Univerzitet u Sarajevu
Elektrotehnički Fakultet

Prepoznavanje Oblika i Obrada Slike

Projektni Zadatak br. 2

Studenti:
Muftić *Belma, 1423/17260*
Lemeš *Lamija, 1474/17070*
Krupalija *Ehlimana, 1431/17461*

Odgovorni asistent:
MoE Sumejja Porča

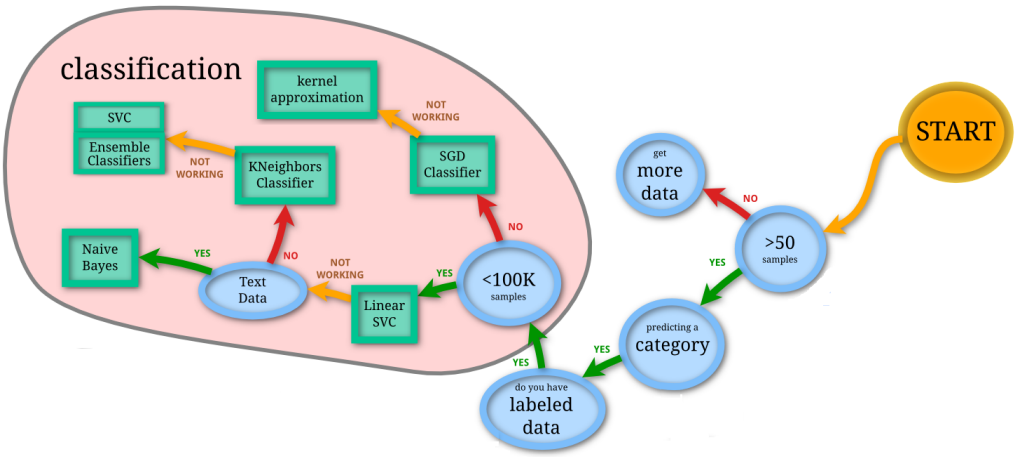
December, 2018

Sadržaj

1 Izbor modela za prepoznavanje

Postoji šest najčešće korištenih klasifikacijskih modela (što je prikazano na Slici 1.):

1. SGD (*Stohastic Gradient Descent*) klasifikacija;
2. *Kernel Approximation* klasifikacija;
3. Linearna SVC (*Support Vector Classification*);
4. SVC (*Support Vector Classification*);
5. KNN (*K-Nearest Neighbors*) klasifikacija;
6. *Naive Bayes* klasifikacija.



Slika 1: Klasifikacijski modeli i vršenje njihovog odabira

Budući da *dataset* ima manje od 100,000 slika, te postoje tekstualni podaci o različitim kategorijama i ROI, koristiti će se ***Naive Bayes*** klasifikacijski model.

2 Izbor i kreiranje deskriptora

Odabran je ***HuMoments*** deskriptor, kako on opisuje oblik željenog objekta, a prepoznavanje ćelije se oslanja prvenstveno na njen oblik. U svrhu toga je kreirana skripta `maskiranje.py` koja izloira samu ćeliju (dakle bez pozadine), jer se *HuMoments* oslanja na analizu slika sa jednim kanalom.

Maskiranje je zasnovano na izdvajanju piksela koji spadaju u određeni rang vrijednosti, nakon čega se radi otvaranje da bi se uklonili "zalutali" pikseli, te se radi i dilatacija da bi se uključile i granice ćelije koje prethodno nisu bile u zadatom rasponu. Nakon maskiranja je pozvana funkcija `HuMoments` koja vraća niz brojeva koji opisuju oblik prethodno dobivene binarne slike. Nakon normalizacije niza, vrijednosti se zapisuju u CSV datoteku, te ako su u pitanju *Train* slike, upisuje se i kojoj klasi slike pripadaju.

