IZVEŠTAJ – PROJEKAT 1, ZADATAK 1

Zorić Ana, 2020/0009

Čubrić Filip, 2020/0077

Prvi korak je učitavanje datoteke, i odabiranje podataka za ulaz i izlaz mreže. Zatim smo podelili ulaz po klasama, i na osnovu toga smo izvršili onehot encoding izlaza (imamo 3 klase te izlaz predstavljamo sa 3 kolone). Zatim smo prikazali ulazne klase na grafiku:

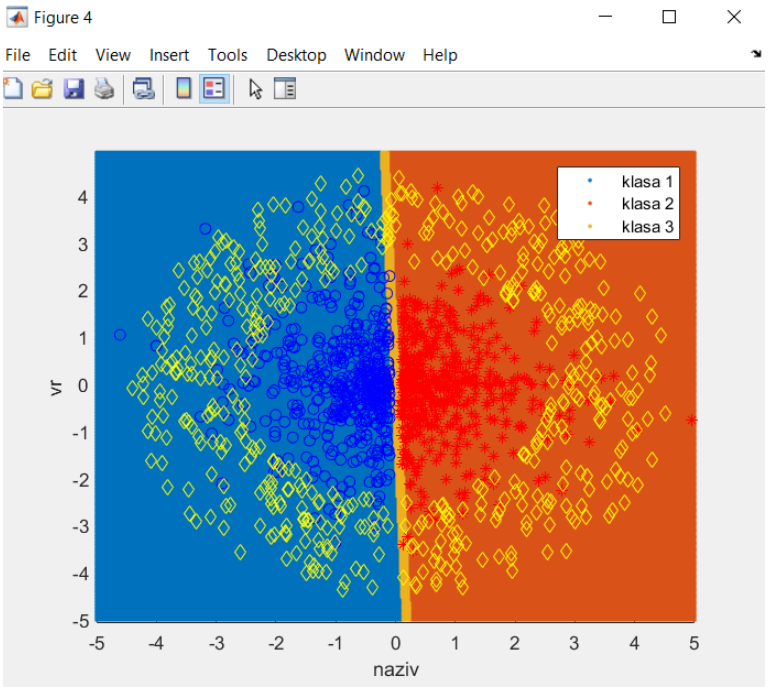
Slika 1: podela podataka na klase

Nakon toga smo nasumično podelili podatke na trening i test skupove. Ovu podelu vršimo jer je loša praksa testirati mrežu na istom skupu koji smo koristili za treniranje. Uvek je bolje podeliti skup ulaza da bi mreža mogla da se testira na nepoznatim ulazima. Sada sledi projektovanje mreže. Da bismo dobili mrežu koja underfittuje, izabrali smo arhitekturu koja je previše mala da bi mreža uspela adekvatno da se obuči (layers= [1,2,3]). Za mrežu koja overfittuje koristili smo arhitekturu koja je prevelikog obima za ovakav problem, te je neuralna mreža “previše dobro naučila” trening skup, a za test skup daje neadekvatne rezultate (layers = [100,100,100]). Za optimalnu mrežu koristili smo arhitekturu [4,4,4,4]. Ove arhitekture smo otprilike podesili na osnovu dobijenih rezultata od nasumično postavljenih arhitektura.

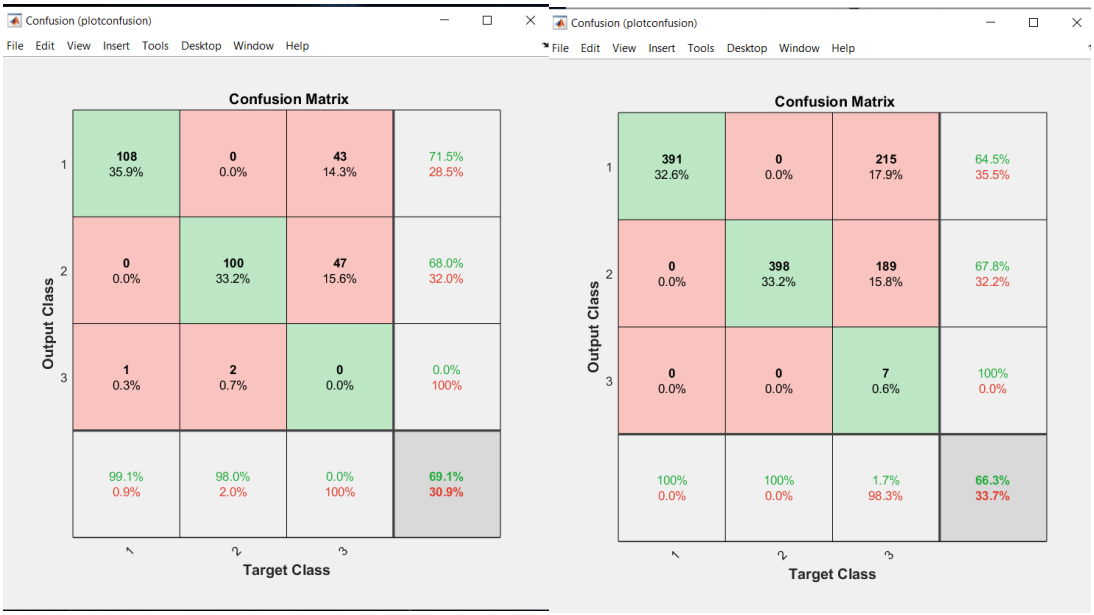
Nakon toga smo obučili mrežu i prikazali rezultate na test i trening skupovima koristeći matrice konfuzije.

