# Analiza skupa podataka Network Intrusion Data

Seminarski rad u okviru kursa
Istraživanje podataka 1
Matematički fakultet

Marina Pilipović 115/2015
Vladana Djordjević 89/2015

Septembar 2018.

## 1 Uvod u podatke

Skup podataka koji ćemo obradjivati korišćen je na KDD (Knowledge Discovery and Data Mining) kupu održanom 1999. godine. Podaci se odnose na napadanje mreža i mogu se naći ovde. Medju navedenim linkovima mi smo koristile 10% ukupnog skupa (kddcup.data_10_percent), radi bržeg izvršavanja algoritama. Za analizu i obradu podataka korišćen je KNIME Analytics Platform.

## 2 Analiza i pretprocesiranje

Podaci se nalaze u jednoj tabeli koja se sastoji od 494021 instance, gde svaka instanca predstavlja jedan zapis o konekciji. Svaka konekcija je opisana pomoću 41 atributa i označena kao napad na mrežu odredjenog tipa ili kao normalan saobraćaj. Svi atributi se mogu podeliti u tri grupe, mi smo za svaku grupu izdvojile najvažnije atribute i prikazale ih tabelarno:

1. osnovni atributi: u ovoj kategoriji se nalaze svi atributi koji mogu biti izdvojeni iz TCP/IP konekcije.

| ime atributa | opis | tip |
|---|---|---|
| protocol_type | tip protokola | diskretan |
| service | internet servis na destinaciji | diskretan |
| src_bytes | broj bajtova podataka od izvora do destinacije | neprekidan |
| flag | status konekcije(normalan ili greška) | diskretan |

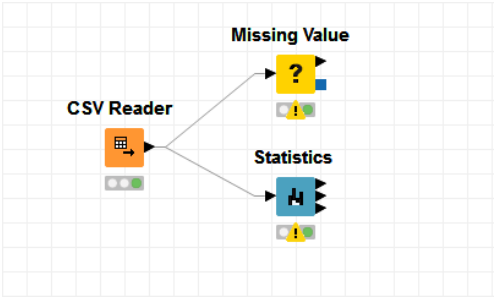
2. atributi saobraćaja: ova kategorija uključuje atribute koji su izračunati uzimajući u obzir vremenski inteval i podeljena je u dve grupe (”same host” i ”same service”).

| ime atributa | opis | tip |
|---|---|---|
| count | broj konekcija ka istom hostu kao i trenutna konekcija u poslednje 2 sekunde | neprekidan |
| serror_rate | procenat konekcija koji imaju SYN greške | neprekidan |
| rerror_rate | procenat konekcija koje imaju REJ greške | neprekidan |
| same_srv_rate | procenat konekcije ka istom servisu | neprekidan |
| diff_srv_rate | procenat konekcije ka različitim servisima | neprekidan |

3. atributi sadržaja: uključuje atribute koji pretražuju sumnjiva ponašanja u podacima.

| ime atributa | opis | tip |
|---|---|---|
| num_failed_logins | broj neuspelih pokušaja prijavljivanja | neprekidan |
| logged_in | 1 - za uspešno prijavljivanje, 0 - inače | diskretan |
| su_attempted | 1 - ako je pokušana komanda "su root", 0 - inače | diskretan |
| num_file_creations | broj operacija kreiranja fajlova | neprekidan |

Analizu skupa podataka započinjemo čvorom CSV Reader (slika 2.1), kojim učitavamo podatke. Da bismo se pozabavili nedostajućim vrednostima moramo prvo da proverimo da li ih ima. U tu svrhu koristimo čvor Missing Value kome sve tri vrednosti polja za obradu nedostajucih vrednosti postavljamo na "Remove Row*". Time obezbedjujemo da će nedostajuće vrednosti, ako se u bilo kom redu jave (bilo kategoričkih, bilo numeričkih atributa), biti uklonjene. Nakon pokretanja čvora uporedjujemo broj redova izlazne i originalne tabele. S obzirom na to da se broj redova nije promenio dolazimo do zaključka da podaci ne sadrže nedostajuće vrednosti.




slika 2.1: analiza i pretprocesiranje skupa podataka

Da bismo se upoznali sa numeričkim karakteristikama podataka koristimo čvor Statistics. Na slikama 2.2-2.5 mogu se videti neke od važnijih karakteristika.

Table "default" - Rows: 38 | Spec - Columns: 16 | Properties | Flow Variables

| Row ID | Min | Max | Mean | Std. d... | Varian... | Skew... | Kurtosis | Overal... | No. mi... | No. N... | No. +cos | No. -cos | Median | Row c... | Histogram |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| duration | 0 | 41,065 | 44.331 | 672.171 | 451,814.51 | 26.314 | 943.65 | 2,190,037 | 0 | 0 | 0 | 0 | ? | 49402 | |
| src_bytes | 0 | 5,133,876 | 1,500.53 | 50,391.708 | 2,539,324... | 92.069 | 8,990.572 | 74,129,195 | 0 | 0 | 0 | 0 | ? | 49402 | |
| dst_bytes | 0 | 5,155,468 | 879.413 | 31,157.287 | 970,776,5... | 135.358 | 20,178.849 | 43,444,782 | 0 | 0 | 0 | 0 | ? | 49402 | |
| land | 0 | 1 | 0 | 0.006 | 0 | 157.161 | 24,698.5 | 2 | 0 | 0 | 0 | 0 | ? | 49402 | |
| wrong_frag... | 0 | 3 | 0.006 | 0.135 | 0.018 | 21.638 | 472.285 | 320 | 0 | 0 | 0 | 0 | ? | 49402 | |
| urgent | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ? | 49402 | |
| hot | 0 | 30 | 0.035 | 0.785 | 0.616 | 32.422 | 1,113.493 | 1,707 | 0 | 0 | 0 | 0 | ? | 49402 | |
| num_failed... | 0 | 3 | 0 | 0.019 | 0 | 116.41 | 15,549.615 | 10 | 0 | 0 | 0 | 0 | ? | 49402 | |
| logged_in | 0 | 1 | 0.149 | 0.356 | 0.127 | 1.976 | 1.904 | 7,341 | 0 | 0 | 0 | 0 | ? | 49402 | |
| num_comp... | 0 | 238 | 0.01 | 1.076 | 1.158 | 219.189 | 48,464.914 | 475 | 0 | 0 | 0 | 0 | ? | 49402 | |
| root_shell | 0 | 1 | 0 | 0.011 | 0 | 90.726 | 8,229.5 | 6 | 0 | 0 | 0 | 0 | ? | 49402 | |

slika 2.2: statistike atributa

Table "default" - Rows: 38 | Spec - Columns: 16 | Properties | Flow Variables

| Row ID | Min | Max | Mean | Std. d... | Varian... | Skew... | Kurtosis | Overal... | No. mi... | No. N... | No. +cos | No. -cos | Median | Row c... | Histogram |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| su_attempted | 0 | 2 | 0 | 0.013 | 0 | 157.161 | 24,698.5 | 4 | 0 | 0 | 0 | 0 | ? | 49402 | |
| num_root | 0 | 268 | 0.011 | 1.223 | 1.495 | 213.309 | 46,715.188 | 561 | 0 | 0 | 0 | 0 | ? | 49402 | |
| num_file_cr... | 0 | 21 | 0.001 | 0.112 | 0.013 | 157.845 | 27,403.96 | 63 | 0 | 0 | 0 | 0 | ? | 49402 | |
| num_shells | 0 | 1 | 0 | 0.008 | 0 | 128.317 | 16,464 | 3 | 0 | 0 | 0 | 0 | ? | 49402 | |
| num_acces... | 0 | 2 | 0.001 | 0.031 | 0.001 | 39.492 | 1,744.405 | 43 | 0 | 0 | 0 | 0 | ? | 49402 | |
| num_outbo... | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ? | 49402 | |
| is_host_login | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ? | 49402 | |
| is_guest_lo... | 0 | 1 | 0.001 | 0.037 | 0.001 | 26.702 | 711.044 | 69 | 0 | 0 | 0 | 0 | ? | 49402 | |
| count | 0 | 511 | 332.331 | 213.083 | 45,404.448 | -0.543 | -1.471 | 16,417,807 | 0 | 0 | 0 | 0 | ? | 49402 | |
| srv_count | 0 | 511 | 292.933 | 246.252 | 60,640.214 | -0.274 | -1.915 | 14,471,473 | 0 | 0 | 0 | 0 | ? | 49402 | |
| serror_rate | 0 | 1 | 0.175 | 0.38 | 0.144 | 1.709 | 0.925 | 8,667.26 | 0 | 0 | 0 | 0 | ? | 49402 | |

slika 2.3: statistike atributa

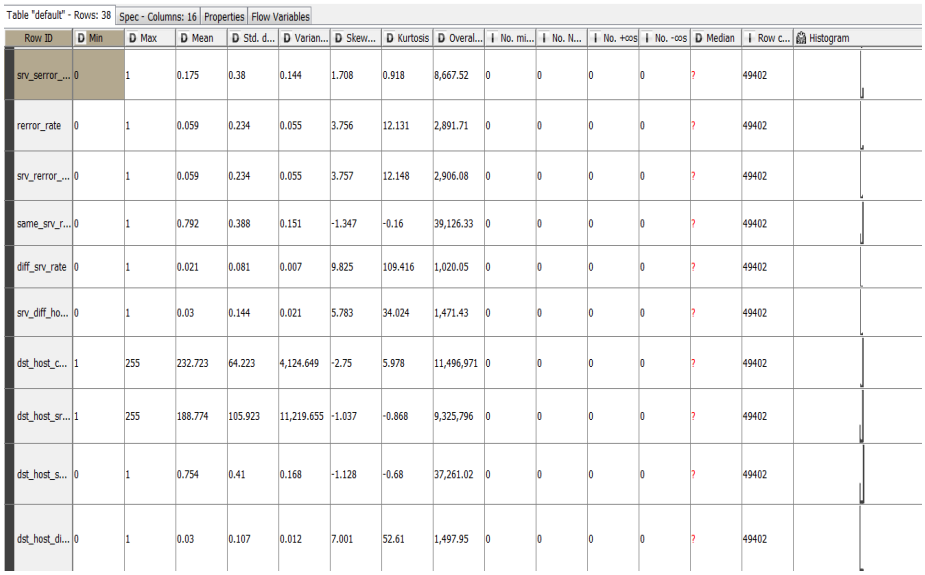


Table "default" - Rows: 38 | Spec - Columns: 16 | Properties | Flow Variables

| Row ID | Min | Max | Mean | Std. d... | Varian... | Skew... | Kurtosis | Overal... | No. mi... | No. N... | No. +os | No. -os | Median | Row c... | Histogram |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| srv_serror_... | 0 | 1 | 0.175 | 0.38 | 0.144 | 1.708 | 0.918 | 8,667.52 | 0 | 0 | 0 | 0 | ? | 49402 | |
| rerror_rate | 0 | 1 | 0.059 | 0.234 | 0.055 | 3.756 | 12.131 | 2,891.71 | 0 | 0 | 0 | 0 | ? | 49402 | |
| srv_rerror_... | 0 | 1 | 0.059 | 0.234 | 0.055 | 3.757 | 12.148 | 2,906.08 | 0 | 0 | 0 | 0 | ? | 49402 | |
| same_srv_r... | 0 | 1 | 0.792 | 0.388 | 0.151 | -1.347 | -0.16 | 39,126.33 | 0 | 0 | 0 | 0 | ? | 49402 | |
| diff_srv_rate | 0 | 1 | 0.021 | 0.081 | 0.007 | 9.825 | 109.416 | 1,020.05 | 0 | 0 | 0 | 0 | ? | 49402 | |
| srv_diff_ho... | 0 | 1 | 0.03 | 0.144 | 0.021 | 5.783 | 34.024 | 1,471.43 | 0 | 0 | 0 | 0 | ? | 49402 | |
| dst_host_c... | 1 | 255 | 232.723 | 64.223 | 4,124.649 | -2.75 | 5.978 | 11,496,971 | 0 | 0 | 0 | 0 | ? | 49402 | |
| dst_host_sr... | 1 | 255 | 188.774 | 105.923 | 11,219.655 | -1.037 | -0.868 | 9,325,796 | 0 | 0 | 0 | 0 | ? | 49402 | |
| dst_host_s... | 0 | 1 | 0.754 | 0.41 | 0.168 | -1.128 | -0.68 | 37,261.02 | 0 | 0 | 0 | 0 | ? | 49402 | |
| dst_host_di... | 0 | 1 | 0.03 | 0.107 | 0.012 | 7.001 | 52.61 | 1,497.95 | 0 | 0 | 0 | 0 | ? | 49402 | |