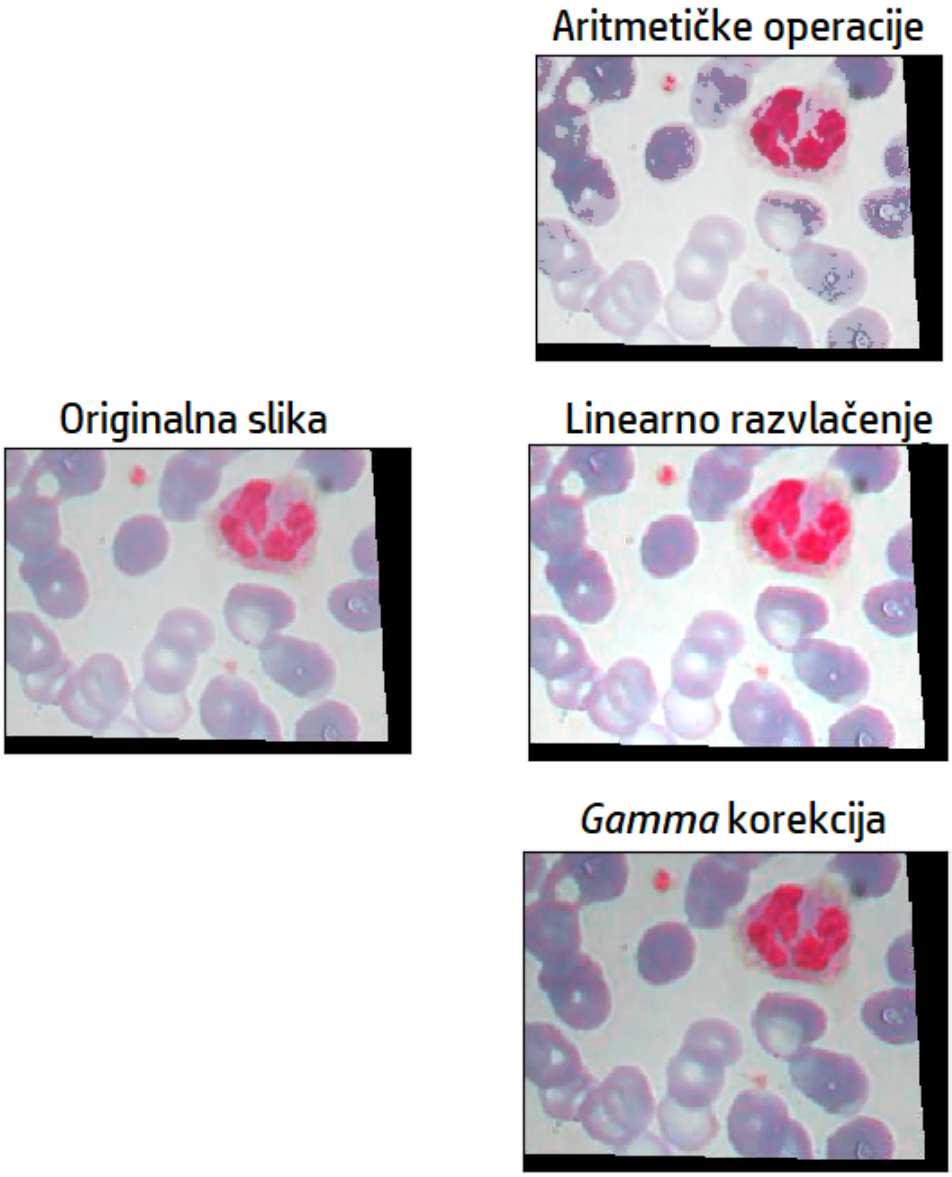
Navedene funkcionalnosti implementirane su na sljedeći način:

```
for i in range(0, 90):
        try:
                Slika = cv2.imread('./Train/CroppedROI{}.jpg'
                                .format(i + 1))
                B = cv2.inRange(Slika, lowerR, upperR)
                if (i > 29):
                        klasa = 2
                if (i > 59):
                        klasa = 3
                print("{}, {}\n".format(i, klasa))
                shape = cv2.getStructuringElement
                                (cv2.MORPH_ELLIPSE, (2, 2))
                nm = cv2.morphologyEx(B, cv2.MORPH_OPEN, shape)
                shape2 = cv2.getStructuringElement
                                (cv2.MORPH_ELLIPSE, (3, 5))
                NewMask = cv2.dilate(nm, shape2)
                B = NewMask
                huMoments = cv2.HuMoments(cv2.moments(B))
                                        .flatten()
                for i in range(0, 7):
                        huMoments[i] = -1 * m.copysign(1.0,
                        huMoments[i]) * m.log10(abs(huMoments[i]))
                f = open("./DeskriptorTrain.csv", "a")
                f.write("{},{},{},{},{},{},{},{}\n"
                .format(huMoments[0],huMoments[1], huMoments[2],
                huMoments[3], huMoments[4], huMoments[5],
                huMoments[6], klasa))
```

