Analiza skupa podataka Children Killed or Injured

- Seminarski rad -

Istraživanje podataka

Matematički fakultet

Zorana Gajić 400/2016
David Dimić 137/2015

Beograd 2018.

Sadržaj

1 Uvod **2**

2 Prikupljanje podataka **2**

3 Ekstrakcija i čišćenje podataka **5**
3.1 Opis podataka . . . 5
3.2 Vizualna reprezentacija polaznih podataka . . . 6
3.2.1 Nedostajuće vrednosti . . . 6
3.2.2 Geografska lokacija incidenta . . . 8
3.3 Priprema podataka . . . 11
3.3.1 Uvod . . . 11
3.3.2 Čišćenje podataka . . . 11
3.3.3 Prikaz . . . 12

4 Analitička obrada i algoritmi **14**
4.1 Pravila pridruživanja . . . 14
4.2 Klasterovanje . . . 16
4.2.1 K-sredina . . . 16
4.2.2 Hijerarhijsko klasterovanje . . . 17
4.2.3 DBSCAN . . . 17
4.3 Klasifikacija . . . 18
4.3.1 Stabla odlučivanja . . . 18
4.3.2 Bajesov algoritam . . . 20
4.3.3 K-najbližih suseda . . . 20
4.3.4 Neuronska mreža . . . 21
4.3.5 SVM . . . 21
4.4 Predikcija klasifikacijom KNN . . . 22

1 Uvod

Istraživanje podataka sastoji iz tri faze:

- prikupljanje podataka
- ekstrakcija i čišćenje podataka
- analitička obrada i algoritmi

Podaci predstavljaju izveštaje o nesrećama i ubistvima dece u gradovima Amerike.

2 Prikupljanje podataka

Podaci su prikupljani sa: http://www.gunviolencearchive.org/reports/ i to:

- Children Killed
- Children Injured
- Accidental Deaths (Children Ages 0-11)
- Accidental Injuries (Children Ages 0-11)

Ponuđeni .csv fajl sadrži premalo atributa, u svakom fajlu po 6 (Incident Date, State, City Or County, Address, Killed, Injured). Dok kada se klikne na View Incident dobije se mnogo više korisnih podataka za taj konkretan incident.

Naš cilj je prvo prikupiti te dodatne podatke i od njih formirati nove .csv fajlove sa nešto pogodnijim atributima za rad, a potom te novoprikupljene dalje obrađivati.
Za to je bilo korisno nekoliko linux alatki iz terminala kao i programski jezik **Python3**.
Za svaku kolonu sa sajta treba skinuti html stranicu do koje se dolazi klikom na View Incident. Pošto u svaki fajl ima oko 500 redova to smo ostvarili rekurzivnim obilaskom sajta pomoću alatke **wget**:

```
wget -np -r -R "index.html*" -l 1 "http://www.gunviolencearchive.org/children-killed"
```

Potrebno je takođe postaviti dubinu rekurzije na 1 (flag -l 1) da se ne bi, ulazeći u druge linkove, krenulo u obilazak celog sajta. Potom je to isto potrebno uraditi za sve stranice, a da se ne bi radilo ručno napravljen je sledeći jednostavan python skript:

```
1 import os
2
3 for i in range(2,18):
4     os.system("wget -np -r -R \"index.html*\" -l 1  \"http://www.
          gunviolencearchive.org/children-killed?page=" + str(i) + "\"")
```

Pošto imamo prikupljene potrebne html strane možemo pristupiti parsiranju html-a i izdvajanju željenih atributa. Korišćeni su regularni izrazi (modul **re**) u pythonu.

Jedan primer dela html strane *Incident* sa koje je potrebno izdvojiti željene podatke:

```
Location

    June 12, 2016
    first block of James Avenue
    Littlestown, Pennsylvania
    Geolocation: 39.7448, -77.092

Participants

    -Type: Victim
    -Name: Codie Powell
    -Age: 9
    -Age Group: Child 0-11
    -Gender: Female
    -Status: Killed

    -Type: Victim
    -Name: Talisaha Powell
    -Age Group: Adult 18+
    -Gender: Female
    -Status: Injured

    -Type: Subject-Suspect
    -Relationship: Family
    -Name: Donald Powell
    -Age: 37
    -Age Group: Adult 18+
    -Gender: Male
    -Status: Killed
```

Odgovarajućim regularnim izrazima možemo izdvojiti željene podatke, ali treba ima na umu da redosled podataka nije uniforman na svim stranicama. Na primer, uočeno je da u nekim redosled *Gender* pa *Status*, dok je u drugim njihov redosled zamenjen.
Takođe, u novim tabelama biće više redova nego u originalnim jer dodajemo redove za svakog od učesnika, kojih može biti više za jedan isti događaj. Baš zbog te osobine dodat je novi atribut: *ID* koji je isti za iste događaje (incidente).

Ilustrujmo male delove python koda koji od odeljka *Participants* formira atribute: Type, Name, Age, Age Group, Gender i Status.

Analizom html-a uočavamo grupaciju koja se odnosi na učesnike i formiramo odgovarajuće regularne izlaze, a potom podatke tražimo unutar tog bloka izdvajanjem cele liste (između tagova $< ul >$. Jedna lista odgovara podacima o jednoj osobi), dok unutar njega grupišemo elemente liste (između tagova $< li >$) i to sve levo i desno od znaka dvotačke [1]:

[1] Poslednji regex je napravljen posebno pošto se koristi na više mesta u kodu

```
#ucesnici
re_participants = re.compile(r"""
     <\s*h2\s*>\s*Participants\s*<\s*\/h2\s*>(.|\s)*?</div
     """, re.VERBOSE)

#ucesnici, suzen izbor
re_regex1 = re.compile(r"""
    <\s*ul\s*>(.|\s)*?<\s*\/ul\s*>
    """, re.VERBOSE)

#type:value
re_regex_type_value = re.compile(r"""
    <li\s*>(?P<type>(.|\s)*?):(?P<value>(.|\s)*?)<
    """, re.VERBOSE)
```

Dalje se ovako definisani regularni izrazi koriste u kodu:

```
#svi ucesnici
#siri kontekst - trazenje dela sa Participants
x = re_participants.search(html_file)
if x != None:
    participants = x.group()

    #uzi kontekst - trazenje pojedinacnih ucesnika
    for z in re_regex1.finditer(participants):
        subject = z.group()

        #pretpostavimo da sve od navedenog za svaku osobu
        #moze da nedostaje
        #ako se dokaze suprotno dodelicemo nadjenu vrednost
        Type = "N/A"
        Name = "N/A"
        Age = "N/A"
        Age_group = "N/A"
        Gender = "N/A"
        Status = "N/A"

        #za svakog ucesnika trazimo atribute
        #type, name, age, age group, gender i status
        for y in re_regex_type_value.finditer(subject):
            key = y.group('type').strip()
            value = y.group('value').strip()

            if key == 'Type':
                Type = value
            elif key == 'Name':
                Name = value
            elif key == 'Age':
                Age = value
            elif key == 'Age Group':
                Age_group = value
            elif key == 'Gender':
                Gender = value
            elif key == 'Status':
                Status = value

         #upisivanje ucesnika u izlaznu datoteku...
```

Ovim smo obezbedili da može imati više osoba i da redosled njihov atributa nije bitan. Slučaj nedostajućih vrednosti takođe je pokriven. Na sličan način izdvoje se i ostali potrebni podaci, a sve to izvršava se u obilasku prosleđenog direktorijuma u kojem se nalaze sve potrebne html stranice, što je moguće izvesti sa os.walk(path) funkcijom.

3 Ekstrakcija i čišćenje podataka

3.1 Opis podataka

Više redova u tabeli predstavlja informacije o jednom događaju, s obzirom da u jednom događaju može učestvovati više ljudi, odnosno žrtve i osumnjičeni.
Na raspolaganju su nam atributi sa tabele:

Podaci			
Atribut	Tip atributa	Moguće vrednosti	Nedostajuće vrednosti
ID	Kvantitativni	0-100000	NE
Incident Date	Kategorički	Datum oblika: Month Date, Year	NE
State	Kategorički		NE
City	Kategorički		NE
Geolocation	Kvantitativni	Koordinate oblika: double, double	DA
Guns Involved	Kvantitativni	1-4	DA
Gun Types	Kategorički	tipovi oružja	DA
Guns Stolen	Kategorički, binarni	Not-stolen, Stolen	DA
Type	Kategorički, binarni	Victim, Subject - Suspect	NE
Name	Kategorički	First Name, Last Name	DA
Age	Kvantitativni	1-82	DA
Age Group	Kategorički, binarni	Child 0-11, Adult 18+	DA
Gender	Kategorički, binarni	Male, Female	DA
Status	Kategorički	Arrested, Injured, Injured Arested, Killed, Unharmed, Unharmed Arested	DA

