezne ekipe. Na podlagi dobljenih rezultatov so v ekipo pripeljali igralce, ki so bili močni v teh elementih statistike. Kljub tretjemu najnižjemu proračunu za plače igralcev so se od uvedbe Moneyballa dvakrat zapored uvrstili v končnico.

Liga NBA že od nekdaj velja za najmočnejšo košarkarsko ligo na svetu. Vsako leto je zbranih več podatkov o posameznih tekmah in igralcih v ligi, kar omogoča vedno boljšo analizo igre. Poleg klasičnih statističnih elementov, kot so procent meta iz igre, procent zadetih prostih metov, so se začeli beležiti tudi položaji, iz katerih so igralci metali na koš. Najnovejši podatki sledijo tudi gibanju igralcev na igrišču, kar omogoča še boljšo analizo posameznih situacij.

Ekipe si ponavadi želijo svoje igralce v napadu postaviti v ugodne situacije za dosego koša, v obrambi pa nasprotnika prisiliti v čim težji met in tako zmanjšati procent uspešnih metov. Mnoge ekipe so že najele tako imenovane analitike, ki ekipam pri tem svetujejo. Tako je recimo ekipa Houston Rockets ugotovila, da se meti za 2 točki iz bližine črte za 3 točke ne izplačajo.

Eden izmed pomembnih faktorjev košarkarske igre je tudi čas. Košarkarska tekma lige NBA je razdeljena na štiri četrtine po 12 minut. V primeru neodločenega izida po pretečenih 48-ih minutah sledijo petminutni podaljški, dokler ena ekipa ne zmaga. Čas za zaključek napada je omejen na 24 sekund, kar preprečuje zavlačevanje.

V očeh trenerjev čas ponavadi ne igra prav pomembne vloge do same končnice tekme. Njihove ekipe v večini primerov igrajo tempo, primeren dogovorjeni taktiki. Seveda se ekipa mora prilagoditi tudi situaciji na igrišču. V primeru da nasprotna ekipa v zelo kratkem času doseže veliko število zaporednih točk, trenerji ponavadni umirijo dogajanje z dolgim napadom. Še en primer manipulacije s časom je, ko se trenerji v zadnji minuti četrtine odločijo

za hiter zaključek napada z namenom, da bodo nekaj sekund pred koncem še enkrat dobili žogo. Dinamika tekme pa se ponavadi precej spremeni v zadnji četrtini. Ekipe, ki so v rezultatskem zaostanku, hitreje zaključujejo napade, medtem ko ekipe, ki so v prednosti, ponavadi igrajo na dolge napade.

Do teh zaključkov smo prišli na podlagi izkušenj, ki smo jih pridobili s spremljanjem preteklih tekem. V tej diplomi nas je predvsem zanimalo, ali se pojavljajo še kakšni drugi vzorci, ki jih s prostim očesom morda ne vidimo.

1.1 Pregled področja

Velika večina raziskovalcev, ki se ukvarjajo s podatkovnim rudarjenjem na področju športa, se osredotoča na napovedovanje rezultatov posameznih tekem [1]. Razlog za to je predvsem industrija športnih stav.

Precej manj jih napoveduje izid posameznih elementov igre, na primer uspešnost zadetih metov v košarki [2] oziroma uspešnost zadetih strelov v nogometu [5]. Trenerji bi lahko z upoštevanjem rezultatov omenjenih raziskav prilagodili taktične postavitve in tako svoje igralce poskušali postaviti v položaj, ko je verjetnost za dosego gola oziroma koša največja.

Na področju dejanskega poteka tekme ni bilo narejenega veliko. Eden izmed redkih primerov je uporaba Markovega modela za simulacijo košarkarskih tekem [8], kjer so bili uporabljeni podatki play-by-play. Z vse večjo multi medijsko pokritostjo posameznih tekem pa se je povečala tudi količina podatkov. Najnovejši podatki namreč sledijo igralčevim premikom po igrišču. Takšni podatki so bili uporabljeni za napovedovanje dogodkov na košarkarski tekmi na podlagi trenutnega stanja na igrišču [9].

1.2 Cilj diplomske naloge

Glavni cilj diplomske naloge je ugotoviti, ali je mogoče iz zapisa zaporednih dogodkov s košarkarske tekme zgraditi model, ki bi uspešno napovedoval pretok časa med posameznima dogodkoma. Naš problem se ne osredotoča le

na vprašanje, ali je model sposoben prepoznati določene vzorce, ampak tudi, ali kakšni vzorci sploh obstajajo.

V uvodnem poglavju je na kratko predstavljen koncept ekipnega športa in košarkarska liga NBA. Navedena sta tudi dva primera uporabe analitike v nekaterih profesionalnih ekipah. Na hitro so opisana tudi področja, s katerimi se strokovnjaki podatkovnega rudarjanja ukvarjajo na področju športa.

V drugem poglevju so predstavljena orodja in knjižnice, s katerimi smo si pomagali pri urejanju podatkov in grajenju različnih napovednih modelov.

Sledi poglavje, v katerem je razloženo, kakšni so naši vhodni podatki. Opišemo tudi težave, s katerimi smo se pri obravnavanju podatkov srečali in kako smo jih rešili.

V četrtem poglavju smo s pomočjo linearne regresije zgradili naš prvi napovedni model. Najprej smo analizirali posamezne neodvisne spremenljivke in preverili njihov doprinos h kvaliteti modelov. Za tem smo testirali različne kombinacije neodvisnih spremenljivk.

V naslednjem poglavju smo modelirali s pomočjo regresijskih dreves. Opisali smo nekaj prednosti in slabosti, ki jih ima ta metoda strojnega učenja, na koncu pa smo podobno kot pri linearnem modeliranju testirali različne kombinacije neodvisnih spremenljivk ter še nekatere druge parametre.

V predzadnjem poglavju smo gradili nevronske mreže. Po kratki predstavitvi, kaj nevronske mreže sploh so, smo testirali različne kombinacije neodvisnih spremenljivk, struktur in ostalih parametrov učenja ter na koncu izbrali model, ki je prinesel najboljše rezultate.

V zaključku smo primerjali najboljše modele posameznih metod strojnega učenja in določili najboljši model. Razložili smo tudi, kako omejenost vhodnih podatkov onemogoča boljše napovedi in kako bi lahko posamezne modele še izboljšali.

Poglavje 2

Orodja in knjižnice

Vsa orodja in knjižnice, ki so uporabljene v diplomi, so odprtokodni.

R (programski jezik): R je programski jezik in okolje, namenjeno predvsem statističnemu računanju. V naši nalogi smo ga uporabili le za branje podatkov tipa RData in prepis teh podatkov v format csv (comma-seperated-values). Podatke smo nato obrabnavali v programskem jeziku Python.

Python: Python zaradi svojega bogatega ekosistema že od nekdaj velja za enega izmed primarnih jezikov na področju podatkovnega rudarjenja. Njegova glavna prednost pred ostalimi visokonivojskimi jeziki je manjše število vrstic kode za realizacijo istega problema.

Numpy: Numpy je modul v programskem jeziku Python, namenjen predvsem numeričnim operacijam. Njegova največja odlika je zelo učinkovito računanje z več dimenzionalnimi polji (ang. array), kar nam je prišlo še kako prav, saj so naši vhodni podatki predstavljeni kot dvo-dimenzionalno polje. Prva dimenzija predstavlja posamezne vhodne primere, druga dimenzija pa točno določene komponente posameznega vhodnega primera.

Scrapy: Scrapy je knjižnica v programskem jeziku Python, ki poenostavlja pisanje tako imenovanih pajkov. Pajek je objekt, s katerim avtomat-

sko preiskujemo želene spletne strani in iz njih pridobivamo podatke, ki nas zanimajo. V našem primeru smo preiskovali stran basketball-reference.com. Iz podstrani, ki vsebujejo podatke o posameznih tekmah, smo izluščili podatke o tempu igre.

Pandas: Python modul pandas vsebuje podatkovno strukturo DataFrame, ki je zelo priročna za predstavitev vhodnih podatkov. Omogoča nam enostavno dodajanje in odstranjevanje stolpcev, spreminjanje tipa podatkov v posameznih stolpcih ter enostavno kreiranje pomožnih spremenljivk.

Statsmodels: Python modul statsmodels smo uporabili za dodajanje konstant v strukturo DataFrame ter za gradnjo linearnih modelov.

Orange: Orange je v programskem jeziku Python napisan odprtokodni program, ki omogoča vizualizacijo in podatkovno rudarjenje v obliki vizualnega programiranja. To pomeni, da je v grafičnem vmesniku potrebno izbrati določene gradnike (vhodne podatke, metode strojnega učenja, vizualizacije podatkov, ...), za računske operacije in samo modeliranje pa poskrbi program sam. Uporabili smo ga za modeliranje regresijskih dreves.

Keras: Za izgradnjo nevronske mreže smo uporabili API imenovan Keras, ki je napisan v programskem jeziku Python. Keras lahko deluje na osnovi Python knjižnic Theano oziroma Tensorflow. Njun namen je učinkovito računanje numeričnih izrazov. Odločili smo se za uporabo knjižnice Theano, saj le ta omogoča ponovljivost rezultatov.

Matplotlib: Python knjižnico Matplotlib smo uporabili za izdelavo grafov.

Poglavje 3

Podatki

Košarka je tako kot vsak ekipni šport sestavljena iz posameznih elementov igre. Met za 2 točki, skok v napadu in podaja so le nekateri izmed njih. Ko se med tekmo zgodi določen element igre, lahko zapišemo dogodek. Tu ni nobenega dvoma.

Potek posamezne tekme je predstavljen v tako imenovanem play-by-play zapisu, kar pomeni, da so dogodki zapisani v kronološkem vrstnem redu. Play-by-play zapisi, ki jih najdemo na uradnih športnih spletnih straneh, ponavadi zabeležijo, kaj se je zgodilo ob koncu posesti žoge določene ekipe in ne, kaj vse se je dogajalo med samo posestjo. Glavni razlog za to je preglednost zapisa, saj so v njem zabeleženi dogodki, ki dejansko vplivajo na potek tekme.

Vhodni podatki so vsebovali play-by-play zapise tekem rednega dela iz sezon 2012/2013, 2013/2014, 2014/2015, 2015/2016. Prvi dve sezoni smo uporabili za učno množico, tretjo za preverjanje najboljših kombinacij parametrov znotraj posameznih metod strojnega učenja, zadnjo pa za primerjavo najboljših modelov, ki smo jih dobili z različnimi metodami strojnega učenja (testna množica). Vhodne podatke smo dobili na spletni strani stats.nba.com in jih zapisali v štiri csv datoteke. Vsaka izmed njih predstavlja eno sezono dogodkov. Posamezne vrstice so v csv datotekah predstavljene kot vrednosti, ločene z vejico. V podatkih imamo v eni vrstici že zabeležena dva zaporedna

dogodka ter tudi čas, ki je pretekel med njima. Format vrstice je naslednji:

COUNTER, DATE, AWAY_NAME, HOME_NAME, QUARTER, TIME, DIFF, PREVACT, ACT, DUR

COUNTER: predstavlja zaporedno številko kombinacije dveh dogodkov v sezoni

DATE: datum, na katerega se je zgodila kombinacija dveh dogodkov - format datuma: LLLLMMDD

AWAY_NAME: kratica gostujoče ekipe

HOME_NAME: kratica domače ekipe

QUARTER: četrtina, v kateri se je kombinacija dveh dogodkov zgodila

TIME: čas od dogodka PREVACT do konca četrtine v sekundah

DIFF: razlika v točkah med ekipama. V primeru vodstva domače ekipe je število pozitivno, v nasprotnem primeru pa negativno. V primeru izenačenega izida je število enako 0

PREVACT: prvi dogodek v kombinaciji dveh dogodkov

ACT: drugi dogodek v kombinaciji dveh dogodkov

DUR: število sekund ki so pretekle med dogodkoma PREVACT in ACT

Primer vrstice v csv datoteki, ki predstavlja kombinacijo dveh dogodkov, lahko vidimo v izpisu 3.1. Gre za 6267. kombinacijo dogodkov v sezoni. Kombinacija se je zgodila 26.11.2012 na tekmi med ekipama New York Knicks (gostujoča ekipa) in Brooklyn Nets (domača ekipa). Prvi dogodek (HINB - izvajanje iz avta domače ekipe) se je zgodil 470 sekund pred koncem 2. četrtine. Deset sekund pozneje se je zgodil drugi dogodek (H2PA - zgrešen met za 2 točki domače ekipe). V času med dogodkoma je bil rezultat izenačen.