slika 2.4: statistike atributa

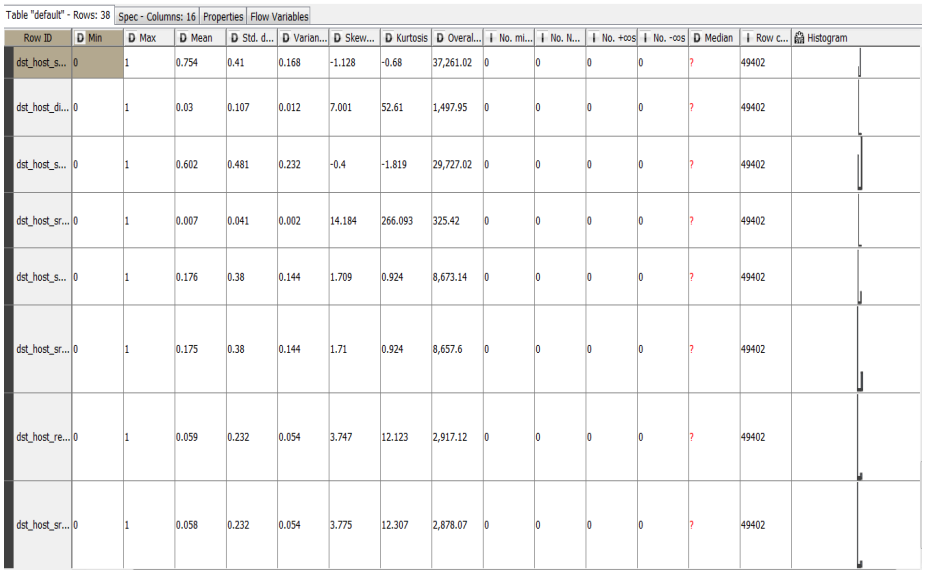


Table "default" - Rows: 38 | Spec - Columns: 16 | Properties | Flow Variables

| Row ID | Min | Max | Mean | Std. d... | Varian... | Skew... | Kurtosis | Overal... | No. mi... | No. N... | No. +os | No. -os | Median | Row c... | Histogram |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| dst_host_s... | 0 | 1 | 0.754 | 0.41 | 0.168 | -1.128 | -0.68 | 37,261.02 | 0 | 0 | 0 | 0 | ? | 49402 | |
| dst_host_di... | 0 | 1 | 0.03 | 0.107 | 0.012 | 7.001 | 52.61 | 1,497.95 | 0 | 0 | 0 | 0 | ? | 49402 | |
| dst_host_s... | 0 | 1 | 0.602 | 0.481 | 0.232 | -0.4 | -1.819 | 29,727.02 | 0 | 0 | 0 | 0 | ? | 49402 | |
| dst_host_sr... | 0 | 1 | 0.007 | 0.041 | 0.002 | 14.184 | 266.093 | 325.42 | 0 | 0 | 0 | 0 | ? | 49402 | |
| dst_host_s... | 0 | 1 | 0.176 | 0.38 | 0.144 | 1.709 | 0.924 | 8,673.14 | 0 | 0 | 0 | 0 | ? | 49402 | |
| dst_host_sr... | 0 | 1 | 0.175 | 0.38 | 0.144 | 1.71 | 0.924 | 8,657.6 | 0 | 0 | 0 | 0 | ? | 49402 | |
| dst_host_re... | 0 | 1 | 0.059 | 0.232 | 0.054 | 3.747 | 12.123 | 2,917.12 | 0 | 0 | 0 | 0 | ? | 49402 | |
| dst_host_sr... | 0 | 1 | 0.058 | 0.232 | 0.054 | 3.775 | 12.307 | 2,878.07 | 0 | 0 | 0 | 0 | ? | 49402 | |

slika 2.5: statistike atributa

Na osnovu histograma možemo da zaključimo da se u većini atributa veliki broj puta pojavljuje jedna do nekoliko različitih vrednosti. Znači da su atributi takvi da se u njima ne javlja mnoštvo različitih vrednosti. To zaključujemo na osnovu toga što se u histogramima ne javljaju vrednosti rasporedjene duž njegove x-ose, već su vrednosti uglavnom skoncentrisane na jednom mestu. Nekoliko atributa predstavlja flegove, čije vrednosti mogu biti jedino 0 i 1. Neki

izražavaju procente u opsegu [0,1].

## 2.1 Obrada outlier-a

Za obradu outlier-a naših podataka KNIME nije bio pogodan izbor, s obzirom na to da je vizualizacija outlier-a pomoću Box Plot čvora bila teško čitljiva. Stoga smo koristile programski jezik Python i biblioteke za mašinsko učenje pandas i numpy da bismo izdvojile vrednosti van granica. Na slikama 2.6 i 2.7 se nalazi kod u Python-u kao i izlaz programa koji se sastoji od imena svake numeričke kolone, percentila te kolone, granica za blage i prave outlier-e kao i izdvojeni pravi outlier-i, ako ih ima.

```
import numpy as np
import pandas as pd

def main():

    df = pd.read_csv('kddcup.csv.data_10_percent_corrected')

    numerical_attribute_list = ['duration', 'src_bytes', 'dst_bytes', 'land', 'wrong_fragment',
                      'urgent', 'hot', 'num_failed_logins', 'logged_in', 'num_compromised', 'root_shell',
                      'su_attempted','num_root', 'num_file_creations', 'num_shells', 'num_access_files',
                      'num_outbound_cmds', 'is_host_login', 'is_guest_login', 'count', 'srv_count',
                      'serror_rate', 'srv_serror_rate', 'rerror_rate', 'srv_rerror_rate', 'same_srv_rate',
                      'diff_srv_rate', 'srv_diff_host_rate', 'dst_host_count', 'dst_host_srv_count',
                      'dst_host_same_srv_rate','dst_host_diff_srv_rate', 'dst_host_same_src_port_rate',
                      'dst_host_srv_diff_host_rate','dst_host_serror_rate','dst_host_srv_serror_rate',
                      'dst_host_rerror_rate', 'dst_host_srv_rerror_rate']


    for i in range(len(numerical_attribute_list)):

        attribute_name = numerical_attribute_list[i]
        print("Attribute name: " + attribute_name)

        percentiles = np.percentile(df[attribute_name], [0, 25, 50, 75, 100])
        print("0%: " + str(percentiles[0]) +
              ", 25%: " + str(percentiles[1]) +
              ", 50%: " + str(percentiles[2]) +
              ", 75%: " + str(percentiles[3]) +
              ", 100%: " + str(percentiles[4]))


        q1 = percentiles[1]
        q3 = percentiles[3]
        IQR = q3 - q1
        f1 = q1 - 3*IQR/2
        f3 = q3 + 3*IQR/2
        F1 = q1 - 3*IQR
        F3 = q3 + 3*IQR

        print ("Mild outliers: [" + str(F1) + ", " + str(f1) + "] and [" + str(f3) + ", " + str(F3) + "]")
        print ("Extreme outliers:  (-" + str(float('inf')) + ", " + str(F1) + ") and (" + str(F3) + ", +" + str(float('inf')) + ")")

        if (F1 != F3):
            print ("Extreme outlier found: ")
            new_df = df[(df[attribute_name] < F1) | (df[attribute_name] > F3)]
            print(new_df[attribute_name])
        else:
            print ("All of the column's values are considered outliers")

        print("\n")

if __name__ == "__main__":
    main()
```

slika 2.6: programski kod u Python-u za obradu outlier-a

```
Attribute name: duration
0%: 0.0, 25%: 0.0, 50%: 0.0, 75%: 0.0, 100%: 58329.0
Mild outliers: [0.0, 0.0] and [0.0, 0.0]
Extreme outliers:  (-inf, 0.0) and (0.0, +inf)
All of the column's values are considered outliers


Attribute name: src_bytes
0%: 0.0, 25%: 45.0, 50%: 520.0, 75%: 1032.0, 100%: 693375640.0
Mild outliers: [-2916.0, -1435.5] and [2512.5, 3993.0]
Extreme outliers:  (-inf, -2916.0) and (3993.0, +inf)
Extreme outlier found:
725        19721
747        15744
766         4031
777        15726
792         9640
951         5519
3996        4173
4006       15716
4029       15688
4030        4034
4046       15310
4069        7022
4089        9597
4103        4056
6850        4247
6884        4247
7531       12813
7559       13859
7608       12804
7611       12762
7641        4876
7649       12490
7672        6433
7707        6471
7742        4875
7772       12748
7785        6210
7788        5208
```

```
7772       12748
7785        6210
7788        5208
15626      13777
15634      14349
           ...
487033      6062
487050    106532
491468     12324
491475      5319
491478     11376
491487      7280
491488     35195
491491     12983
491548      6601
491549      6062
491659     15377
491661      9178
491670      5150
491677     12324
491697     10073
491699     15377
491708      7162
491716     14052
491717      5558
491729      5661
491731     36530
491734      5161
491735      4793
491736      4316
491745     12983
491747      7162
491750      7321
491773      7940
491799     66896
491807      8766
Name: src_bytes, Length: 4056, dtype: int64


Attribute name: dst_bytes
0%: 0.0, 25%: 0.0, 50%: 0.0, 75%: 0.0, 100%: 5155468.0
Mild outliers: [0.0, 0.0] and [0.0, 0.0]
Extreme outliers:  (-inf, 0.0) and (0.0, +inf)
```

```
Attribute name: land
0%: 0.0, 25%: 0.0, 50%: 0.0, 75%: 0.0, 100%: 1.0
Mild outliers: [0.0, 0.0] and [0.0, 0.0]
Extreme outliers:  (-inf, 0.0) and (0.0, +inf)
All of the column's values are considered outliers


Attribute name: wrong_fragment
0%: 0.0, 25%: 0.0, 50%: 0.0, 75%: 0.0, 100%: 3.0
Mild outliers: [0.0, 0.0] and [0.0, 0.0]
Extreme outliers:  (-inf, 0.0) and (0.0, +inf)
All of the column's values are considered outliers


Attribute name: urgent
0%: 0.0, 25%: 0.0, 50%: 0.0, 75%: 0.0, 100%: 3.0
Mild outliers: [0.0, 0.0] and [0.0, 0.0]
Extreme outliers:  (-inf, 0.0) and (0.0, +inf)
All of the column's values are considered outliers


Attribute name: hot
0%: 0.0, 25%: 0.0, 50%: 0.0, 75%: 0.0, 100%: 30.0
Mild outliers: [0.0, 0.0] and [0.0, 0.0]
Extreme outliers:  (-inf, 0.0) and (0.0, +inf)
All of the column's values are considered outliers


Attribute name: num_failed_logins
0%: 0.0, 25%: 0.0, 50%: 0.0, 75%: 0.0, 100%: 5.0
Mild outliers: [0.0, 0.0] and [0.0, 0.0]
Extreme outliers:  (-inf, 0.0) and (0.0, +inf)
All of the column's values are considered outliers


Attribute name: logged_in
0%: 0.0, 25%: 0.0, 50%: 0.0, 75%: 0.0, 100%: 1.0
Mild outliers: [0.0, 0.0] and [0.0, 0.0]
Extreme outliers:  (-inf, 0.0) and (0.0, +inf)
All of the column's values are considered outliers
```

