Peta domača naloga

Časovne vrste

Napredno strojno učenje 2023

Dejan Perić

Vpisna številka: 27222015

15. avgust 2023

1 Časovne vrste

1.1 Priprava podatkov

V Python datotekah `RNN_model.py`, `GRU_model.py` in `LSTM_model.py` se nahaja koda za učenje in shranjevanje napovedi v mapo `predictions`. V `grafi_in_napake.ipynb` se nahaja koda za izračun natančnosti ter izris in shranjevanje grafov v mapo `graphs`.

Datoteko `okuzeni.csv` uvozimo v svojo python datoteko z okoljem `Pandas`. Iz vseh občin izberemo stolpce z mestnimi občinami, pri čemer ugotovimo, da manjkajo podatki za 26 dni za mestno občino Slovenj Gradec. Teh 26 dni je zaporednih in so obdani z dnevi z minimalnim številom okužb (ena ali dve okužbi na dan), zato zamenjamo mankajoče vrednosti z 0. Število dni v podatkih je 800, odločimo se, da je prvih 600 dni učna množica, zadnjih 200 pa testna.
V prvih poskusih uporabe določenega modela ugotovimo, da bi verjetno bilo dobro podatke normirati. Za to uporabimo Z-normalizacijo. Tudi iz slike lahko sklepamo, da nam bo vmesni visoki vrh okužb znal delati težave pri učenju modela. Ker načeloma vseh podatkov ne poznamo vnaprej, bomo normalizacijo podatkov izvedli po sekvencah. Funkcijo za sekvence definiramo podobno kot na vajah. Zatem v učni zanki modela vsakič pred izvedbo korako sekvenco normaliziramo:

$$X' := \frac{X - \overline{X}_{sekvence}}{\sigma_{sekvence}}.$$

Izračunamo še normalizirano dejansko vrednost, pri čemer uporabimo povprečje in standardno deviacijo prejšnje sekvence ($\overline{X}_{sekvence}$ in $\sigma_{sekvence}$). V testni fazi strojnega učenja napovedane podatke za prihajajoče dni odnormiramo, da dobimo dejanske vrednosti števila okužb:

$$y' := \sigma_{sekvence} \cdot y + \overline{X}_{sekvence}.$$

V ta namen sta v Pythonu definirani funkciji *normiraj(sez, glavni_sez)* in *odnormiraj(sez, glavni_sez)*, kjer je *sez* seznam, ki se (od)normira, *glavni_sez* pa seznam čigar povprečje in standardna deviacija se vzame za (od)normiranje.

1.2 Implementacija modela in določitev hiperparametrov

Našo Python skripto smo skušali napisati podobno kot se je pisala na vajah. Z uporabo knjižnice `torch.nn` implementiramo modele *RNN*, *GRU* in *LSTM*. Funkcijo *create_sequences()* uporabimo v isti obliki kot na vajah. V učni fazi strojnega učenja imamo isto zanko kot na vajah, koda se tu ne prireja nič temu dejstvu, da imamo napovedi za 7 oz. za 30 dni naprej in ne samo za en dan. Sprememba pa je v testni fazi. Tu se najprej naredi napoved za prvi dan, zatem se to napoved doda na konec sekvence in se potem napove vrednost za drugi dan z uporabe sekvence brez prvega elementa v tej sekvenci. To induktivno ponavljamo, dokler ne pridemo do 7./30. dne.
Vse napovedi zatem zapakiramo v slovar in shranimo kot `.json` datoteko, ena datoteka za izbrane hiperparametre. Hiperparametri so naslednji.

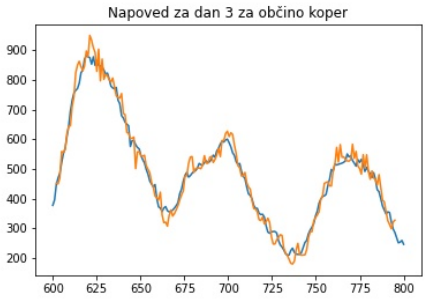
- **epohi** (epochs): Število epohov je 10 in 20.
- **stopnja učenja** (lr): Stopnje učenja so pri nas 0.01, 0.001 in 0.0001.
- **število skritih plasti** (hidden): Število skritih plasti v nevronski mreži je pri nas 16, 32 in 64.
- **dolžina sekvence** (seq_len): Dolžina zaporedja dnevov, iz katerih model napove vrednost za prihodnji dan. Za napoved sedmih dni vnaprej so te vrednosti 5, 10 in 15, pri napovedi 30 dni vnaprej pa so te vrednosti 10, 15 in 20.
- **metoda**: Na izbiro imamo modele RNN, GRU in LSTM.

