Tehnike dubokog učenja primjenjene kod analize prijetnji od korona virusa

Vugić, Martin

Undergraduate thesis / Završni rad

2020

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Mechanical Engineering and Naval Architecture / Sveučilište u Zagrebu, Fakultet strojarstva i brodogradnje**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:235:072803

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-04-19**

Repository / Repozitorij:

Repository of Faculty of Mechanical Engineering and Naval Architecture University of Zagreb

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

ZAVRŠNI RAD

Martin Vugić

Zagreb, 2020.

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE

ZAVRŠNI RAD

Mentori:

Doc. dr. sc. Tomislav Stipančić, dipl. ing.

Student:

Martin Vugić

Zagreb, 2020.

Izjavljujem da sam ovaj rad izradio samostalno koristeći znanja stečena tijekom studija i navedenu literaturu.

Zahvaljujem se svojem mentoru doc. dr. sc. Tomislavu Stipančiću na stručnim savjetima, pristupačnosti i pruženoj pomoći pri izradi ovog rada.

Posebnu zahvalu upućujem svojoj obitelji, djevojci i prijateljima na podršci, strpljenju i razumijevanju tijekom mojeg obrazovanja.

Martin Vugić

SVEUČILIŠTE U ZAGREBU
FAKULTET STROJARSTVA I BRODOGRADNJE
Središnje povjerenstvo za završne i diplomske ispite
Povjerenstvo za završne ispite studija strojarstva za smjerove:
proizvodno inženjerstvo, računalno inženjerstvo, industrijsko inženjerstvo i menadžment, inženjerstvo materijala i mehatronika i robotika

Sveučilište u Zagrebu Fakultet strojarstva i brodogradnje	
Datum:	Prilog:
Klasa: 602 - 04 / 20 - 6 / 3	
Ur. broj: 15 - 1703 - 20 -	

ZAVRŠNI ZADATAK

Student: **Martin Vugić** Mat. br.: 0035202291

Naslov rada na hrvatskom jeziku: **Tehnike dubokog učenja primijenjene kod analize prijetnji od korona virusa**

Naslov rada na engleskom jeziku: **Deep learning techniques applied in coronavirus threat analysis**

Opis zadatka:

Svjetska zdravstvena organizacija (WHO) je 31. prosinca 2019. obaviještena o nekoliko slučajeva atipične virusne upale pluća u gradu Wuhanu u kineskoj provinciji Hubei. Pronađeni virus (COVID-19) nije odgovarao nijednom drugom do sada poznatom virusu što je izazvalo zabrinutost ljudi širom svijeta budući da do tada nije bio istražen njegov utjecaj na ljude. U međuvremenu se virus proširio na ostatak svijeta.
Znanstveni podaci o broju oboljelih, umrlih i izliječenih na dnevnoj bazi te informacije o širenju virusa na globalnoj razini mogu dati uvide u različite trendove ponašanja virusa te mogu omogućiti predviđanja o utjecaju virusa na svjetsku populaciju.
U radu je potrebno:

a) identificirati dostupne COVID-19 znanstvene baze podataka,
b) odrediti i objasniti metodologiju duboke analize podataka primjenjivu na ovoj problematici,
c) temeljem definirane metodologije i dostupnih podataka predvidjeti trendove širenja koronavirusa u svijetu,
d) temeljem definirane metodologije i dostupnih podataka predvidjeti broj potvrđenih, umrlih te izliječenih slučajeva vezanih za ovu pandemiju,
e) dati kritički osvrt na dobivene rezultate.

Analiza i evaluacija rezultata mora biti temeljena na programskom jeziku Python.
U radu je potrebno navesti korištenu literaturu i eventualno dobivenu pomoć.

Zadatak zadan:
15. svibnja 2020.

Datum predaje rada:
2. rok (izvanredni): 1. srpnja 2020.
3. rok: 17. rujna 2020.

Predviđeni datumi obrane:
2. rok (izvanredni): 3.7.2020.
3. rok: 21.9. - 25.9.2020.

Zadatak zadao:

Predsjednik Povjerenstva:

Doc. dr. sc. Tomislav Stipančić

Prof. dr. sc. Branko Bauer

SADRŽAJ

SADRŽAJ I
POPIS SLIKA II
POPIS TABLICA III
SAŽETAK IV
SUMMARY V
1. UVOD 1
2. KORONAVIRUSNA BOLEST (COVID-19) 2
2.1. Utjecaj COVID-19-a u svijetu 4
2.2. Utjecaj COVID-19-a u Hrvatskoj 5
3. INDUSTRIJA 4.0 7
3.1. Primjena metoda umjetne inteligencije u zdravstvu 9
4. PROGRAMSKI JEZIK PYTHON 10
4.1. Analiza vremenske serije 10
4.2. Python-ove biblioteke 11
4.2.1. Pandas 11
4.2.2. NumPy 11
4.2.3. SciPy 12
4.2.4. FbProphet 12
4.2.5. Statsmodels 13
4.2.6. Keras 13
5. PROJEKTNI ZADATAK 14
5.1. Predviđanje trenda širenja koronavirusne bolesti (COVID-19) u Python-u pomoću biblioteke *fbprophet* 14
5.2. Predviđanje trenda širenja koronavirusne bolesti (COVID-19) u Python-u pomoću biblioteke *statsmodels* 21
5.2.1. Autoregresivni (AR) model 22
5.2.2. ARIMA model 26
5.2.3. SARIMA model 31
5.2.4. Usporedba AR, ARIMA i SARIMA modela koristeći biblioteku SciPy 34
5.3. Predviđanje trenda širenja koronavirusne bolesti (COVID-19) u Python-u pomoću biblioteke *keras* 35
6. ZAKLJUČAK 39
LITERATURA 40
PRILOZI 42
DODACI 43

POPIS SLIKA

Slika 1.	Industrijske revolucije	7
Slika 2.	Prednosti industrije 4.0	7
Slika 3.	Ciklus industrije 4.0 [6]	8
Slika 4.	Slučajevi koronavirusne bolesti u periodu od 22.1. do 16.3.2020.	15
Slika 5.	Prognoza potvrđenih slučajeva	17
Slika 6.	Potvrđeni slučajevi u odnosu na predviđene slučajeve	18
Slika 7.	Prognoza smrtnih slučajeva	19
Slika 8.	Potvrđeni smrtni slučajevi u odnosu na predviđene smrtne slučajeve	19
Slika 9.	Prognoza izlječenih slučajeva	20
Slika 10.	Potvrđeni izlječeni slučajevi u odnosu na predviđene izlječene slučajeve	20
Slika 11.	Prognoza slučajeva koronavirusne bolesti (*fbprophet* model)	21
Slika 12.	Potvrđeni slučajevi u odnosu na predviđene slučajeve (AR model)	23
Slika 13.	Smrtni slučajevi u odnosu na predviđene smrtne slučajeve (AR model)	24
Slika 14.	Izlječeni slučajevi u odnosu na predviđene izlječene slučajeve (AR model)	25
Slika 15.	Usporedba grafova (AR model)	25
Slika 16.	ACF i PACF dijagram za potvrđene slučajeve	27
Slika 17.	Dijagram potvrđenih slučajeva u odnosu na predviđene slučajeve	28
Slika 18.	ACF i PACF dijagrami za smrtne slučajeve	29
Slika 19.	Dijagram smrtnih slučajeva u odnosu na predviđene smrtne slučajeve (ARIMA model)	29
Slika 20.	ACF i PACF dijagrami za izlječene slučajeve	30
Slika 21.	Dijagram izlječenih slučajeva u odnosu na predviđene izlječene slučajeve (ARIMA model)	30
Slika 22.	Usporedba stvarnih podataka koronavirusne bolesti sa predviđenim podacima ARIMA modela	31
Slika 23.	Dijagram potvrđenih slučajeva u odnosu na predviđene slučajeve (SARIMA model)	32
Slika 24.	Dijagram smrtnih slučajeva u odnosu na predviđene smrtne slučajeve (SARIMA model)	32
Slika 25.	Dijagram izlječenih slučajeva u odnosu na predviđene izlječene slučajeve (SARIMA model)	33
Slika 26.	Usporedba stvarnih i predviđenih podataka (SARIMA model)	33
Slika 27.	Dijagram stvarnog broja slučajeva u odnosu na predviđeni broj slučajeva	37
Slika 28.	Dijagram za stvarne smrtne slučajeve u odnosu na predviđene smrtne slučajeve	38
Slika 29.	Dijagram za stvarne izlječene slučajeve u odnosu na predviđene izlječene slučajeve	38