Slede prikazane granice odlučivanja i matrice konfuzije za sve tri mreže:

Mreža koja underfittuje:

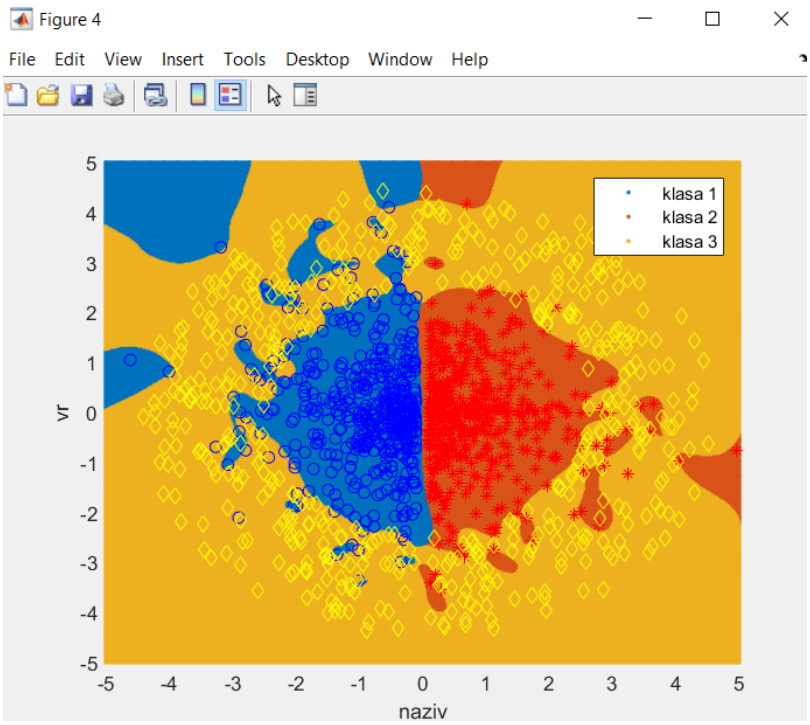


Slika 2a: granica odlučivanja

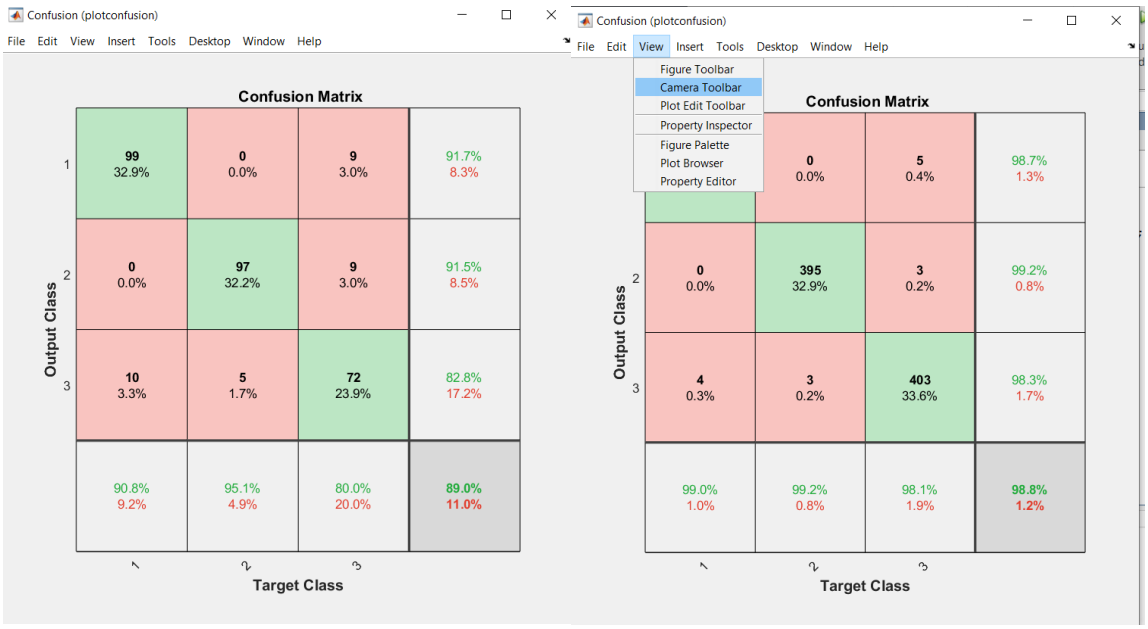


Slika 2b: matrice konfuzije na test skupu (levo) i na trening skupu (desno)

Mreža koja overfittuje:

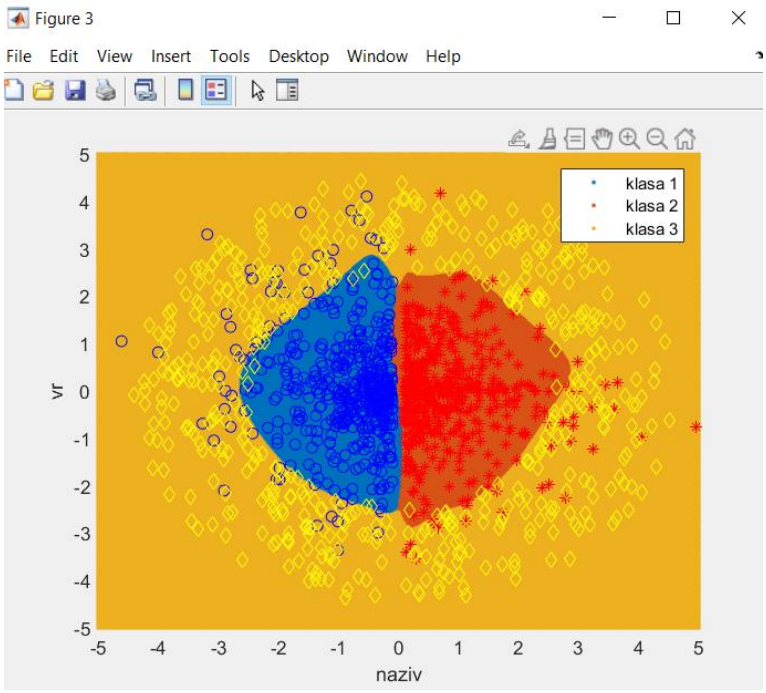


Slika 3a: granica odlučivanja

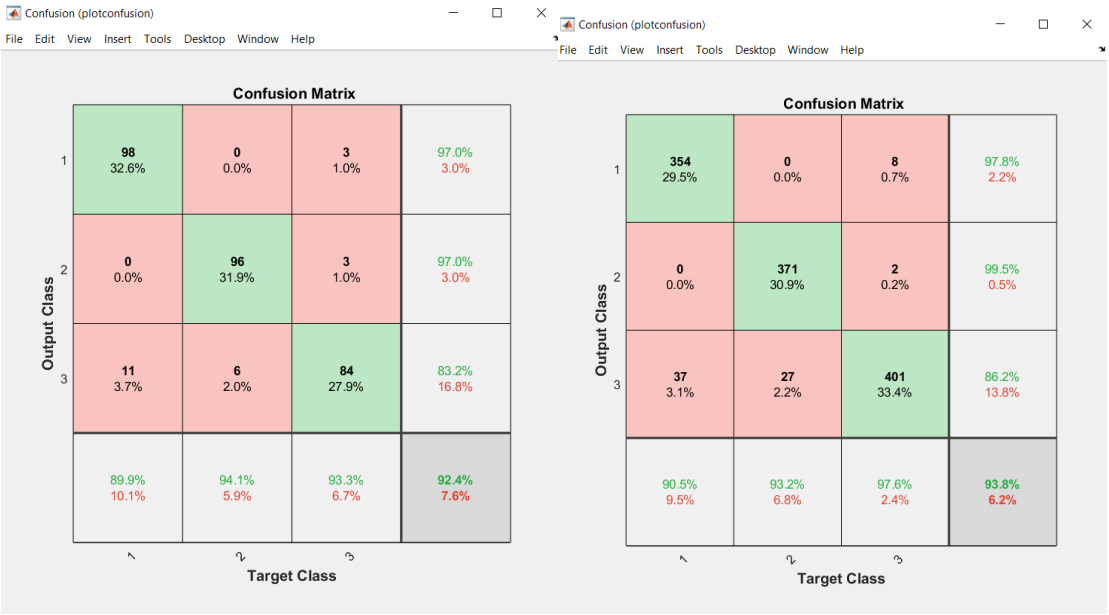


Slika 3b: matrice konfuzije na test skupu (levo) i na trening skupu (desno)

Mreža koja radi optimalno:



Slika 4a: granica odlućivanja



Slika 4b: matrice konfuzije na test skupu (levo) i na trening skupu (desno)

Komentar na performanse:

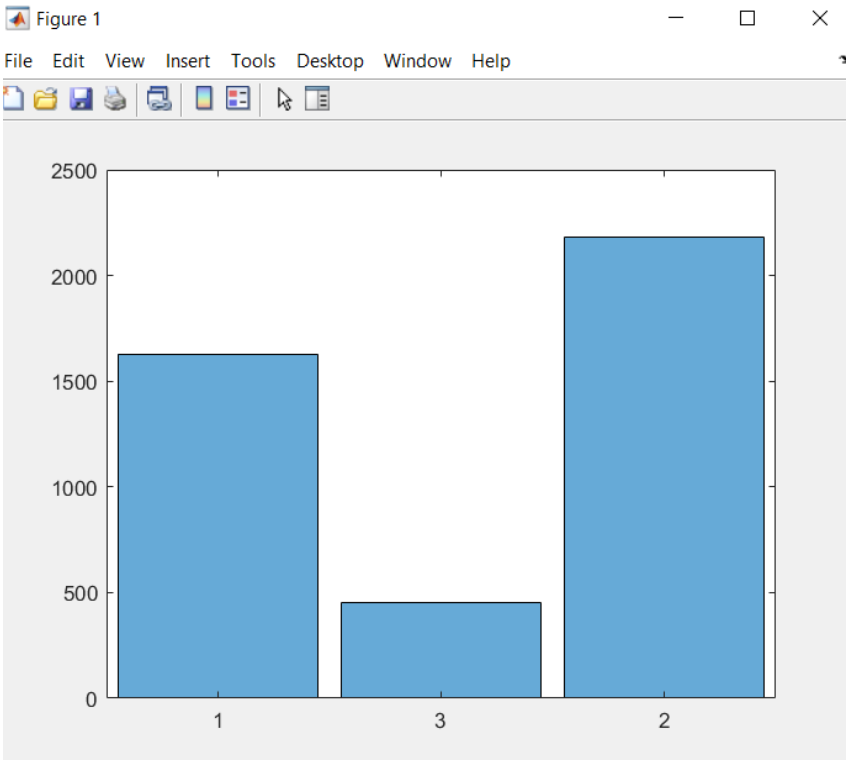
1. Mreža koja underfittuje:
   Granica odlučivanja uopšte nije prilagođena jednoj od klasa. Druge dve su samo linearno odvojene međusobno. Posledica ovoga je da je mreža dobro klasifikovala klasu1 i klasu2, a klasu 3 sa 0% tačnosti.
   Uspešnost mreže na trening skupu je 66%, dok je uspešnost na test skupu 69%
2. Mreža koja overfittuje:
   Granica odlučivanja previše dobro klasifikuje podatke (ukoliko je u ulaznim podacima postojala greška, ova mreža će se takođe prilagoditi tim podacima). Posledica ovoga jeste da je mreža odlično klasifikovala sve podatke iz trening skupa, dok performans opada na skupu za testiranje.
   Uspešnost mreže na trening skupu je 98%, dok je uspešnost na test skupu 89%
3. Optimalna mreža:
   Granica odlučivanja je glatka i nema nepravilnosti (pogrešni podaci će biti zanemareni u dovoljnoj meri). Mreža klasifikuje podatke iz trening i test skupa sa sličnom uspešnošću.
   Uspešnost mreže na trening skupu je 93%, dok je uspešnost na test skupu 92%

Na osnovu dobijenih rezultata zaključujemo da optimalna mreža ima najbolji rezultat na test skupu, što nam je i cilj.