Tabela 1: Atributi i njihove karakteristike

3.2 Vizualna reprezentacija polaznih podataka

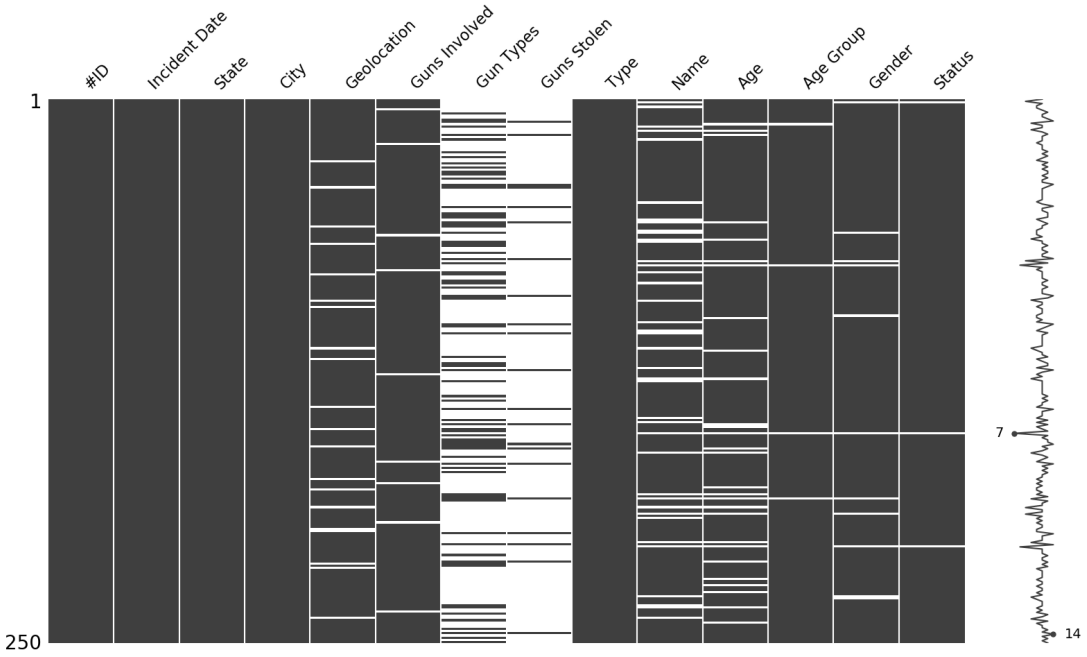
Da bismo stekli bolji osećaj i uvid u to sa kakvim podacima radimo vizualno smo prestavili samo neke interesantne njihove aspekte pre ikakve obrade. Ovim može da se stekne bolja slika kojim putem krenuti u obradu i koje algoritme primeniti.

3.2.1 Nedostajuće vrednosti

Koristeći python modul **missingno** reprezentovali smo nedostajuće vrednosti nad slučajnim uzorkom od 250 redova [2] za svaku od četiri tabele.

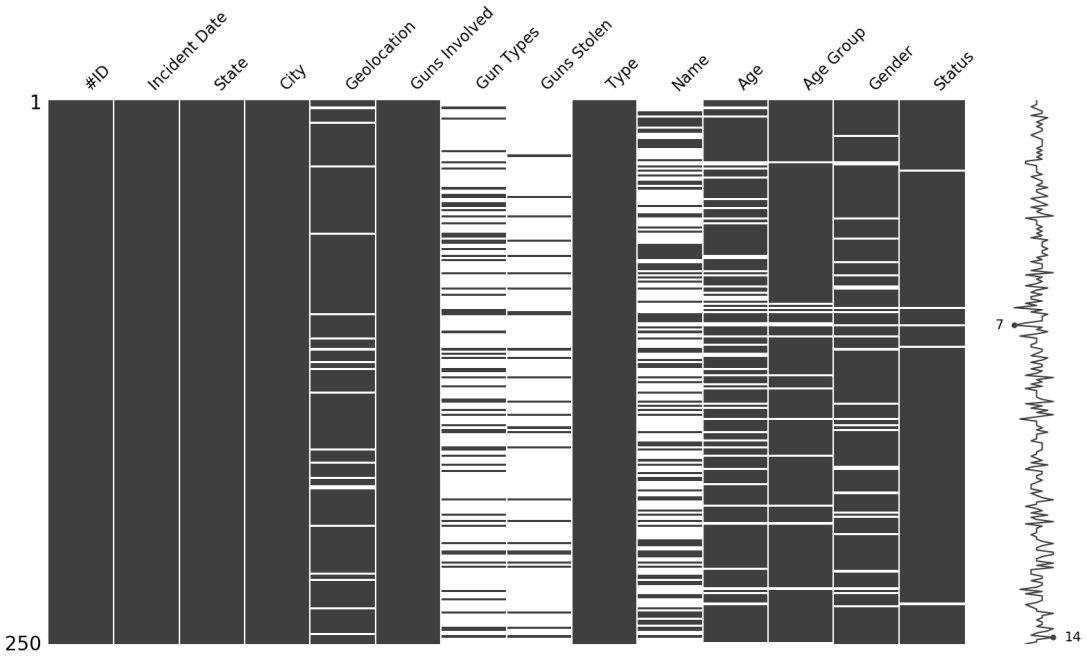
```
import sys
import pandas
import missingno
import matplotlib.pyplot as plt

args = sys.argv
if len(args) != 2:
    sys.exit(f"usage: ./{args[0]} /path/to/.csv")

#ucitavanje iz prosledjenog csv fajla
df = pandas.read_csv(args[1])
#plotovanje missingo rezultata
plt.show(missingno.matrix(df.sample(250)))
```

Dobijamo sledeće grafikone na kojima se jasno vidi kakva nam je raspodela nedostajućih vrednosti:

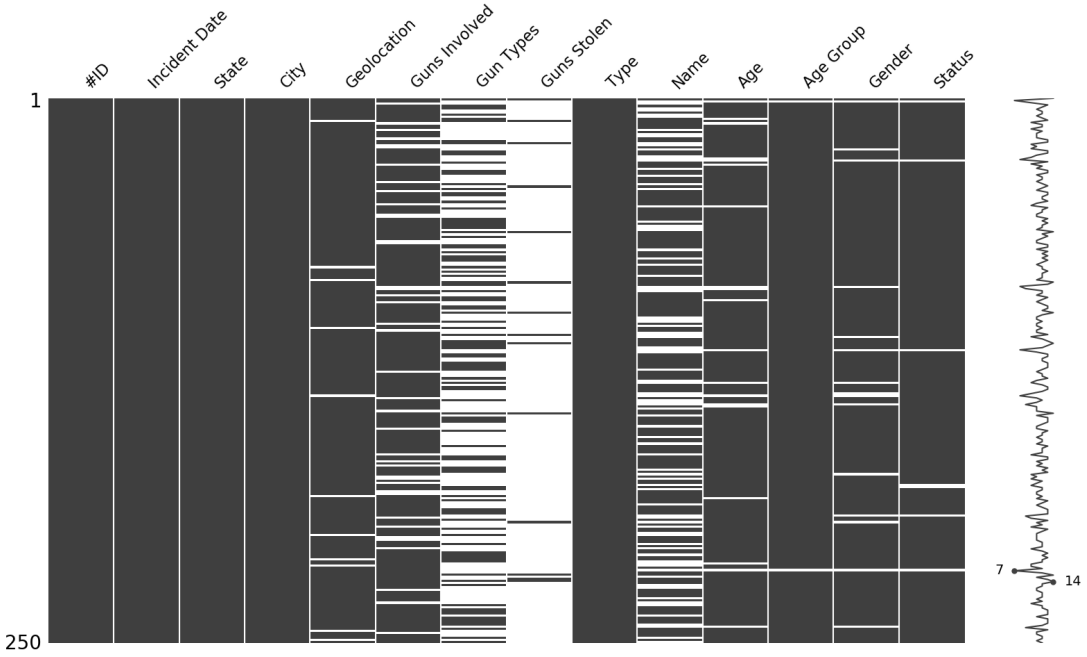


Slika 1: Children Killed Missing Values

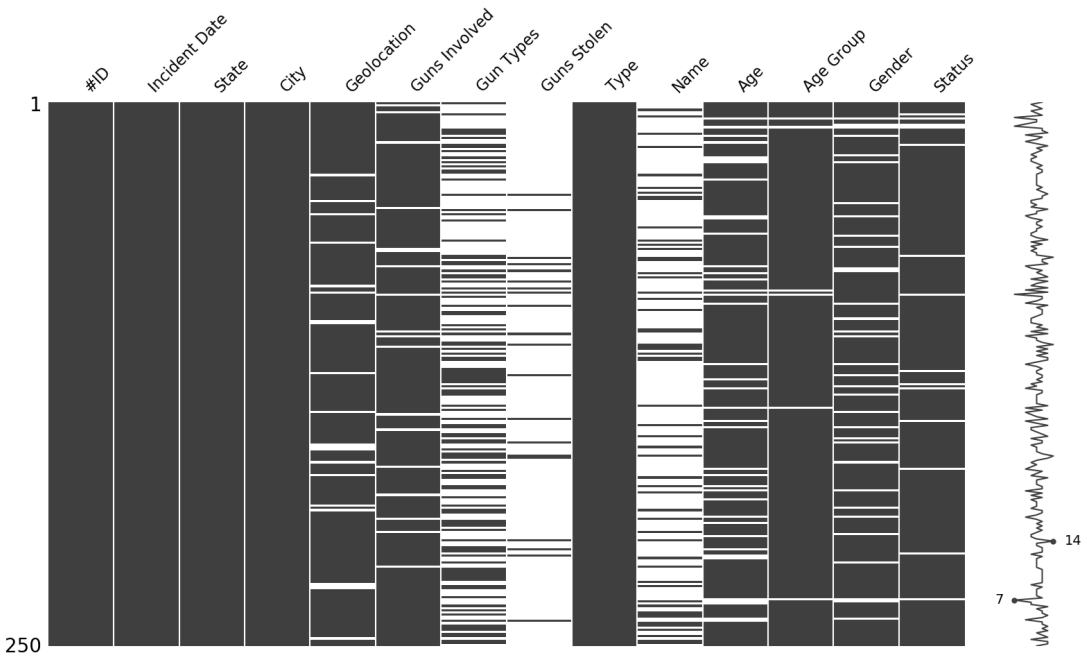
[2] Preporuka korišćenog modula



Slika 2: Children Injured Missing Values



Slika 3: Accidental Deaths Missing Values



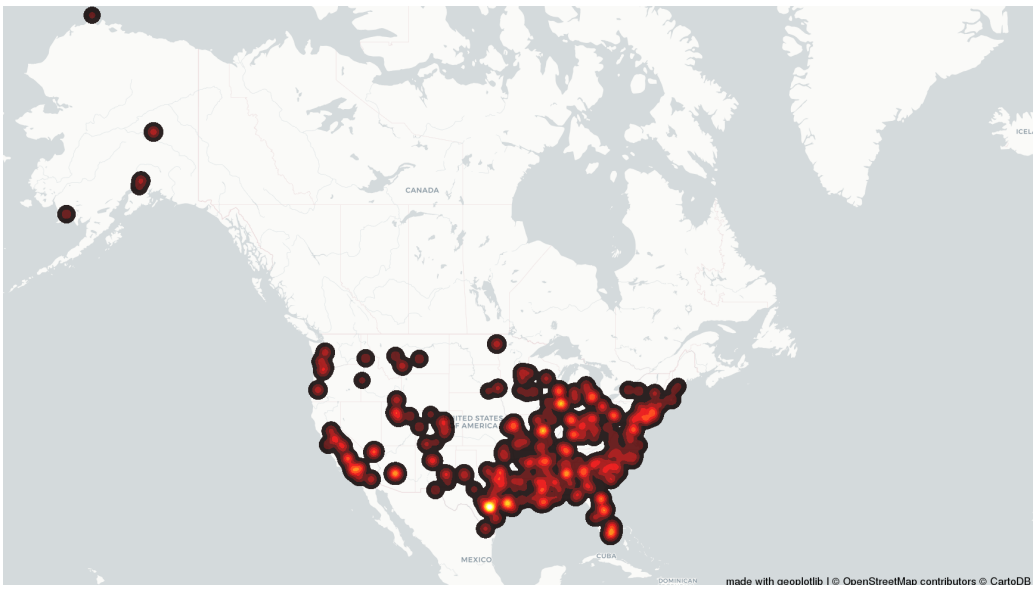
Slika 4: Accidental Injuries Missing Values

Praznine označavaju nedostajuće vrednosti. Kao što smo i očekivali, neki atributi nemaju nedostajućih vrednosti kao što je naznačeno u tabeli 3.1, ali sada, za ostale vidimo koji imaju manje, a koji više.

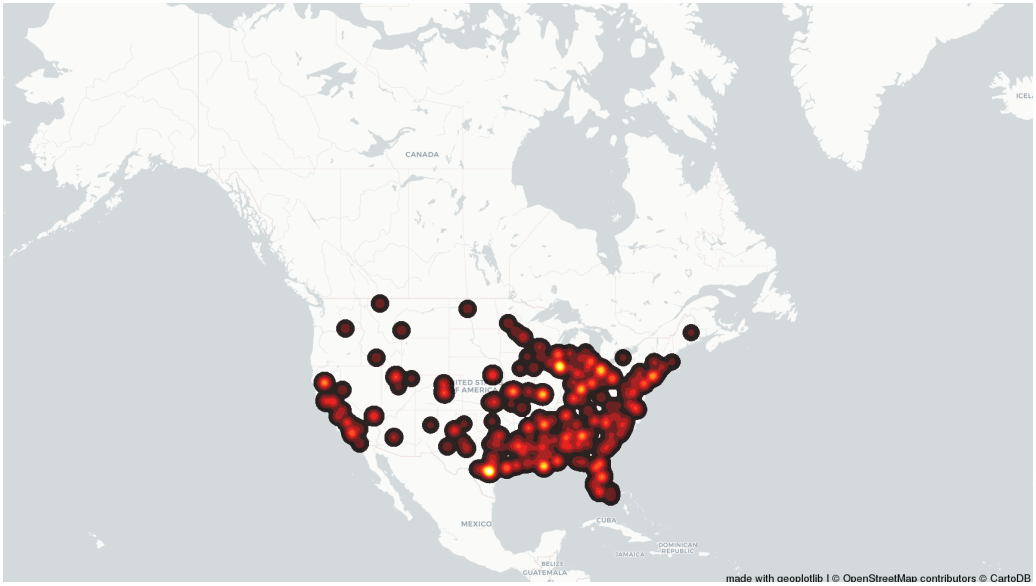
3.2.2 Geografska lokacija incidenta

U podacima su nam obezbeđene i geografska dužina i širina, pa bi iz njihovog vizualnog prikaza moglo da se vidi učestalost i raspoređenost na mapi. Te mogućnosti pruža modul **geoplotlib** sa funkcijom *kde* za prikaz mape gustine, ograničeno na teritoriju USA gde su se dogodili incidenti:

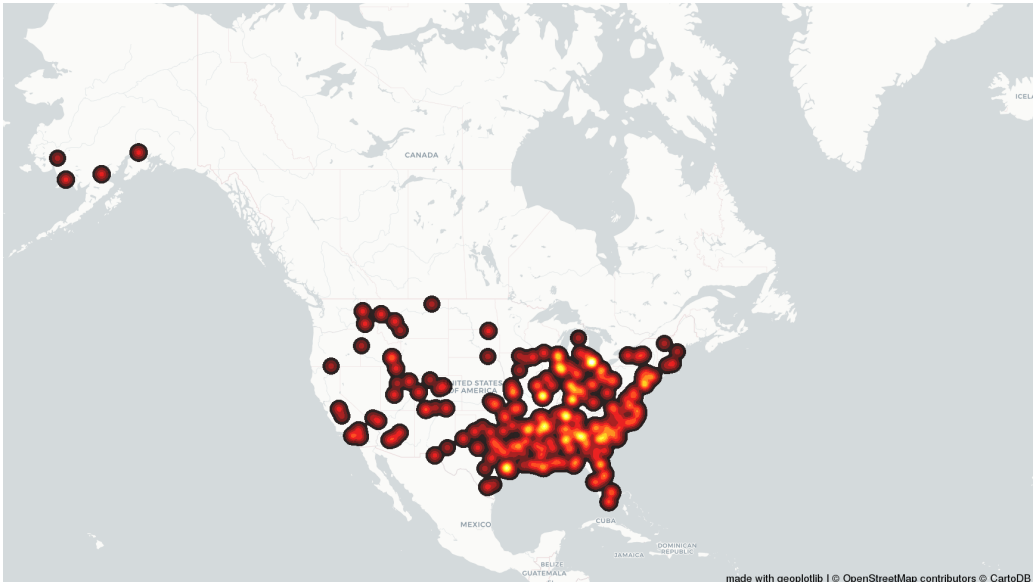
```
import geoplotlib
from geoplotlib.utils import read_csv, BoundingBox
import sys

args = sys.argv
if len(args) != 2:
    sys.exit(f"usage: ./{args[0]} /path/to/.csv")

data = read_csv(args[1])
geoplotlib.kde(data, 5, cut_below=1e-4, show_colorbar=False)
geoplotlib.set_bbox(BoundingBox.USA)
geoplotlib.show()
```



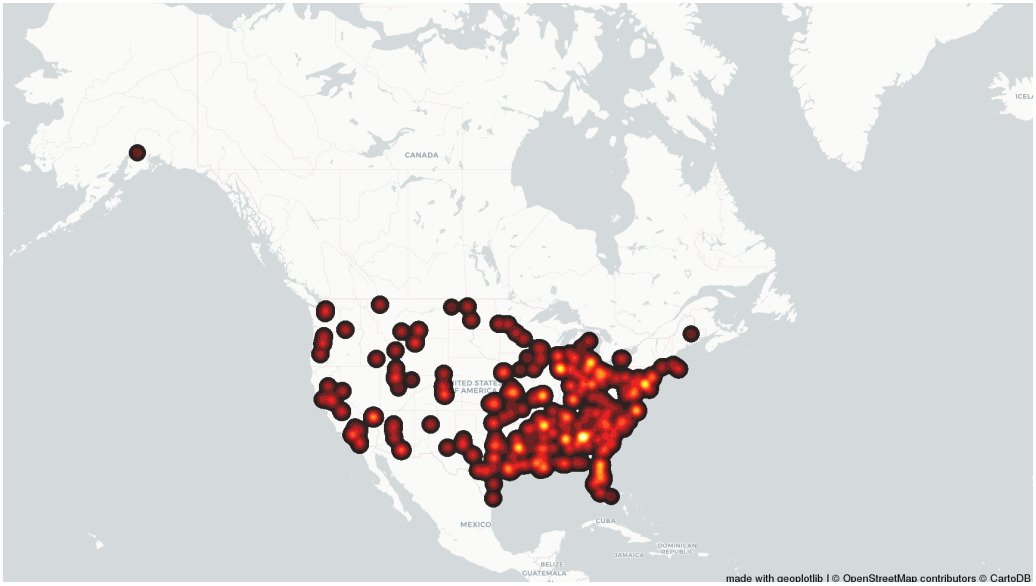
Slika 5: Children Killed Geolocation



Slika 6: Children Injured Geolocation



Slika 7: Accidental Deaths Geolocation



Slika 8: Accidental Injuries Geolocation

Također, KNIME alatom možemo dobiti u kojoj državi i u kom gradu imamo najviše ubistava, što potvrđuje prethodni prikaz na mapi. [3]