POPIS TABLICA

Tablica 1. Predviđena i stvarna vrijednost potvrđenih slučajeva u rasponu dana od 11.4. do 15.4.2020. pomoću *fbprophet* modela.....17
Tablica 2. Predviđena i stvarna vrijednost potvrđenih slučajeva u rasponu dana od 11.4. do 15.4.2020. pomoću AR modela 23

SAŽETAK

Kako tehnologija napreduje, a globalna populacija raste, raste i količina podataka i informacija koje svakodnevno stvaramo. Njihovo sortiranje, skupljanje i analiziranje može biti poprilično težak i mukotrpan zadatak. Jedna od najzastupljenijih i najnovijih grana strojnog učenja, duboko učenje (*deep learning)* upravo se bavi tom problematikom. Duboko učenje se temelji na predstavljanju podataka složenim algoritmima na visokom stupnju apstrakcije do kojih se dolazi slijedom stečenih nelinearnih transformacija. Metode dubokog učenja svoju primjenu pronalaze u važnim područjima kao što su: umjetna inteligencija, obrade prirodnog jezika, u bioinformatici i sl., no u okviru ovog rada, koristit ćemo je u analiziranju i predviđanju trendova bolesti zastupljenom korona virusom ili skraćeno COVID-19, uz stečene podatke i statistike u programskom jeziku Python.

Ključne riječi: Python; duboko učenje: COVID-19; predviđanje; vremenske serije

SUMMARY

As technology advances and the world's population grows, so does the amount of data we create every day. Collecting, sorting and analyzing them can be quite a difficult and demanding task. One of the most common and newest branches of machine learning, deep learning, deals with this kind of issues. Deep learning is based on the presentation of complex data algorithms at a high degree of abstraction which are given by learned nonlinear transformations. Deep learning methods find their application in important areas such as artificial intelligence, natural language processing, bioinformatics, etc., but in this paper, we will use it to analyze and predict trends with acquired data and statistics for disease aquired by corona virus, or shortened COVID-19 in the Python programming language.

Key words: Python; deep learning; COVID-19; forecasting; time series

1. UVOD

Koronavirusna bolest (COVID-19) je bolest koja ima ogroman utjecaj na svjetskoj razini, sa više od 25 milijuna zaraženih u više od 190 država. Rastući broj zaraženih teško je zaustaviti jer se virus lako prenosi neposrednom blizinom zaraženog, najčešće kapljičnim putem kao posljedica kašljanja, kihanja, pa čak i kod razgovora s oboljelim. Upravo zbog tih razloga, zaustavljanje kretanja virusa nije lagan posao zbog česte migracije ljudi zbog poslovnih obaveza ili iz osobnih razloga, pa je teško znati kada će pandemija koronavirusa završiti. Ozbiljnost situacije prepoznalo je dosta država uvodeći razne preventivne mjere, poput nošenja maska, određene udaljenosti između ljudi (najčešće 2 metra između pojedinih osoba), zabrana masovnih okupljanja, skraćeno radno vrijeme trgovina i sl. koje određuju stručni stožeri civilne zaštite pojedinih država. Uz poštivanje mjera sigurnosti i odgovornim ponašanjem, možemo uvelike pomoći usporavanju širenja virusa u državi, pa naposljetku i na globalnoj razini. Zato je prognoziranje širenja trenda COVID-19-a jako važan i značajan posao jer tehnikama dubokog učenja i njihovim algoritmima koje smo primjenili u ovom radu, možemo prognozirati daljnje širenje virusa, tj. pandemije koju je uzrokovao virus. Integrirali smo epidemiološke podatke o COVID-19-u za prva tri mjeseca od pojavljivanja prvog zaraženog u Europi 22.1.2020. u logistički model kako bi opisali trend širenja virusa pomoću programskog jezika Python i njegovih biblioteka (eng. *library*) koji se bave analizom podataka. Korištene biblioteke su: *pandas*, *numpy*, *scipy, fbprophet*, *statsmodels* i *keras.* Naravno, kako se ovaj rad temelji na prognozi trenda širenja bez posebnih slučajeva na koja ne možemo utjecati ili očekivati, poput pronalaska cjepiva ili postroženja mjera sigurnosti određenih država koji će naglo zaustaviti širenje trenda, njih ćemo izopćiti iz jednadžbe.

2. KORONAVIRUSNA BOLEST (COVID-19)

Koronavirusna bolest ili skraćeno, COVID-19, virusna je bolest uzrokovana novonastalim koronavirusom SARS-CoV-2. Koronavirusi su virusi koji cirkuliraju među životinjama, ali neki od njih mogu zaraziti i ljude. Nakon što prijeđu sa životinja na čovjeka mogu se prenositi među ljudima. Šišmiši se smatraju prirodnim domaćinima ovih virusa, no velik broj životinja mogu biti nositelji koronavirusa [1].

Novopećena bolest nastala je u gradu Wuhanu, pokrajne Hubei u Kini tijekom prosinca 2019. godine, dok je prvi slučaj zabilježen 17. studenog iste godine. Povodom rapidnog širenja bolesti, ne samo u Kini, već i u ostatku svijeta gdje su najviše nastradala područja Europe, Sjeverne i Južne Amerike i Azije, Svjetska zdravstvena organizacija (eng. *World Health organization;* skraćeno *WHO*) proglasila je pandemiju. Prema stanju od 29. kolovoza 2020. potvrđeno je 25,2 milijuna slučajeva, od toga 17,5 milijuna izliječenih i 850 tisuća mrtvih, no ti brojevi i dalje rastu alarmantnom stopom [2].

Bolest se prenosi velikom lakoćom, što uzrokuje dodatne probleme kod širenja pandemije. Primarno se širi kapljičnim putem kao posljedica kihanja, kašljanja ili razgovora s oboljelim kao i indirektnim putem kontaminiranih ruku izlučevinama zaražene osobe s obzirom da virus može preživjeti i do nekoliko sati na kontaminiranim površinama ili lebdjeći u zraku. Dosta je teško uočiti prve simptome bolesti ako je osoba zaražena što dodatno poboljšava trend širenja pandemije. Najčešći slučaj pojave prvih simptoma bolesti je nakon tri dana, iako je moguće i prije. Trenutni epidemiološki podaci potvrđuju da se virus izuzetno brzo i lako širi među ljudima, te uz procjenu epidemiologa, jedna zaražena osoba u prosjeku može zaraziti i do tri osobe, ako ne i više. Na broj novooboljelih može se značajno utjecati nizom preventivnih mjera, poput pranja ruku, ranom detekcije bolesti, izbjegavanje kontakta sa zaraženim i dr.