Izpis 3.1: Primer vrstice v csv datoteki

```
6267,20121126,NYK,BKN,2,470,0,HINB,H2PA,10
```

Ti podatki so že zelo dobro urejeni, a še vedno vsebujejo napake. Razloga za napake sta dva. Prvi se pojavi, ko vhodne podatke uredimo po nekih vnaprej napisanih pravilih, ki pa ne veljajo za vsako kombinacijo zapisa dogodkov. Recimo po zadetem zadnjem prostem metu je v naših podatkih avtomatsko dodan dogodek izvajanje iz avta. Toda v primeru, da je zadet zadnji prosti met predstavljal tehnični prosti met, potem mu lahko sledi tudi skok za žogo in ne izvajanje iz avta. Lahko bi si vzeli ogromno časa in ročno preverili ter popravili napačne kombinacije, a tako bi krepko presegli količino časa, ki smo si ga rezervirali za to diplomsko delo.

Drugi razlog za napake pa je lahko že nepravilen originalen play-by-play zapis. Tu ne moremo storiti ničesar, saj brez ogleda tekme ne moremo določiti, kaj se je dejansko zgodilo in v kakšnem vrstnem redu. To lahko povzroči veliko zmedo, saj določen zapis zaporedja dogodkov lahko predstavlja dve povsem različni, a še vedno logični zaporedji dogodkov.

Eden izmed takih primerov nastane, ko po zgrešenem metu na koš igralca iz nasprotnih ekip poskrbita za mrtvo žogo. Sledi skok za žogo in igralec, ki dobi skok, bo v play-by-play zapisu avtomatsko nagrajen še z dobljenim skokom za prvotno zgrešen met. V play-by-play zapisu je trajanje med skokom za žogo in avtomatsko pripisanim skokom vedno 0 sekund. Toda kaj se zgodi, ko je vrstni red teh dveh dogodkov zapisan napačno, torej najprej skok igralca po zgrešenem metu in šele potem skok za žogo. Ta zapis dejansko pomeni povsem drugačno dinamiko dogodkov. Lahko bi igralec dobil skok za žogo in recimo 10 sekund kasneje bi prišlo do mrtve žoge.

3.1 Identifikacija dogodkov

V zanki smo se sprehodili preko učne množice in izluščili vse različne zapise dogodkov. Izkazalo se je, da se skozi 2 sezoni pojavi 38 različnih zapisov. Pri pozornem pregledu dogodkov smo opazili, da prva črka dogodka predstavlja

ekipo, ki je pri dogodku sodelovala, preostale črke pa predstavljajo dejanski dogodek. Črka H pomeni domačo ekipo, črka A pa gostujočo. Iz tega lahko sklepamo, da je vseh dogodkov dejansko 19, pri vsakem izmed njih pa lahko sodeluje ali domača ali gostujoča ekipa. Na primer zapis INB predstavlja dogodek izvajanje iz avta. Če iz avta izvaja domača ekipa, je dogodek zapisan kot HINB, če pa gostujoča, pa AINB. Sledi seznam dogodkov:

1. INB - izvajanje iz avta
2. 2PA - zgrešen met za 2 točki
3. 2PM - zadet met za 2 točki
4. 3PA - zgrešen met za 3 točke
5. 3PM - zadet met za 3 točke
6. FT3A - zgrešen prvi od treh prostih metov
7. FT3M - zadet prvi od treh prostih metov
8. FT2A - zgrešen predzadnji prosti met
9. FT2M - zadet predzadnji prosti met
10. FT1A - zgrešen zadnji prosti met
11. FT1M - zadet zadnji prosti met
12. DRB - skok v obrambi
13. ORB - skok v napadu
14. DREB - ekipni skok v obrambi
15. OREB - ekipni skok v napadu
16. F - osebna napaka
17. V - kršitev pravil (a ne osebna napaka)

18. TO - izgubljena žoga

19. JB - skok za žogo

V tabeli 3.1 je prikazano, kolikokrat se kateri dogodek pojavi na katerem mestu v kombinaciji dveh zaporednih dogodkov.

Tabela 3.1: Število pojavitev dogodkov v spremenljivkah prvi dogodek v kombinaciji in drugi dogodek v kombinaciji.

	Prvi dogodek v kombinaciji		Drugi dogodek v kombinaciji	
	Domača ekipa	Gostujoča ekipa	Domača ekipa	Gostujča ekipa
INB	144709	147523	141998	144667
2PA	77304	79032	77304	79032
2PM	74562	72211	74853	72551
3PA	32119	33264	32119	33265
3PM	18246	18068	18428	18229
FT3A	111	88	111	88
FT3M	373	353	373	353
FT2A	6700	6506	6700	6506
FT2M	18542	17609	18542	17609
FT1A	7257	6949	7257	6949
FT1M	24539	23535	24539	23535
DRB	77272	74828	78481	75943
ORB	27247	26809	27370	26942
DREB	5094	5080	5245	5188
OREB	6044	6320	7909	8322
F	50499	51912	50503	51924
V	1074	1217	1074	1217
TO	35283	36012	35541	36260
JB	2165	2086	831	784

3.2 Kombinacije dogodkov

Ker naš problem temelji na napovedovanju časa med dvema dogodkoma, je zelo pomembno obravnavati vse možne kombinacije dogodkov. Ko vidimo, da zaporedje dveh dogodkov ni mogoče, ga lahko izločimo in tako ne uvrstimo med vhodne primere. Dodatna prednost obravnavanja vseh primerov je določitev kombinacij dogodkov, med katerimi vedno preteče 0 sekund. Tako se lahko metode strojnega učenja osredotočijo na preostale primere.

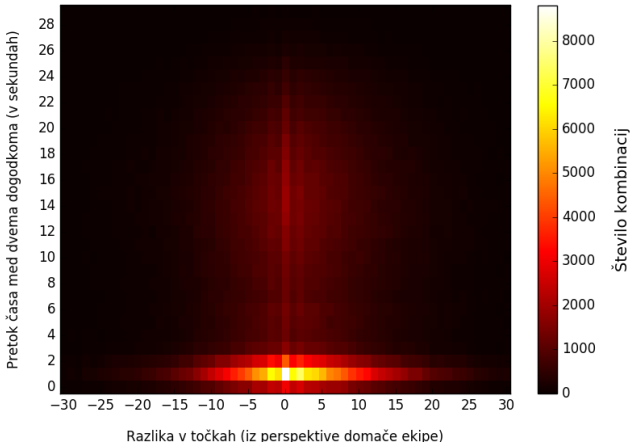
Za vsakega izmed 19-ih dogodkov iz tabele 3.1 smo v ločeno tekstovno datoteko izpisali vse kombinacije, kjer se obravnavani dogodek pojavi na prvem mestu v kombinaciji. Za vsako kombinacijo smo izpisali tudi število pojavitev skozi dve sezoni in pa povprečen pretok časa med obema dogodkoma v kombinaciji.

Izločili smo kombinacije, ki se na tekmi ne morejo zgoditi. Prav tako smo izločili tudi vse kombinacije, kjer med dogodkoma mine 30 sekund ali več. Čas napada je v ligi NBA omejen na 24 sekund, vzeli pa smo še 5 sekund rezerve.

Število vseh različnih kombinacij je 457, od katerih pa je 127 napačnih. To pomeni, da bomo pretok časa napovedovali za 330 različnih kombinacij. V košarki se zgodi veliko kombinacij dogodkov, kjer med dogodkoma vedno mine 0 sekund. Takšne kombinacije smo določili s pomočjo predznanja košarke. Za njih smo pretok časa napovedovali ročno, saj vemo, da med dogodkoma vedno preteče 0 sekund. V naših vhodnih podatkih je takšnih kombinacij 149.

Sledila je vizualizacija števila kombinacij pri določeni točkovni razliki v odvisnosti od pretoka časa med dogodkoma v kombinaciji. Graf je prikazan na sliki 3.1.

Vidimo lahko, da se največ kombinacij zgodi, ko je razlika v točkah med ekipama majhna. Zagotovo pa najbolj zbode v oči število kombinacij dogodkov, med katerima preteče ena sekunda. Po ponovnem pregledu analize vseh kombinacij smo ugotovili, da gre večinoma za kombinacije zgrešenih metov in skokov za žogo.



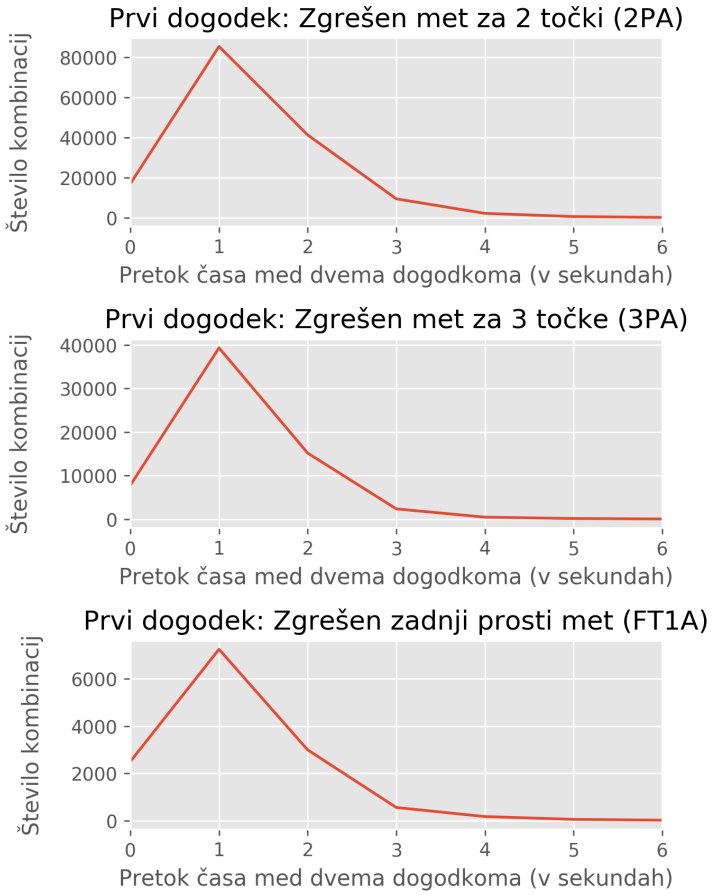
Slika 3.1: Število kombinacij pri določeni točkovni razliki v odvisnosti od pretoka časa med dogodkoma v kombinaciji.

Naslednja vizualizacija predstavlja število kombinacij v odvisnosti od pretoka časa, ko je prvi dogodek v kombinaciji zgrešen met. Drugi dogodek ni vedno skok. Lahko je tudi osebna napaka ali pa kršitev pravil (ang. violation). Na sliki 3.2 so prikazani trije grafi. Prvi predstavlja kombinacije dogodkov, kjer je prvi dogodek zgrešen met za 2 točki, drugi kombinacije dogodkov, kjer je prvi dogodek zgrešen met za 3 točke, tretji pa kombinacije dogodkov, kjer je prvi dogodek zgrešen zadnji prosti met.

Kot lahko vidimo, je porazdelitev pretoka časa v vseh treh primerih skorajda identična. Takšnih kombinacij je 46.