Row ID	State	Victim ...
Row41	Texas	102
Row9	Florida	57
Row4	California	55
Row34	Ohio	50
Row25	Missouri	47
Row37	Pennsylv...	39
Row10	Georgia	35
Row13	Illinois	32
Row18	Louisiana	31
Row32	North Car...	31

Slika 9: Broj ubijenih po državama

Row ID	City	Victim ...
Row78	Chicago	19
Row194	Houston	17
Row359	Saint Louis	17
Row92	Columbus	14
Row404	Sutherla...	14
Row211	Kansas City	13
Row266	Memphis	13
Row274	Milwaukee	13
Row368	San Anto...	13
Row245	Louisville	12

Slika 10: Broj ubijenih po gradovima

[3] Za nezgode pogledati originalni KNWF: children_analysis

3.3 Priprema podataka

3.3.1 Uvod

U fazi pripreme podataka smo kreirali 4 odvojene datoteke koje koristimo u fazi obrađivanja.
Prvi korak: grupisanje podataka po accidental deaths and injuries i killed and injured, zbog potencijalne analize između te dve grupe i unutrašnjih podgrupa.
Podatke u ove dve grupe smo grupisali po ID, s obzirom na to da nam je svaki događaj, odnosno svaka vrsta u tabeli imala svoj ID.
Kao što vidimo iz tabele 3.1 podaci nisu obrađeni, s obzirom na to da imamo nedostajućih vrednosti u više atributa.

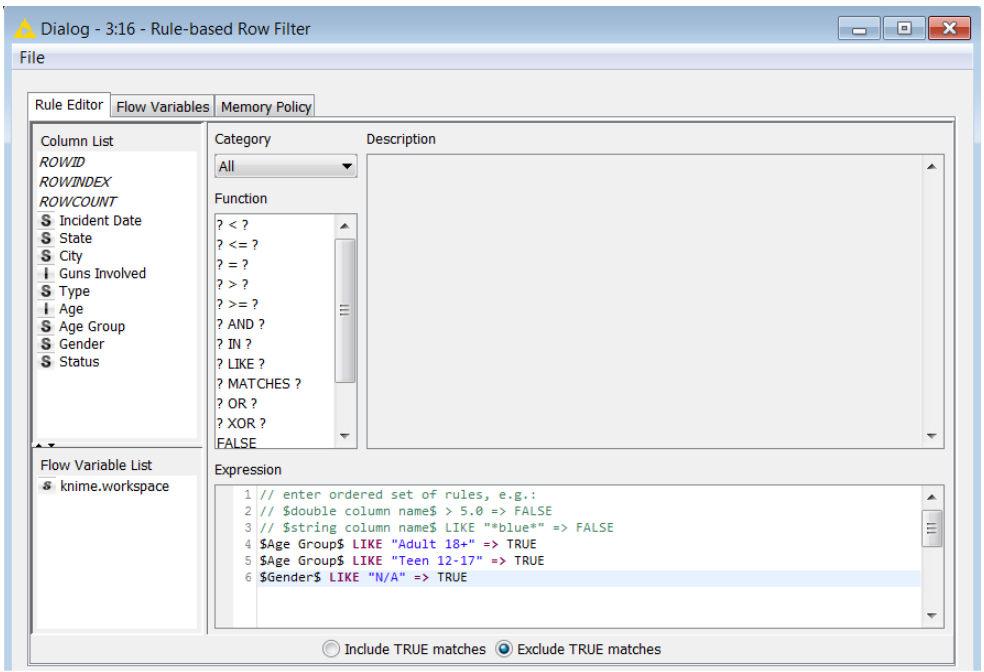
3.3.2 Čišćenje podataka

Format u kojem smo dobili podatke (children_killed.csv i children_injured.csv) nije prilagođen za rad. Naime, imamo problem sa nedostajućim vrednostima i normalizacijom.

S obzirom na to da radimo sa podacima koji nisu trivijalni, kao što je na primer skup tačaka, gde možemo proceniti nedostajuće vrednosti, ovde smo se ipak odlučili za varijantu eliminisanja problematičnih instanci.

Imamo nedostajuće vrednosti u sledećim atributima: Guns Involved, Age. S obzirom da takvih instanci nema puno, eliminisali smo ih.
Takođe imamo instance koje za atribute Gun Types, Gun Stolen i Gender imaju vrednosti N/A. Za atribut Gender ćemo takve instance isto ukloniti, jer ih ima malo, međutim kod Gun Types i Gun Stolen atributa primećujemo da su skoro sve instance sa vrednostima N/A. Tako nešto nam neće pomoći, zato ćemo ukloniti te atribute kompletno.

Naš zadatak se sastoji u analizi smrti i povreda, konkretno dece, odnosno Age group - Children 0-11, ali mi u našim podacima imamo i više od toga. Zato smo čvorom Rule-based Row Filter zadali pravilo prikazano na slici 3.3.2



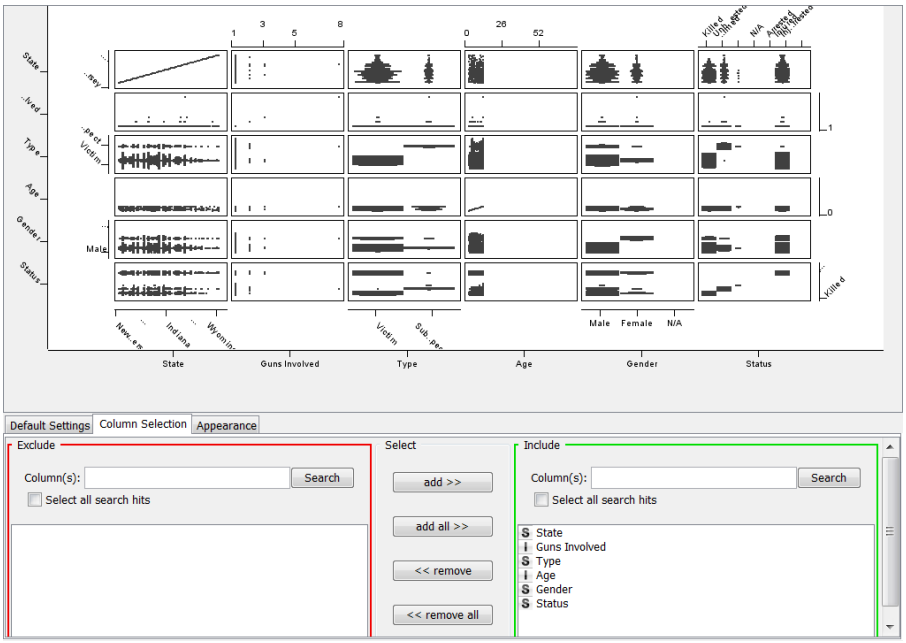
Slika 11: Filtriranje podataka

Time smo se oslobodili nepotrebnih instanci. Sada kada smo to uradili, možemo ukloniti i atribut Age group.

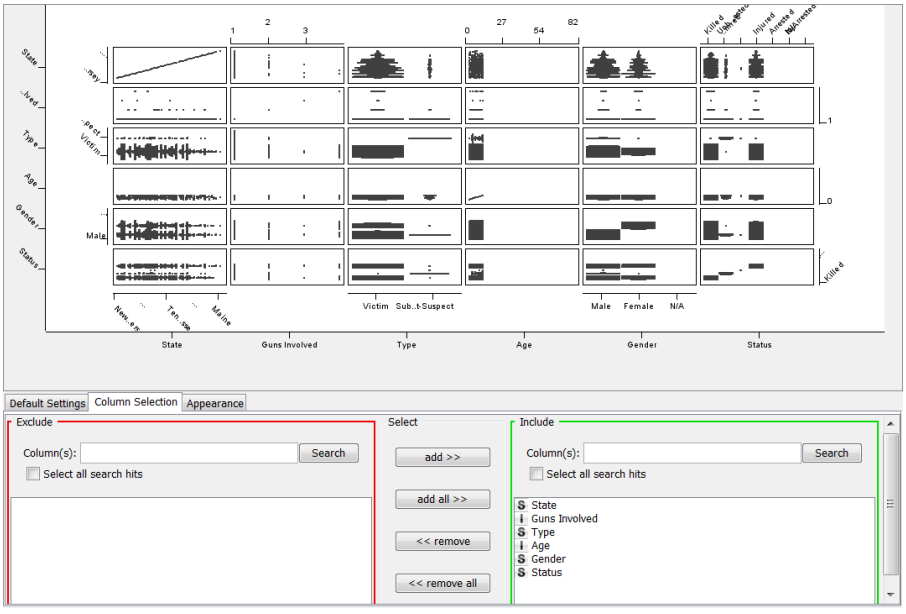
Kada izvršimo istu pripremu nad podacima accidental_deaths.csv i accidental_injuries.csv vidimo da nam atribut Guns Involved za sve instance ima vrednost 1, što nije slučaj od pre.