Simptomi bolesti ponekad mogu biti slični običnoj gripi, a to su:

- Vručica
- Umor
- Kašalj
- Otežano disanje
- Gubitak okusa
- Gubitak mirisa
- Bolovi u mišićima

a u težim slučajevima javlja se i upala pluća, sepsa, septički šok, akutni sindrom respiratornog distresa koji mogu imati fatalne posljedice kod zaraženih. Osobe koje boluju od težih oblika kroničnih bolesti podložnije su težim oboljenjima. Zbog toga što koronavirusna bolest u početku ima slične simptome kao i obična gripa, znatno je teže dijagnosticirati oboljelog bez stručne pomoći i potrebnih pretraga [1].

Kako bi se smanjila stopa zaraze, moramo se ponašati odgovorno i poštivati mjere zaštite preporučene od strane stručnih ljudi i stožera civile zaštite pojedinih država. Neke od preventivnih mjera koje su zastupljene u Hrvatskoj, pa i drugim državama su:

- Nošenje maske
- Držanje razmaka između ljudi
- Zabrana okupljanja većeg broja ljudi na javnim površinama
- Zabrana okupljanja ljudi u zatvorenim prostorima (do određenog broja)
- Smanjeno radno vrijeme trgovina i drugih uslužnih djelatnosti
- Zabranjeno rukovanje
- Zabrana ili povečanje mjera opreza kod javnih događaja (koncerti, razne priredbe, sportski događaji i sl.)
- Zabrana migracije ljudi unutar države (osim u nužnim slučajevima)
- Zabrana migracije ljudi van države (osim u nužnim slučajevima)
- Mjerenje temperature pri ulasku u određene ustanove (poslovni objekti, fakulteti, medicinske ustanove i sl.)
- Održavanje poslovnih sastanaka preko informacijsko-komunikacijskih davatelja usluga (eng. *Information and communication technologies*; skraćeno *ICT*)

i dr. Također, moramo povećati i osobnu higijenu, pa tako učestalo prati i dezinficirati ruke jer rukama se najlakše prenosi virus zbog toga što virus ulazi u tijelo kroz oči, nos i usta. Radne površine se isto moraju češće dezinficirati, pogotovo u trgovinama, školama, fakultetima, vrtićima i sličnim ustanovama gdje je prolazi veći broj ljudi.

Nažalost, nije moguće sa sigurnošću predvidjeti koliko će pandemija trajati i kako će se ona odvijati. Zbog toga što se radi o relativno novom virusu, nema dovoljno informacija o tome kako ga suzbiti, pa naposljetku i kompletno iskorijeniti. Znanstvenici diljem svijeta pokušavaju pronaći cjepivo, no zbog nedovoljno informacija o virusu i dugoročnog procesa nastanka

cjepiva (eksperimentiranje, testiranje, pa u konačnici masovna proizvodnja i prodaja) nemoguće je odrediti kada će cjepivo biti dostupno za lječenje oboljelih.

2.1. Utjecaj COVID-19-a u svijetu

Bolest se prvi put pojavljuje krajem prosinca 2019. u Wuhanu u kineskoj provinciji Hubei. Prvi slučaj nastaje prenošenjem virusa sa životinje na čovjeka , sumnja se da je to preneseno sa šišmiša na gradskoj tržnici *Huanan Seafood Wholesale Market* u Wuhanu. U prvim rezultatima sekvenciranja genoma virusa, znanstvenici i članovi instituta za virusologiju u Wuhanu pokazuju da je genom virusa 96% identičan koronavirusu kod šišmiša. Iako podatci pokazuju da su šišmiši prenosnici virusa, podatak još uvijek nije potpuno utvrđen. Nakon prvog oboljelog, sve više ljudi u Wuhanu oboljeva i dobiva simptome od tada još neidentificirane bolesti koja je slična običnoj gripi ili upali pluća [2].

Radi sankcioniranja i sprječavanja širenja koronavirusa, kineske su vlasti zatvorile tržnicu, uveli karantenu u Wuhanu i drugim gradovima, ograničili međunarodni i međugradski zračni prijevoz, kao i zabranu drugih oblika javnog prijevoza, te provođenje mjera masovne dezinfekcije javnih prostora i površina. Unatoč tome epidemija se brzo proširila i na druge kineske pokrajine, ali i izvan Kine. 30. siječnja 2020. Svjetska zdravstvena organizacija (eng. *World Health Organization*) proglasila je epidemiju koronavirusa zdravstvenom prijetnjom od međunarodnog značaja zbog brzine širenja epidemije [1, 2].

Virus se ubrzo počeo širiti diljem svijeta, pa su tako 25. siječnja 2020. godine u Francuskoj potvrđeni prvi slučajevi koronavirusa (COVID-19) u Europi. Nakon toga, najveća žarišta u Europi su bile Italija i Španjolska. Također, virus je zahvatio i područja Sjeverne i Južne Amerike gdje je stopa rasta među najvećima u svijetu, pogotovo u SAD-u, Brazilu, Kolumbiji i Peruu [2].

Zbog cijele situacije povezane sa koronavirusom i širenjem pandemije, dolazi do negativnog utjecaja na globalnu ekonomiju koju stručnjaci opisuju kao najgorim padom ekonomije unzad par desetljeća, pa tako i padom BDP-a država zahvaćene koronom [2]. Pokušaji obuzdavanja virusa značajno je usporio ekonomiju svake države, zabrane putovanja i karantene smanjile su prodaju i potražnju dok se iz navedenih razloga zatvaraju tvornice gdje najviše stradavaju obitelji otpuštenih radnika. Svaka grana gospodarstva države pogođena je krizom nastalom od širenja koronavirusa gdje se ističe i turizam koja je jedna od glavnih grana gospodarstva

Hrvatske. Pad tržišta ovakvog raspona implicira veliku vjerojatnost da će u idućih 12 mjeseci doći do recesije, no državne organizacije, kao i svjetske, pokušavaju suzbiti najgore.

2.2. Utjecaj COVID-19-a u Hrvatskoj

Pandemija COVID-19-a proširila se i Hrvatskom 25. veljače 2020. Prvi slučaj oboljelog potvrđen je u Zagrebu nakon boravljenja u talijanskom gradu Milanu, tadašnjim žarištem koronavirusa u Italiji [2, 3]. Ubrzo nakon prvog zaraženog, počelo se zabilježavati sve više i više slučajeva. Zbog preventivnih mjera stručnog stožera civilne zaštite navedenih u poglavlju 2.1., samim početkom pandemije u Hrvatskoj, broj zaraženih je umjereno rastao dok je u ljetnim mjesecima stopa rasta značajno porasla, ali i dalje osrednjim trendom, pa je tako 19.2.2020 zabilježeno više od sto slučajeva, a za samo dva dana, broj se duplicirao. Početkom travnja zabilježeno je više od tisuću slučajeva, a krajem 8. mjeseca zabilježeno je više od deset tisuća zaraženih. Prema istraživanjima na Sveučilištu u Oxfordu, Hrvatska je među državama s najstrožim restrikcijama i mjerama za smanjenje zaraze, pa zbog toga bilježimo blagi rast zaraženih slučajeva i stopu smrtnosti oko 1,2 promila (120 umrlih na 100 tisuća prokuženog stanovništva) iz čega proizlazi da se djelovanje zdravstvenog sustava i stožera civile zaštite pokazalo adekvatnim za razliku od drugih država Europe. U prvim mjesecima pandemije u Hrvatskoj zbog velikih dnevnih migracija stanovništva unutar države, Vlada Republike Hrvatske donijela je odluku o zatvaranju osnovnih i srednjih škola, vrtića i fakultetskih ustanova kao i zakon za samoizolaciju na onaj dio stanovništva koji dolaze u Hrvatsku iz drugih zemalja. Nakon toga kreću faze popuštanja mjera koje se odvijaju u tri faze:

- Prva faza – otvaranje trgovina, gradskog prijevoza, treninzi
- Druga faza – puni rad zdravstvenog sustava, frizera, kozmetičara i sl.
- Treća faza – okupljanje do 10 osoba na jednom mjestu, rad trgovačkih centara, vrtića, razredne nastave

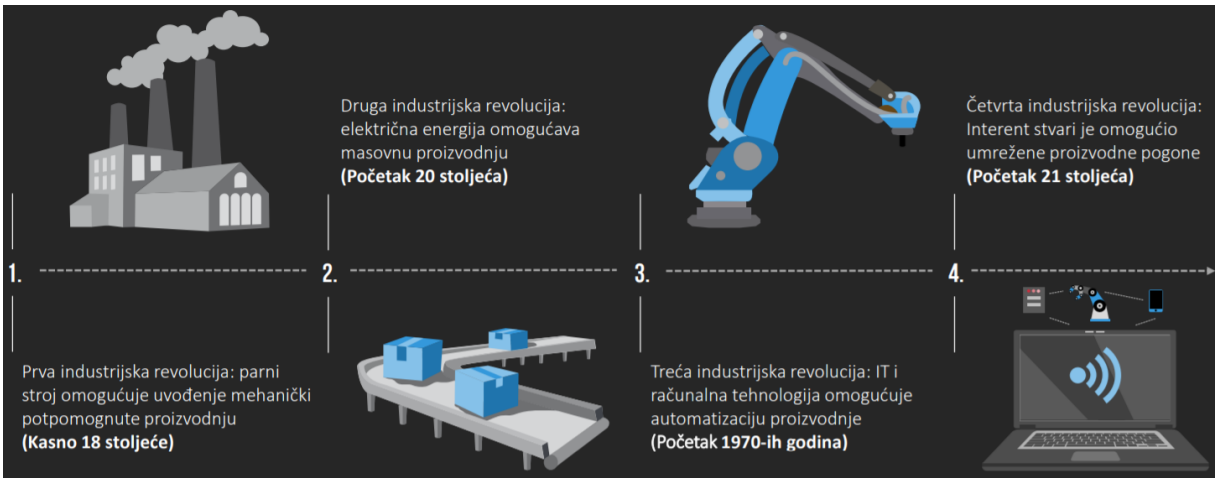
Prva faza počela se odvijati krajem travnja. Omogućuje se rad u svim poslovnim objektima koji obavljaju djelatnost trgovine, osim velikih trgovačkih centara. Također, omogućuje se i nastavak rada u svim objektima koji obavljaju uslužne djelatnosti, osim onih koji moraju ostvarivati bliski kontakt s klijentima (frizeri, kozmetičari i sl.) kao i uvođenje javnog gradskog i prigradskog prijevoza. Omogućuju se rad knjižnica, muzeja, galerija i sl. kao i treninzi sportaša i sportašica. U drugoj fazi omogućuje se rad javnog zdravstvenog sustava u punom opsegu,

privatni zdravstveni sustav, te otvaranje poslovnih objekata koji obavljaju uslužne djelatnosti u kojima se ostvaruje bliski kontakt s klijentima. U trećoj fazi omogućuje se okupljanje do deset osoba na jednom mjestu, uz poštivanje razmaka između ljudi. Omogućuje se rad trgovačkih centara i razredna nastava, kao i održavanje vježbi te praktični rad u malim grupama u visokom obrazovanju. Popuštaju se mjere međužupanijskih prijevoznih linija i linije zračnog prometa. Rad ugostiteljskih objekata isključivo se održava u vanjskim prostorima ili terasama uz poštivanje epidemioloških mjera te se omogućuje rad nacionalnih parkova i parkova prirode[3].

Kako je svjetska ekonomija pogođena krizom zbog koronavirusa, ni Hrvatska nije izuzetak. Hrvatska ekonomija će u ovoj godini zabilježiti pad od čak pet posto, tako da je sasvim izvjestan pad u recesiju, kao i pad BDP-a. Na to upozoravaju i analitičari koji kažu da širenje koronavirusa već sad ima negativan utjecaj na hrvatsko gospodarstvo, koje u značajnoj mjeri ovisi o turizmu. No, osim turizma, i drugi sektori hrvatskog gospodarstva su na udaru, posebno industrija, promet i ugostiteljstvo kao i robna razmjena i tržište rada. Posljedica svega toga bit će manji raspoloživi dohodak, pad povjerenja potrošača i manja osobna potrošnja [4].

3. INDUSTRIJA 4.0

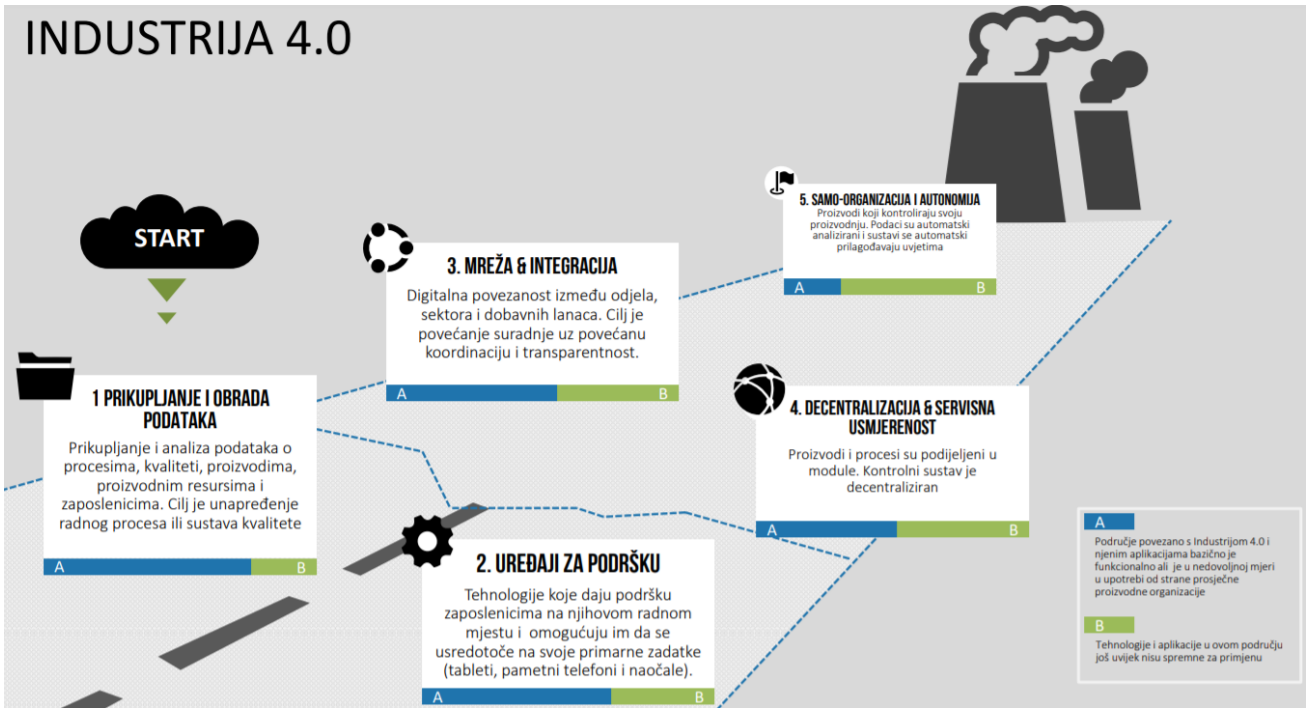
Industrija 4.0 promatra se kao nastavak na prethodne tri industrijske revolucije i odnosi se na transformaciju koja se temelji na digitalizaciji industrije, odnosno inteligentnom umrežavanju strojeva i uređaja pomoću naprednih informacijsko – komunikacijskih tehnologija.