3.3 Testiranje modelov

Cilj diplomske naloge je bila čim boljša napoved pretoka časa med dogodki na košarkarski tekmi. Pretok časa bi lahko za vse kombinacije dogodkov napovedovali po občutku s pomočjo predznanja dinamike košarkarske igre. Nekatere napovedi bi bile točne, saj je pri določenih kombinacijah pretok časa konstanten. Pri kombinacijah, kjer pretok časa ni konstanten, pa verjetno



Slika 3.2: Porazdelitev pretoka časa v kombinacijah, ko je prvi dogodek zgrešen met.

ne bi imeli možnosti proti raznim računskim modelom. To je tudi razlog, da smo vhodne primere razdelili na tiste, ki jih napovedujejo modeli in tiste, kjer je pretok časa napovedujemo ročno.

Pretok časa smo napovedovali za 330 različnih kombinacij dogodkov, ki so razdeljene v naslednje tri skupine:

Vhodni primeri, kjer je pretok časa vedno enak 0 sekund: Te primere smo vedno napovedovali ročno (149 kombinacij).

Vhodni primeri, o katerih nimamo nobenega predznanja: Ti vhodni primeri vsebujejo 135 kombinacij dogodkov, za katere pretok časa vedno napovedujejo modeli.

Vhodni primeri, kjer je prvi dogodek zgrešen met: Ti vhodni primeri vsebujejo kombinacije, kjer je opažena neka konstantna porazdelitev pretoka časa med dogodkoma (46 kombinacij). Za vsako kombinacijo posebej smo na učni množici izračunali povprečen pretok časa, ki bo uporabljen kot napoved pri ročnem napovedovanju pretoka časa za te kombinacije.

Vsako tehniko strojnega učenja smo testirali na dveh vrstah podatkov:

Podatki 1: Modeli napovedujejo vhodne primere, o katerih nimamo nobenega predznanja. Ročno napovedujemo vhodne primere, kjer med dogodkoma vedno mine 0 sekund in vhodne primere, kjer je prvi dogodek zgrešen met.

Podatki 2: Modeli napovedujejo vhodne primere, o katerih nimamo nobenega predznanja in vhodne primere, kjer je prvi dogodek zgrešen met, ročno pa napovedujemo samo vhodne primere, kjer je pretok časa vedno enak 0 sekund.

Razlog za takšno testiranje je naše predznanje o košarki. Po zgrešenem metu je v veliki večini primerov naslednji dogodek skok za žogo. Pretok časa naj ne bi bil odvisen od ostalih neodvisnih spremenljivk, temveč od odboja

žoge od obroča (lahko tudi primer, ko žoga ne zadane obroča oziroma table). Ker podatkov o odboju nimamo (dolg odboj, kratek odboj, več odbojev, brez odboja, samo odboj od table, ...), lahko smatramo, da je naključen. To pomeni, da bi bila najbolj smiselna napoved povprečen pretok časa za posamezno kombinacijo, izmerjen na učni množici.

Ker pa nismo povsem prepričani, da je pretok časa po zgrešenem metu neodvisen od ostalih neodvisnih spremenljivk, smo se odločili testirati oba načina.

Kot merilo kvalitete modela smo vzeli povprečno absolutno napako napovedi, merjeno v sekundah.

3.4 Neodvisne spremenljivke

Neodvisne spremenljivke so lahko nominalne ali pa zvezne. Vrednosti prvih so ponavadi oznake in med seboj niso v nikakršnem količinskem razmerju, vrednosti drugih pa so numerične. Pri določenih metodah strojnega učenja je potrebno zvezne spremenljivke standardizirati, saj v nasprotnem primeru pride do težav.

3.4.1 Nominalne spremenljivke

Nominalne spremenljivke morajo biti v nekaterih algoritmih strojnega učenja (linearna regresija, nevronske mreže, ...) predstavljene v numerični obliki. To omogoči pretvorba v pomožne spremenljivke. Poleg vsake neodvisne spremenljivke je zapisano, koliko pomožnih spremenljivk je potrebnih za predstavitev te spremenljivke. Število pomožnih spremenljivk je za ena manjše od števila vseh možnih vrednosti neodvisne spremenljivke.

Množici vhodnih podatkov Podatki 1 in Podatki 2 vključujeta različne vhodne primere, zato se na položaju spremenljivk predhodnih in pred predhodnih dogodkov pojavljajo različne vrednosti. Pri spremenljivkah, ki vključujejo kakršno koli obliko predhodnega ali pred predhodnega dogodka, velja prvo število pomožnih spremenljivk za Podatke 1, število v oklepaju pa za Po-

datke 2. V seznamu možnih vrednosti so najprej navedene vrednosti, ki se pojavijo v Podatkih 1 in Podatkih 2, v oklepaju pa samo tiste, ki se pojavijo v Podatkih 2.

Prvi dogodek v kombinaciji (s podatkom o ekipi)

Število pomožnih spremenljivk: 13
Množica možnih vrednosti: FIRST-AOREB, FIRST-HOREB, FIRST-AINB, FIRST-HDREB, FIRST-AJB, FIRST-HINB, FIRST-HTO, FIRST-HDRB, FIRST-ADREB, FIRST-HJB, FIRST-HORB, FIRST-ATO, FIRST-ADRB, FIRST-AORB

Prvi dogodek v kombinaciji je dogodek, od katerega merimo pretok časa do naslednjega dogodka (drugi dogodek v kombinaciji).

Drugi dogodek v kombinaciji (s podatkom o ekipi)

Število pomožnih spremenljivk: 15
Množica možnih vrednosti: SECOND-A2PA, SECOND-AV, SECOND-AJB, SECOND-A3PM, SECOND-HJB, SECOND-A2PM, SECOND-AF, SECOND-H3PM, SECOND-HF, SECOND-H3PA, SECOND-HTO, SECOND-H2PA, SECOND-A3PA, SECOND-ATO, SECOND-H2PM, SECOND-HV

Drugi dogodek v kombinaciji je dogodek, ki se zgodi za prvim dogodkom v kombinaciji.

Predhodni dogodek (s podatkom o ekipi)

Število pomožnih spremenljivk: 30 (34)
Množica možnih vrednosti: PREV-HDREB, PREV-A3PA, PREV-ADRB, PREV-HORB, PREV-ATO, PREV-H2PA, PREV-HJB, PREV-AFT1A, PREV-H3PM, PREV-H2PM, PREV-AF, PREV-HOREB, PREV-H3PA, PREV-ADREB, PREV-HFT1A, PREV-AFT1M, PREV-A2PM, PREV-AORB, PREV-HFT1M, PREV-HV, PREV-HF, PREV-HDRB, PREV-A3PM, PREV-None, PREV-A2PA, PREV-AINB, PREV-AJB, PREV-AOREB, PREV-HINB, PREV-

AV, PREV-HTO, (PREV-HFT2M, PREV-AFT2M, PREV-HFT2A, PREV-AFT2A)

V spremenljivko predhodni dogodek zapišemo dogodek, ki se je zgodil pred dogodkom, zapisanem v spremenljivki prvi dogodek v kombinaciji. Gre za dogodek, ki se v predhodnem vhodnem primeru nahaja v spremenljivki prvi dogodek v kombinaciji. V primeru, da vhodni primer opisuje prvo kombinacijo dogodkov v četrtini oziroma podaljšku, ima spremenljivka predhodni dogodek vrednost None. To pomeni, da predhodnega dogodka ni.

Pred predhodni dogodek (s podatkom o ekipi)

Število pomožnih spremenljivk: 34 (38)
Množica možnih vrednosti: PP-HDRB, PP-HF, PP-HDREB, PP-H3PM, PP-H2PA, PP-HOREB, PP-A2PA, PP-AV, PP-AFT1M, PP-H3PA, PP-ADREB, PP-AFT1A, PP-HFT1M, PP-A3PA, PP-ATO, PP-HFT2A, PP-HFT1A, PP-A3PM, PP-A2PM, PP-HORB, PP-HTO, PP-AORB, PP-AF, PP-AFT2M, PP-None, PP-HINB, PP-AOREB, PP-AJB, PP-HJB, PP-ADRB, PP-H2PM, PP-AINB, PP-HV, PP-HFT2M, PP-AFT2A, (PP-HFT3M, PP-AFT3M, PP-HFT3A, PP-AFT3A)

Spremenljivka pred predhodni dogodek predstavlja dogodek, ki se je zgodil pred dogodkom, ki se nahaja v spremenljivki predhodni dogodek. Podobno kot spremenljivka predhodni dogodek ima lahko tudi ta spremenljivka vrednost None.

Prvi dogodek v kombinaciji (brez podatka o ekipi)

Število pomožnih spremenljivk: 6
Množica možnih vrednosti: FIRST-TO, FIRST-DREB, FIRST-INB, FIRST-OREB, FIRST-ORB, FIRST-DRB, FIRST-JB

Drugi dogodek v kombinaciji (brez podatka o ekipi)

Število pomožnih spremenljivk: 7

Množica možnih vrednosti: SECOND-F, SECOND-3PM, SECOND-3PA, SECOND-TO, SECOND-2PM, SECOND-2PA, SECOND-JB, SECOND-V

Predhodni dogodek (brez podatka o ekipi)

Število pomožnih spremenljivk: 15 (17)

Množica možnih vrednosti: PREV-DRB, PREV-JB, PREV-FT1A, PREV-DREB, PREV-ORB, PREV-F, PREV-3PM, PREV-INB, PREV-None, PREV-2PM, PREV-OREB, PREV-3PA, PREV-TO, PREV-2PA, PREV-V, PREV-FT1M, (PREV-FT2M, PREV-FT2A)

Pred predhodni dogodek (brez podatka o ekipi)

Število pomožnih spremenljivk: 17 (19)

Množica možnih vrednosti: PP-3PM, PP-FT2M, PP-TO, PP-2PM, PP-OREB, PP-ORB, PP-None, PP-INB, PP-FT1A, PP-2PA, PP-DREB, PP-FT1M, PP-DRB, PP-V, PP-FT2A, PP-JB, PP-3PA, PP-F, (PP-FT3M, PP-FT3A)

Četrtina

Število pomožnih spremenljivk: 3

Množica možnih vrednosti: Q-1, Q-2, Q-3, Q-4

Košarkarska tekma v ligi NBA je razdeljena na štiri četrtine po dvanajst minut. V primeru da je izid po koncu rednega dela izenačen, se igrajo 5-minutni podaljški, dokler ni znan zmagovalec.

V stolpcu QUARTER (četrtina) so mogoče vrednosti od 1 do 4 (četrtine v rednem delu tekme), ter od 5 naprej v primeru podaljškov. Vrednost 7 torej pomeni 3. podaljšek. Ker je podaljškov malo, smo jih obravnavali enako kot zadnjih pet minut rednega dela tekme. To pomeni, da spremenljivka vedno zavzame vrednosti od 1 do 4.

Čas do konca četrtine

Število pomožnih spremenljivk: 11
Množica možnih vrednosti: TQ-0, TQ-1, TQ-2, TQ-3, TQ-4, TQ-5, TQ-6, TQ-7, TQ-8, TQ-9, TQ-10, TQ-11

Četrtino smo razdelili na 12 minutnih odsekov. Spremenljivka dobi vrednost tistega odseka, v katerem se je zgodil prvi dogodek iz kombinacije.

Čas do konca tekme

Število pomožnih spremenljivk: 47
Množica možnih vrednosti: T-0, T-1, T-2, T-3, T-4, T-5, T-6, T-7, T-8, T-8, T-10, T-11, T-12, T-13, T-14, T-15, T-16, T-17, T-18, T-19, T-20, T-21, T-22, T-23, T-24, T-25, T-26, T-27, T-28, T-29, T-30, T-31, T-32, T-33, T-34, T-35, T-36, T-37, T-38, T-39, T-40, T-41, T-42, T-43, T-44, T-45, T-46, T-47

Tekmo smo razdelili na 48 minutnih odsekov. Spremenljivka dobi vrednost tistega odseka, v katerem se je zgodil prvi dogodek iz kombinacije.