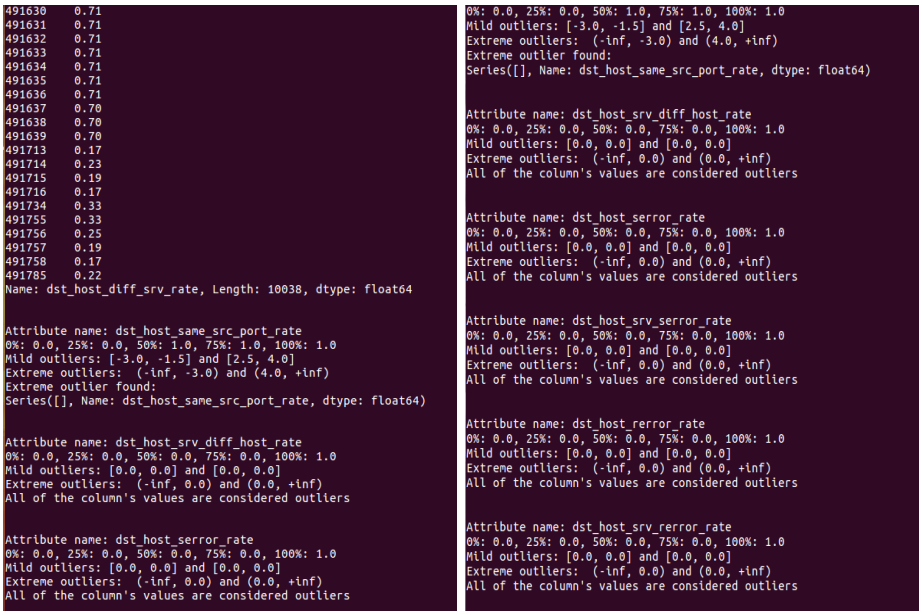
```
All of the column's values are considered outliers


Attribute name: num_compromised
0%: 0.0, 25%: 0.0, 50%: 0.0, 75%: 0.0, 100%: 884.0
Mild outliers: [0.0, 0.0] and [0.0, 0.0]
Extreme outliers:  (-inf, 0.0) and (0.0, +inf)
All of the column's values are considered outliers


Attribute name: root_shell
0%: 0.0, 25%: 0.0, 50%: 0.0, 75%: 0.0, 100%: 1.0
Mild outliers: [0.0, 0.0] and [0.0, 0.0]
Extreme outliers:  (-inf, 0.0) and (0.0, +inf)
All of the column's values are considered outliers


Attribute name: su_attempted
0%: 0.0, 25%: 0.0, 50%: 0.0, 75%: 0.0, 100%: 2.0
Mild outliers: [0.0, 0.0] and [0.0, 0.0]
Extreme outliers:  (-inf, 0.0) and (0.0, +inf)
All of the column's values are considered outliers


Attribute name: num_root
0%: 0.0, 25%: 0.0, 50%: 0.0, 75%: 0.0, 100%: 993.0
Mild outliers: [0.0, 0.0] and [0.0, 0.0]
Extreme outliers:  (-inf, 0.0) and (0.0, +inf)
All of the column's values are considered outliers


Attribute name: num_file_creations
0%: 0.0, 25%: 0.0, 50%: 0.0, 75%: 0.0, 100%: 28.0
Mild outliers: [0.0, 0.0] and [0.0, 0.0]
Extreme outliers:  (-inf, 0.0) and (0.0, +inf)
All of the column's values are considered outliers


Attribute name: num_shells
0%: 0.0, 25%: 0.0, 50%: 0.0, 75%: 0.0, 100%: 2.0
Mild outliers: [0.0, 0.0] and [0.0, 0.0]
Extreme outliers:  (-inf, 0.0) and (0.0, +inf)
All of the column's values are considered outliers
```

```
Attribute name: num_access_files
0%: 0.0, 25%: 0.0, 50%: 0.0, 75%: 0.0, 100%: 8.0
Mild outliers: [0.0, 0.0] and [0.0, 0.0]
Extreme outliers:  (-inf, 0.0) and (0.0, +inf)
All of the column's values are considered outliers


Attribute name: num_outbound_cmds
0%: 0.0, 25%: 0.0, 50%: 0.0, 75%: 0.0, 100%: 0.0
Mild outliers: [0.0, 0.0] and [0.0, 0.0]
Extreme outliers:  (-inf, 0.0) and (0.0, +inf)
All of the column's values are considered outliers


Attribute name: is_host_login
0%: 0.0, 25%: 0.0, 50%: 0.0, 75%: 0.0, 100%: 0.0
Mild outliers: [0.0, 0.0] and [0.0, 0.0]
Extreme outliers:  (-inf, 0.0) and (0.0, +inf)
All of the column's values are considered outliers


Attribute name: is_guest_login
0%: 0.0, 25%: 0.0, 50%: 0.0, 75%: 0.0, 100%: 1.0
Mild outliers: [0.0, 0.0] and [0.0, 0.0]
Extreme outliers:  (-inf, 0.0) and (0.0, +inf)
All of the column's values are considered outliers


Attribute name: count
0%: 0.0, 25%: 117.0, 50%: 510.0, 75%: 511.0, 100%: 511.0
Mild outliers: [-1065.0, -474.0] and [1102.0, 1693.0]
Extreme outliers:  (-inf, -1065.0) and (1693.0, +inf)
Extreme outlier found:
Series([], Name: count, dtype: int64)


Attribute name: srv_count
0%: 0.0, 25%: 10.0, 50%: 510.0, 75%: 511.0, 100%: 511.0
Mild outliers: [-1493.0, -741.5] and [1262.5, 2014.0]
Extreme outliers:  (-inf, -1493.0) and (2014.0, +inf)
Extreme outlier found:
Series([], Name: srv_count, dtype: int64)
```

```
Attribute name: serror_rate
0%: 0.0, 25%: 0.0, 50%: 0.0, 75%: 0.0, 100%: 1.0
Mild outliers: [0.0, 0.0] and [0.0, 0.0]
Extreme outliers:  (-inf, 0.0) and (0.0, +inf)
All of the column's values are considered outliers


Attribute name: srv_serror_rate
0%: 0.0, 25%: 0.0, 50%: 0.0, 75%: 0.0, 100%: 1.0
Mild outliers: [0.0, 0.0] and [0.0, 0.0]
Extreme outliers:  (-inf, 0.0) and (0.0, +inf)
All of the column's values are considered outliers


Attribute name: rerror_rate
0%: 0.0, 25%: 0.0, 50%: 0.0, 75%: 0.0, 100%: 1.0
Mild outliers: [0.0, 0.0] and [0.0, 0.0]
Extreme outliers:  (-inf, 0.0) and (0.0, +inf)
All of the column's values are considered outliers


Attribute name: srv_rerror_rate
0%: 0.0, 25%: 0.0, 50%: 0.0, 75%: 0.0, 100%: 1.0
Mild outliers: [0.0, 0.0] and [0.0, 0.0]
Extreme outliers:  (-inf, 0.0) and (0.0, +inf)
All of the column's values are considered outliers


Attribute name: same_srv_rate
0%: 0.0, 25%: 1.0, 50%: 1.0, 75%: 1.0, 100%: 1.0
Mild outliers: [1.0, 1.0] and [1.0, 1.0]
Extreme outliers:  (-inf, 1.0) and (1.0, +inf)
All of the column's values are considered outliers


Attribute name: diff_srv_rate
0%: 0.0, 25%: 0.0, 50%: 0.0, 75%: 0.0, 100%: 1.0
Mild outliers: [0.0, 0.0] and [0.0, 0.0]
Extreme outliers:  (-inf, 0.0) and (0.0, +inf)
All of the column's values are considered outliers


Attribute name: srv_diff_host_rate
```

```
Attribute name: srv_diff_host_rate
0%: 0.0, 25%: 0.0, 50%: 0.0, 75%: 0.0, 100%: 1.0
Mild outliers: [0.0, 0.0] and [0.0, 0.0]
Extreme outliers:  (-inf, 0.0) and (0.0, +inf)
All of the column's values are considered outliers


Attribute name: dst_host_count
0%: 0.0, 25%: 255.0, 50%: 255.0, 75%: 255.0, 100%: 255.0
Mild outliers: [255.0, 255.0] and [255.0, 255.0]
Extreme outliers:  (-inf, 255.0) and (255.0, +inf)
All of the column's values are considered outliers


Attribute name: dst_host_srv_count
0%: 0.0, 25%: 46.0, 50%: 255.0, 75%: 255.0, 100%: 255.0
Mild outliers: [-581.0, -267.5] and [568.5, 882.0]
Extreme outliers:  (-inf, -581.0) and (882.0, +inf)
Extreme outlier found:
Series([], Name: dst_host_srv_count, dtype: int64)


Attribute name: dst_host_same_srv_rate
0%: 0.0, 25%: 0.41, 50%: 1.0, 75%: 1.0, 100%: 1.0
Mild outliers: [-1.360000000000003, -0.4750000000000014] and [1.885000000000002, 2.7700000000000005]
Extreme outliers:  (-inf, -1.3600000000000003) and (2.7700000000000005, +inf)
Extreme outlier found:
Series([], Name: dst_host_same_srv_rate, dtype: float64)


Attribute name: dst_host_diff_srv_rate
0%: 0.0, 25%: 0.0, 50%: 0.0, 75%: 0.04, 100%: 1.0
Mild outliers: [-0.12, -0.06] and [0.1, 0.16]
Extreme outliers:  (-inf, -0.12) and (0.16, +inf)
Extreme outlier found:
206      0.25
207      0.17
218      0.40
219      0.20
725      0.17
729      0.20
746      0.17
```

```
725        0.17
729        0.20
746        0.17
759        0.67
776        0.75
777        0.21
780        0.40
783        0.30
787        0.20
822        0.22
914        0.33
916        0.17
922        1.00
923        0.29
929        0.33
930        0.19
937        0.50
938        0.25
945        0.22
954        0.50
955        0.25
960        0.75
961        0.36
962        0.21
3582       0.27
3583       0.19
           ...
491620     0.70
491621     0.70
491622     0.70
491623     0.70
491624     0.70
491625     0.71
491626     0.71
491627     0.71
491628     0.71
491629     0.70
491630     0.71
491631     0.71
491632     0.71
491633     0.71
491634     0.71
491635     0.71
```

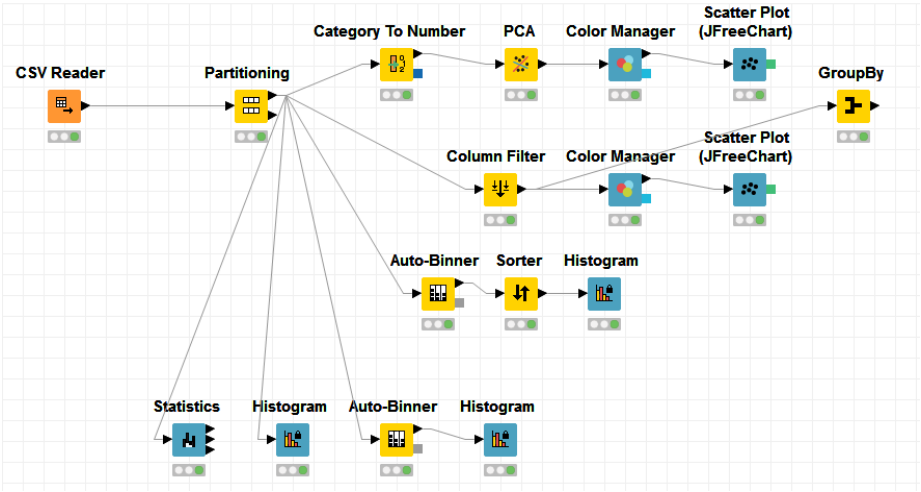



slika 2.7: izlaz iz programa, podaci za svaki atribut

Kao što vidimo mnogi atributi se sastoje pretežno od 0, što dovodi do toga da obe granice za blage i prave outlier-e budu 0, a iz toga sledi da su sve vrednosti tih atributa zapravo outlier-i. U kodu smo ograničile da nam se na izlazu ne prikazuju outlier-i atributa čije se sve vrednosti smatraju outlier-ima, jer bi nam se onda prikazivale cele kolone, a to nije poenta. Umesto toga smo samo ispisale poruku posle atributa da se sve vrednosti atributa smatraju outlier-ima. Kod atributa čije granice outlier-a nisu isti brojevi ili smo dobile prazan skup outlier-a, što znači da ih nemaju (npr. atributi: count, srv_count, dst_host_same_srv_rate i dst_host_same_srv_port_rate) ili smo dobile prave outlier-e koji su podskup skupa vrednosti atributa (npr. atributi: src_bytes i dst_host_diff_srv_rate).