3 Izbor metoda poboljšanja

3.1 Poboljšavanje kontrasta

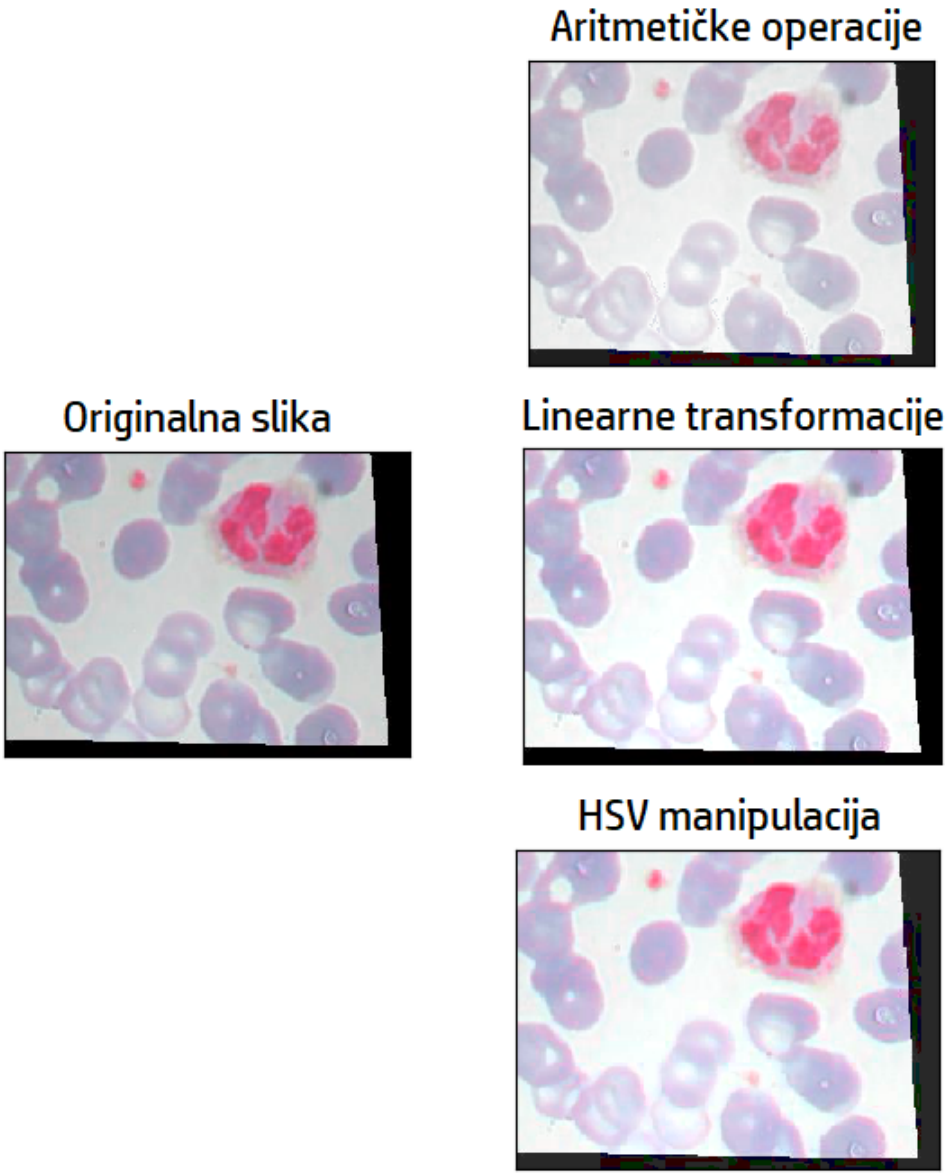
Od tri implementirane metode, najbolje rezultate pokazala je metoda **linearnog razvlačenja kontrasta**. Metoda aritmetičkih operacija previše je narušila strukturu slike, dok metoda *gamma* korekcije nije izvršila dovoljnu modifikaciju kontrasta. Iz ovog razloga metoda linearnog razvlačenja kontrasta koristiti će se za poboljšavanje slika prije njihove dalje obrade.