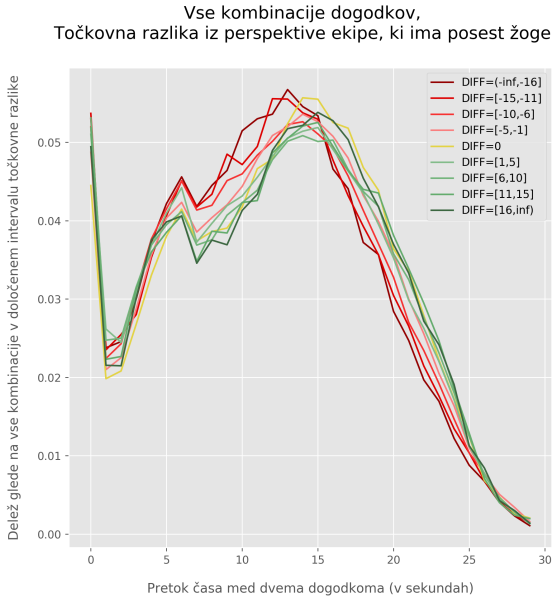
Točkovna razlika (s perspektive ekipe, ki ima posest žoge)

Število pomožnih spremenljivk: 8
Množica možnih vrednosti: DIFF-POS=(-inf,-16], DIFF-POS=[-15,-11], DIFF-POS=[-10,-6], DIFF-POS=[-5,-1], DIFF-POS=0, DIFF-POS=[1,5], DIFF-POS=[6,10], DIFF-POS=[11,15], DIFF-POS=[16,inf)

S kombinacijo spremenljivk posest žoge in točkovna razlika lahko ugotovimo, ali vodilne ekipe resnično igrajo daljše napade. Posest žoge smo s predhodnim znanjem o košarki določili ročno.

Točkovno razliko smo razdelili v devet kategorij. Prva izmed njih je izenačen rezultat, druga izmed njih vodstvo za več kot 15 točk, tretja pa vodstvo nasprotne ekipe za več kot 15 točk. Ostalih šest kategorij so intervali po 5 točk od -15 do +15.

Da vodilne ekipe resnično igrajo daljše napade, potrjuje graf na sliki 3.3.



Slika 3.3: Pretok časa med dvema dogodkoma v odvisnosti od točkovne razlike s perspektive ekipe, ki ima posest žoge.

Še bolj pa je to opazno v zadnji četrtini tekme, kar lahko vidimo na grafu na sliki 3.4

3.4.2 Zvezne spremenljivke

Čas do konca četrtine

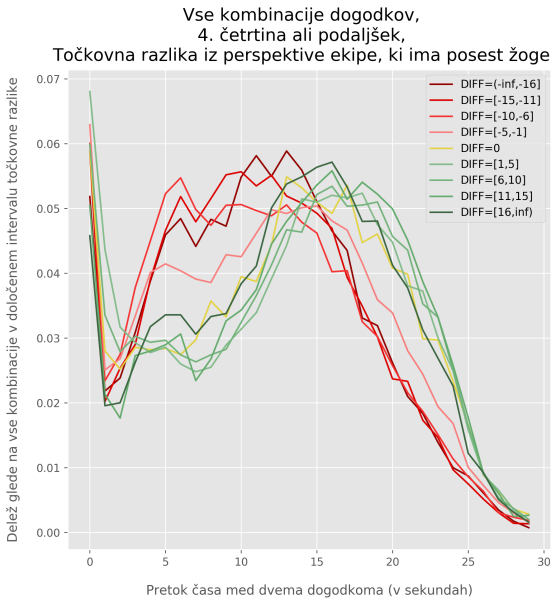
Interval vrednosti: od 0 do 720

Vrednosti so predstavljene v sekundah. Ker četrtina v košarkarski ligi NBA traja 12 minut, je maksimalno število sekund 720.

Čas do konca tekme

Interval vrednosti: od 0 do 2880

Tekma traja 48 minut plus morebitni podaljški. Ker je podaljškov malo, so obravnavani kot zadnjih 5 minut tekme rednega dela.



Slika 3.4: Pretok časa med dvema dogodkoma v zadnji četrtini oziroma podaljšku v odvisnosti od točkovne razlike s perspektive ekipe, ki ima posest žoge.

Točkovna razlika (s perspektive ekipe, ki ima posest žoge)

Interval vrednosti: od -56 do 54

Tempo igre ekipe, ki ima posest žoge

Interval vrednosti: od 86,874 do 103,100

Tempo igre je ocena števila posesti žoge posamezne ekipe v 48-ih minutah. S pajkom smo se sprehodili po spletnih straneh z obnovami tekem in izluščili informacije o tempu igre. Za vsako ekipo smo izračunali povprečen sezonski tempo igre od začetka sezone do vsake njihove tekme v sezoni.

Pretok časa med dogodkoma v predhodni kombinaciji

Interval vrednosti: od 0 do 63

V tej spremenljivki se nahaja število pretečenih sekund med dogodkoma iz predhodne kombinacije. Upoštevani so tudi pretoki časov predhodnih kombinacij, kjer je med dogodkoma minilo več kot 30 sekund.

Čas do konca četrtine (standardizirano)

Interval vrednosti: od -1,713 do 1,731

Čas do konca tekme (standardizirano)

Interval vrednosti: od -1.717 do 1.730

Točkovna razlika (s perspektive ekipe, ki ima posest žoge) (standardizirano)

Interval vrednosti: od -5.315 do 5.241

Tempo igre (standardizirano)

Interval vrednosti: od -2.214 do 4.076

Pretok časa med dogodkoma v predhodni kombinaciji (standardizirano)

Interval vrednosti: od -1.719 do 7.321

3.4.3 Kratice spremenljivk

V spodnjem seznamu so navedene kratice, ki smo jih uporabili pri predstavitivi posameznih modelov.

A1 - prvi dogodek v kombinaciji - brez podatka o ekipi (nominalna)

A2 - drugi dogodek v kombinaciji - brez podatka o ekipi (nominalna)

PA - predhodni dogodek - brez podatka o ekipi (nominalna)

PPA - pred predhodni dogodek - brez podatka o ekipi (nominalna)

A1P - prvi dogodek v kombinaciji - s podatkom o ekipi (nominalna)

A2P - drugi dogodek v kombinaciji - s podatkom o ekipi (nominalna)

PAP - predhodni dogodek - s podatkom o ekipi (nominalna)

PPAP - pred predhodni dogodek - s podatkom o ekipi (nominalna)

Q - četrtina (nominalna)

TQD - čas do konca četrtine (nominalna)

TGD - čas do konca tekme (nominalna)

DD - točkovna razlika s perspektive ekipe, ki ima posest žoge (nominalna)

TQ - čas do konca četrtine (zvezna)

TG - čas do konca tekme (zvezna)

D - točkovna razlika - s perspektive ekipe, ki ima posest žoge (zvezna)

P - tempo igre ekipe, ki ima posest žoge (zvezna)

PT - pretok časa med dogodkoma v predhodni kombinaciji (zvezna)

TQS - čas do konca četrtine (standardizirana zvezna)

TGS - čas do konca tekme (standardizirana zvezna)

DS - točkovna razlika s perspektive ekipe, ki ima posest žoge (standardizirana zvezna)

PS - tempo igre ekipe, ki ima posest žoge (standardizirana zvezna)

PTS - pretok časa med dogodkoma v predhodni kombinaciji (standardizirana zvezna)

3.4.4 Spremenljivke za opis ekip

Edina neodvisna spremenljivka, ki smo jo uporabili za opis ekip, je tempo igre. Ta nam neposredno pove, katere ekipe rade igrajo hitre in katere počasne napade. Poznamo še veliko spremenljivk, ki opisujejo kvaliteto ekip. Preprostejše opisujejo vse mogoče ekipne statistike, kot na primer število doseženih točk na tekmo, število doseženih skokov na tekmo in tako naprej. Ena izmed spremenljivk bi lahko bila tudi razmerje ekipe med zmagami in porazi.

Pri sorodnem klasifikacijskem problemu, napovedovanja naslednjega dogodka na tekmi poznamo tudi informativne ekspertne atribute (npr. four factors), ki se uspešno uporabljajo pri modeliranju košarke. Pri napovedovanju časa med dogodki na košarkarski tekmi pa teh atributov zaenkrat še ni. Atributi, ki opisujejo in ocenjujejo kvaliteto posameznih ekip, v svojih formulah ne vsebujejo časovne komponente, saj se na ta način enako obravnavajo ekipe, ki rade igrajo hitre oziroma počasne akcije.

Empirične raziskave so pokazale, da niti ekspertni atributi four factors niti neke druge statistike za opis ekip (na primer razmerje med hitrimi in počasnimi napadi) ne izboljšajo napovedi modela, ki uporablja samo čas do

konca tekme in trenutno razliko v rezultatu (vsaj ne pri modeliranju tako kakovostne in izenačene lige, kot je NBA). Meritve so bile opravljene na regresijskih drevesih, ki napovedi generirajo z vzorčenjem primerov v listih.

Poglavje 4

Linearni model

Večkratna (multipla) linearna regresija je metoda strojnega učenja, ki proučuje odvisnosti med večimi neodvisnimi spremenljivkami in eno odvisno spremenljivko. Linearna regresija poišče hiperravnino, ki se najbolj prilega podatkom in tako minimizira vsoto kvadratnih napak. V linearni regresiji kvadratno napako predstavlja kvadrirana razlika med dejansko vrednostjo odvisne spremenljivke in napovedano vrednostjo odvisne spremenljivke. Linearni model je dober, ko je povprečna kvadratna napaka dovolj majhna.

Več o linearni regresiji si lahko preberete v knjigi Introduction to Linear Regression Analysis [6].

4.1 Analiza neodvisnih spremenljivk

Variabilnost odvisne spremenljivke je vsota z modelom pojasnjene variabilnosti in nepojasnjene variabilnosti. Delež pojasnjene variance je kvocient med pojasnjeno varianco in celotno varianco. Označimo ga z r^2, imenujemo pa ga determinacijski koeficient. Njegova vrednost nam pove, kolikšen delež variance odvisne spremenljivke pojasnijo neodvisne spremenljivke, vključene v model. Večji kot je delež pojasnjene variabilnosti, boljši je model.

V tabeli 4.1 so rezultati, ki predstavljajo delež variance, ki jo pojasni model, v katerem je le ena neodvisna spremenljivka.

Tabela 4.1: Vrednosti r^2 predstavljajo delež variance, ki jo pojasni model, v katerem je le ena neodvisna spremenljivka.

Neodvisna spremenljivka	r^2 (Podatki 1)	r^2 (Podatki 2)
A1	0.335	0.614
A2	0.053	0.452
PA	0.320	0.549
PPA	0.050	0.322
A1P	0.336	0.614
A2P	0.053	0.452
PAP	0.320	0.549
PPAP	0.050	0.322
Q	0.000	0.000
TQD	0.019	0.014
TGD	0.023	0.015
DD	0.001	0.088
TQ	0.007	0.006
TG	0.000	0.000
D	0.001	0.053
P	0.004	0.002
PT	0.003	0.006
TQS	0.006	0.005
TGS	0.000	0.000
DS	0.001	0.027
PS	0.004	0.002
PTS	0.003	0.006

V Podatkih 2 kombinacije, kjer je prvi dogodek zgrešen met, napoveduje model. Pretok časa pri kombinacijah, kjer je prvi dogodek zgrešen met, je zelo predvidljiv. Posledično lahko tudi model dokaj natančno napove pretok časa za takšne kombinacije. Pojasnjena varianca je pri modelih grajenih na Podatkih 1 manjša, ker ti podatki vsebujejo manj predvidljivih kombinacij.