3.3.3 Prikaz

Nakon sređivanja podataka u *Knime-u* možemo prikazati dobijene rezultate u formi *Scatter Matrix* i *Pie chart*, kao i statistički prikaz podataka sa *Box Plot* i *Statistics*.

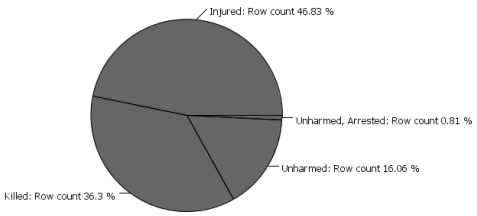


Slika 12: Accidental Deaths and Injuries Scatter Matrix

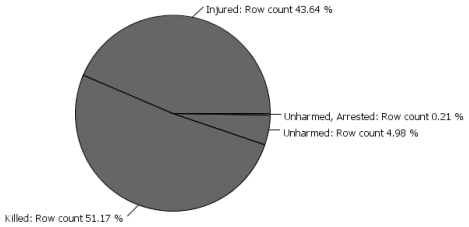


Slika 13: Children Killed and Injured Scatter Matrix

Sa priloženih grafikona uočavamo sličnost procenata povređene dece za obe grupacije, što je oko 43%, dok možemo primetiti znatnu razliku u procentima ubijene dece za ove grupacije. Razlikuju se do na 15%. Interesantno je primetiti da je mnogo više slučajnih smrti nego ubistava. Međutim možemo zaključiti da je odnos uhapšenih i neuhapšenih počinitelja znatno različit i to 0.21%, 0.81%, respektivno. Što znači da su počinitelji zločina i privedini.

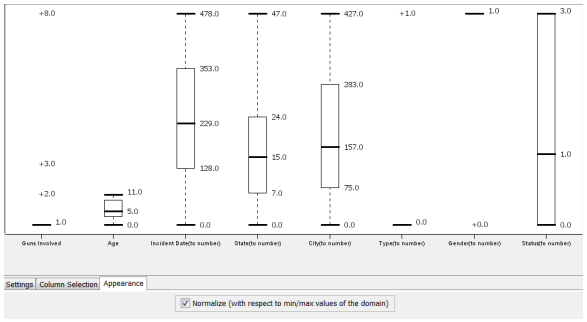


Slika 14: Accidental Deaths and Injuries Pie Chart

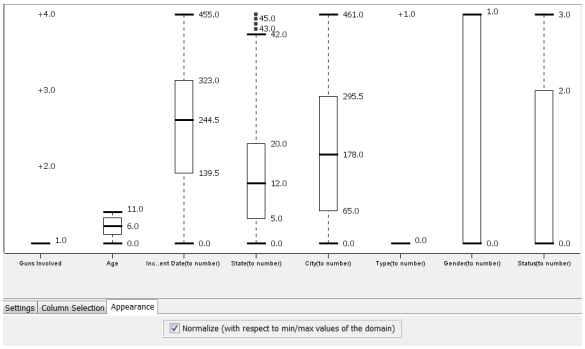


Slika 15: Children Killed and Injured Pie Chart

Pogledajmo sada sledeće prikaze *Box Plot*. Na slici 16 imamo elemente van granica u atributu *State*. Ali ovo je i očekivano jer smo u geografskom prikazu lokacije incidenata početnih podataka [4] videli izmeštenost nekoliko instanci na području Aljaske, dok u skupu *Accidental Deaths and Injuries* nemamo elemenata van granica.



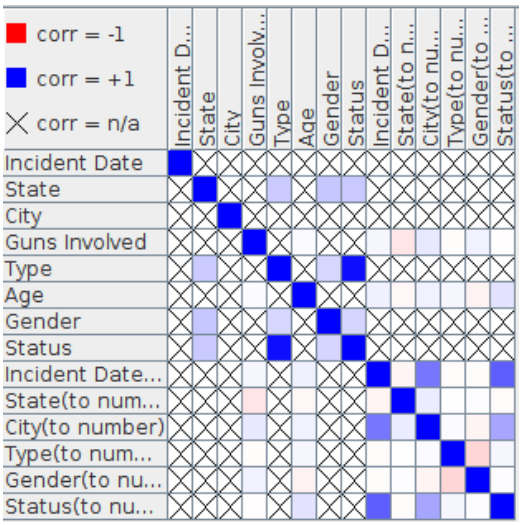
Slika 16: Accidental Deaths and Injuries Box Plot



Slika 17: Children Killed and Injured Box Plot

[4] Pogledati geografski prikaz na slikama: 3.2.2, 5, 3.2.2 i 7

Takođe možemo posmatrati linearnu korelaciju izmedju atributa, i ono što možemo primetiti da je za obe grupe koje analiziramo uočena korelacija izmedju atributa State i Type, što znači da bismo mogli eliminisati jedan atribut, ali jednostavnije je da to u ovom trenutku ne radimo.



Slika 18: Korelacija za Children Killed and Injured

4 Analitička obrada i algoritmi

4.1 Pravila pridruživanja

Prvi primenjen algoritam je Apriori algoritam, koji spada u algoritam za pravila pridruživanja. Cilj nam je da vidimo imamo li nekih interesantnih pravila. Mera interesantnosti za naše podatke može biti Lift mera.

Prvi način: korišćenje KNIME cvora *Apriori(3.7)*, sa sledećim podešavanjima:
minimalna podrska = 0.2
metricType = Confidence od mogućih Lift, Conviction.

Zapravo, menjanjem ovih vrednosti (povećavanjem/smanjenjem minimalne podrške...) nemamo interesantnije podatke. Dobijen je sledeći izlaz:

```
Apriori
=======

Minimum support: 0.2 (189 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 16

Generated sets of large itemsets:

Size of set of large itemsets L(1): 5

Size of set of large itemsets L(2): 6

Size of set of large itemsets L(3): 2

Best rules found:

 1. Gender=Male Status=Killed 304 ==> Type=Victim 303    <conf:(1)> lift:(1.05) lev:(0.02) [15] conv:(8.37)
 2. Status=Killed 483 ==> Type=Victim 481    <conf:(1)> lift:(1.05) lev:(0.03) [24] conv:(8.87)
 3. Status=Injured 412 ==> Type=Victim 410    <conf:(1)> lift:(1.05) lev:(0.02) [20] conv:(7.56)
 4. Gender=Female 339 ==> Type=Victim 337    <conf:(0.99)> lift:(1.05) lev:(0.02) [16] conv:(6.22)
 5. Gender=Male Status=Injured 253 ==> Type=Victim 251    <conf:(0.99)> lift:(1.05) lev:(0.01) [11] conv:(4.65)
 6. Gender=Male 605 ==> Type=Victim 555    <conf:(0.92)> lift:(0.97) lev:(-0.02) [-16] conv:(0.65)
```

Slika 19: Children Killed and Injured Apriori

Kako konkretno izgleda čvor ne možemo prikazati jer ne postoji taj čvor na novijoj verziji KNIME alata, već samo na virtualnoj mašini pri staroj verziji.

Drugi način: Ručno kreiranje čestih skupova stavki i pravila pridruživanja uz pomoć čvora *Association Rule Learner*, iz kojeg možemo zaključiti sledeće:

Broj čestih skupova = 23
Najmanja podrška = 0.314
Najveća podrška = 0.962

Row ID	Support(0-1):	Items
item set 0	0.962	[1]
item set 12	0.945	[Victim]
item set 1	0.91	[1,Victim]
item set 18	0.641	[Male]
item set 7	0.618	[1,Male]
item set 13	0.588	[Male,Victim]
item set 2	0.568	[1,Male,Victim]
item set 20	0.512	[Killed]
item set 15	0.51	[Killed,Victim]
item set 9	0.493	[1,Killed]
item set 4	0.49	[1,Killed,Victim]

Slika 20: Česti skupovi stavki za Children Killed

Broj pravila pridruživanja = 54
Najmanja podrška = 0.253
Najveća podrška = 0.91

Row ID	Support	Confidence	Lift	Consequent	implies	Items
rule40	0.91	0.963	1.001	1	<---	[Victim]
rule41	0.91	0.946	1.001	Victim	<---	[1]
rule38	0.618	0.964	1.002	1	<---	[Male]
rule39	0.618	0.642	1.002	Male	<---	[1]
rule36	0.588	0.917	0.971	Victim	<---	[Male]
rule37	0.588	0.622	0.971	Male	<---	[Victim]
rule33	0.568	0.966	1.004	1	<---	[Male,Victim]
rule34	0.568	0.919	0.973	Victim	<---	[1,Male]
rule35	0.568	0.624	0.974	Male	<---	[1,Victim]
rule31	0.51	0.996	1.054	Victim	<---	[Killed]
rule32	0.51	0.539	1.054	Killed	<---	[Victim]
rule29	0.493	0.963	1.001	1	<---	[Killed]
rule30	0.493	0.512	1.001	Killed	<---	[1]
rule27	0.49	0.996	1.054	Victim	<---	[1,Killed]
rule26	0.49	0.963	1.001	1	<---	[Killed,Victim]

Slika 21: Pravila pridruživanja za Children Killed and Injured

Neka od pravila su:
Type Victim => *Guns Involved* 1 sa podrškom *0.91* i lift merom 1.001.
Gender Male => *Guns Involved* 1 sa podrškom *0.618* i lift merom 1.002.