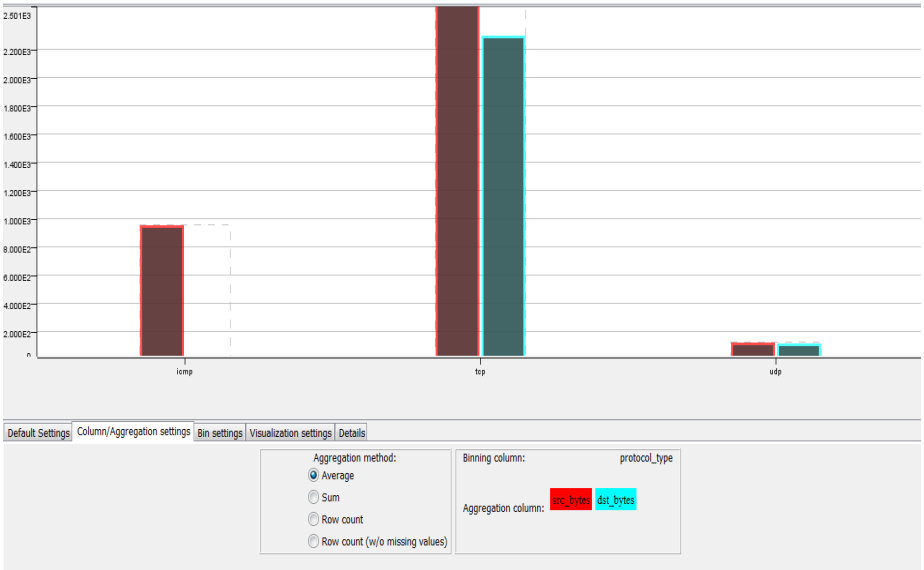
# 3 Vizualizacija

Ponovo učitavamo podatke koristeći CSV Reader. S obzirom na to da obrada skoro 500000 instanci zahteva mnogo vremena, koristimo čvor Partitioning da bismo izdvojili 10% celokupnog skupa, čime smanjujemo vreme izvršavanja. Kao parametar smo zadali stratifikovano uzorkovanje na osnovu ciljne klase (class).




slika 3.1: vizualizacija

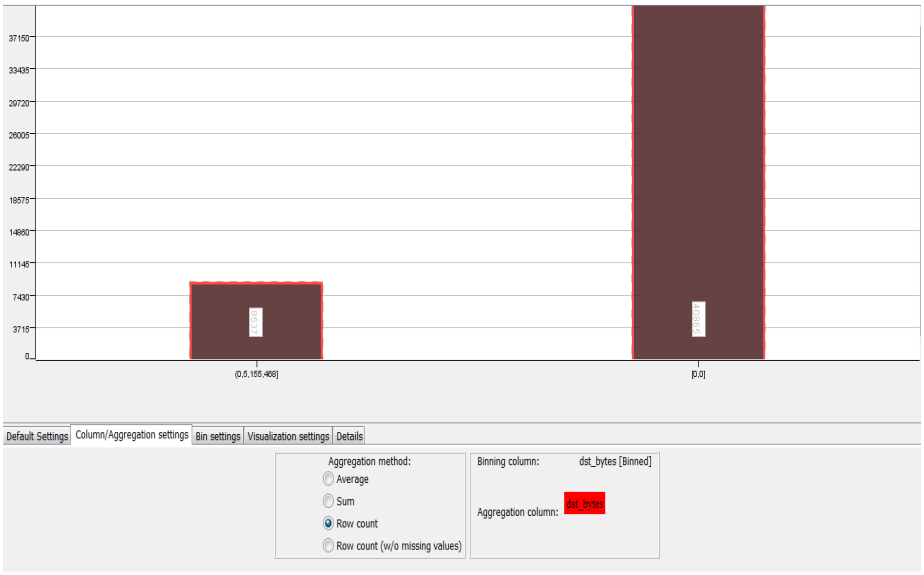
Za vizualizaciju podataka koristimo čvor Histogram. Želimo da vidimo koliko se u proseku šalje i prima bajtova u odnosu na različite protokole. Zato smo kao kolonu koju ćemo "razložiti" na različite vrednosti stavili atribut protocol_type (tip protokola), a kolone po kojima ćemo vršiti agregaciju smo stavili da budu src_bytes (broj bajtova od izvora do odredišta) i dst_bytes (broj bajtova od odredišta do izvora).




slika 3.2: Prosečan broj poslatih i primljenih bajtova po protokolu

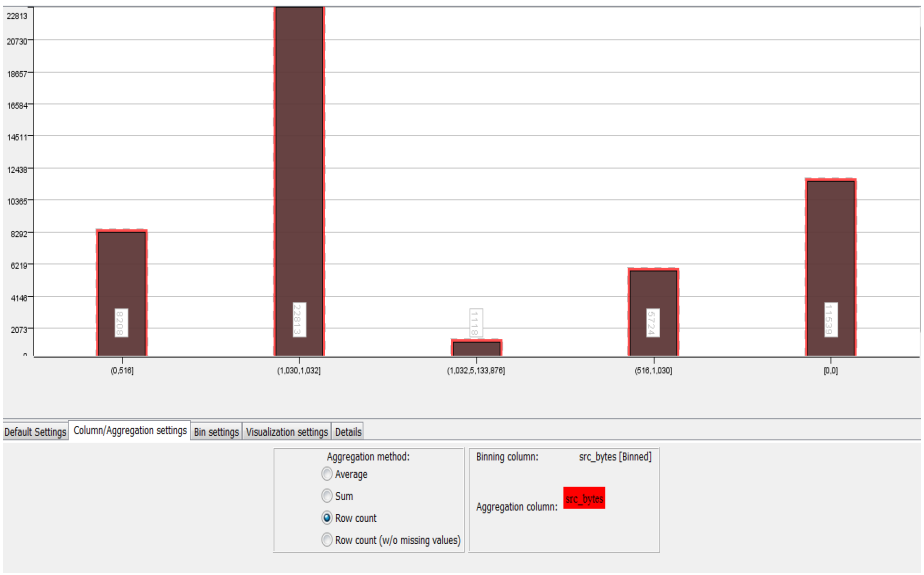
Uočavamo da se mnogo veći broj bajtova po proseku prenosi preko tcp protokola (i icmp u odredjenoj meri) nego preko udp. Takodje, kod konekcija koje

su koristile icmp protokol nije bilo primljenih bajtova (od odredišta do izvora). Sve to može da nam bude zanimljivo, s obzirom na to da se neki napadi sastoje isključivo od ogromnog broja poslatih bajtova. Na ovaj način možemo dovesti u vezu protokole konekcije sa napadima na mreže.
Potom smo podelile sve moguće vrednosti atributa dst_bytes u 5 različitih intervala po frekvenciji koristći čvor Auto-Binner i te vrednosti prosledili Histogram-u jer smo želele da vidimo u kom opsegu se nalazi broj bajtova koji se najviše prenosi po konekciji.



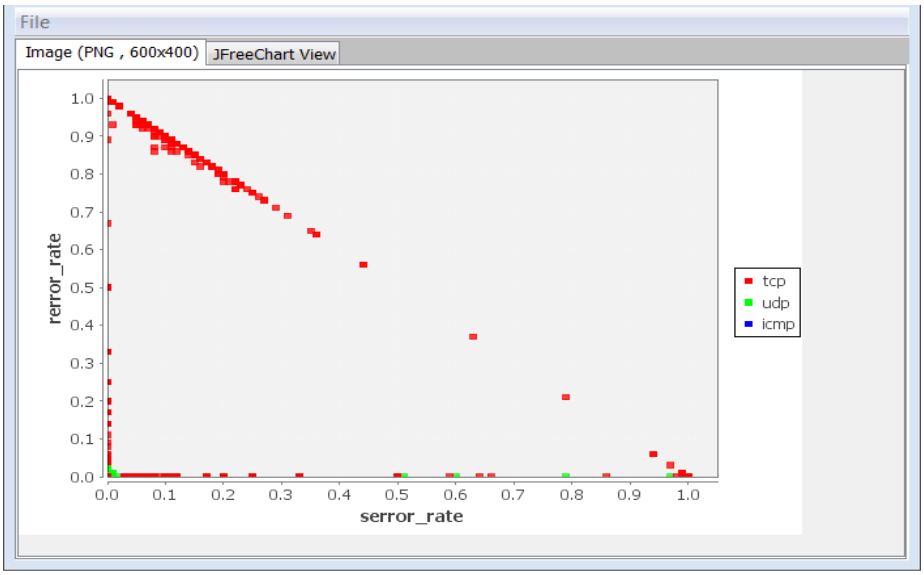

slika 3.3: Broj instanci po različitom opsegu vrednosti atributa dst_bytes

Kao rezultat smo dobile dve intervala, jer ovaj atribut ne sadrži veliki broj različitih vrednosti da bi se mogao na toliko intervala razložiti. Takodje, vidimo da se u najvećem broju konekcija (instanci) šalje 0 bajtova od odredišta do izvora. Isti postupak smo primenile na atribut src_bytes.




slika 3.4: Broj instanci po različitom opsegu vrednosti atributa src_bytes

Ovde već jesmo dobile 5 različitih intervala usled činjenice da ovaj atribut sadrži veći broj različitih vrednosti.
Nakon toga, zanimao nas je odnos tipa protokola (protocol_type) u odnosu na procenat "SYN" (serror_rate) i "REJ" (rerror_rate) greške. Prvo smo pomoću čvora Column Filter izdvojili pomenute atribute, nakon toga smo u čvoru Color Manager dodelili boje različitim tipovima protokola, a onda smo rezultate iscrtale pomoću čvora Scatter Plot (JFreeChart).



slika 3.5: Različiti protokoli u odnosu na procenat "SYN" i "REJ" greške

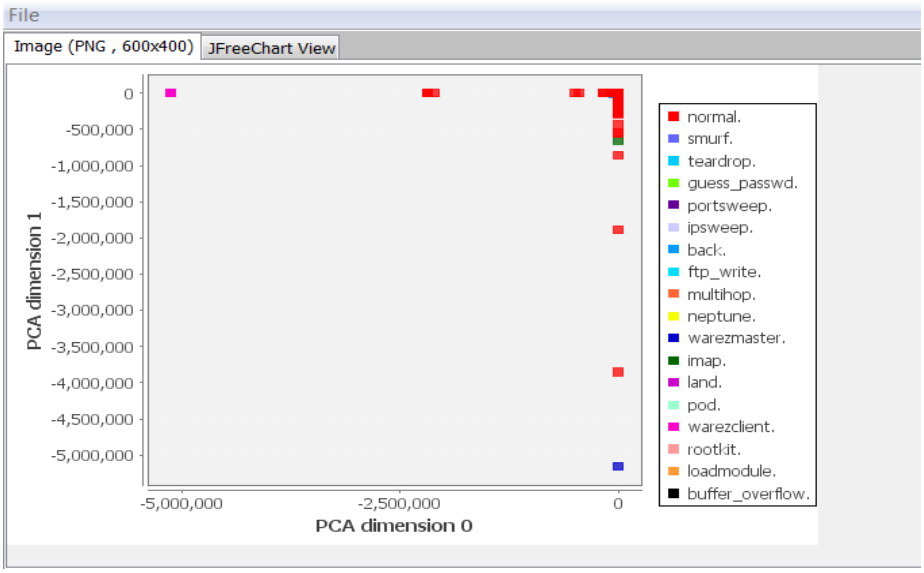
Primećujemo da se većina grešaka javila u okviru tcp protokola, a nijedna u okviru icmp protokola. Da bismo to proverile iskoristile smo čvor GroupBy koji grupiše redove tabele po jedinstvenim vrednostima u izabranim atributima, a mi smo izabrale upravo tri pomenuta atributa.

Table "default" - Rows: 112 | Spec - Columns: 3 | Prope

| Row ID | protoc... | serror... | rerror... |
|---|---|---|---|
| Row0 | icmp | 0 | 0 |
| Row1 | tcp | 0 | 0 |
| Row2 | tcp | 0 | 0.04 |
| Row3 | tcp | 0 | 0.05 |
| Row4 | tcp | 0 | 0.06 |
| Row5 | tcp | 0 | 0.08 |
| Row6 | tcp | 0 | 0.09 |
| Row7 | tcp | 0 | 0.11 |
| Row8 | tcp | 0 | 0.14 |
| Row9 | tcp | 0 | 0.17 |
| Row10 | tcp | 0 | 0.2 |
| Row11 | tcp | 0 | 0.25 |
| Row12 | tcp | 0 | 0.33 |
| Row13 | tcp | 0 | 0.5 |
| Row14 | tcp | 0 | 0.67 |
| Row15 | tcp | 0 | 0.89 |
| Row16 | tcp | 0 | 0.96 |
| Row17 | tcp | 0 | 0.99 |
| Row18 | tcp | 0 | 1 |
| Row19 | tcp | 0.01 | 0.93 |
| Row20 | tcp | 0.01 | 0.99 |
| Row21 | tcp | 0.02 | 0.98 |
| Row22 | tcp | 0.03 | 0 |
| Row23 | tcp | 0.04 | 0 |
| Row24 | tcp | 0.04 | 0.96 |
| Row25 | tcp | 0.05 | 0 |
| Row26 | tcp | 0.05 | 0.93 |
| Row27 | tcp | 0.05 | 0.94 |
| Row28 | tcp | 0.05 | 0.95 |
| Row29 | tcp | 0.06 | 0 |
| Row30 | tcp | 0.06 | 0.92 |
| Row31 | tcp | 0.06 | 0.93 |
| Row32 | tcp | 0.06 | 0.94 |
| Row33 | tcp | 0.07 | 0 |
| Row78 | tcp | 0.25 | 0 |
| Row79 | tcp | 0.25 | 0.75 |
| Row80 | tcp | 0.26 | 0.74 |
| Row81 | tcp | 0.27 | 0.73 |
| Row82 | tcp | 0.29 | 0.71 |
| Row83 | tcp | 0.31 | 0.69 |
| Row84 | tcp | 0.33 | 0 |
| Row85 | tcp | 0.35 | 0.65 |
| Row86 | tcp | 0.36 | 0.64 |
| Row87 | tcp | 0.44 | 0.56 |
| Row88 | tcp | 0.5 | 0 |
| Row89 | tcp | 0.59 | 0 |
| Row90 | tcp | 0.63 | 0.37 |
| Row91 | tcp | 0.64 | 0 |
| Row92 | tcp | 0.66 | 0 |
| Row93 | tcp | 0.79 | 0.21 |
| Row94 | tcp | 0.86 | 0 |
| Row95 | tcp | 0.94 | 0.06 |
| Row96 | tcp | 0.97 | 0.03 |
| Row97 | tcp | 0.98 | 0 |
| Row98 | tcp | 0.99 | 0 |
| Row99 | tcp | 0.99 | 0.01 |
| Row100 | tcp | 1 | 0 |
| Row101 | udp | 0 | 0 |
| Row102 | udp | 0 | 0.02 |
| Row103 | udp | 0 | 0.03 |
| Row104 | udp | 0 | 0.04 |
| Row105 | udp | 0.01 | 0 |
| Row106 | udp | 0.01 | 0.01 |
| Row107 | udp | 0.02 | 0 |
| Row108 | udp | 0.51 | 0 |
| Row109 | udp | 0.6 | 0 |
| Row110 | udp | 0.79 | 0 |
| Row111 | udp | 0.97 | 0 |

slika 3.6: Deo tabele koji predstavlja grupisane instance po jedinstvenim vrednostima atributa