4.2 Analiza posameznih modelov

Za gradnjo modelov smo uporabili spremenljivke, opisane pri analizi posameznih spremenljivk. Uspešnost modelov smo določili z merjenjem napake napovedi pri posameznih kombinacijah v sezoni 2014/2015. Rezultati testiranj so prikazani v tabeli 4.3.

Model 0 predstavlja trivialni model. V njem je napoved za vsako kombinacijo enaka povprečnemu pretoku časa, izmerjenem na vseh vhodnih primerih učne množice.

Najprej smo testirali spremenljivke, ki predstavljajo dogodke. Pri prvih osmih modelih ugotovimo, da v linearnem modelu ni pomembno, ali imajo dogodki podatek o ekipi ali ne. Testiranje smo zato nadaljevali s spremenljivkami, ki nimajo podatka o ekipi, saj model vsebuje manjše število pomožnih spremenljivk, kar posledično pomeni preprostejši model.

Pri modelih 9, 10, 11, 12 smo testirali najboljšo kombinacijo predstavitve časa do konca tekme. Najboljšo kombinacijo pri Podatkih 1 vsebuje model 12, a ker nas je modul statsmodels opozoril na multikolinearnost, smo testiranje nadaljevali z modelom 10. Pri Podatkih 2 je najboljši rezultat dosegel model 11, zato smo testiranje nadaljevali s tem modelom.

Pri modelih 13, 14, 15 smo testirali najboljšo kombinacijo spremenljivk, ki predstavljajo točkovno razliko med ekipama. Za nadaljnje testiranje smo pri obojih podatkih uporabili model 13.

Pri modelih 16 in 17 smo dodali še spremenljivki tempo igre in pretok časa v predhodni kombinaciji. Obe spremenljivki sta pomagali izboljšati napoved.

V modelu 18 smo spremenljivke z dogodki brez podatka o ekipi zame-

njali s spremenljivkami, ki predstavljajo dogodke s podatkom o ekipi. Kot lahko vidimo, še vedno ni nobene razlike med dogodki s podatkom o ekipi in dogodki brez podatka o ekipi.

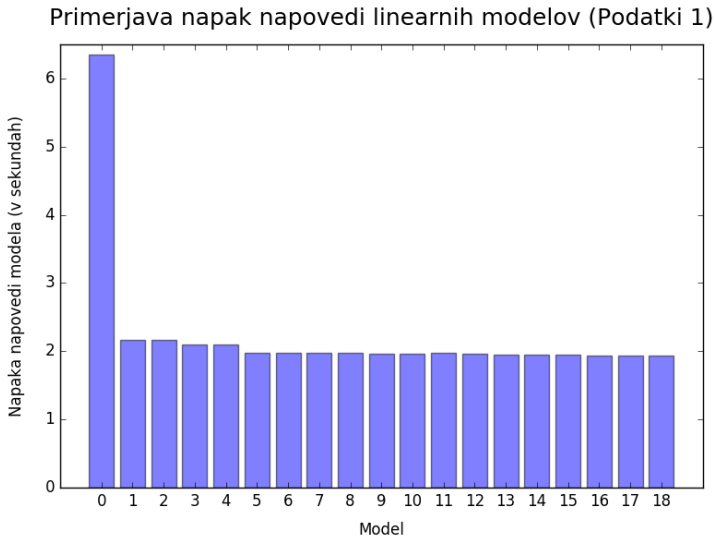
Tabela 4.2: Množice neodvisnih spremenljivk, ki jih predstavljajo posamezne številke modela. Pri zadnjih šestih modelih je za Podatke 1 uporabljena spremenljivka TGD, za Podatke 2 pa TGS.

Številka modela	Neodvisne spremenljivke
0	Trivialni model
1	A1
2	A1P
3	A1, A2
4	A1P, A2P
5	A1, A2, PA
6	A1P, A2P, PAP
7	A1, A2, PA, PPA
8	A1P, A2P, PAP, PPAP
9	A1, A2, PA, PPA, Q, TQD
10	A1, A2, PA, PPA, TGD
11	A1, A2, PA, PPA, TGS
12	A1, A2, PA, PPA, TGD, TQD
13	A1, A2, PA, PPA, TGD/TGS, DD
14	A1, A2, PA, PPA, TGD/TGS, DS
15	A1, A2, PA, PPA, TGD/TGS, DD, DS
16	A1, A2, PA, PPA, TGD/TGS, DD, PS
17	A1, A2, PA, PPA, TGD/TGS, DD, PS, PTS
18	A1P, A2P, PAP, PPAP, TGD/TGS, DD, PS, PTS

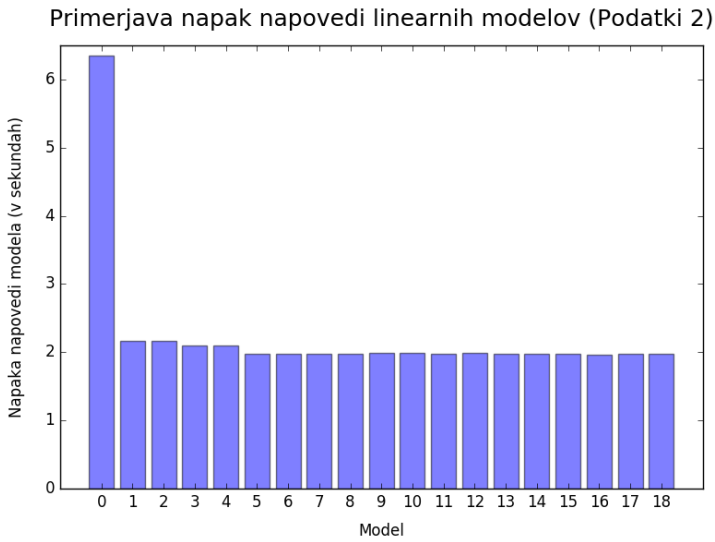
Rezultati za Podatke 1 so grafično predstavljeni na sliki 4.1, za Podatke 2 pa na sliki 4.2.

Tabela 4.3: Kombinacije neodvisnih spremenljivk, ki jih predstavljajo števila v stolpcu Model, se nahajajo v tabeli 4.2. Vrednosti v stolpcih Napaka so merjene na sezoni 2014/2015, predstavljene pa so v sekundah.

Model	r^2 (Podatki 1)	Napaka (Podatki 1)	r^2 (Podatki 2)	Napaka (Podatki 2)
0	-	6.357	-	6.357
1	0.335	2.168	0.614	2.169
2	0.336	2.168	0.614	2.169
3	0.379	2.100	0.640	2.100
4	0.379	2.099	0.640	2.100
5	0.443	1.982	0.676	1.983
6	0.443	1.982	0.676	1.983
7	0.445	1.979	0.677	1.981
8	0.445	1.979	0.677	1.981
9	0.458	1.962	0.682	1.993
10	0.461	1.957	0.683	1.990
11	0.445	1.979	0.677	1.981
12	0.461	1.957	0.683	1.990
13	0.465	1.952	0.679	1.976
14	0.463	1.954	0.678	1.980
15	0.465	1.952	0.679	1.976
16	0.469	1.940	0.681	1.970
17	0.469	1.939	0.681	1.971
18	0.470	1.939	0.681	1.971



Slika 4.1: Primerjava napak napovedi linearnih modelov v sezoni 2014/2015 na Podatkih 1.



Slika 4.2: Primerjava napak napovedi linearnih modelov v sezoni 2014/2015 na Podatkih 2.

4.3 Končni model

Kot lahko vidimo, smo boljše rezultate dobili z modeli naučenimi na Podatkih 1. To pomeni, da je ročno napovedovanje pretoka časa pri dogodkih, kjer je prvi dogodek zgrešen met, pomagalo pri kvaliteti napovedi.

Za najboljši model smo izbrali model 17, ker ima od vseh modelov z enakim najboljšim časom najmanj vhodnih atributov. Ima osem neodvisnih spremenljivk, kar po pretvorbi v pomožne spremenljivke pomeni 102 vhodna atributa. Uporabljene so spremenljivke prvi dogodek v kombinaciji, drugi dogodek v kombinaciji, predhodni dogodek, pred predhodni dogodek (vse v nominalni obliki brez podatka o ekipi), čas do konca tekme, točkovna razlika (obe v nominalni obliki), tempo igre ter pretok časa v predhodni kombinaciji (obe v standardizirani zvezni obliki).

Končno uspešnost modela smo testirali na sezoni 2015/2016, kjer napaka napovedi modela znaša 1.927 sekunde, varianca napake 7.684 sekunde, relativna napaka glede na napako trivialnega modela pa 0.305.

Poglavje 5

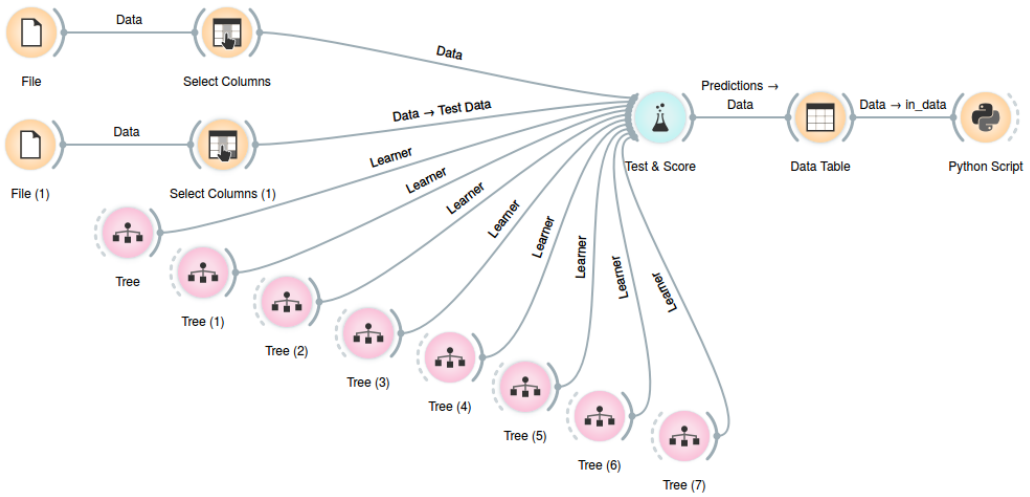
Regresijska drevesa

Regresijska drevesa so vrsta odločitvenih dreves, kjer je odvisna spremenljivka zvezna. Za razliko od linearne regresije so sposobna modelirati nelinearne relacije med odvisnimi in neodvisnimi spremenljivkami. Linearni modeli modelirajo celotno množico vhodnih primerov, medtem ko odločitvena drevesa razdelijo celotno množico vhodnih primerov v manjše podmnožice, ki so potem modelirane vsaka posebej. Zaporedje odločitev od korena do lista je samo pot, ki za vsak vhodni primer določi napoved. Več o odločitvenih drevesih si lahko preberete v knjigi Machine Learning and Data Mining [4].

5.1 Orange

Za modeliranje dreves smo uporabili program Orange, ki pri gradnji dreves uporablja vnaprejšnje rezanje (ang. pre-pruning). Implementacija omogoča gradnjo klasifikacijskih in regresijskih dreves. Za naš problem smo uporabili regresijska, saj je odvisna spremenljivka zvezna. Zelo dobra lastnost implementacije je sposobnost obravnave tako zveznih kot tudi nominalnih neodvisnih spremenljivk. Prav tako je za naš problem koristna sposobnost gradnje nebinarnih dreves, saj imajo nekatere nominalne neodvisne spremenljivke veliko možnih vrednosti. V listih dreves se nahaja točkovni model.