Slika 1. Industrijske revolucije

Primarni zadatak je omogućavanje autonomnog povezivanja među uređajima, analiziranje i pohranjivanje velike količine podataka, autonomno donošenje odluka, praćenje imovine i procesa u stvarnom vremenu, stvaranje dodane vrijednosti te vertikalna i horizontalna integracija. Osnova je integracija informacijskih tehnologija s operativnim aktivnostima, što dovodi do jače proizvodne organizacije. Transformacija poslovnih modela u sklopu Industrije 4.0 rezultira sljedećim poboljšanjima u poduzeću, koja vode do povećanja dobiti, smanjenja troškova, poboljšanih iskustava kupaca i inovacija kao što prikazuje slika 2. [5]

Slika 2. Prednosti industrije 4.0



Slika 3. Ciklus Industrije 4.0 [6]

Industrija 4.0 ovisi o nizu novih i inovativnih tehnologija, pa tako i o razvitku računarskih znanosti te njihovim proučavanjem teoretskih osnova informacije i računanja i njihovim implementacijama i primjenama u računalnim sustavima. Metode umjetne inteligencije uz računalne mreže i pamete senzore važan su dio Industrije 4.0 Ponajviše ćemo se baviti primjenom metoda umjente inteligencije koje se bavi razvojem sposobnosti računala da obavljaju zadaće za koje je potreban neki oblik inteligencije, tj. da se mogu snalaziti u novim prilikama, učiti nove koncepte, donositi zaključke [5, 6]. Funkcije inteligentnoga sustava su

- prikupljanje i obradba informacija,
- interakcija s radnom okolinom, komunikacija s čovjekom ili s drugim inteligentnim sustavima,
- obradba znanja,
- zaključivanje i planiranje.

3.1. Primjena metoda umjetne inteligencije u zdravstvu

Primjena metoda umjetne inteligencije u javnom sektoru tek je u počecima, iako se već pojavljuju projekti u kojima se istražuju mogućnosti primjene umjetne inteligencije. Kako bismo pokrenuli projekte istraživanja u područjima javnog sektora, moramo ostvariti ne samo direktnu primjenu, već i mogućnost stvaranja konkurentnih rješenja. U ovom radu, stavit ćemo naglasak na dio javnog sektora, a to je javno zdravstvo. Umjetna inteligencija danas zauzima važnu ulogu u omogućavanju dionicima u zdravstvu da svoje odluke donose brže i kvalitetnije. Primjenjuje se kod personalizacije liječenja, uvodi se u primjeni automatizacije dijagnostike i postupaka, napredak tehnologije omogućuje automatizaciju i robotizaciju postupaka i njihovu implementaciju, pa tako i stvaranje raznih modela za predviđanje trenda širenja koronavirusne bolesti uz pomoć stečenih podataka. Podatci su jedan od važnijih preduvjeta za primjenu umjetne inteligencije. Količina i kvaliteta podataka određuju mogućnosti koje se mogu primjeniti u umjetnoj inteligenciji, pa pristupom javnih i otvorenih podataka koje objavljuje javni sektor omogućuju kompleksne, ali i bitne koncepte u kojima se jasno razumije prednost umjetne inteligencije. Pristup javnim podacima omogućuje se kontinuirano putem programa upravljanja podataka kroz koje je potrebno objavljivati sve dostupne konačne i kronološke setove podataka. Potrebno je definirati i usvojiti programe upravljanja podacima na nivou organizacije, što uključuje obavezu stvaranja kataloga podataka i meta-podataka za svaki proces i programsko rješenje te ih objaviti putem portala otvorenih podataka [7].

4. PROGRAMSKI JEZIK PYTHON

Python je programski jezik visoke razine i opće namjene kojeg je osmislio i napravio Guido van Rossum 1991. godine. Njegove jezične konstrukcije i objektno orijentirani pristup pomažu programerima pri konstrukciji logičnog koda za male projekte i projekte visokih razina. Ugrađene podatkovne strukture na visokoj razini, u kombinaciji s dinamičkim tipkanjem i dinamičkim povezivanjem, čine ga vrlo poželjnim programskim jezikom za brzi razvoj aplikacija, kao i za upotrebu kod skriptiranja ili za povezivanje postojećih komponenata. Relativno nove grane kao što su znanost o podacima, umjetna inteligencija, robotika i analitika podataka, zajedno s tradicionalnim zanimanjima poput web razvoja i znanstvenog istraživanja, prihvaćaju Python kao odabrani programski jezik. Također, Python se može koristiti za brzo prototipiranje ili za razvoj softvera spremnog za proizvodnju. Budući da ne postoji korak kompilacije (eng. *Compilation step*), ciklus uređivanja, ispitivanja i otklanjanja pogrešaka nevjerojatno je brz. Otklanjanje pogrešaka u Python-u je jako jednostavno, greška ili loš unos nikada neće uzrokovati grešku u segmentaciji. Umjesto toga, kada interpreter otkrije pogrešku, pokreće iznimku. Kada program ne uhvati iznimku, tumač ispisuje *stack-trace*. Program za ispravljanje pogrešaka na izvornoj razini omogućuje inspekciju lokalnih i globalnih varijabli, procjenu proizvoljnih izraza, postavljanje točaka prekida, prelazak koda kroz liniju odjednom itd. Koristi se u raznim operativnim sustavima poput Windows-a, Mac-a, Linux-a, Raspberry pi-ju i sl. U ovom radu, koristimo najažurniju verziju Python-a, verziju 3.8 koja je izdana 14. listopada 2019. u Windows operacijskom sustavu [8, 9].

4.1. Analiza vremenske serije

Analiza vremenskih serija služi za analizu podataka vremenskih serija za izdvajanje značajnih podataka ili generiranje drugih značajnih spoznaja koje se primjenjuju u određenim situacijama. Općenito, podaci vremenskih serija su sklop promatranja pohranjenih u vremenskom redoslijedu.

Analiza vremenskih serija pomaže u razumijevanju vremenskih nalaza skupa metričkih podatkovnih točaka. Tehnike predviđanja vremenskih serija mogu se koristiti za razne stvari u poslovnom svijetu što se tiće ekonomije i ostalih grana gospodarstva, no u ovom radu će se koristiti isključivo kod prognoze trenda širenja koronavirusa u svijetu. Osnovni cilj analize

vremenskih serija je obično odrediti model koji opisuje uzorak vremenske serije da ga se može koristiti za predviđanje.

Tehnike prognoziranja klasičnih vremenskih serija temelje se na statističkim modelima koji oduzimaju puno vremena pri podešavanje modela i podataka. Moraju se podesiti parametri modela s obzirom na specifični problem kada model predviđanja ne funkcionira kako treba. Kalibriranje i podešavanje tih modela zahtijeva temeljito razumijevanje načina funkcioniranja modela temeljnih vremenskih serija [10].

4.2. Python-ove biblioteke

4.2.1. Pandas

Python pandas je biblioteka (eng. *library*) otvorenog koda (eng. *open source*) stvorena za programski jezik Python. Omogućuje jednostavno korištenje, a služi za analizu i vizualizaciju podataka. Nudi rad sa podatkovnim strukturama i operacijama za manipuliranje numeričkim tablicama i vremenskim nizovima. Alat se koristi u širokom rasponu, uključujući akademske i komercijalne domene, financije, ekonomiju, statistiku i analitiku . Naziv Pandas izveden je iz „panel data“ ekonometrijskog pojma za skupove podataka koji uključuju promatranja kroz više vremenskih razdoblja. Pandas biblioteka koriste podatke pisane u CSV (*Comma-seperated values*) obliku koji je korišten u ovom radu i stvara Python objekt s redovima i stupcima, podatkovni okvir (eng. *data frame*), koji izgleda vrlo slično tablici u statističkim softverima poput Excel-a ili SPSS-a [11].