Kao što možemo da vidimo na osnovu tabele icmp zaista ima 0 procenata i "SYN" i "REJ" greške kao i da se u udp konekciji javlja mali procenat istih. Dakle, ove vrste grešaka se najviše javljaju u okviru konekcije sa tcp protokolom, a one su karakteristične za DoS napade. Ono što je vrlo interesantno je to da se napadi na mreže mogu više dovesti u vezu sa tcp protokolom, koji omogućava sigurniju vezu, nego sa udp protokolom koji nije toliko siguran.
Hteli smo i da proverimo na koji način smanjivanje dimenzionalnosti našim podacima utiče na krajnji ishod. U tu svrhu smo iskoristile čvor PCA, kome smo kao parametar stavile da želimo 95% informacija da sačuva, medjutim, pre njegove primene, čvorom Category To Number smo tri kategorička atributa - protocol_type, service i flag - prebacile u numeričke jer PCA algoritam radi samo sa numeričkim vrednostima. Na taj način smo omogućile da PCA koristi sve atribute u procesu smanjivanja dimenzionalnosti. Nakon toga smo povezale čvor Color Manager koji je dodelio boju svim mogućim vrednostima ciljne klase i potom smo rezultat prikazali pomoću čvora Scatter Plot (JFreeChart).

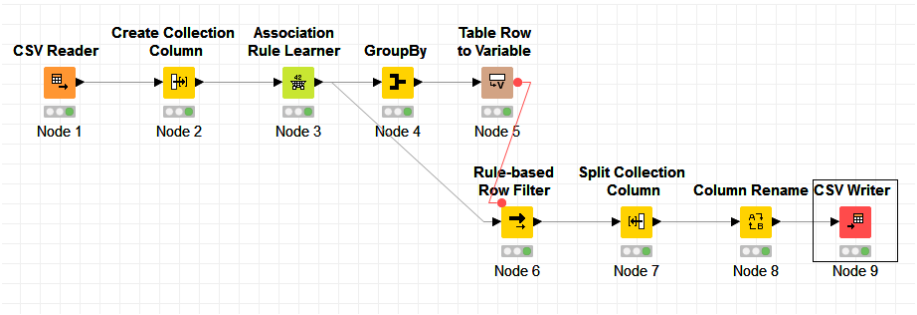



slika 3.7: Različiti protokoli u odnosu na procenat "SYN" i "REJ" greške

Možemo da zaključimo da se u najvećem broju slučajeva javlja normalna konekcija. Osim normalne konekcije javljaju se i napadi tipa warezclient, imap i warezmaster.

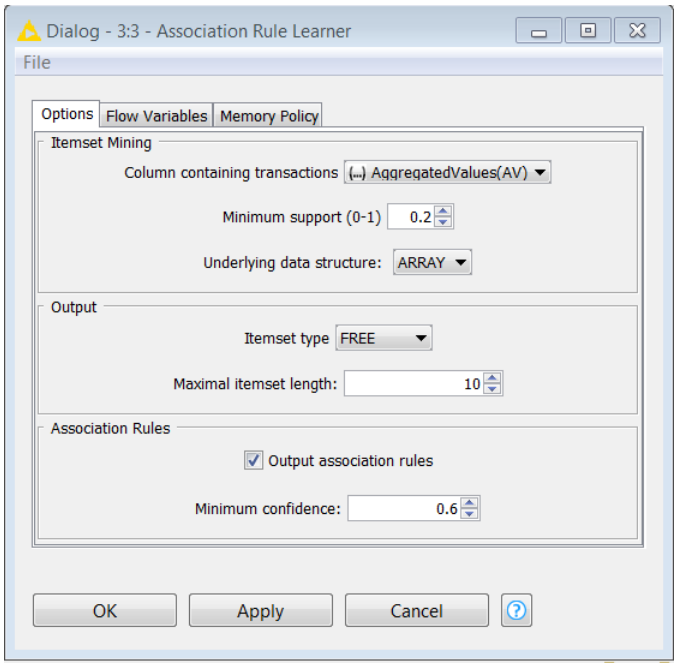
# 4 Pravila pridruživanja

Da bismo primenili pravila pridruživanja na naše podatke potrebno je prvo da ih učitamo, za šta koristimo CSV Reader, a potom da napravimo kolekcijsku kolonu koja sadrži elemente koji su nam od značaja za pravila pridruživanja. Kolekcijsku kolonu pravimo koristeći čvor Create Collection Column kome zadajemo da uključi atribute: protocol_type, service, flag (status konekcije), count (broj konekcija ka istom hostu kao i trenutna konekcija u poslednje dve sekunde) i serror_rate (procenat konekcija koje imaju "SYN" grešku). Ovo su atributi koji su značajni za DoS napad, pa nas zanima njihov medjusobni odnos zarad boljeg razumevanja samih napada.




slika 4.1: pravila pridruživanja

Nakon toga smo pokrenuli čvor Association Rule Learner kome smo parametre podesili na sledeći način:




slika 4.2: podešavanje parametara čvora Association Rule Learner

Kao rezultat smo dobili sledeća pravila pridruživanja:

Table "default" - Rows: 21 | Spec - Columns: 6 | Properties | Flow Variables

| Row ID | D Support | D Confid... | D Lift | ? Conse... | S implies | (...) Items |
|---|---|---|---|---|---|---|
| rule0 | 0.207 | 0.923 | 2.4 | tcp | <--- | [private] |
| rule1 | 0.46 | 1 | 1.305 | SF | <--- | [0.0,ecr_i,icmp,...] |
| rule2 | 0.46 | 1 | 1.22 | 0.0 | <--- | [SF,ecr_i,icmp,...] |
| rule3 | 0.46 | 1 | 1.742 | icmp | <--- | [0.0,SF,ecr_i,...] |
| rule4 | 0.46 | 1 | 1.756 | ecr_i | <--- | [0.0,SF,icmp,...] |
| rule5 | 0.46 | 0.807 | 1.75 | 511 | <--- | [0.0,SF,ecr_i,...] |
| rule6 | 0.46 | 1 | 1.305 | SF | <--- | [0.0,511] |
| rule7 | 0.46 | 1 | 1.22 | 0.0 | <--- | [SF,511] |
| rule8 | 0.46 | 0.601 | 1.304 | 511 | <--- | [0.0,SF] |
| rule9 | 0.46 | 0.997 | 1.301 | SF | <--- | [511] |
| rule10 | 0.46 | 0.6 | 1.301 | 511 | <--- | [SF] |
| rule11 | 0.46 | 0.997 | 1.216 | 0.0 | <--- | [511] |
| rule12 | 0.57 | 1 | 1.305 | SF | <--- | [0.0,ecr_i,icmp] |
| rule13 | 0.57 | 1 | 1.22 | 0.0 | <--- | [SF,ecr_i,icmp] |
| rule14 | 0.57 | 1 | 1.742 | icmp | <--- | [0.0,SF,ecr_i] |
| rule15 | 0.57 | 0.992 | 1.742 | ecr_i | <--- | [0.0,SF,icmp] |
| rule16 | 0.574 | 1 | 1.305 | SF | <--- | [0.0,icmp] |
| rule17 | 0.574 | 1 | 1.22 | 0.0 | <--- | [SF,icmp] |
| rule18 | 0.574 | 0.751 | 1.308 | icmp | <--- | [0.0,SF] |
| rule19 | 0.764 | 0.933 | 1.218 | SF | <--- | [0.0] |
| rule20 | 0.764 | 0.998 | 1.218 | 0.0 | <--- | [SF] |

slika 4.3: izlaz iz čvora Association Rule Learner

U ovim pravilima nailazimo na neke podatke koje smo već videli, npr. ukoliko se koristi icmp protokol procenat greske u konekciji će biti 0. Takodje, nailazi se na neke očekivane rezultate - ako je konekcija završena sa uspehom (flag = SF), procenat greške je 0 i obrnuto. Zanimljivo je u pravilima koju su vezana za uspešne konekcije javlja 511 kao broj konekcija ka istom hostu kao i trenutna konekcija u poslednje dve sekunde. Taj broj uslovljava uspešne konekcije, kao što i uspešne konekcije uslovljavaju njega. Slično kao i broj 511, tako se i ecr_i servis javlja u okviru uspešno izvršenih konekcija. Iz dobijenih pravila se, medjutim, ne može izvući šta uslovljava napade, već su samo dobijene informacije o normalnom saobraćaju.
Zanima nas koja su pravila najbolja po podršci i lift meri, tj. koja pravila imaju najveću vrednost tih "statistika". Da bismo ih izdvojili prvo pronalazimo maksimalnu vrednost podrške i lift mere, uz pomoć čvora GroupBy. U podešavanjima postavljamo da ne uključujemo nijedan atribut i da izdvajamo Maximum za Support i Lift. Kao izlaz iz tog čvora dobijamo:

Table "default" - Rows: 1 | Spec - Columns: 2 | Properties | Flow Variables

| Row ID | D Max*(Support) | D Max*(Lift) |
|---|---|---|
| Row0 | 0.764 | 2.4 |

slika 4.4: maksimalne vrednosti support i lift mere

Nakon toga koristimo čvor Table Row to Variable koji će nam dobijene vrednosti pretvoriti u promenljive kojima možemo da baratamo u čvoru Rule-based Row Filter. U njemu zadajemo da nam se u rezultat uključe samo oni redovi koji sadrže date maksimalne vrednosti podrške i lift mere. Dobijamo:

Table "default" - Rows: 3 | Spec - Columns: 6 | Properties | Flow Variables

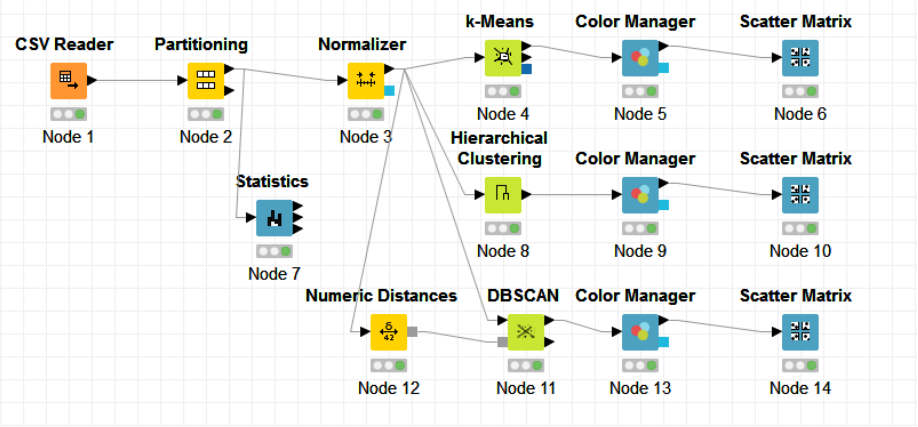
| Row ID | D Support | D Confid... | D Lift | ? Conse... | S implies | (...) Items |
|---|---|---|---|---|---|---|
| rule0 | 0.207 | 0.923 | 2.4 | tcp | <--- | [private] |
| rule19 | 0.764 | 0.933 | 1.218 | SF | <--- | [0.0] |
| rule20 | 0.764 | 0.998 | 1.218 | 0.0 | <--- | [SF] |

slika 4.5: najbolja pravila po support i lift meri

Ako dobijene rezultate želimo da ispišemo u csv fajl moramo prvo da transformišemo podatke u tabeli. Prvo je potrebno kolekcijsku kolonu razbiti na različite kolone. Bez obzira na to sto naša kolekcija u svakom polju sadrži samo jedan element, CSV Writer ne zna to da ispše na izlaz dokle god je kolekcijskog tipa, dakle razbijamo je čvorom Split Collection Column. Primećujemo da je tip kolone Consequent nepoznat, što je logično s obzirom na to da se u njoj nalaze vrednosti različitih tipova. Potrebno je postaviti tip kolone da bi se tabela mogla ispisati na izlaz, za to koristimo čvor Column Rename koji sadrži opciju za promenu tipa kolone. Nakon što smo tip Consequent kolone postavili na String možemo uz pomoć čvora CSV Writer da ispišemo rezultate na izlaz.