Slika 2: Rezultati korištenja različitih metoda za poboljšavanje kontrasta

3.2 Poboljšavanje osvjetljenja

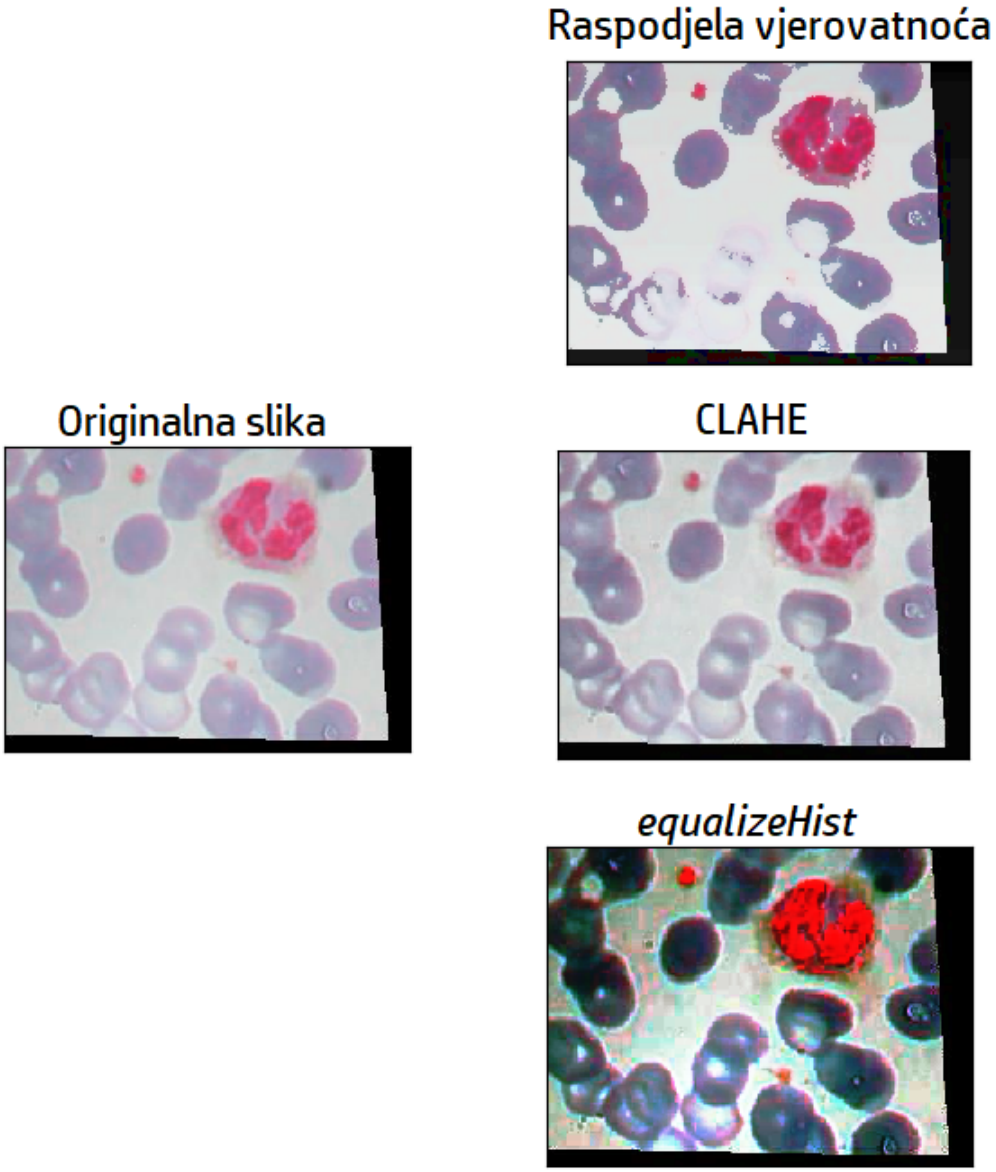
Od tri implementirane metode, najbolje rezultate pokazala je metoda **linearnih transformacija**. Metoda aritmetičkih operacija nije izvršila dovoljnu promjenu osvjetljenja, dok je metoda manipulacije HSV slikom izvršile preveliko povećanje osvjetljenja, te će se iz ovog razloga metoda linearnih transformacija koristiti za poboljšavanje slika prije njihove dalje obrade.



Slika 3: Rezultati korištenja različitih metoda za poboljšavanje osvjetljenja

3.3 Ujednačavanje histograma

Od tri implementirane metode, najbolje rezultate pokazala je metoda **CLAHE**. Metoda raspodjele vjerovatnoća previše je degradirala strukturu slike (pri čemu su neke ćelije potpuno nestale sa slike, što je nedopustivo), dok je metoda `equalizeHist` degradirala strukturu slike u pogledu boje. Iz ovog razloga metoda CLAHE koristiti će se za poboljšavanje slika prije njihove dalje obrade.