Vidimo da je Lift mera oko 1 i pravila su krajnje trivijalna, naime ako je neko žrtva logično je da je u ubistvu učestvovalo bar jedno oružje. Iz ovoga možemo zaključiti da dobijena pravila nisu zanimljiva.

Slični rezultati su dobijeni primenom algoritma nad podacima *accidental_deaths* i *accidental_injuries*. Dobijeno je manje čestih skupova i pravila pridruživanja, takodje je najveća podrška manja od malopre.

4.2 Klasterovanje

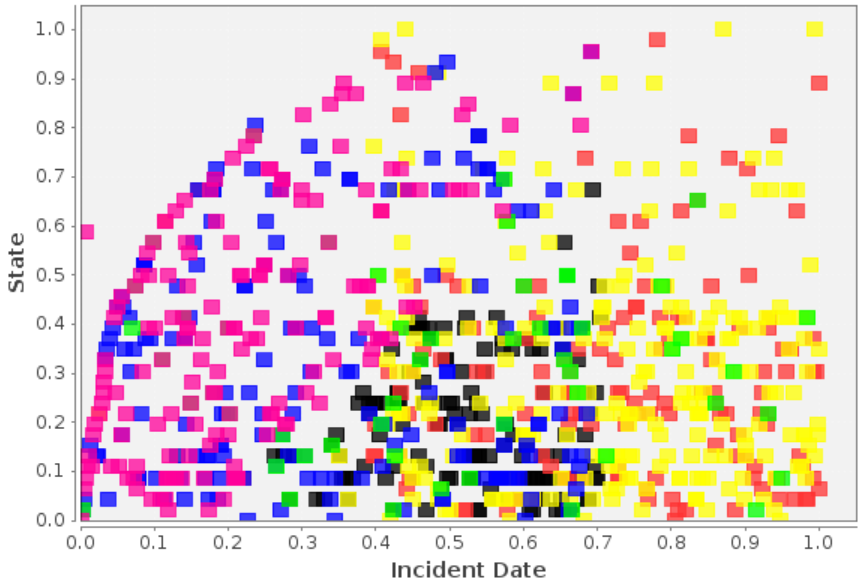
Klasterovanje spada u tehniku prepoznavanja klastera bez poznavanja izlaza. Zato ćemo videti, da se klasifikacija ponaša mnogo bolje od klasterovanja.
Želimo da nam ciljni atribut bude atribut Status. Odnosno, želimo da na osnovu svih podataka zaključimo da li je osoba ubijena, povređena, uhapšena itd.

Zaključak koji smo doneli na osnovu sledeća tri algoritma je da ni u jednom nismo uspeli da klasterujemo podatke i dobijemo očekivani status.

4.2.1 K-sredina

Ova tehinka klasterovanja se zasniva na centru i particionisanju, koja pokušava da pronadje korisnički definisan broj klastera. S obzirom da je najveća mana ovog algoritma koji broj klastera uzeti, mi ćemo napraviti petlju gde ćemo taj parametar klastera, odnosno k, prošetati kroz moguće vrednosti od 2 do naprimer 7 i videti šta time dobijamo preko Scatter-Plot-a.

S obzirom da K-sredina radi samo sa numeričkim podacima, potrebno je konvertovati kategoričke podatke i normalizovati.

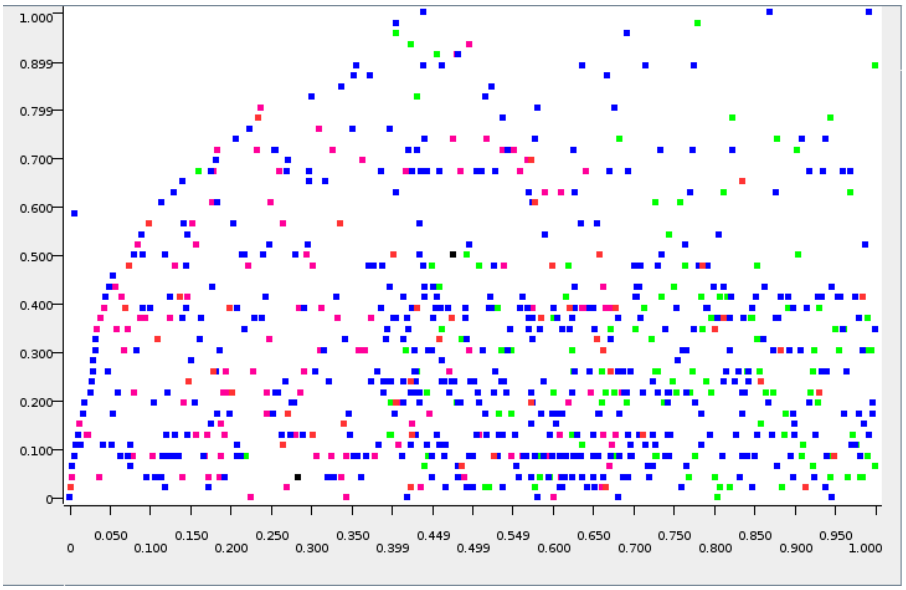


Slika 22: K-sredina za broj klastera 6

Eksportovali smo rezultate klasterovanja u folder kMeans_childrenKI iz kojeg možemo zaključiti da se ništa nije klasterovalo kao što bismo očekivali, svi klasteri su izmešani za svako k (koje predstavlja broj klastera).

4.2.2 Hijerarhijsko klasterovanje

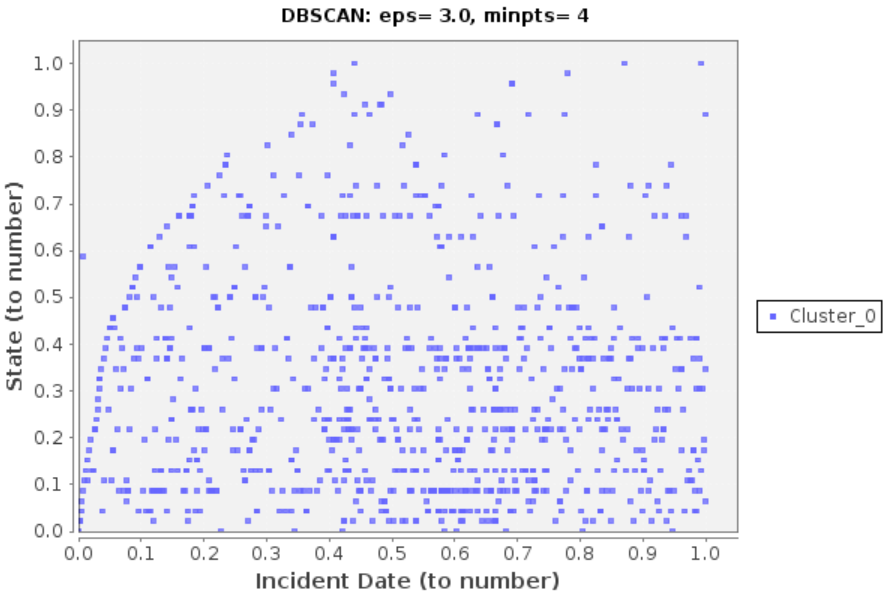
Hijerarhijskim klasterovanjem, kao i kod K najbližih suseda, nismo dobili dobro klasterovane podatke, kao što možemo videti u prilogu.



Slika 23: Hijerarhijsko klasterovanje za broj klastera 5

4.2.3 DBSCAN

Još jedan algoritam klasterovanja je DBSCAN klasterovanje. Stavili smo petlje, u nadi da će se pokazati bolje od ostalih algoritama klasterovanja, ali iz foldera dbscan_childrenKI u kojem su ekspportovane slike za različite brojeve epsilon i minpts, vidimo da je sve klasterovano u jedan klaster.
Korišćeni parametri: $epsilon \in [1..0.5..3]$ i $minpts \in [3, 5]$



Slika 24: DBSCAN

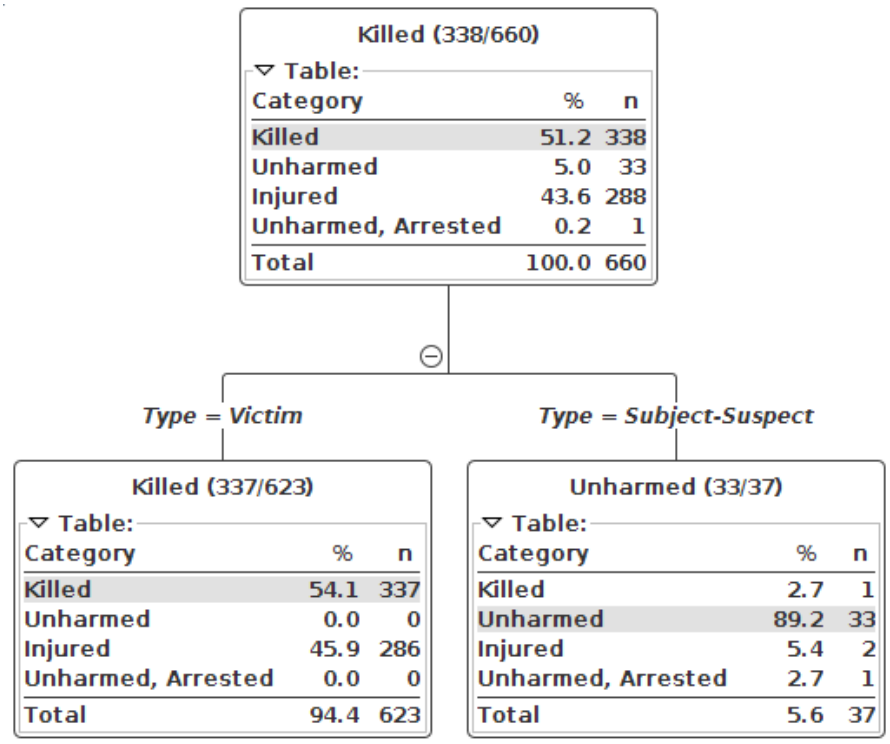
4.3 Klasifikacija

U svim algoritmima nam je i dalje cilj da se pogađa status, ali s obzirom da se radi o klasifikaciji, uzima se u obzir i željeni izlaz.
U svim algoritmima se prvo podaci moraju razdvojiti na trening (70%) i test skup, korišćenjem slojevitog uzorkovanja *(engl. stratified sampling)* pomoću KNIME čvora *Partitioning*.