# 5 Klasterovanje

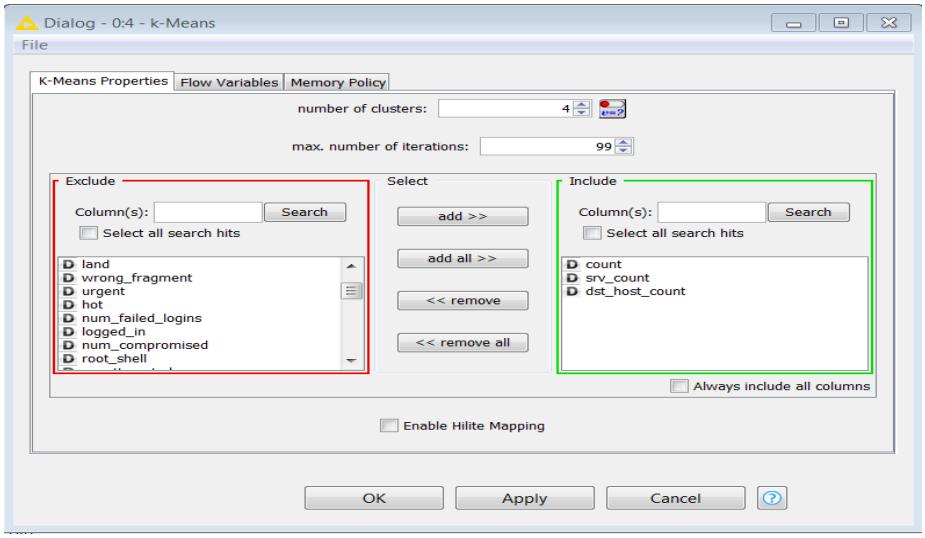
Pre primene algoritama za klasterovanje smo smanjile broj instanci zarad bržeg izvršavanja algoritama. To smo postigle uz pomoć čvora Partitioning kome smo kao parametre zadale da želimo 0.2% skupa i stratifikovano uzorkovanje. S obzirom na to da koristimo algoritme koji koriste Euklidsko rastojanje bilo je potrebno prethodno normalizovati podatke, čvorom Normalizer, da ne bi neki atributi više uticali na proces klasterovanja.



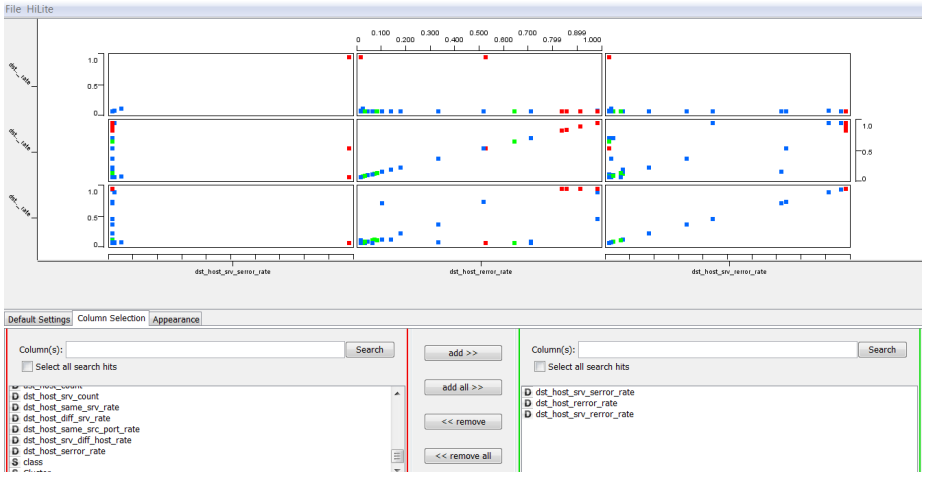

slika 5.1: klasterovanje

Prvo smo primenile čvor k-Means koji primenjuje algoritam k-sredina. Kao atribute po kojima želimo da vršimo klasterovanje smo izabrale count, srv_count, dst_host_count jer su jedine vrednosti koji nisu ili 0 ili 1, ili u opsegu [0,1].

Smatrale smo da će nam to lepše vizuelno prikazati klastere, jer ti atributi sadrže različitije vrednosti. Parametre smo postavile na sledeći način:

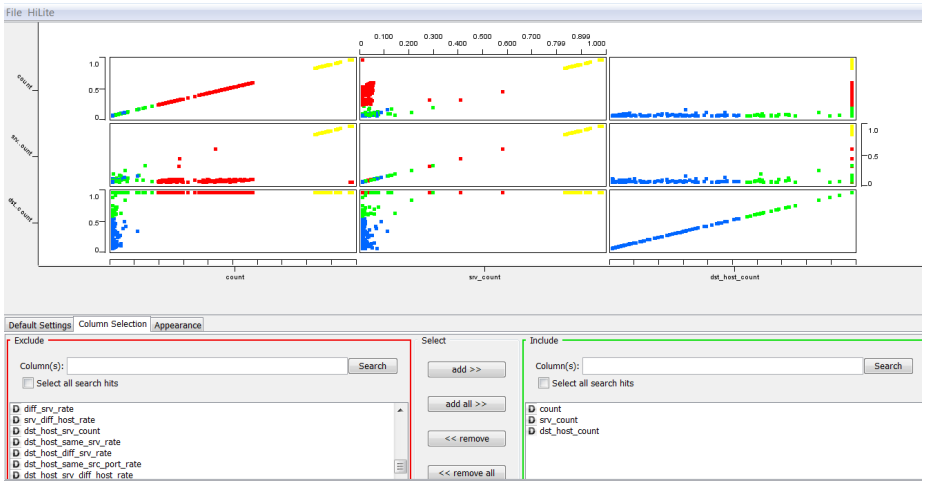



slika 5.2: podešavanje parametara čvora k-Means



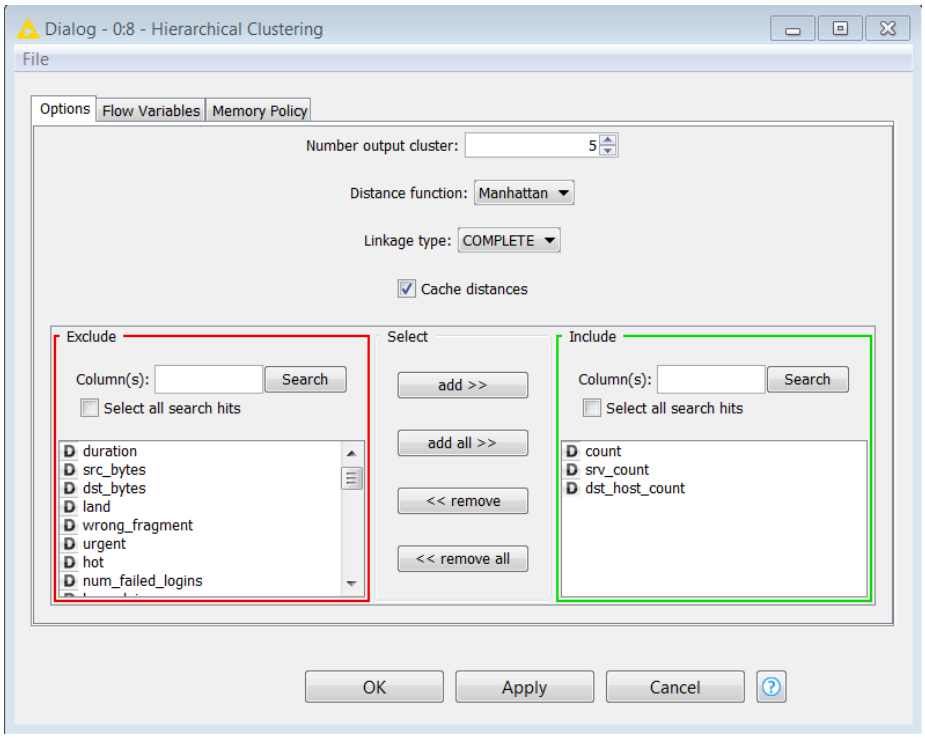

slika 5.3: primer lošijeg modela klasterovanja

Nakon primene ovog algoritma čvorom Scatter Matrix smo prikazale atribute dst_host_srv_serror_rate, dst_host_serror_rate, dst_host_srv_rerror_rate. U ovom slučaju smo dobile relativno loš model. Klasteri su razbacani i ne mogu se izdvojiti neke korisne informacije.

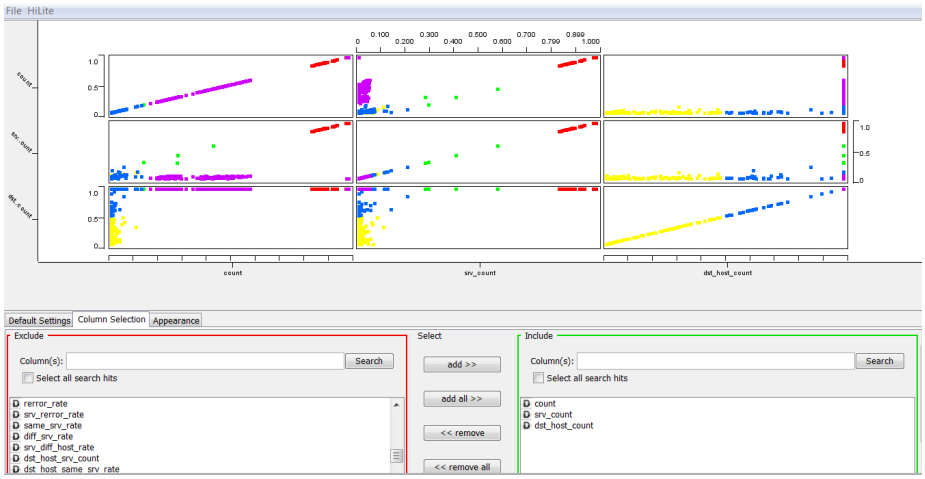



slika 5.4: primer boljeg modela klasterovanja

Kada smo kao atribute koje želimo da prikažemo izabrale upravo atribute koje smo koristile za pravljenje klastera - count, srv_count, dst_host_count - dobile smo bolji model. Na slici se mogu uočiti razdvojene celine koje predstavljaju klastere. Najbolja podela se vidi u donjem levom uglu matrice po atributima count i dst_host_count. Podaci se prostiru duž y-ose i duž prave $x = 1$, jer većina podataka ima vrednosti 0 ili 1, ili eventualno neku vrednost izmedju.
Nakon toga smo primenili hijerarhijsko klasterovanje uz pomoć čvora Hierarchical Clustering. Ponovo smo izabrale iste atribute, zbog gore navedenog razloga. Takodje, kao rastojanje smo izabrale Manhattan, prvenstveno da bismo videle razliku u odnosu na Euklidsko koje je korišćeno u k-sredina, a i zato što nam se čini da obliku naših podataka više odgovara to rastojanje.

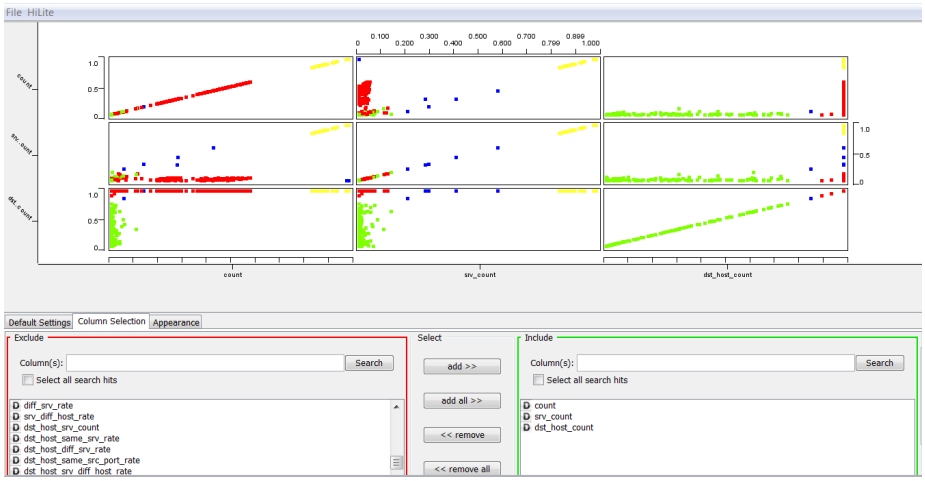



slika 5.5: podešavanje parametara čvora Hierarchical Clistering




slika 5.6: primer modela dobijenog hijerarhijskim klasterovanjem

Dobili smo isti rezultat kao i primenom algoritma k-sredina.
Da bismo primenile algoritam DBSCAN potrebno je prvo zadati model čvorom Numeric Distances. Biramo iste atribute kao i u prethodna dva puta, a kao meru rastojanja smo izabrale Maximum. U čvoru DBSCAN epsilon postavimo na 0.1, a minimalan broj tačaka na 6.