Slika 4: Rezultati korištenja različitih metoda za ujednačavanje histograma

4 Testiranje modela

Kreirani deskriptori koriste se za treniranje i testiranje modela, pri čemu veoma važan aspekt predstavlja varijabla `confusionMatrix`, koja sadrži sve relevantne informacije o performansama modela. Na Slici 5. vidljivo je da je tačnost modela oko **76 %**, te da najveću specifičnost pokazuje **klasa 1**, dok najveću senzitivnost pokazuje **klasa 2**. Najmanje vrijednosti specifičnosti i senzitivnosti ima **klasa 3**.

```
Accuracy sa accuracy_score za citav model: 0.7619047619047619


Confusion matrix:
[5, 0, 2]
[0, 7, 0]
[0, 3, 4]


Klasa 1
Acc: 0.7619047619047619
Sen: 0.7142857142857143
Spec: 1.0


Klasa 2
Acc: 0.7619047619047619
Sen: 1.0
Spec: 0.75


Klasa 3
Acc: 0.7619047619047619
Sen: 0.5714285714285714
```

Slika 5: Rezultati pokretanja modela za prepoznavanje

5 Poboljšavanje performansi modela za prepoznavanje

5.1 Izmjena parametara modela

Vrijednosti parametara tačnosti, specifičnosti i senzitivnosti za različite parametre modela prikazane su u sljedećim tabelama:

Priors	***Klasa***	***Accuracy***	***Sensitivity***	***Specificity***
Inicijalni	Klasa 1	0.76	0.71	1.0
	Klasa 2	0.76	1.0	0.75
	Klasa 3	0.76	0.57	0.86

Priors	***Klasa***	***Accuracy***	***Sensitivity***	***Specificity***
0.7	Klasa 1	0.76	0.71	1.0
0.15	Klasa 2	0.76	1.0	0.75
0.15	Klasa 3	0.76	0.57	0.86

Priors	***Klasa***	***Accuracy***	***Sensitivity***	***Specificity***
0.8	Klasa 1	0.81	0.86	1.0
0.1	Klasa 2	0.81	1.0	0.77
0.1	Klasa 3	0.81	0.57	0.93

Priors	***Klasa***	***Accuracy***	***Sensitivity***	***Specificity***
0.9	Klasa 1	0.86	1.0	1.0
0.05	Klasa 2	0.86	1.0	0.79
0.05	Klasa 3	0.86	0.57	1.0

Priors	***Klasa***	***Accuracy***	***Sensitivity***	***Specificity***
0.95	Klasa 1	0.76	1.0	0.82
0.025	Klasa 2	0.76	1.0	0.75
0.025	Klasa 3	0.76	0.29	1.0

Tačnost modela pokazala je tendenciju poboljšavanja s povećanjem vrijednosti prvog parametra za *priors* (povećanjem ostalih parametara tačnost modela je značajno degradirala, te iz tog razloga rezultati za iste nisu ni prikazani, budući da je tačnost modela u tim slučajevima iznosila oko 33 %). Najbolji rezultat dobiven je za vrijednost *priors* jednaku `[0.9, 0.05, 0.05]`, za koju je postignuta tačnost modela od **86 %**. U tom slučaju, 4 od 6 vrijednosti specifičnosti i senzitivnosti jednake su maksimalnoj mogućoj (1.0), a senzitivnost klase 3 iznosi 0.57, dok je specifičnost klase 2 jednaka 0.79.

5.2 Drukčija podjela podataka

5.3 Izbacivanje *outlier*-a slika

Za bolju raspodjelu podataka u postojeće klase implementirana je funkcija `kMeansClustering` koja vrši raspodjelu podataka u *cluster*-e u okviru iterativnog postupka, koji se sastoji iz sljedećih akcija:

1. Inicijalizacija centroida koristeći nasumične vrijednosti;
2. Dodjela svih podataka *cluster*-ima na osnovu najmanje euklidske udaljenosti od centroida;
3. Provjera da li je uslov završavanja ispunjen (konvergencija podataka ili maksimalan broj iteracija).