Poleg hiperparametrov lahko še izberemo število napovednih dni za število okuženih (M = 7, 30) in velikost testne množice (test_size = 200). Za ocenjevanje napake modelov pa bomo uporabljali R2 metriko. Glede na to napako bomo skušali dobiti model in hiperparametre, ki nam bojo najbolje ustrezali.

1.3 Analiza rezultatov za M = 7

Po zagonu kode pogledamo, kateri modeli in hiperparametri so najboljši za našo nevronsko mrežo. Najprej se osredotočimo na napovedi za sedem dni vnaprej. Izračunamo povprečno R2 vrednost vseh občine za vse evaluirane modele. Ugotovimo, da ima najboljši R2 vrednost model shranjen v datoteki `GRU_20_0.01_16_15_7_200_po_sekvencah.json`, R2 vrednost je 0.90398, kar glede na lastnosti R2 metrike označuje zelo dobro napoved modela. Torej tu smo imeli model GRU, 20 epohov, stopnjo učenja 0.01, 16 skritih plasti in dolžino sekvence 15. Izmed občin je najbolšo R2 vrednost dosegla mestna občina Koper (0.94344), najslabšo pa mestna občina Ptuj (0.82340).

Slika 1: Napoved za 1. dan za občino Koper.



Slika 2: Napoved za 3. dan za občino Koper.

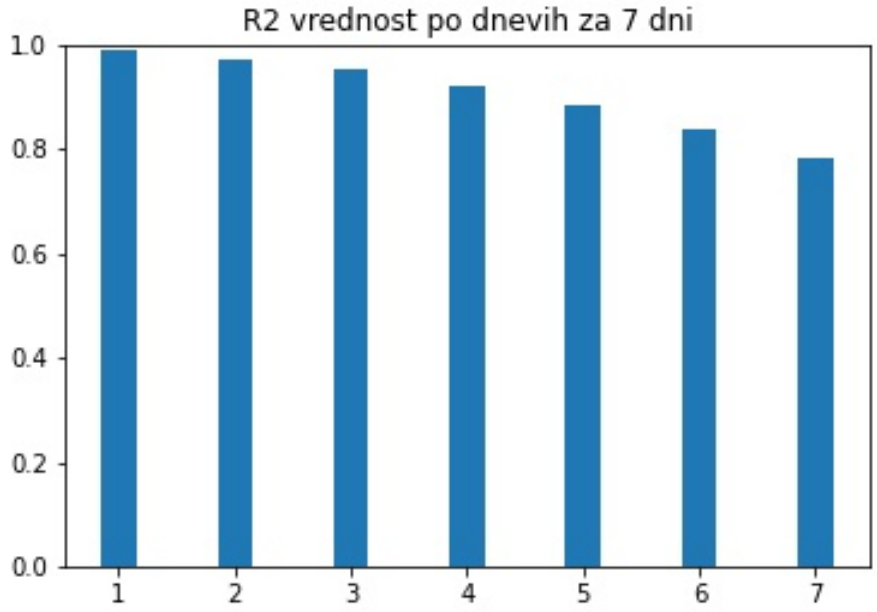
Slika 3: Napoved za 7. dan za občino Koper.

Slika 4: Napoved za 1. dan za občino Ptuj.

Slika 5: Napoved za 3. dan za občino Ptuj.

Slika 6: Napoved za 7. dan za občino Ptuj.

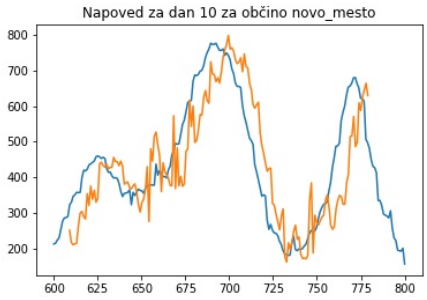
Neka velika razlika med natančnosti napovedi za dve različni občini iz grafov ni vidna. Pričakovano iz grafov opazimo, da se natančnost napovedi z vsakim dnem slabša. Poglejmo, kako se R2 vrednost spreminja glede na oddaljenost dneva. Za vsak dan posebej izračunamo povrečno natančnost. Za prvi dan je povprečna R2 vrednost0.98869, za sedmi pa 0.78063.