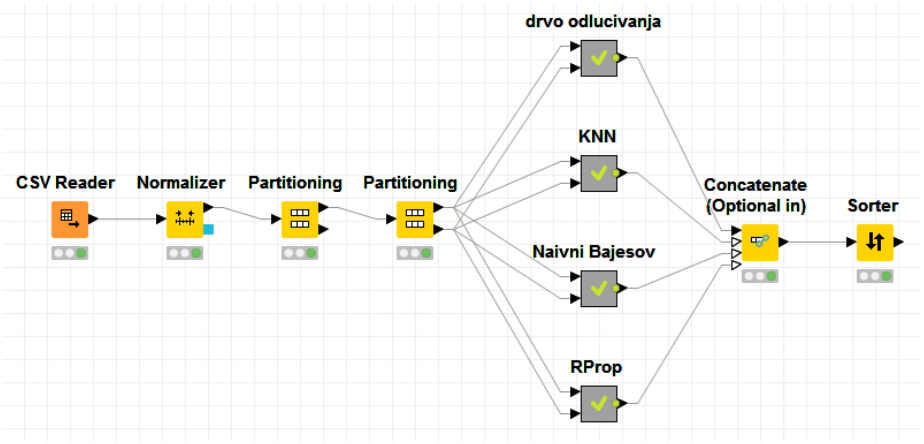



slika 5.7: primer modela dobijenog primenom DBSCAN algoritma

Dobile smo drugačiji model u odnosu na prethodna dva puta, s obzirom na to da je DBSCAN algoritam zasnovan na gustini.
Sva tri algoritma su nam dala dobre modela, koliko je moguće uzevši u obzir prirodu naših podataka.

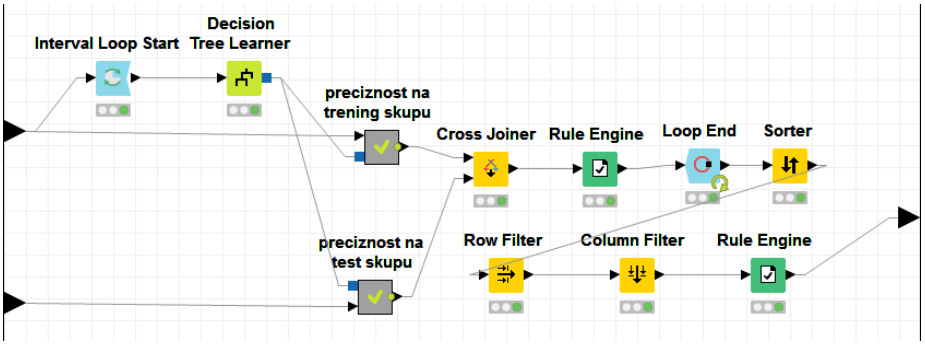
# 6 Klasifikacija

Pre klasifikacije prvo smo učitali podatke, potom smo izdvojili 10% skupa zbog brzine, a potom smo podelili podatke na trening i test skup, gde smo podesili da trening iznosi 70% ukupnog skupa, a test 30%. Bilo je neophodno izvršiti normalizaciju zbog knn algoritma i nakon toga smo mogli da primenimo algoritme za klasifikaciju. Koristile smo četiri algoritma, kojima smo izdvojile preciznost i spojile ih u jednu tabelu da bismo lakše mogle da vidimo koji algoritam je najbolje klasifikovao naše podatke.



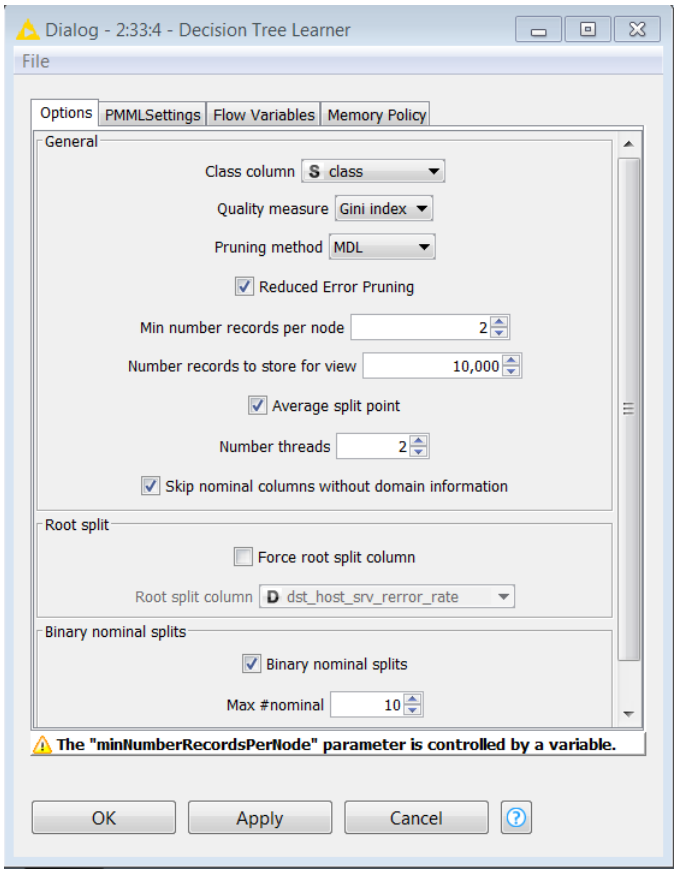

slika 6.1: klasifikacija

Prvi algoritam za klasifikaciju koji primenjujemo je drvo odlučivanja. Ne znamo na koliko treba da postavimo vrednost parametra Min number records per node tako da prvo u petlji (čvor Interval Loop Start) postavljamo da nam se vrednost promenljive menja u svakoj iteraciji od 1 do 10.



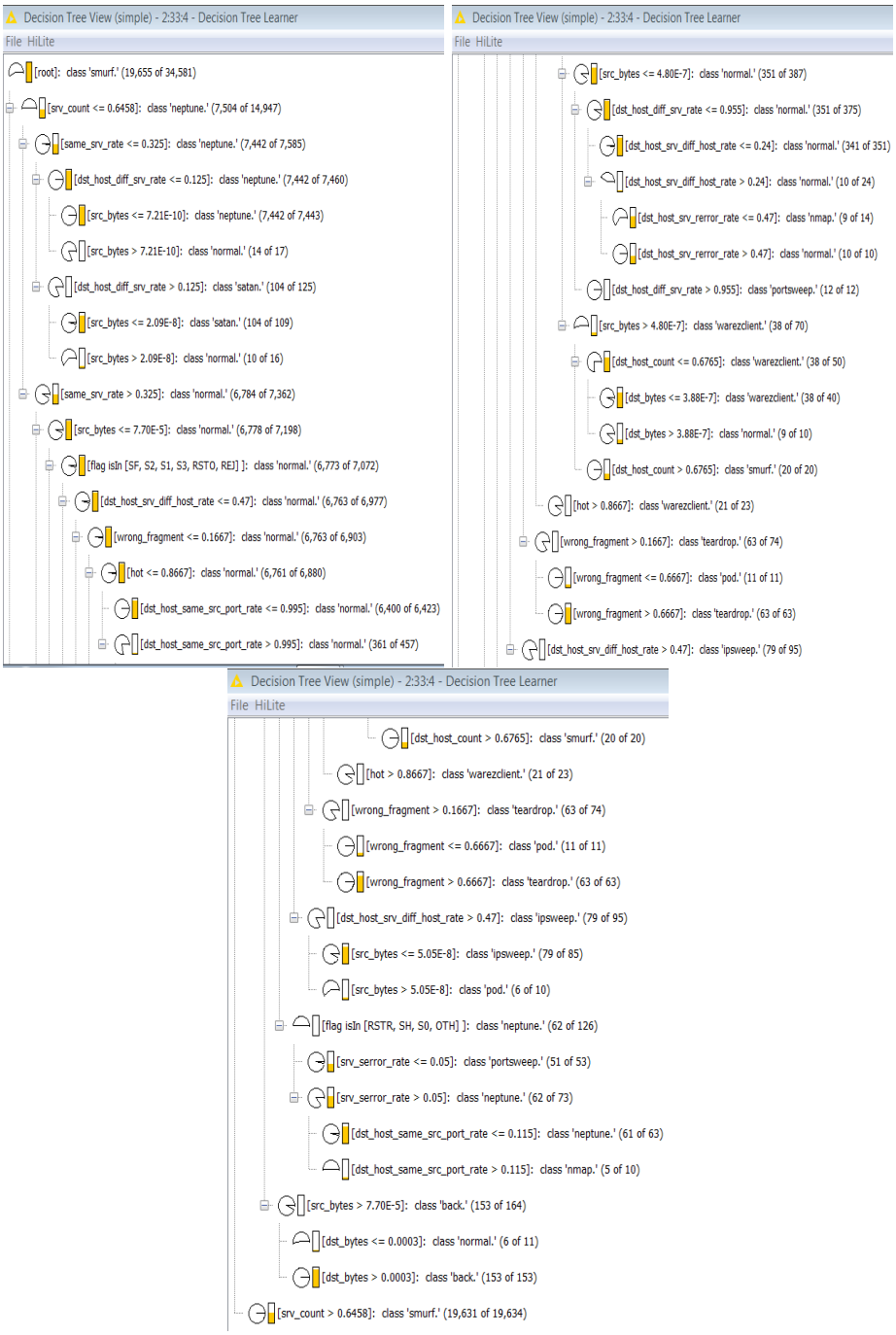

slika 6.2: drvo odlučivanja

Nakon toga pokrećemo čvor Decision Tree Learner kome smo prethodno parametre podesile na sledeći način:




slika 6.3: podešavanje parametara čvora Decision Tree Learner

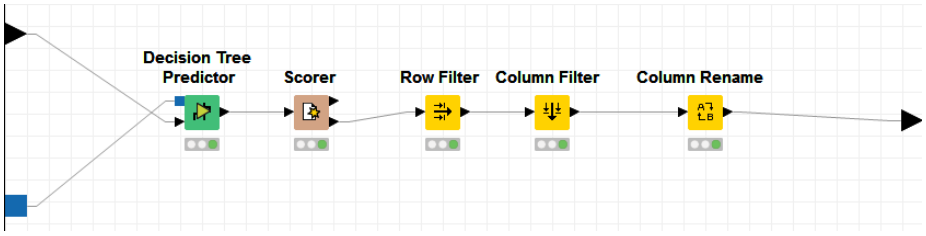
Kao rezultat dobijamo sledeće drvo odlučivanja:



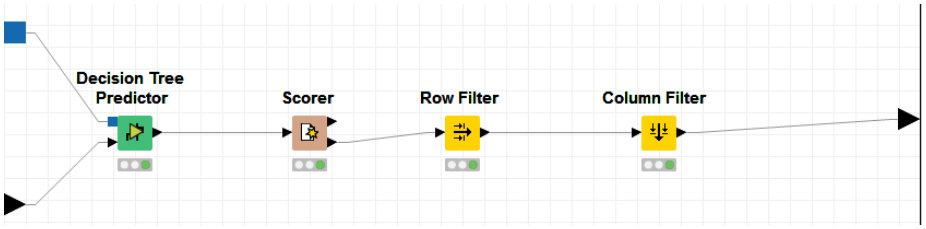

slika 6.4: izlaz iz čvora Decision Tree Learner - drvo odlučivanja

Sledeće što želimo je da dobijeni model primenimo na trening i test podatke

i izdvojimo preciznost modela. U oba slučaja (slika 6.5 i slika 6.6) smo primenili sličan metod. Uz pomoć čvora Decision Tree Predictor smo dobijeni model primenuli na trening (test) podatke, a potom smo Scorer-om izdvojili matricu konfuzije i važne statistike modela. S ozbzirom na to da se u koloni Accuracy nalazi samo jedan red sa vrednošću preciznosti našeg modela, a ostala polja su nedostajuće vrednosti, čvorom Row Filter smo izdvojili samo taj relevantan red. Column Filter-om smo izdvojili samo kolonu Accuracy a potom je Column Rename-om preimenovali u AccuracyTrening da bismo kasnije razlikovali preciznost trening od test skupa. Jedina razlika izmedju ova dva skupa je to što za test skup nismo koristili Column Rename jer nije bilo potrebe, ako smo jednu preciznost nazvali AccuracyTrening drugu ćemo znati da je preciznost test skupa.