IZVEŠTAJ – PROJEKAT 1, ZADATAK 2

Prvi korak je ponovo učitavanje datoteke sa podacima i nakon toga izdvojiti kolone koje su izlaz i kolone koje predstavljaju ulaz u mrežu.

Potrebno je takođe i transformisati izlaz tako da u matrici izlaza umesto tekstualnih podataka budu numerički podaci koji su pogodni za rad. Sada se nakon podele podataka po klasama, podaci dele na test, validacioni i trening skup. Pošto nije isti broj odbiraka koji pripadaju svakoj klasi, morali smo da napravimo posebno ta tri skupa za svaku od klasa i zatim da napravimo konačni validacioni, test i trening skup spojivši pojedinačne skupove od svake klase. Nakon toga se podaci izmešaju da se ne bi desilo da se prvo obučavaju da prepoznaju jednu klasu, pa zatim drugu i tako dalje.



Slika 5: histogram koji pokazuje količinu podataka po klasi

Hiperpatametri koje smo mi uzeli u obzir su arhitektura mreže, konstanta obučavanje i regularizacija:

```
arhs={[6,5,4],[8,9,10,7],[10,9,3,5,6],[12,15,8,4,3]};
lrs=[0.2,0.06,0.037,0.4];
regs=[0.25,0.45,0.7];
```

Uticaj na performanse:

1. Arhitektura: potrebno je naći arhitekturu koja neće biti ni previše kompleksna ni previše jednostavna, kako se ne bi desilo da tokom obučavanja greška na validacionom skupu

bude mnogo veća od greške na trening skupu, jer se u tom slučaju obučavanje zaustavlja i mreža neće biti dovoljno obučena.

2. Konstanta obučavanja: ako je premala, presporo se približavamo minimumu kriterijumske funkcije, a ukoliko je prevelika može da se desi da oscilujemo oko minimuma i ne dostižemo ga (overshooting)
3. Regularizacija: što je veći ovaj parametar to je veća generalizacija među podacima, tako da ne treba da bude prevelika da mreža ne bi previše generalizovala, a ako bi bila premala desilo bi se da se ponovo greške u ulaznom skupu previše uzimaju u obzir, tako da je opet optimalno rešenje naći 'nešto između'.

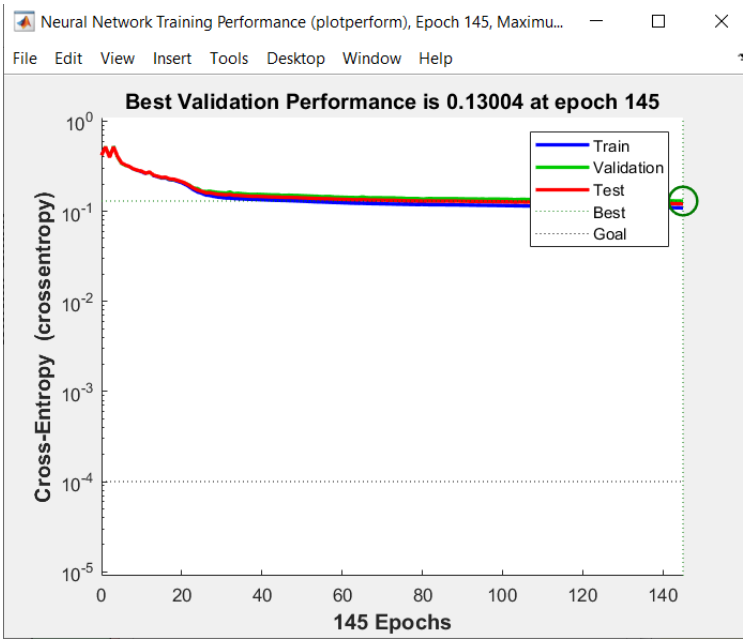
Kada smo odabrali po nekoliko vrednosti za svaki od ovih parametara, napravili smo ugnježdene for petlje koje isprobaju svaku kombinaciju ovih parametara, i ako je performans bolji od najoptimalnijeg u tom trneutku, taj se uzima za najoptimalniji i parametri koji su učestvovali u njemu se čuvaju kao najbolji do tad. Nama se kao najbolja pokazala sledeća kombinacija:

Konstanta obučavanja: 0.2

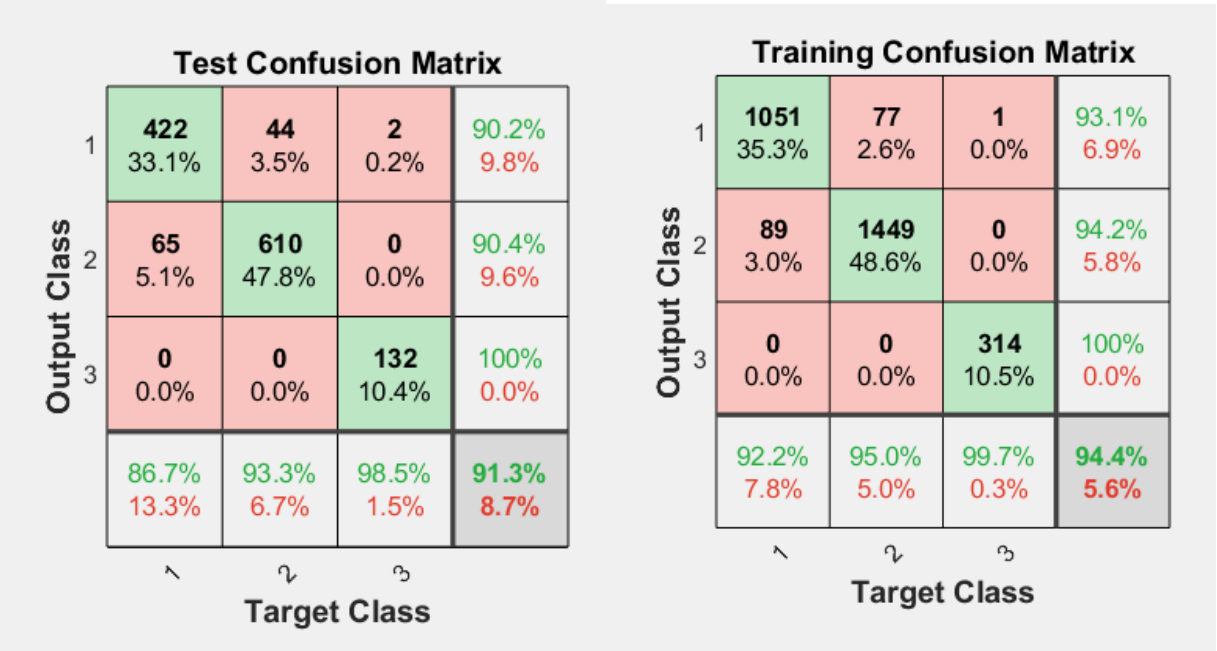
Koeficijent regularizacije: 0.25

Arjhitektura mreže: [10,9,3,5,6]

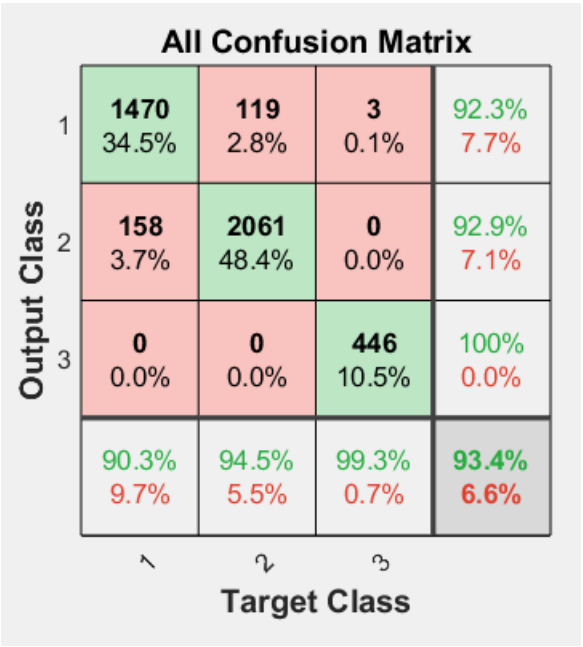
Što se tiče performansi, zadovoljavajuće su za test skup (91.3%).



Slika 6: greška na sva tri skupa u zavisnosti od epohe



Slika 7: matrice konfuzije na test (levo) i trening skupu(desno)



Slika 8: prikazana osetljivost i preciznost za svaku klasu

Osetljivosti se redom za klase 1,2 i 3 nalaze u prva tri polja poslednje vrste matrice. Preciznosti za te klase se redom odozgo na dole nalaze u poslednjoj koloni matrice.