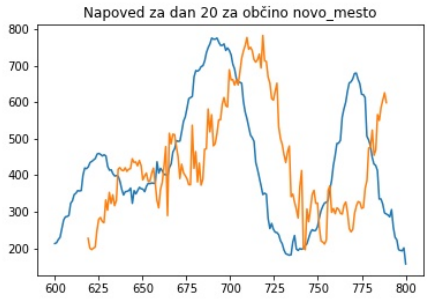
Slika 7: R2 vrednost po dnevih za 7 dni.

1.4 Analiza rezultatov za M = 30

Sedaj pa analizirajmo še modele pri napovedi 30 dni vnaprej. Ugotovimo, da noben naš model se ne prilagaja dobro, saj ima najvišjo R2 vrednost model shranjen v datoteki `GRU_20_0.0001_16_15_30_200_po_sekvencah.json`. R2 vrednost tega modela je -1.2160, kar pomeni zelo slabo prileganje dejanskim vrednostim. Za primer si oglejmo kako izgleda za ta model graf napovedi za 10. in 20. dan za mestno občino Novo mesto, ki ima v tem primeru največjo R2 vrednost (-0.015522).

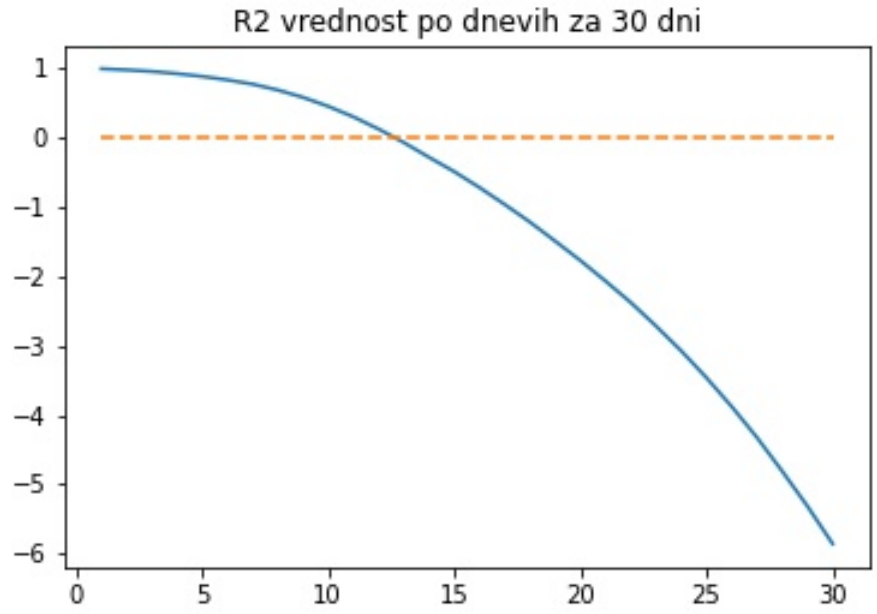


Slika 8: Napoved za 10. dan za občino Novo mesto



Slika 9: Napoved za 20. dan za občino Novo mesto

Opazimo, da zaradi majhne stopnje učenja (lr = 0.001) sta napovedi bolj kot ne zgolj zamik dejanskega grafa za 10 oz. 20 dni v desno. Tako da ugotovimo, da nismo uspeli najti ustreznega modela za napoved število okuženih do 30 dni vnaprej.



Slika 10: R2 vrednost po dnevih za 30 dni.

Vidimo, da model od začetka nekako še v redu napoveduje. Prvi dan je namreč R2 vrednost 0.98981, zadnja R2 vrednost, ki je še nekako sprejemljiva, je 10. dan, tu je vrednost 0.44893. Za 30. dan imamo R2 vrednost -5.85505, ki je preslaba in glede na naklon krivulje na grafu vidimo, da bi z nadaljnimi napovednimi dnevi še strmeje padla.

1.5 Zaključek

Predpostavljali smo, da bojo napovedi za 30 dni vnaprej slabše kot za 7 dni vnaprej. Vendarle pa je ta razlika točnosti precej velika in je dejansko naš model za napoved 30 dni naprej precej neuporaben. Sklepamo, da je za 30-dnevno napoved vnaprej potrebna še kaka informacija več in tudi drugače zasnovan model (model, ki upošteva letni čas, vreme, odloke vlade ipd.).