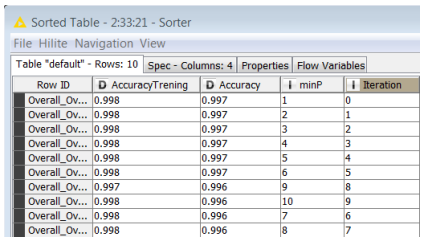



slika 6.5: preciznost na trening skupu




slika 6.6: preciznost na test skupu

Nakon izračunatih preciznosti oba skupa spajamo ih u jednu tabelu Cross Joiner-om, a Rule Engine-om dodajemo kolonu u tabelu koja sadrži vrednost promenljive Min number records per node. Čvor Loop End prikuplja podatke iz svih iteracija petlje i dobijamo preciznost trening i test skupa za sve vrednosti promenljive, a potom čvor Sorter sortira redove opadajuće po preciznosti test i trening skupa, respektivno. Rezultate možemo videti na slici 6.7.


Sorted Table - 2:33:21 - Sorter

File Hilite Navigation View

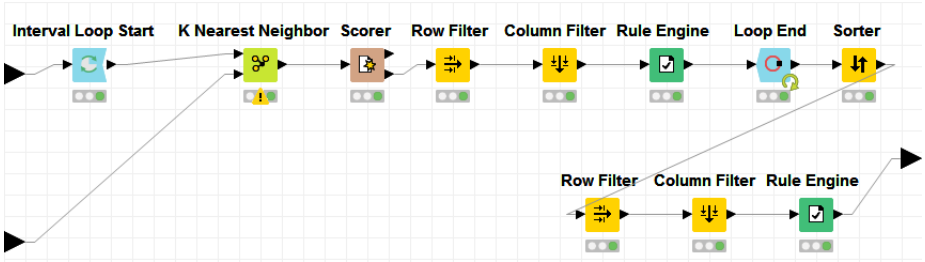
Table "default" - Rows: 10 | Spec - Columns: 4 | Properties | Flow Variables

| Row ID | AccuracyTrening | Accuracy | minP | Iteration |
|---|---|---|---|---|
| Overall_Ov... | 0.998 | 0.997 | 1 | 0 |
| Overall_Ov... | 0.998 | 0.997 | 2 | 1 |
| Overall_Ov... | 0.998 | 0.997 | 3 | 2 |
| Overall_Ov... | 0.998 | 0.997 | 4 | 3 |
| Overall_Ov... | 0.998 | 0.997 | 5 | 4 |
| Overall_Ov... | 0.998 | 0.997 | 6 | 5 |
| Overall_Ov... | 0.997 | 0.996 | 9 | 8 |
| Overall_Ov... | 0.998 | 0.996 | 10 | 9 |
| Overall_Ov... | 0.998 | 0.996 | 7 | 6 |
| Overall_Ov... | 0.998 | 0.996 | 8 | 7 |

slika 6.7: preciznost test i trening skupa u odnosu na vrednost promenljive

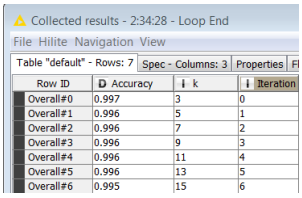
Vidimo da je preciznost najveća za vrednosti 1-6 promenljive Min number records per node.
Potom Row Filter-om izdvajamo prvi red - red sa najboljom preciznošću, Column Filter-om izdvajamo kolonu Accuracy, a Rule Engin-om dodajemo kolonu koja označava model koji smo koristili. Dakle, preciznosti smo dodali model za koji smo je dobili. Izlaz iz ovog čvora predstavlja ulaz u čvor Concatenate (Optional in) u koji ćemo ubacivati i preciznosti ostalih modela kako ih budemo dobijali.

Sledeći algoritam koji primenjujemo je k najbližih suseda (KNN). Procedura je veoma slična kao i kod drveta odlučivanja sa razlikom što smo ovde kroz petlju menjale vrednost k, i to od 3 do 15 sa korakom 2 (da bi vrednost uvek bila neparna). Takodje, K Nearest Neighbors radi samo na test skupu tako da smo ovde na kraju izdvojile samo preciznost test skupa, koja nam je, u suštini, jedina i bitna.




slika 6.8: KNN

Loop End je ponovo prikupio rezultate svih iteracija petlje i Scorer-om smo sortirali tako da na vrhu bude najbolja preciznost.


Collected results - 2:34:28 - Loop End

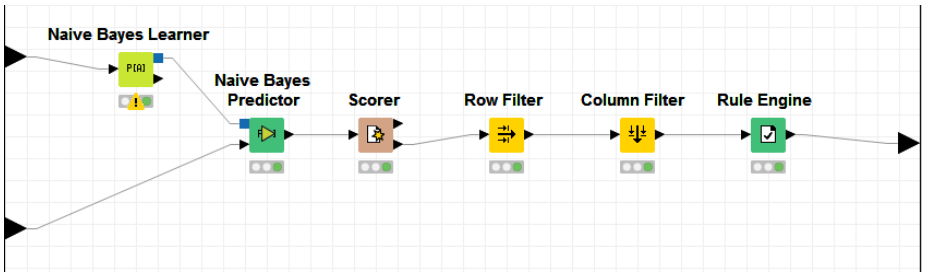
File Hilite Navigation View

Table "default" - Rows: 7 | Spec - Columns: 3 | Properties

| Row ID | Accuracy | k | Iteration |
|---|---|---|---|
| Overall#0 | 0.997 | 3 | 0 |
| Overall#1 | 0.996 | 5 | 1 |
| Overall#2 | 0.996 | 7 | 2 |
| Overall#3 | 0.996 | 9 | 3 |
| Overall#4 | 0.996 | 11 | 4 |
| Overall#5 | 0.996 | 13 | 5 |
| Overall#6 | 0.995 | 15 | 6 |

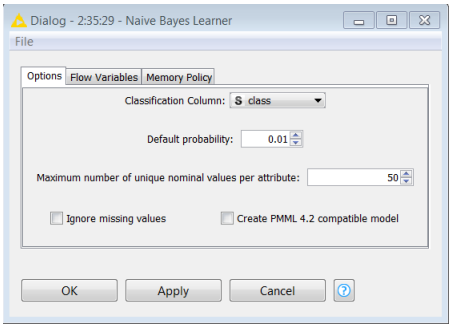
slika 6.9: preciznost test skupa u odnosu na vrednost parametra k

Naredni algoritam je Naivni Bajesov. Procedura je skoro ista kao i kod drveta odlučivanja, sem što nemamo petlju u kojoj menjamo vrednost parametra.



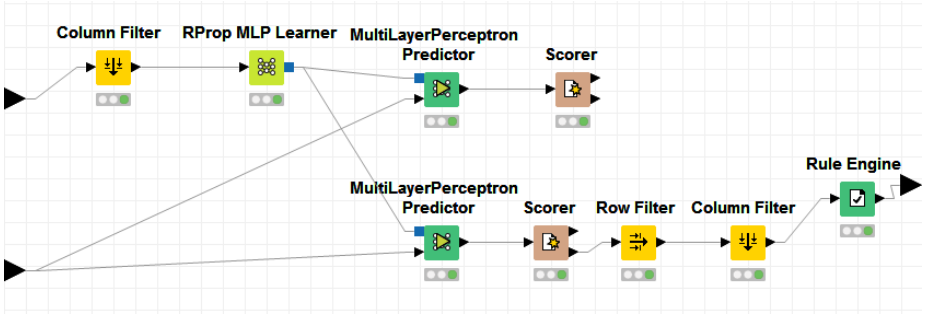

slika 6.10: Naivni Bajesov

Parametre čvora Naive Bayes Learner postavljamo na sledeći način:



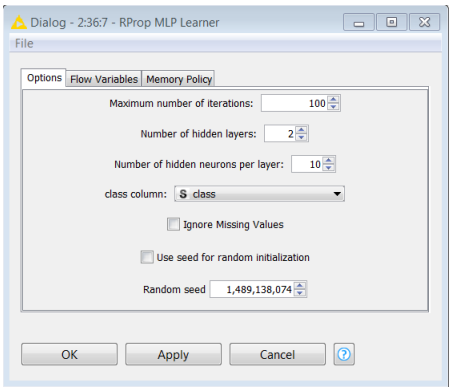

slika 6.11: podešavanje parametara čvora Naive Bayes Learner

Poslednji algoritam klasifikacije koji smo koristile je algoritam neuronskih mreža RProp MLP. S obzirom na to da on radi isključivo sa numeričkim podacima prvo je bilo potrebno izbaciti kategoričke atribute, što smo postigli čvorom Column Filter.




slika 6.12: RProp

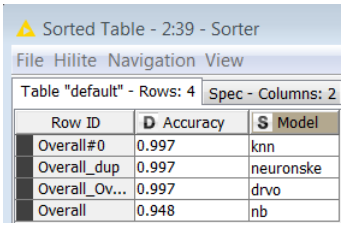
Parametre čvora Naive Bayes Learner smo postavili na sledeći način:




slika 6.13: podešavanje parametara čvora RProp MLP Learner

U ovom slučaju smo model primenili i na trening i na test skup (kao i u slučaju drveta odlučivanja). Preciznost trening skupa je 0.997, ali nam ona nije bitna za dalji tako da smo je samo spomenuli ovde. Što se preciznosti test skupa tiče procedura je u potpunosti ista kao i u prethodnim algoritmima.

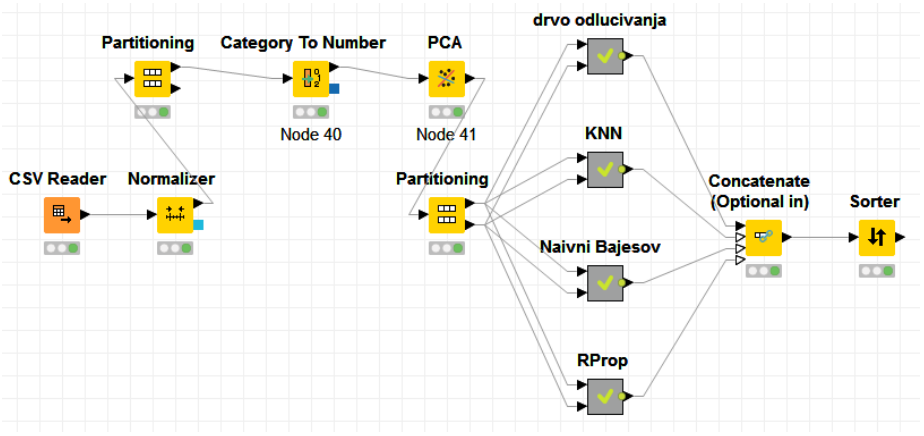
Na kraju je jedino preostalo da spojimo dobijene preciznosti svih modela u jednu tabelu pomoću čvora Concatenate (Optional in) i da ih Sorter-om sortiramo opadajuće po preciznosti. Dobijeni su rezultati:




| Row ID | D Accuracy | S Model |
|---|---|---|
| Overall#0 | 0.997 | knn |
| Overall_dup | 0.997 | neuronske |
| Overall_Ov... | 0.997 | drvo |
| Overall | 0.948 | nb |

slika 6.14: sortirane preciznosti dobijenih modela

Najbolju preciznost dobijamo u algoritmima KNN, RProp MLP, drvo odlučivanja, a neznatno manju u Naivnom Bajesovom. Ovako visok stepen nivo preciznosti nije iznenadjujuće s obzirom na to da imamo ciljnu klasu u tabeli i veliki broj numeričkih atributa sa kojima može da se barata. Medjutim, s obzirom na to da imamo veliki broj atributa, da su neki kategorički, a neki algoritmi ne rade sa kategoričkim atributima, nas je zanimalo da li možemo da neki način da poboljšamo preciznost naših modela. Odlučile smo da upotrebimo algoritam PCA koji smanjuje dimenzionalnost podataka. Skup čvorova kojim smo vršili klasifikaciju smo izmenili na sledeći način:




slika 6.15: klasifikacija, ali sa prethodno smanjenom dimenzionalnošću PCA algoritmom

Sem toga, ništa više nije bilo potrebe menjati, samo smo sve pokrenule ispočetka i na kraju izdvojile rezultate:

Sorted Table - 2:39 - Sorter

File Hilite Navigation View

Table "default" - Rows: 4 | Spec - Columns: 2

| Row ID | D Accuracy | S Model |
|---|---|---|
| Overall#0 | 0.993 | knn |
| Overall_Ov... | 0.992 | drvo |
| Overall_dup | 0.985 | neuronske |
| Overall | 0.98 | nb |

slika 6.16: sortirane preciznosti dobijenih modela uz korišćenje PCA

Preciznost se nije popravila u odnosu na klasifikaciju bez PCA algoritma, izuzev Naivnog Bajesovog algoritma. Medjutim, što se ostalih ălgoritama tiče nije se drastično smanjila, već neznatno malo. Ovaj primer nam zapravo ukazuje na to koliko je PCA algoritam moćan. Uspeo je 41 atribut da svede na 2, a preciznost klasifikacije je ostala veoma velika. Takodje, primetno je mnogo brže izvršavanje algoritama u odnosu na postupak bez PCA.