Zaključak iz svih navedenih algoritama da su se najbolje pokazali algoritmi K najbližih suseda i Neuronske mreže sa preciznošću na test skupu čak 85%.

4.3.1 Stabla odlučivanja

Prvi algoritam klasifikacije koji smo primenili je stablo odlučivanja. Korišćena mera kvaliteta je Gini indeks.
Ono što možemo zaključiti je da je algoritam sveo stablo na dubinu 2 što je dobro, jer na osnovu jednog atributa (Type) možemo zakljuciti status. Naime, ako je osoba žrtva onda je ona i ubijena (54%), a ako je osumnjičen onda je i nepovređen (čak 89%).



Slika 25: Stablo odlučivanja

Ali, na osnovu preciznosti možemo reći da ovo baš nije merodavno klasifikovano, jer je preciznost na test skupu kao i na trening skupu oko 50%.

Row ID	AccuracyTrening	AccuracyTest	minRec
Overall_Overall#3	0.561	0.553	20
Overall_Overall#4	0.561	0.553	25
Overall_Overall#0	0.664	0.55	5
Overall_Overall#1	0.656	0.539	10
Overall_Overall#2	0.642	0.529	15

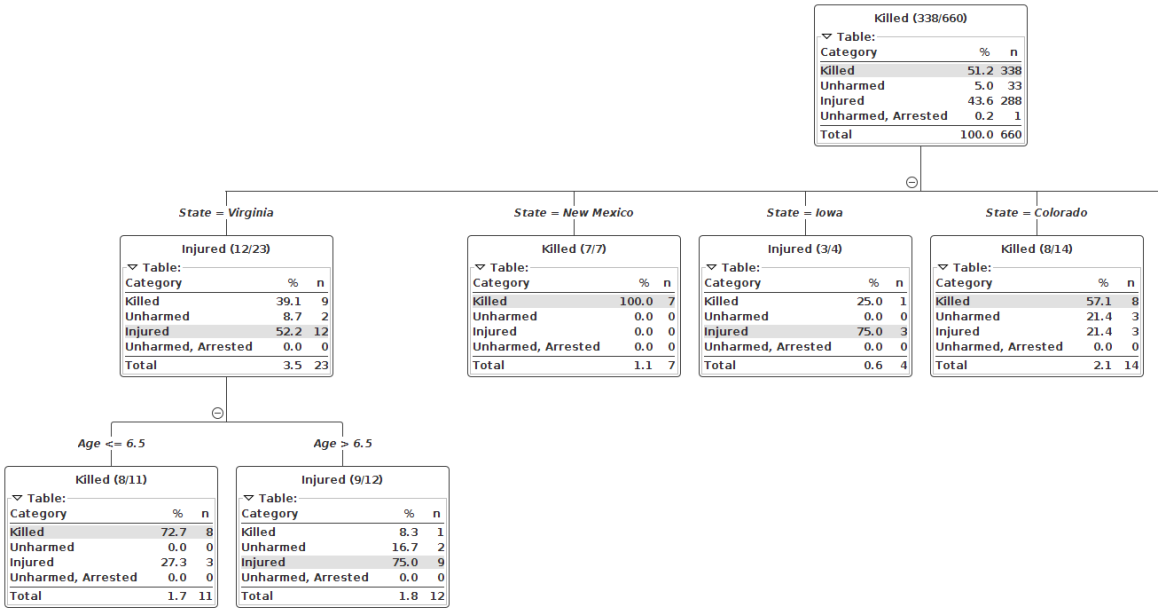
Slika 26: Preciznosti za test i trening skup za stablo odlučivanja

Na sledećoj slici imamo prikazanu matricu konfuzije za test skup. Iz nje možemo zaključiti da je atribut killed klasifikovan veoma dobro, dok je za atribut injured potpuno pogrešno klasifikovao. Svi koji treba da budu povređeni su klasifikovani kao ubijeni.

Row ID	Killed	Unhar...	Unhar...	Injured	Arrest...	N/A	Injure...
Killed	144	1	0	0	0	0	0
Unharmed	1	13	0	0	0	0	0
Unharmed...	0	1	0	0	0	0	0
Injured	124	0	0	0	0	0	0
Arrested	0	0	0	0	0	0	0
N/A	0	0	0	0	0	0	0
Injured, Ar...	0	0	0	0	0	0	0

Slika 27: Matrica konfuzije za test skup za stablo odlučivanja

Ako podesimo u algoritmu parametar da se stablo mora razviti po nekom atributu, na primer state, dobijamo sledeće stablo. Vidimo da nije mogao kao malo pre algoritam u jednom koraku da zaključi ciljni status, već se negde grana po atributu godine, i da preciznost nije bolja.



Slika 28: Stablo odlučivanja po atributu *State*

4.3.2 Bajesov algoritam

Bajesov algoritam radi sa verovatnoćama i vidimo da se pokazao malo bolje nego stabla odlučivanja. Korišćen parametar je podrazumevana verovatnoća (0.001) koja nam omogućava malo precizniju klasifikaciju.

Preko matrice konfuzije, vidimo da je lošije klasifikovan atribut killed u odnosu na stabla odlučivanja, ali je zato atribut injured bolje klasifikovan.

Row ID	Killed	Unhar...	Injured	Unhar...	Arrest...	N/A	Injure...
Killed	88	5	52	0	0	0	0
Unharmed	2	10	2	0	0	0	0
Injured	43	1	80	0	0	0	0
Unharmed...	0	1	0	0	0	0	0
Arrested	0	0	0	0	0	0	0
N/A	0	0	0	0	0	0	0
Injured, Ar...	0	0	0	0	0	0	0

Slika 29: Matrica konfuzije za test skup

Dobili smo da je preciznost na trening skupu 93% dok je na test skupu 62.7%.
Iako je preciznost na trening skupu velika, to nam ne garantuje za nove podatke (test skup) da se snađe kako treba jer se previše ugledao na ciljne atribute iz trening skupa.

4.3.3 K-najbližih suseda

Za ovaj algoritam smo iskoristili petlju po k, koje predstavlja broj najbližih suseda.
$k \in [1, 15]$
Vidimo da je za ovaj algoritam za sad preciznost najbolja, čak 85% u 6. iteraciji za k=9.

Row ID	Accuracy	k	Iteration
Overall#6	0.852	9	6
Overall#0	0.849	3	0
Overall#4	0.849	7	4
Overall#8	0.849	11	8
Overall#1	0.845	4	1
Overall#2	0.845	5	2
Overall#9	0.845	12	9
Overall#5	0.842	8	5
Overall#10	0.842	13	10
Overall#3	0.838	6	3
Overall#7	0.835	10	7
Overall#12	0.835	15	12
Overall#11	0.827	14	11

Slika 30: Preciznost za K najbližih suseda

U skladu sa velikom preciznošću vidimo na matrici konfuzije da su atributi i killed i injured ovog puta veoma dobro klasifikovani, a takodje i unharmed.

Row ID	Killed	Unhar...	Unhar...	Injured	Arrest...	N/A	Injure...
Killed	134	0	0	11	0	0	0
Unharmed	2	12	0	0	0	0	0
Unharmed...	0	1	0	0	0	0	0
Injured	33	0	0	91	0	0	0
Arrested	0	0	0	0	0	0	0
N/A	0	0	0	0	0	0	0
Injured, Ar...	0	0	0	0	0	0	0

Slika 31: Matrica konfuzije za K najbližih suseda

4.3.4 Neuronska mreža

Primenom ovog algoritma iskoristili smo petlju po broju skrivenih slojeva.
$num\_hidden\_layers \in [1, 5]$ Ono što možemo videti da je isto veoma dobra preciznost i na trening i na test skupu, oko 85%.

Row ID	AccuracyTrening	AccuracyTest	num_hidden_layers	Iteration
Overall_Ov...	0.945	0.884	3	2
Overall_Ov...	0.956	0.845	4	3
Overall_Ov...	0.976	0.835	5	4
Overall_Ov...	0.88	0.835	1	0
Overall_Ov...	0.906	0.806	2	1

Slika 32: Preciznost za neuronske mreže

Vidimo da je podjednako dobra klasifikacija kao i kod K najbližih suseda.