Celotna shema uporabljenih gradnikov je prikazana na sliki 5.1. Gra-



Slika 5.1: Shema gradnikov v programu Orange.

dnika File vsebujeta učno in testno množico. Gradnika Select columns nam omogočata izbiro neodvisnih spremenljivk, ki jih želimo uporabiti v posameznih modelih. Različne kombinacije neodvisnih spremenljivk smo testirali na binarnih in nebinarnih drevesih. Binarizacija se nastavi v gradniku Tree, kjer se prav tako nastavijo vrednosti še nekaterih drugih parametrov. Vsak izmed osmih gradnikov Tree je uporabil svojo kombinacijo parametrov iz tabele 5.1.

Tabela 5.1: Kombinacije parametrov za modeliranje regresijskih dreves.

Minimalno število primerov v listih	Ne razcepi podmnožic, manjših od
20	50
40	100
50	125
60	150
80	200
100	250
120	280
150	375

Gradnik Test & Score ocenjuje kvaliteto modelov, napovedi na testni množici so podane v gradniku Data Table, povprečno napako napovedi pa smo izračunali v gradniku Python Script, ki omogoča pisanje Python kode.

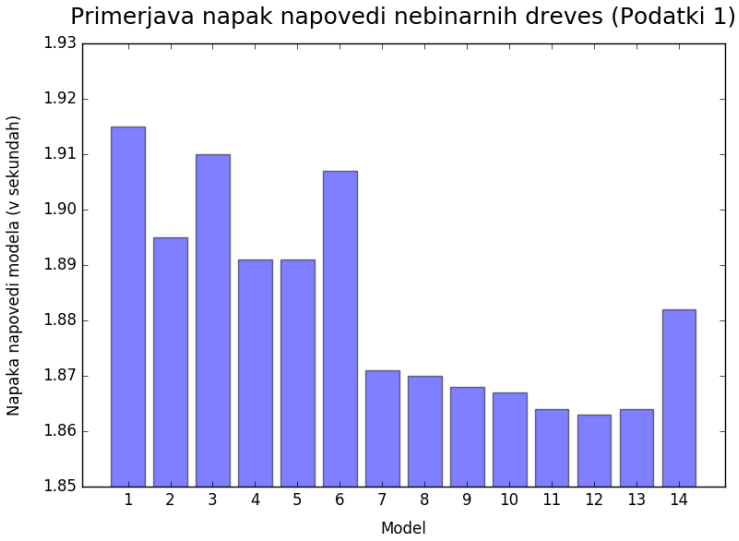
5.2 Nebinarna regresijska drevesa

Nebinarna odločitvena drevesa omogočajo razcep na več kot dve veji v posameznem vozlišču. Rezultati testiranj različnih kombinacij neodvisnih spremenljivk na modelih nebinarnih regresijskih dreves so predstavljeni v tabeli 5.2.

Tabela 5.2: Vrednosti v stolpcih Napaka so merjene na sezoni 2014/2015, predstavljene pa so v sekundah. P1 pomeni Podatki 1, P2 pa Podatki 2.

	Neodvisne spremenljivke	Napaka (P1)	Napaka (P2)
1	A1, A2, PA, TGD	1.915	1.914
2	A1, A2, PA, TG	1.895	1.893
3	A1, A2, PA, TG, DD	1.910	1.908
4	A1, A2, PA, TG, D	1.891	1.889
5	A1, A2, PA, TGS, DS	1.891	1.889
6	A1P, A2P, PAP, TG, D	1.907	1.905
7	A1, A2, PA, TG, TGD, D	1.871	1.869
8	A1, A2, PA, TG, TGD, D, DD	1.870	1.868
9	A1, A2, PA, TG, TGD, D, P	1.868	1.866
10	A1, A2, PA, TG, TGD, D, PT	1.867	1.859
11	A1, A2, PA, TG, TGD, D, P, PT	1.864	1.856
12	A1, A2, PA, PPA, TG, TGD, D, P, PT	1.863	1.857
13	A1, A2, PA, TG, TGD, DD, D, P, PT	1.864	1.857
14	A1P, A2P, PAP, TG, TGD, D, P, PT	1.882	1.875

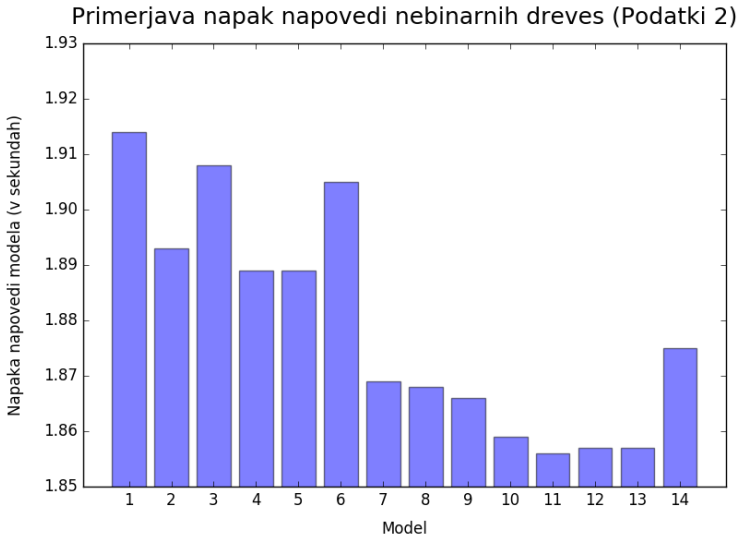
Modeli 1, 2, 3 in 4 pokažejo, da spremenljivki čas do konca tekme in točkovna razlika dajeta boljše rezultate v zvezni obliki. Model 5 potrdi, da pri odločitvenih drevesih standardizacija neodvisnih spremenljivk ne spremeni ničesar. Pri šestem modelu vidimo, da so pri neodvisnih spremenljivkah, ki predstavljajo dogodke, veliko boljša izbira tiste brez podatka o ekipi.



Slika 5.2: Primerjava napak napovedi nebinarnih dreves v sezoni 2014/2015 na Podatkih 1.

V modelu 7 dodamo spremenljivko čas do konca tekme tudi v nominalni obliki, kar prinese precejšnjo izboljšavo rezultatov. Podobno v modelu 8 poskusimo s spremenljivko točkovna razlika v nominalni obliki, a izboljšava ni tako očitna. V modelih 9, 10, 11 so testirane različne kombinacije spremenljivk tempo igre in pretok časa v predhodni kombinaciji dogodkov. Obe še dodatno znižata napako napovedi. Modelu 12 je dodana spremenljivka pred predhodni dogodek, ki pri Podatkih 1 izboljša rezultat za 0.00012 sekunde, pri Podatkih 2 pa ga poslabša. Trinajsti model ponovno poskusi dodati spremenljivko točkovna razlika v nominalni obliki, a izboljšave rezultatov ni. Zadnji model uporabi enake neodvisne spremenljivke kot model 11, le da dogodkovne spremenljivke vsebujejo podatek o ekipi. Napovedi so pričakovano precej slabše. Rezultati za Podatke 1 so grafično predstavljeni na sliki 5.2, za Podatke 2 pa na sliki 5.3.

Pri nebinarnih regresijskih drevesih boljše rezultate dobimo na Podatkih 2, kar pomeni, da drevesa kombinacije, kjer je prvi dogodek zgrešen met, napovedujejo bolje kot ročna napoved, ki temelji na predznanju košarke.



Slika 5.3: Primerjava napak napovedi nebinarnih dreves v sezoni 2014/2015 na Podatkih 2.

Za najbolši model izberemo model številka 11. Uporabljene spremenljivke so prvi dogodek v kombinaciji, drugi dogodek v kombinaciji, predhodni dogodek, čas do konca tekme (vse v nominalni obliki brez podatka o ekipi), čas do konca tekme, točkovna razlika, tempo igre in pretok časa v predhodni kombinaciji (vse v zvezni obliki).

5.3 Binarna regresijska drevesa

Program Orange pri gradnji binarnih dreves preprečuje uporabo nominalnih spremenljivk, pri katerih se pojavlja več kot 16 različnih vrednosti. Pri Podatkih 2 je tako onemogočena uporaba spremenljivke predhodni dogodek. Gre za eno izmed najpomembnejših spremenljivk, brez katere so rezultati občutno slabši. Posledično smo se odločili, da binarnih dreves na Podatkih 2 ne bomo gradili. Pri Podatkih 1 smo lahko od dogodkovnih spremenljivk uporabili le spremenljivke prvi dogodek v kombinaciji, drugi dogodek v kombinaciji ter predhodni dogodek (vse brez podatka o ekipi). Prav tako nismo

mogli uporabiti spremenljivke čas do konca tekme v nominalni obliki, saj le ta vsebuje 48 različnih vrednosti. Rezultati gradnje binarnih modelov na Podatkih 1 so prikazani v tabeli 5.3.

Tabela 5.3: Vrednosti v stolpcu Napaka so merjene na sezoni 2014/2015, predstavljene pa so v sekundah.

	Neodvisne spremenljivke	Napaka (Podatki 1)
1	A1, A2, PA, TG, D	1.865
2	A1, A2, PA, TG, DD	1.865
3	A1, A2, PA, TG, DD, D	1.867
4	A1, A2, PA, TG, DD, P	1.874
5	A1, A2, PA, TG, DD, PT	1.863
6	A1, A2, PA, TG, DD, P, PT	1.869

Prvi trije modeli pokažejo, da je predstavitev spremenljivke točkovna razlika najuspešnejša v obliki ene nominalne spremenljivke. V modelu 5 dodamo spremenljivko pretok časa v predhodni kombinaciji, kar prinese izboljšano napoved. Četrti in šesti model pokažeta, da je spremenljivka tempo igre pri binarnih drevesih povsem neuporabna.

Za najboljšega se izkaže peti model.

5.4 Končni model

Najboljšo povprečno napoved pretoka časa je prinesel 11. model pri nebinarnih regresijskih drevesih. V njem so uporabljene neodvisne spremenljivke prvi dogodek v kombinaciji, drugi dogodek v kombinaciji, predhodni dogodek, čas do konca tekme (vse v nominalni obliki brez podatka o ekipi), čas do konca tekme, točkovna razlika, tempo igre in pretok časa v predhodni kombinaciji (vse v zvezni obliki).
Vrednosti ostalih parametrov:

- Število notranjih vozlišč: 14020

- Število listov: 8303
- Minimalno število instanc v listih: 60
- Ne razcepi podmnožic manjših od: 150

Končno uspešnost modela smo testirali na sezoni 2015/2016, kjer napaka napovedi modela znaša 1.859 sekunde, varianca napake 7.152 sekunde, relativna napaka glede na napako trivialnega modela pa 0.294.

Poglavje 6

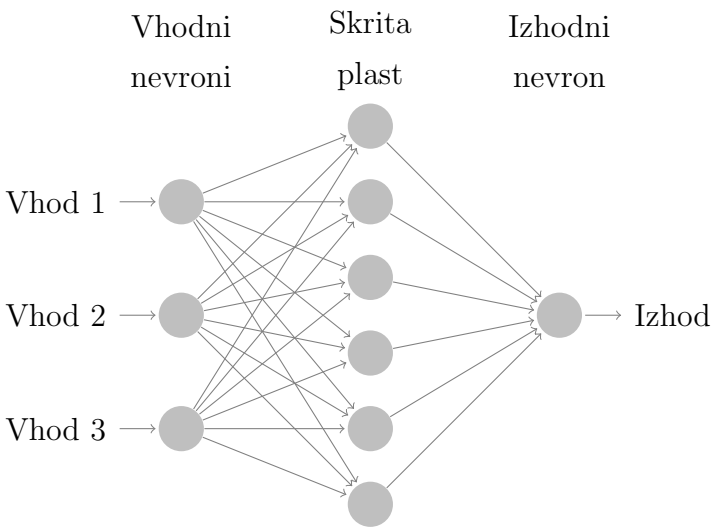
Nevronske mreže

Standardni algoritmi sledijo zaporedju ukazov in nas na koncu pripeljejo do rešitve določenega problema. Pomankljivost takega pristopa je omejenost na reševanje problemov, ki jih že znamo rešiti. Zelo uporabno bi bilo reševati tudi probleme, za katere še ne vemo, kako bi jih rešili.