U skladu s postupkom, funkcija `kMeansClustering` prima sljedeće parametre:

1. `matrix`: podaci koje je potrebno dodijeliti *cluster*-ima;
2. `noOfClasses`: broj *cluster*-a;
3. `outlierPercentage`: ukoliko postotak broja podataka u određenom *cluster*-u bude manji od ovog parametra, svi podaci koji pripadaju tom *cluster*-u biti će obrisani, jer predstavljaju izolovani skup podataka udaljen od ostalih;
4. `maxNoOfIterations`: maksimalni broj iteracija nakon kojeg će se postupak *clustering*-a završiti, ukoliko prethodno ne dođe do konvergencije podataka.

Za testni set podataka koji je prethodno korišten pri treniranju i testiranju izvršena su sljedeća testiranja vršenja *clustering*-a:

`noOfClasses`	`outlierPercentage`	`maxNoOfIterations`	**Rezultujući *cluster*-i**
3	0.1	10	[14, 16, 13]
3	0.1	100	[15, 22, 33]
3	0.1	1000	[19, 21, 20]
3	0.2	1000	[19, 20, 21]
3	0.5	1000	[0, 0, 0]

Iz navedenih rezultata vidljivo je da se s povećanjem broja iteracija dobiva veća pouzdanost *clustering*-a, budući da početna konfiguracija centroida ovisi o nasumičnim vrijednostima. Iz tog razloga testiranja za veći broj klasa (kako bi se demonstriralo brisanje *outlier*-a) biti će izvršeno koristeći vrijednost iteracija koja je veoma velika, da bi broj dodijeljenih elemenata u svim *cluster*-ima bio pretežno identičan za sve različite veličine postotka za *outlier*-e.

`noOfClasses`	`outlierPercentage`	`maxNoOfIterations`	**Rezultujući *cluster*-i**
4	0.1	1000	[14, 10, 20, 16]
4	0.2	1000	[19, 0, 21, 13]
4	0.3	1000	[0, 0, 20, 0]
6	0.1	1000	[0, 14, 14, 0, 16, 10]
6	0.2	1000	[16, 14, 0, 0, 0, 0]
6	0.3	1000	[0, 0, 0, 0, 0, 0]

Iz navedenih rezultata vidljivo je da se s povećanjem postotka koji se koristi kako bi se odredilo da je određeni *cluster* potrebno obrisati, kao posljedica povećanja klasa i sve manjeg broja podataka u pojedinačnim *cluster*-ima, povećava i broj obrisanih *cluster*-a, odnosno *cluster*-a koji se proglašavaju *outlier*-ima.
Nakon vršenja *clustering*-a izvršeno je novo testiranje modela, te su postignute sljedeće performanse:

```
Klasa 1
Acc: 0.42857142857142855
Sen: 0.14285714285714285
Spec: 0.7272727272727273


Klasa 2
Acc: 0.42857142857142855
Sen: 0.7142857142857143
Spec: 0.4


Klasa 3
Acc: 0.42857142857142855
Sen: 0.42857142857142855
Spec: 0.6666666666666666
```

Slika 6: Rezultati pokretanja modela za prepoznavanje nakon vršenja *clustering*-a

Vidljivo je da je tačnost modela pala na čak **42.9 %**, što je i očekivano, budući da je pri *clustering*-u korištena metoda Euklidske distance, koja nije adekvatno mjerilo pripadnosti deskriptora određenim klasama.

Treniranje modela urađeno je i bez vršenja dodjele klasa podacima nakon *clustering*-a, pri čemu je postignuta tačnost modela od **86 %** (koja je identična tačnosti modela bez vršenja *clustering*-a) za sljedeće vrijednosti parametara:

```
noOfClasses = 4 - 12, outlierPercentage = 0.05
```

Za sve druge vrijednosti navedenih parametra postignute su gore performanse (npr. za broj klasa jednak 13 tačnost modela je oko 81 %). Ovo je indikacija da se uklanjanjem određenog dijela podataka na ovaj način ne mogu postići bolje performanse od inicijalnih, budući da podaci ne samo da su ispravno grupisani, već uklanjanje jednog dijela njih za posljedicu ima nedovoljnu istreniranost modela kako bi testiranje bilo uspješno.

```
Klasa 1
Acc: 0.8571428571428571
Sen: 1.0
Spec: 1.0


Klasa 2
Acc: 0.8571428571428571
Sen: 1.0
Spec: 0.7857142857142857


Klasa 3
Acc: 0.8571428571428571
Sen: 0.5714285714285714
Spec: 1.0
```

Slika 7: Rezultati pokretanja modela za prepoznavanje nakon vršenja *clustering*-a bez dodjele klasa