Row ID	Killed	Unhar...	Injured	Unhar...	Arrest...	N/A	Injure...
Killed	126	0	19	0	0	0	0
Unharmed	1	13	0	0	0	0	0
Injured	25	1	98	0	0	0	0
Unharmed...	0	1	0	0	0	0	0
Arrested	0	0	0	0	0	0	0
N/A	0	0	0	0	0	0	0
Injured, Ar...	0	0	0	0	0	0	0

Slika 33: Matrice konfuzije za neuronske mreže

4.3.5 SVM

SVM klasifikacija je zasnovana na ideji vektorskih prostora, gde želimo naći razdvajajuću hiper-ravan, tako da su svi podaci iz date klase na jednoj strani ravni.
Ovaj algoritam je zasnovan na binarnim podacima, ali iz svih ovih algoritama, videli smo da nam se podaci najčešce klasifikuju u killed ili injured.
U parametrima možemo da biramo različite kernele, odnosno

- Polynomial kernel
  Za power=1, bias=1, gamma=1 preciznost je 0.2% za trening i 0.4% za test skup.
- HyperTangent
  Preciznost ista kao i za polinomijalni kernel.
- RBF
  Za sigma=0.1 preciznost za trening skup je oko 50% i 60% za test skup.

Najbolje se pokazao RBF kernel.

Row ID	Killed	Unhar...	Unhar...	Injured	Arrest...	N/A	Injure...
Killed	140	0	0	5	0	0	0
Unharmed	2	1	0	11	0	0	0
Unharmed...	0	0	0	1	0	0	0
Injured	78	5	8	33	0	0	0
Arrested	0	0	0	0	0	0	0
N/A	0	0	0	0	0	0	0
Injured, Ar...	0	0	0	0	0	0	0

Slika 34: Matrice konfuzije za SVM koristeći RBF kernel

4.4 Predikcija klasifikacijom KNN

Interesantno bi bilo, koristeći algoritam K-najbližih suseda, izvršiti predikciju o statusu osobe na osnovu nekih datih podataka. Ilustrujmo takav primer u Python-u uz korišćenje modula **sklearn**.

`Prilog: knnClassifier.py`

Prvo treba učitati željeni csv fajl i izbaciti atribute koji nam nisu od značaja, kao *ID* ili *Name* koji su uvek različiti, a potom i nedostajuće vrednosti:

```
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report, confusion_matrix
import pandas as pd
import numpy as np
from sys import exit, argv

df = pd.read_csv(argv[1])
#izbacujemo neke nevazne atribute
df = df.drop(['ID', 'Name', 'Age Group', 'Incident Date', 'Geolocation', '
    Gun Types', 'Guns Stolen'], axis=1)
#izbacujemo instance sa nedostajucim vrednostima
df = df.replace("N/A", np.nan)
df = df.dropna()
#prikaz pocetnih podataka
print(df.head())
```

Treba izbaciti i *Status* jer on predstavlja ciljni atribut za koji vršimo predikciju. Dalje, sve kategoričke podatke predstavljene stringovima treba zameniti jedinstvenim brojem, što se lako može ostvariti preko mape $int \rightarrow string$ kao strukture podataka.[5] Pogledajmo na primeru za Status. Za ostale atribute je slično:

```
final_attr = 'Status'
#izbacujemo Status, njega pogadjamo
X = df.drop([final_attr], axis=1)
#ciljni atribut je Status
y = df[[final_attr]]

#sve menjamo jedinstvenim brojevima
status = df[final_attr].unique()
status_dict = dict(zip(status, range(len(status))))
y = y.replace(status_dict)

#invertovana mapa za status
inv_dict = {v: k for k, v in status_dict.items()}
```

[5] Treba za ciljni atribut napraviti i inverznu mapu da bismo od dobijenog (int) rezultata dobili koja je to (string) kategorija u našem slučaju

Sada možemo podeliti skup na trening (70%) i test (30%), pozvati algoritam za klasifikaciju primenom K-najbližih suseda sa podešenim parametrima i izvršiti predikciju za trening i test skup:

```
#podela na trening i test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3)

#algoritam k suseda
knn = KNeighborsClassifier(n_neighbors = 15, weights = 'distance')
knn.fit(X_train, y_train.values.ravel())

#prediktovane vrednosti
y_train_predicted = knn.predict(X_train)
y_test_predicted = knn.predict(X_test)
```

Sa `sklearn.metrics.classification_report(y_test, y_test_predicted)` vidimo kakva je preciznost, odziv, f1-mera i podrška na test ili trening skupu. Za podatke children_killed vidimo odličnu preciznost na trening skupu, blizu 100% i oko 60% na testu.
Matrice konfuzije dobijamo pozivom funkcije `sklearn.metrics.confusion_matrix(y_test, y_test_predicted)`.

Sada ćemo izvšiti predikciju ona osnovu: *State, City, Guns Involved, Type, Age, Gender* koje nam unosi korisnik, prilagoti ulaz funkciji predict i inverznom mapom ispisati kojoj kategoriji pripada uneta osoba:

```
in_data = []
describe = X.describe()

print("\nUnesi sledece podatke:")
for col in describe.columns:
    print('{:20} min: {:6}, max: {:6}'.format(col, describe[col]['min'],
        describe[col]['max']))

try:
    in_state = State_dict[input()]
    in_city = City_dict[input()]
    in_guns = int(input())
    in_type = Type_dict[input()]
    in_age =  int(input())
    in_gender = Gender_dict[input()]
except KeyError as e:
    print("Dati podatak nije validan!")
    exit(1)

in_data = [in_state, in_city, in_guns, in_type, in_age, in_gender]
person = np.array(in_data)
person = person.reshape(1, -1)
print("Uneta osoba je: ", inv_dict[knn.predict(person)[0]])
```

Pogledajmo jedan primer izlaza ovog programa za neke konkretne unete podatke, kao i pomenute statistike i matrice konfuzije:

```
report_train: precision    recall  f1-score   support

          0       0.99      1.00      1.00       496
          1       1.00      1.00      1.00        20
          2       0.99      1.00      0.99        92
          3       0.98      0.94      0.96        48
          4       1.00      0.95      0.98        21
          5       1.00      0.83      0.91         6

avg / total       0.99      0.99      0.99       683
report_test:  precision    recall  f1-score   support

          0       0.78      0.97      0.86       205
          1       0.00      0.00      0.00        13
          2       0.73      0.51      0.60        47
          3       0.00      0.00      0.00        20
          4       0.00      0.00      0.00         8
          5       0.00      0.00      0.00         1

avg / total       0.66      0.76      0.70       294
confusion matrix train:
 [[496   0   0   0   0   0]
 [  0  20   0   0   0   0]
 [  0   0  92   0   0   0]
 [  3   0   0  45   0   0]
 [  0   0   1   0  20   0]
 [  0   0   0   1   0   5]]
confusion matrix test:
 [[198   0   4   3   0   0]
 [ 12   0   1   0   0   0]
 [ 19   0  24   1   3   0]
 [ 18   0   2   0   0   0]
 [  5   1   2   0   0   0]
 [  1   0   0   0   0   0]]

Unesi sledece podatke:
State                min:    0.0, max:   44.0
City                 min:    0.0, max:  318.0
Guns Involved        min:    1.0, max:    4.0
Type                 min:    0.0, max:    1.0
Age                  min:    0.0, max:   82.0
Gender               min:    0.0, max:    1.0
> New Jersey
> Stratford
> 1
> Subject-Suspect
> 45
> Male
Uneta osoba je:  Unharmed, Arrested
```

Slike

1 Children Killed Missing Values . . . 6
2 Children Injured Missing Values . . . 7
3 Accidental Deaths Missing Values . . . 7
4 Accidental Injuries Missing Values . . . 8
5 Children Killed Geolocation . . . 9
6 Children Injured Geolocation . . . 9
7 Accidental Deaths Geolocation . . . 10
8 Accidental Injuries Geolocation . . . 10
9 Broj ubijenih po državama . . . 10
10 Broj ubijenih po gradovima . . . 10
11 Filtriranje podataka . . . 11
12 Accidental Deaths and Injuries Scatter Matrix . . . 12
13 Children Killed and Injured Scatter Matrix . . . 12
14 Accidental Deaths and Injuries Pie Chart . . . 13
15 Children Killed and Injured Pie Chart . . . 13
16 Accidental Deaths and Injuries Box Plot . . . 13
17 Children Killed and Injured Box Plot . . . 13
18 Korelacija za Children Killed and Injured . . . 14
19 Children Killed and Injured Apriori . . . 14
20 Česti skupovi stavki za Children Killed . . . 15
21 Pravila pridruživanja za Children Killed and Injured . . . 15
22 K-sredina za broj klastera 6 . . . 16
23 Hijerarhijsko klasterovanje za broj klastera 5 . . . 17
24 DBSCAN . . . 17
25 Stablo odlučivanja . . . 18
26 Preciznosti za test i trening skup za stablo odlučivanja . . . 18
27 Matrica konfuzije za test skup za stablo odlučivanja . . . 19
28 Stablo odlučivanja po atributu *State* . . . 19
29 Matrica konfuzije za test skup . . . 20
30 Preciznost za K najbližih suseda . . . 20
31 Matrica konfuzije za K najbližih suseda . . . 20
32 Preciznost za neuronske mreže . . . 21
33 Matrice konfuzije za neuronske mreže . . . 21
34 Matrice konfuzije za SVM koristeći RBF kernel . . . 21

Tabele

1 Atributi i njihove karakteristike . . . 5