Nevronska mreža je računski model, ki računalniku omogoča učenje na podlagi opazovanih podatkov. Koncept njihovega delovanja temelji na delovanju bioloških možganov. Sestavljene so iz vhodnih nevronov, ene ali več skritih plasti nevronov in izhodnih nevronov. Ti nevroni so med seboj povezani. Informacije niso procesirane linearno, temveč paralelno skozi mrežo nevronov. Po povezavah med nevroni se širijo različno močni aktivacijski signali. Če je kombinacija vhodnih signalov pri določenem nevronu dovolj močna, se ta aktivira, signal pa potuje v nadaljnje nevrone, ki so povezani z njim. Primer nevronske mreže je prikazan na sliki 6.1.

Nevronske mreže imajo sposobnost izluščiti pomen iz zapletenih in netočnih podatkov. Prepoznajo lahko različne vzorce in trende, ki jih človek oziroma ostali algoritmi niso zmožni. Ker nevronske mreže same rešujejo probleme, je lahko njihovo delovanje nepredvidljivo. Trenutno ponujajo najboljše rezultate pri problemih prepoznavanja slik, govora in naravnih jezikov.

Več o nevronskih mrežah si lahko preberete v knjigi Neural Networks and Deep Learning [7].



Slika 6.1: Primer nevronske mreže s tremi vhodnimi nevroni, šestimi nevroni v edini skriti plasti ter enim izhodnim nevronom

6.1 Izbor strukture mreže

Kot smo že omenili, so nevronske mreže sestavljene iz vhodnih nevronov, ene ali več skritih plasti nevronov in izhodnih nevronov. Število vhodnih nevronov je odvisnih od v model vključenih neodvisnih spremenljivk. Izhoden nevron je en sam, ker je naš problem regresijski. Testiramo lahko torej različne kombinacije števila skritih plasti in števila nevronov v posamezni skriti plasti.

Vsako kombinacijo neodvisnih spremenljivk smo testirali na štirih različnih strukturah nevronskih mrež, predstavljenih v tabeli 6.1.

6.2 Izbor neodvisnih spremenljivk

Nevronske mreže se same učijo iz vhodnih podatkov. Ker na postopek učenja dejansko nimamo veliko vpliva, smo preverili rezultate za številne kombinacije neodvisnih spremenljivk.

Za testiranje najboljše kombinacije neodvisnih spremenljivk smo upora-

Tabela 6.1: Število vhodnih nevronov je odvisno od izbora neodvisnih spremenljivk.

Število skritih plasti	Število nevronov v posamezni skriti plasti
1	1 * število vhodnih nevronov
1	2 * število vhodnih nevronov
2	1 * število vhodnih nevronov
2	2 * število vhodnih nevronov

bili naslednje parametre:

- struktura mreže: vsako kombinacijo smo testirali na štirih strukturah, predstavljenih v tabeli 6.1.
- aktivacijska funkcija nevronov: ReLU
- optimizacijski algoritem: Adam (učni korak 0.001)
- inicializacija uteži: normal
- število prehodov preko učne množice: 40
- velikost paketa: 100

V tabeli 6.3 je vsaka kombinacija predstavljena s strukturo mreže, ki je dosegla najnižjo povprečno napako napovedi.

Testiranje modelov 1 in 2 potrdi, da je spremenljivka čas do konca tekme daje boljše rezultate v nominalni obliki. Model 3 nam pokaže, da spremenljivka točkovna razlika prav tako daje boljše rezultate v nominalni obliki. V modelu 4 dodamo spremenljivko pretok časa v predhodni kombinaciji, ki še dodatno izboljša napoved. V modelu 5 dodana spremenljivka pred predhodni dogodek napovedi ne izboljša. Modeli 6, 7 in 8 nam lepo pokažejo, da boljše rezultate dajejo tiste dogodkovne spremenljivke, ki ne vsebujejo podatkov o ekipi.

Nadaljnja testiranja modelov zato vsebujejo dogodkovne spremenljivke, ki ne vsebujejo podatkov o ekipi. Zvezne spremenljivke v modelih od 1 do 8 so

Tabela 6.2: Množice neodvisnih spremenljivk, ki jih predstavljajo posamezne številke modela.

Model	Neodvisne spremenljivke
1	A1, A2, PA, TGD, DS, PS
2	A1, A2, PA, TGS, DS, PS
3	A1, A2, PA, TGD, DD, PS
4	A1, A2, PA, TGD, DD, PS, PTS
5	A1, A2, PA, PPA, TGD, DD, PS, PTS
6	A1P, A2P, PAP, TGD, DD, PS
7	A1P, A2P, PAP, TGD, DD, PS, PTS
8	A1P, A2P, PAP, PPAP, TGD, DD, PS, PTS
9	A1, A2, PA, TGD, DD, P, PT
10	A1, A2, PA, PPA, TGD, DD, P, PT

bile standardizirane. V modelu 9 poskusimo z zveznimi spremenljivkami brez standardizacije. Izkaže se, da so rezultati boljši, če zvezne spremenljivke niso standardizirane. Zadnji model doda spremenljivko pred predhodni dogodek, ki sedaj, ko so zvezne spremenljivke nestandardizirane, izboljša rezultate.

Pri nekaterih kombinacijah neodvisnih spremenljivk, dajejo boljše rezultate mreže, ki uporabljajo Podatke 1, pri drugih pa mreže, ki uporabljajo Podatke 2.

Najboljšo kombinacijo neodvisnih spremenljivk vsebuje model 10, ki uporablja Podatke 2. Najboljše rezultate doseže s strukturo (114, 114, 114, 1). To pomeni, da ima model 114 vhodnih atributov, dve skriti plasti s po 114-imi nevroni in en izhoden nevron. Kombinacijo neodvisnih spremenljivk smo testirali še na strukturah (114, 342, 342, 1), (114, 114, 114, 114, 1) in (114, 228, 228, 228, 1), a nobena izmed njih ni izboljšala rezultatov.

Model vsebuje spremenljivke prvi dogodek v kombinaciji, drugi dogodek v kombinaciji, predhodni dogodek, pred predhodni dogodek (vse v nominalni obliki brez podatka o ekipi), čas do konca tekme, točkovna razlika (obe v no-

Tabela 6.3: Rezultati testiranja različnih kombinacij neodvisnih spremenljivk in struktur nevronske mreže. Kombinacije neodvisnih spremenljivk, ki jih predstavljajo števila v stolpcu Model se nahajajo v tabeli 6.2. Vrednosti v stolpcih Napaka so merjene na sezoni 2014/2015, predstavljene pa so v sekundah. V stolpcih Struktura vsako število predstavlja število nevronov v posamezni plasti. Prvo število predstavlja vhodno plast, zadnje število predstavlja en izhoden nevron, vsako vmesno število pa predstavlja po eno skrito plast. P1 pomeni Podatki 1, P2 pa Podatki 2.

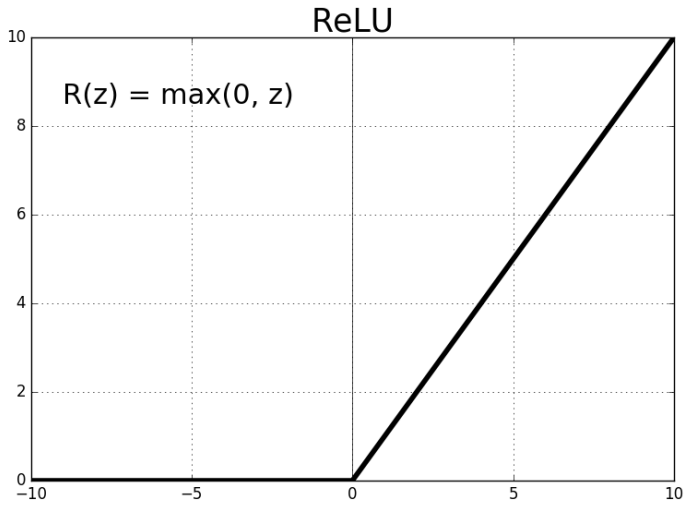
Model	Struktura (P1)	Napaka (P1)	Struktura (P2)	Napaka (P2)
1	(78, 78, 78, 1)	1.885	(87, 87, 87, 1)	1.881
2	(32, 64, 64, 1)	1.926	(41, 41, 41, 1)	1.920
3	(85, 85, 85, 1)	1.875	(94, 94, 94, 1)	1.871
4	(86, 86, 86, 1)	1.866	(95, 95, 95, 1)	1.862
5	(103, 103, 1)	1.866	(114, 114, 114, 1)	1.868
6	(115, 115, 1)	1.880	(133, 133, 133, 1)	1.878
7	(116, 116, 1)	1.871	(134, 134, 134, 1)	1.877
8	(150, 150, 1)	1.870	(172, 172, 1)	1.880
9	(86, 172, 172, 1)	1.850	(95, 190, 190, 1)	1.854
10	(103, 206, 206, 1)	1.844	(114, 114, 114, 1)	1.843

minalni obliki), tempo igre in pretok časa v predhodni kombinaciji (obe v zvezni nestandardizirani obliki). S tem modelom smo tudi nadaljevali nadaljnja testiranja.

6.3 Aktivacijska funkcija nevrona

Aktivacijske funkcije v nevronske mreže vključijo nelinearne lastnosti. Gre za zelo pomemben koncept, saj z nevronskimi mrežami želimo predstaviti kakršno koli funkcijo in ne samo linearne. V primeru, da ne bi uporabljali nelinearnih aktivacijskih funkcij, bi na koncu ne glede na število skritih plasti vedno dobili linearno funkcijo.

Za učenje nevronskih mrež se največ uporablja algoritem imenovan stan-



Slika 6.2: Funkcija ReLU

dardno vzvratno razširjanje (ang. backpropagation). Ker v postopku tega algoritma med drugim pride tudi do odvajanja aktivacijske funkcije, mora biti ta odvedljiva. Za aktivacijsko funkcijo posameznega nevrona smo izbrali zelo pogosto uporabljeno funkcijo ReLU (ang. Rectified linear unit). Kot lahko vidimo na grafu 6.2, pa ta funkcija ni odvedljiva v točki $x = 0$. Ena izmed možnih rešitev je ročna/samodejna nastavitev vrednosti odvoda v točki $x = 0$. Pogosto uporabljene vrednosti so 0, 0,5 in 1.

Omenjena funkcija v nasprotju s sigmoidno funkcijo ne vsebuje računsko zahtevnih operacij, kar posledično pomeni hitrejše učenje. Poleg tega se izogne tudi problemu izginjajočega gradienta, ki prav tako onemogoča hitrejše učenje. Funkcijo lahko opišemo s preprosto enačbo:

$$R(z) = max(0, z)$$

6.4 Testiranje optimizacijskega algoritma

Za optimizacijo smo uporabili algoritem, imenovan Adam [3]. Gre za napredno različico gradientnega spusta, ki osveži vrednost uteži glede na posamezne vhodne aribute. To pomeni, da se pogosto pojavljajoči atributi osvežujejo počasneje, kot pa tisti, ki se pojavljajo manj pogosto. Za vsak atribut v določenem trenutku je uporabljen drugačen učni korak. Ta temelji na preteklih izračunih gradienta za ta atribut. Posledično je ročno prilagajanje učnega koraka nepotrebno. Poleg tega Adam računa tudi spremembe momentuma za posamezne atribute, kar še dodatno izboljša optimizacijo.

Vrednost parametra učni korak določa hitrost posodabljanja uteži. Testirali smo različne vrednosti tega parametra. Rezultati so prikazani v tabeli 6.4.

Tabela 6.4: Rezultati testiranja učnega koraka pri algoritmu Adam. Vrednosti v stolpcu Napaka so merjene na sezoni 2014/2015, predstavljene pa so v sekundah.

Učni korak	Napaka
0.0001	1.871
0.001	1.843
0.01	1.862
0.1	1.963
0.2	4.342
0.3	4.344

Najboljši rezultat smo dobili z učnim korakom 0.001, zato smo testiranje nadaljevali s to vrednostjo.

6.5 Testiranje različnih inicializacij uteži

Včasih je veljalo preprosto pravilo, da se za začetne vrednosti uteži uporabijo majhne naključne vrednosti, danes pa knjižnice za strojno učenje ponujajo

vrsto različnih inicializacij začetnih vrednosti uteži. V tabeli 6.7 so rezultati različnih inicializacij, ki jih ponuja API Keras.

Inicializacija ničle (ang. zero) začetne vrednosti uteži nastavi na 0. Normalna inicializacija (ang. normal) naključne začetne vrednosti generira na podlagi normalne verjetnostne porazdelitve. Ena izmed izpeljank normalne inicializacije je okrnjena normalna inicializacija (ang. truncated normal initialitation), kjer so vrednosti, ki so od pričakovane vrednosti oddaljene več kot dve enoti standardnega odklona, zavržene. Vse izpeljanke normalne inicializacije imajo prikačkovano vrednost 0. Spreminja se le standardni odklon. Zadnji dve inicializaciji v tabeli 6.5 generirata vrednosti na osnovi okrnjene normalne inicializacije.

Zvezno enakomerna inicializacija (ang. uniform) začetne vrednosti generira na podlagi zvezno enakomerne porazdelitve. Spreminjata se le zgornja in spodnja meja, ki določata interval vrednosti (tabela 6.6).

Tabela 6.5: Izpeljanke normalne inicializacije. Spremenljivka n_{in} predstavlja število vhodnih nevronov v nevron, za katerega se generira vrednost uteži. Spremenljivka n_{out} predstavlja število nevronov, v katere gre aktivacija nevrona, za katerega se generira vrednost.

Inicializacija uteži	Standardni odklon
normal	0.05
truncated_normal	0.05
he_normal	$\sqrt{\frac{2}{n_{in}}}$
glorot_normal	$\sqrt{\frac{2}{n_{in}+n_{out}}}$

Najboljši rezultat smo dobili z inicializatorjem imenovanim He normal, ki smo ga potem uporabili pri nadaljnjih testiranjih.

Tabela 6.6: Izpeljanke zvezno enakomerne inicializacije. Spremenljivka n_{in} predstavlja število vhodnih nevronov v nevron, za katerega se generira vrednost uteži. Spremenljivka n_{out} predstavlja število nevronov, v katere gre aktivacija nevrona, za katerega se generira vrednost.

Inicializacija uteži	Zgornja meja	Spodnja meja
uniform	-0.05	$+0.05$
lecun_uniform	$-\sqrt{\frac{3}{n_{in}}}$	$+\sqrt{\frac{3}{n_{in}}}$
he_uniform	$-\sqrt{\frac{6}{n_{in}}}$	$+\sqrt{\frac{6}{n_{in}}}$
glorot_uniform	$-\sqrt{\frac{6}{n_{in}+n_{out}}}$	$+\sqrt{\frac{6}{n_{in}+n_{out}}}$

6.6 Testiranje iteracij v postopku učenja

Velikost paketa (ang. batch size) pri metodi najhitrejšega spusta nam pove, koliko vhodnih primerov je pokazanih nevronski mreži preden se zgodi posodobitev uteži. Število prehodov preko učne množice (ang. epochs) pa nam pove, kolikokrat so nevronski mreži pokazani vsi vhodni primeri. Rezultati različnih kombinacij teh dveh parametrov so prikazani v tabeli 6.8. Kot lahko vidimo, je najboljša kombinacija še vedno:

- število prehodov preko učne množice: 40
- velikost paketa: 100

6.7 Končni model

Končni model vsebuje neodvisne spremenljivke prvi dogodek v kombinaciji, drugi dogodek v kombinaciji, predhodni dogodek, pred predhodni dogodek (vse v nominalni obliki brez podatka o ekipi), čas do konca tekme, točkovna razlika (obe v nominalni obliki), tempo igre, pretok časa v predhodni kombinaciji (obe v nestandardizirani zvezni obliki).
Vrednosti ostalih parametrov:

Tabela 6.7: Rezultati testiranja različnih inicializacij uteži. Vrednosti v stolpcu Napaka so merjene na sezoni 2014/2015, predstavljene pa so v sekundah.

Inicializacija uteži	Napaka
zero	4.340
normal	1.843
truncated_normal	1.854
he_normal	1.841
glorot_normal	1.846
uniform	1.847
lecun_uniform	1.845
he_uniform	1.852
glorot_uniform	1.853

- struktura mreže: (114, 114, 114, 1) - 114 vhodnih nevronov, dve skriti plasti s po 114-imi nevroni ter en izhoden nevron
- aktivacijska funkcija: ReLU
- optimizacijski algoritem: Adam (učni korak = 0.001)
- inicializacija uteži: He normal
- Število prehodov preko učne množice: 40
- Velikost paketa: 100

Končno uspešnost modela smo testirali na sezoni 2015/2016, kjer napaka napovedi modela znaša 1.838 sekunde, varianca napake 7.099 sekunde, relativna napaka glede na napako trivialnega modela pa 0.291.

Tabela 6.8: Rezultati testiranja različnih kombinacij števila prehodov preko učne množice in velikosti paketa. Vrednosti v stolpcu Napaka so merjene na sezoni 2014/2015, predstavljene pa so v sekundah.

Število prehodov preko učne množice	Velikost paketa	Napaka
40	100	1.841
10	10	1.878
10	40	1.911
10	70	1.922
10	100	1.896
30	10	1.843
30	40	1.878
30	70	1.889
30	100	1.891
50	10	1.886
50	40	1.845
50	70	1.846
50	100	1.850
70	10	1.876
70	40	1.890
70	70	1.888
70	100	1.864

Poglavje 7

Zaključek

7.1 Končni rezultati

V tabeli 7.1 so predstavljeni najboljši modeli posameznih metod strojnega učenja. Kot lahko vidimo, je najboljši rezultat dosegla nevronska mreža, nekoliko slabšega pa regresijsko dreveso. Pričakovano se je najslabše odrezal linearni model, saj so bile v našem problemu relacije med odvisnimi in neodvisnimi spremenljivkami nelinearne.

Pri kombinacijah, kjer so prvi dogodki zgrešeni meti, je ročna napoved na podlagi košarkarskega predznanja precej bolje napovedala pretok časa, kot pa linearni model. Pri nevronskih mrežah so si bili rezultati zelo podobni, medtem ko so regresijska drevesa premagala ročno napoved pri omenjenih kombinacijah. To se morda zdi nekoliko presenetljivo, saj naj bi bil odboj od obroča glede na naše vhodne podatke naključen, a bolj kot dejanski odboj je pomemben zapis posameznega dogodka. Zgrešenemu metu v zadnji sekundi lahko sledi zapis ekipnega skoka v napadu. Pretok časa med dogodkoma je torej 0 sekund, čeprav dejanskega skoka sploh ni bilo. Regresijska drevesa so verjetno prepoznala takšne in podobne vzorce, medtem ko je ročna napoved napovedala povprečen pretok časa.

Tabela 7.1: Primerjava rezultatov različnih metod strojnega učenja. Vrednosti v stolpcu Napaka so merjene na sezoni 2015/2016, predstavljene pa so v sekundah. Relativna napaka je merjena glede na napako Trivialnega modela, ki vedno napove povprečen čas med dvema dogodkoma.

Metoda strojnega učenja	Napaka	Relativna napaka	Varianca napake
Trivialni model	6.325	1	13.607
Linearni model (Podatki 1)	1.927	0.305	7.684
Regresijsko drevo (Podatki 2)	1.859	0.294	7.152
Nevronska mreža (Podatki 2)	1.838	0.291	7.099

7.2 Sklepna misel

Ugotovili smo, da so najpomembnejše spremenljivke, ki smo jih imeli na voljo za napovedovanje pretoka časa med dogodki, kar sami dogodki. Napadi se naprimer hitreje zaključujejo po skoku v obrambi, kot pa po izvajanju iz avta po doseženem košu nasprotne ekipe. Eden izmed razlogov za to je več časa za postavitev obrambe. Napadi po skoku v napadu se v povprečju zaključujejo v le nekaj sekundah, saj igralec pogosto ujame žogo v bližini koša, kar pomeni ugodno pozicijo za met na koš. Prav tako smo pokazali, da ekipe v vodstvu ponavadi igrajo daljše napade, še posebej na koncu tekme. Nekatere ekipe igrajo hitrejšo napadalnejšo igro, spet druge počasnejšo, kar tudi vpliva na pretok časa med dogodki.

Vse zgoraj omenjene ugotovitve so zelo splošne ugotovitve dinamike košarkarske igre. V realnosti pa je pretok časa med dogodkoma zelo odvisen od situacije na igrišču, najpogosteje od postavitve igralcev. Če ekipa ukrade žogo pod svojim obročem, je precej manjša verjetnost za protinapad, kot pa če bi žogo ukradli na nasprotnikovi polovici. Tudi kadar ima ekipa prostega igralca pod nasprotnikovem košem, niti malo ni pomembno, kateri dogodek je pripeljal do omenjene situacije. V (skoraj) vsakem primeru bo naslednja podaja namenjena njemu za hitro dosego lahkega koša.

Prostora za izboljšave je še dovolj, vendar so le te večinoma odvisne od vhodnih podatkov, ki jih imamo na voljo. Dodatna neodvisna spremenljivka

bi lahko bil recimo igralec, ki je pri dogodku sodeloval. Nekateri igralci v določenih situacijah sprejemajo drugačne odločitve kot drugi. Morda bi model lahko prepoznal stil igre posameznih igralcev in tako še izboljšal napoved. Kot smo omenili že v uvodu, obstajajo tudi že podatki, ki sledijo premikanju igralcev po igrišču. Ti podatki bi zelo dobro opisali situacijo na igrišču in verjetno še dodatno izboljšali napoved.

Literatura

[1] Maral Haghighat, Hamid Rastegari, and Nasim Nourafza. A review of data mining techniques for result prediction in sports. *Advances in Computer Science : an International Journal*, 2(5):7–12, 2013.

[2] Mark Harmon, Patrick Lucey, and Diego Klabjan. Predicting shot making in basketball using convolutional neural networks learnt from adversarial multiagent trajectories. *arXiv preprint arXiv:1609.04849*, 2016.

[3] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[4] Igor Kononenko and Matjaz Kukar. *Machine Learning and Data Mining: Introduction to Principles and Algorithms*. Horwood Publishing Limited, 2007.

[5] Patrick Lucey, Alina Bialkowski, Mathew Monfort, Peter Carr, and Iain Matthews. quality vs quantity: Improved shot prediction in soccer using strategic features from spatiotemporal data. In *Proc. 8th Annual MIT Sloan Sports Analytics Conference*, pages 1–9, 2014.

[6] D.C. Montgomery, E.A. Peck, and G.G. Vining. *Introduction to Linear Regression Analysis*. Wiley Series in Probability and Statistics. Wiley, 2015.

[7] Michael A. Nielsen. *Neural Networks and Deep Learning*. Determination press, 2015. `http://neuralnetworksanddeeplearning.com/`, [dostopano 8.1.2018].

[8] Erik Štrumbelj and Petar Vračar. Simulating a basketball match with a homogeneous markov model and forecasting the outcome. *International Journal of Forecasting*, 28(2):532–542, 2012.

[9] Yisong Yue, Patrick Lucey, Peter Carr, Alina Bialkowski, and Iain Matthews. Learning fine-grained spatial models for dynamic sports play prediction. In *2014 IEEE International Conference on Data Mining*, pages 670–679. IEEE, 2014.