4.2.2. NumPy

Numerical Python (skraćeno *NumPy*) je Python biblioteka otvorenog koda (eng. *open source*) koja se koristi za rad s nizovima (eng. *arrays*). Također ima funkcije za rad u domeni linearne algebre, Fourierove transformacije i matrica. Zašto koristimo NumPy? U Pythonu imamo popise koji služe u svrhu nizova, ali se sporo obrađuju. NumPy želi pružiti objekt niza koji je i do 50 puta brži od tradicionalnog popisa Python. Nizovi se vrlo često koriste u znanosti o podacima, gdje su brzina i resursi vrlo važni što je vrlo korisno u ovom radu. Također, NumPy nizovi pohranjeni su na jednom neprekidnom mjestu u memoriji, za razliku od popisa, tako da

im procesi mogu vrlo učinkovito pristupiti i njima upravljati. Takvo se ponašanje u računalnoj znanosti naziva lokalitet referenci. To je glavni razlog zašto je NumPy brži od popisa. Također je optimiziran za rad s najnovijim CPU arhitekturama [11].

4.2.3. SciPy

SciPy je besplatna open source Python biblioteka koja se koristi za znanstveno i tehničko računanje. SciPy sadrži module za optimizaciju, linearnu algebru, integraciju, interpolaciju, posebne funkcije, FFT (eng. *Fast Fourier transform*), obradu signala i slika i druge zadatke uobičajene u znanosti i inženjerstvu. SciPy se nadovezuje na NumPy biblioteku i stoga uključuje alate poput Matplotliba i Pandas biblioteke te širi skup znanstvenih računalnih biblioteka [11].

4.2.4. Fbprophet

Tehnike strojnog učenja za algoritme predviđanja grana su računalne znanosti koja se obučava na temelju povijesnih podataka, poput umjetnih neuronskih mreža, dubokog učenja, stabala odlučivanja i Bayesovih mreža. Ideja algoritma je odabrati prikladan model prema karakteristikama povijesnih podataka i koristiti ga za predviđanje budućih rezultata promatranja. Ovu smo tehniku primijenili na predviđanje COVID-19 u stvarnom svijetu. Prophet je *open source* software Facebooka za predviđanje vremenskih serija zasnovan na aditivnom modelu koji je otvoren za javnost 2017. Nelinearni Prophet-ovi trendovi opremljeni su godišnjim, tjednim i dnevnim sezonalnostima, kao i blagdanima. Prophet funckija može ne samo predvidjeti budućnost, već i ispuniti nedostajuće vrijednosti i otkriti anomalije. U Prophet-u, model predviđanja sastoji se od jednadžbe superpozicije (1)

$$\boldsymbol{y}(\boldsymbol{t}) = \boldsymbol{g}(\boldsymbol{t}) + \boldsymbol{s}(\boldsymbol{t}) + \boldsymbol{h}(\boldsymbol{t}) + \boldsymbol{\epsilon t} \quad \textbf{(1)}$$

gdje su varijable

- *g(t)* - funkcija trenda koja se koristi za analizu rasta za modeliranje neperiodičnih promjena u vremenskim serijama

- *s(t)* - funkcija koja odražava periodične promjene, poput periodičnosti dana, tjedna ili godine
- *h(t)* - utjecaj praznika ili dana s nepravilnim rasporedima
- *εt* - je pojam propuštenog razmatranja učinka pogreške modela

Koristeći vrijeme kao glavni objekt, korisnik pokušava spojiti linearne i nelinearne funkcije vremena kao komponente. Korisnik opisuje problem predviđanja kao vježbu prilagođavanja krivulji, umjesto da eksplicitno gleda na vremensku ovisnost svakog promatranja unutar vremenske serije [12].

4.2.5. Statsmodels

Statsmodels je Python-ova biblioteka koji omogućuje korisnicima istraživanje podataka, procjenu statističkih modela i provođenje statističkih testova. Opsežan popis opisne statistike, statističkih testova, funkcija crtanja i statistika rezultata dostupan je za različite vrste podataka i svakog procjenitelja. Statsmodels je dograđivan pomoću numeričkih biblioteka NumPy i SciPy, te se integrira s Pandas bibliotekom za rukovanje podacima. Grafičke funkcije temelje se na Matplotlib biblioteci, te pruža statističku pozadinu za ostale Python biblioteke [13].

4.2.6. Keras

Keras je aplikacijsko programsko sučelje (eng. *application programming interface*, skraćeno; *API*) napisan u Python-u koji je nadograđivan na sučelje strojskog učenja, *TenserFlow*. Keras je u početku razvijen kao dio istraživačkog napora projekta ONEIROS (*Open-ended Neuro-Electronic Intelligent Robot Operating System*). Razvijen je s naglaskom na omogućavanje brzog eksperimentiranja, te je vrlo prihvatljivo i produktivno sučelje za rješavanje problema strojnog učenja s naglaskom na moderno duboko učenje. Pruža osnovne apstrakcije i gradivne blokove za razvoj i isporuku rješenja s velikom brzinom ponavljanja. Keras omogućuje inženjerima i istraživačima da u potpunosti iskoriste skalabilnost i mogućnost različitih platformi [14].

5. PROJEKTNI ZADATAK

5.1. Predviđanje trenda širenja koronavirusne bolesti (COVID-19) u Python-u pomoću biblioteke Fbprophet

Ovaj rad predstavlja problem analize vremenskih serija. Moramo pronaći njegov trend, pa ćemo u ovom dijelu zadatka većinom koristiti jednu od najpopularnijih biblioteka opisane u odlomku 3.2.3, Fbprophet koju je kreirao Facebook. Odlikuje se velikom jednostavnošću, pa čak i za nove korisnike. U našem rješenju koristimo podatke iz baze podataka Sveučilišta John Hopkins [15] koji ažuriraju broj potvrđenih, smrtnih i izlječenih slučajeva koronavirusne bolesti na dnevnoj bazi. Ova datoteka sadrži informacije o tome koliko je potvrđenih slučajeva, slučajeva smrti i oporavka počevši od 22. siječnja 2020. do 16. ožujka 2020. u određenom gradu, provinciji, državi ili regiji [15]. Naša predviđanja izrađena su za travanj 2020. koja ćemo na kraju usporediti sa stvarnim podatcima kako bi usporedili vrijednosti naše prognoze.

U ovom dijelu zadatka koristit ćemo navedene biblioteke iz odlomka 3.2. a to su:

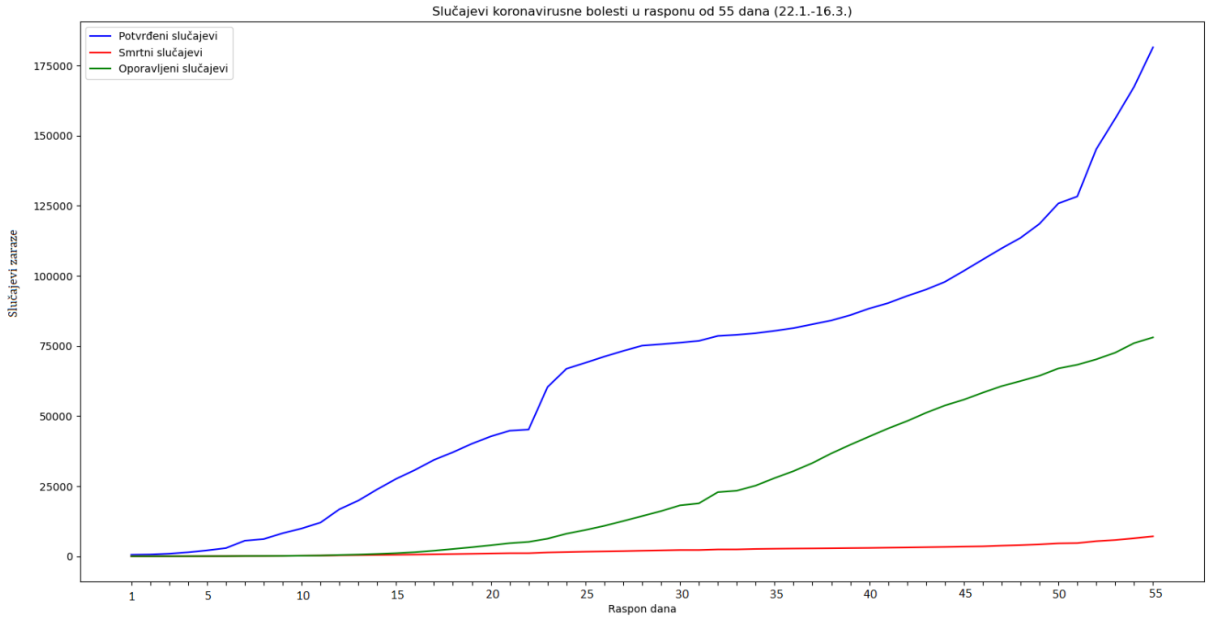
- Pandas
- NumPy
- Matplotlib
- FbProphet

Prvo što moramo napraviti je uvesti te biblioteke u sam program Python-a. U ovom radu koristimo besplatnu distribuciju Python programskoj jezika, Anaconda u PyCharm sučelju. Nakon što smo uvezli biblioteke u Python, učitavamo podatke pomoću biblioteke Pandas.

```
df_datum = df.groupby(["datum"])[['potvrđeni', 'smrtni',
'izlječeni']].sum().reset_index()
df_država = df.groupby(["država"])[['potvrđeni', 'smrtni',
'izlječeni']].sum().reset_index()

datum_x_ticks = [] #datum
država_x_ticks = [] #država
datum_potvrđeni=[] #potvrđenih slučajeva na dnevnoj bazi
datum_smrtni=[]
datum_izlječeni=[]
država_potvrđeni = [] #po državi potvrđenih slučajeva za jedan datum
država_smrtni = []
država_izlječeni = []

for index, row in df_datum.iterrows():
```

```
    datum_x_ticks.append(row['datum'])
    datum_potvrđeni.append(row['potvrđeni'])
    datum_smrtni.append(row['smrtni'])
    datum_izliječeni.append(row['izliječeni'])


for index, row in df_država.iterrows():

    država_x_ticks.append(row['država'])
    država_potvrđeni.append(row['potvrđeni'])
    država_smrtni.append(row['smrtni'])
    država_izliječeni.append(row['izliječeni'])
```

Uz pomoć navedenog dijela koda organiziramo datum u varijabli *df_datum* dok za države radimo varijablu *df_država*. *Groupby* operacija vrši se za kombinaciju razdvajanja objekta, primjene funkcije i kombiniranja rezultata. To se može koristiti za grupiranje velikih količina podataka i izračunavanje operacija u tim skupovima. U ovom dijelu to koristimo kod sumiranja zaraženih, smrtnih i oporavljenih slučajeva po datumima i državama [16, 17].



Slika 4. Slučajevi koronavirusne bolesti u periodu od 22.1. do 16.3.2020.

Sljedeću stvar koju ćemo napraviti je urediti podatke tako da ih biblioteka Fbprophet može prepoznati. Stvaramo primjerak klase Prophet, a zatim pozivamo metode uklapanja i predviđanja. Ulaz u Prophet biblioteku uvijek je vremenska serija s dvije značajke: datum *ds* (eng. data stamp) i vrijednost *y*. U ovom radu, *ds* je datum dana, a *y* akumulirani slučajevi u određenoj zemlji [16].

```
datum_potvrđeni_prophet = df_datum[['datum', 'potvrđeni']]
datum_smrtni_prophet = df_datum[['datum', 'smrtni']]
datum_izliječeni_prophet = df_datum[['datum', 'izliječeni']]

datum_potvrđeni_prophet.columns = ['ds', 'y']
datum_smrtni_prophet.columns = ['ds', 'y']
datum_izliječeni_prophet.columns = ['ds', 'y']
#
model_potvrđeni = Prophet(interval_width=0.99)
model_potvrđeni.fit(datum_potvrđeni_prophet)
budućnost_potvrđeni = model_potvrđeni.make_future_dataframe(periods=30)
prognoza_potvrđeni = model_potvrđeni.predict(budućnost_potvrđeni)
```

Gornjim kodom izrađujemo prognozu za potvrđene slučajeve. Prvo, kreiramo model i kažemo da želimo interval pouzdanosti od 99%. Zatim podatke obrađujemo metodom *fit()*, pa onda dodamo razdoblje za koje želimo predvidjeti podatke, u našem slučaju to je sljedećih 30 dana.

Posljednji korak je pozivanje metode *predict()* koja će izvršiti predviđanje. Prognoza ima mnogo atributa, ali nas zanima atribut *y* koji predviđa slučajeve [16]. U tablici 1. su prikazane dobivene vrijednosti za zadnjih 5 dana naše prognoze (period od 11.4. do 15.4.2020.)

Relativnu razliku predviđenih i stvarnih vrijednosti ćemo računati pomoću jednadžbi (1), (2) i (3) koje glase:

$$\overline{x} = \frac{x_1 + x_2}{2} \tag{2}$$

$$\Delta x = x_1 - x_2 \tag{3}$$

$$\frac{\Delta x}{\overline{x}} \cdot 100\% \tag{4}$$

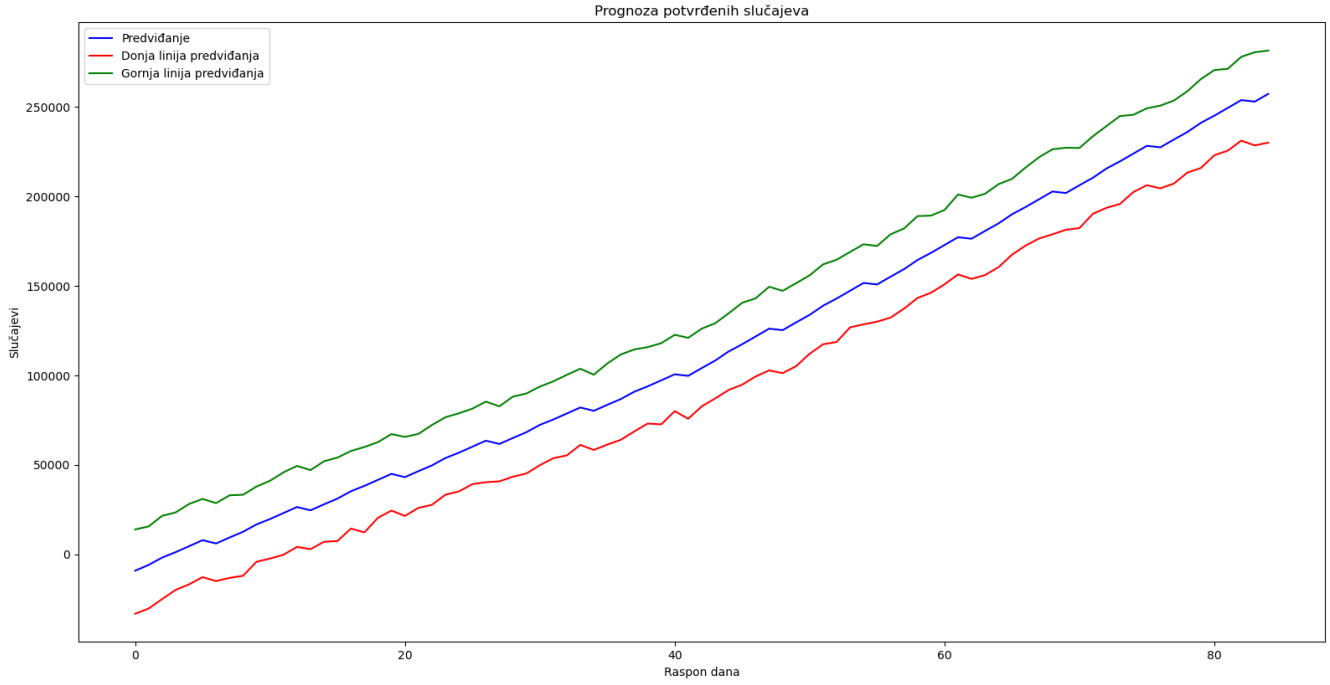
gdje su:

x_1 – stvarna vrijednost

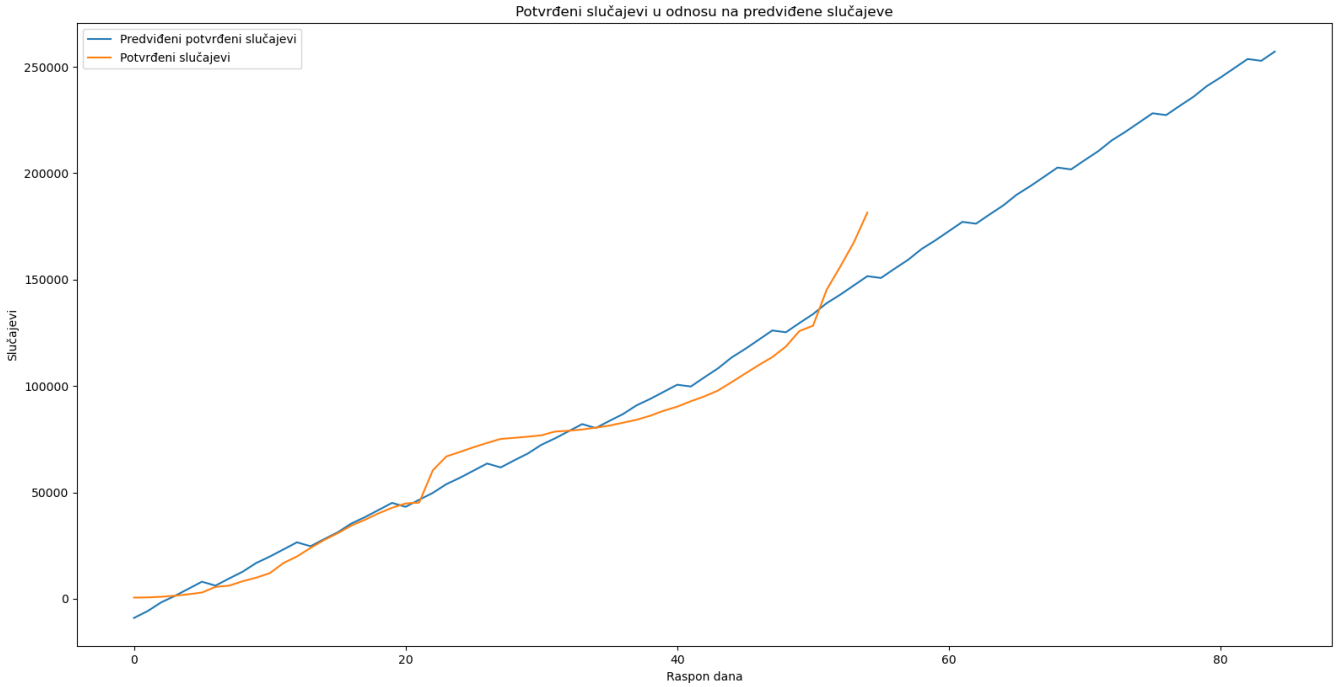
x_2 – predviđena vrijednost

Tablica 1. Predviđena i stvarna vrijednost potvrđenih slučajeva u rasponu dana od 11.4. do 15.4.2020. pomoću *fbprophet* modela

Datum	Predviđena vrijednost potvrđenih slučajeva	Stvarna vrijednost potvrđenih slučajeva (prema *John Hopkins University* bazi podataka)	Relativna razlika između stvarne i predviđene vrijednosti
11.4.2020.	245002	1804228	152,18%
12.4.2020.	249345	1872735	153%
13.4.2020.	253710	1942828	153,8%
14.4.2020.	252862	2016325	155,43%
15.4.2020.	257198	2096106	156,28%

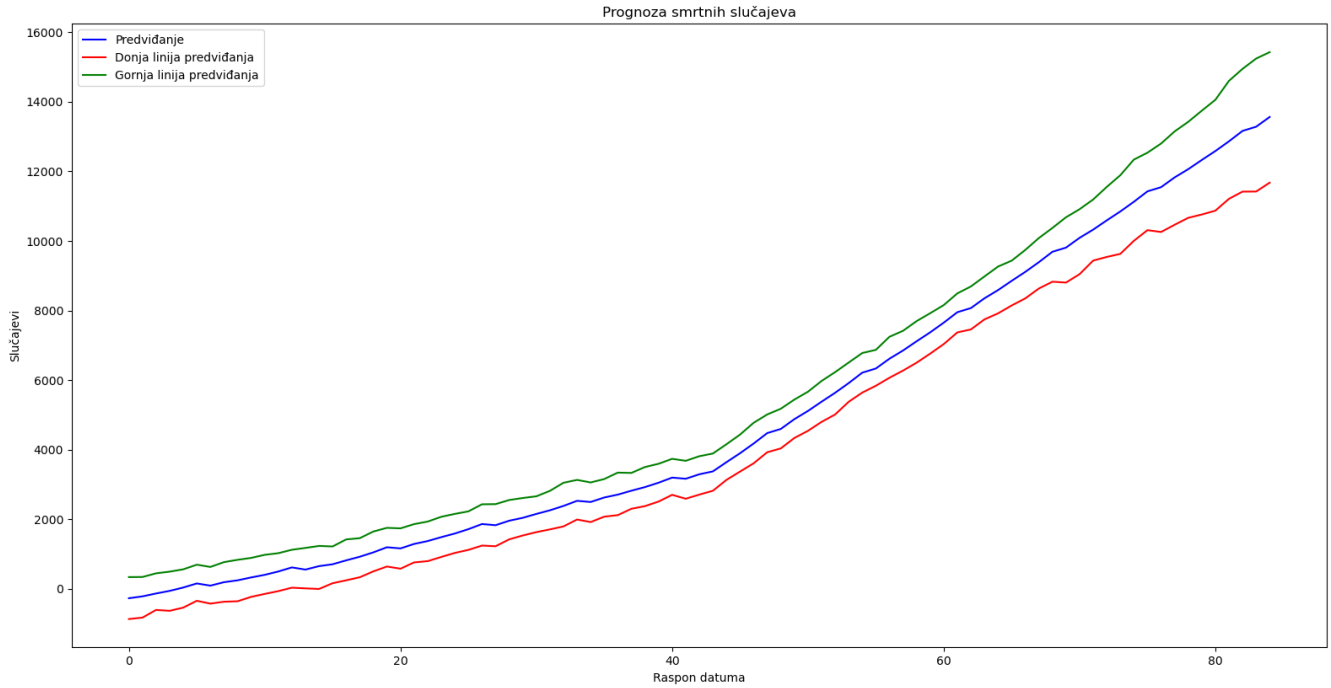


Slika 5. Prognoza potvrđenih slučajeva

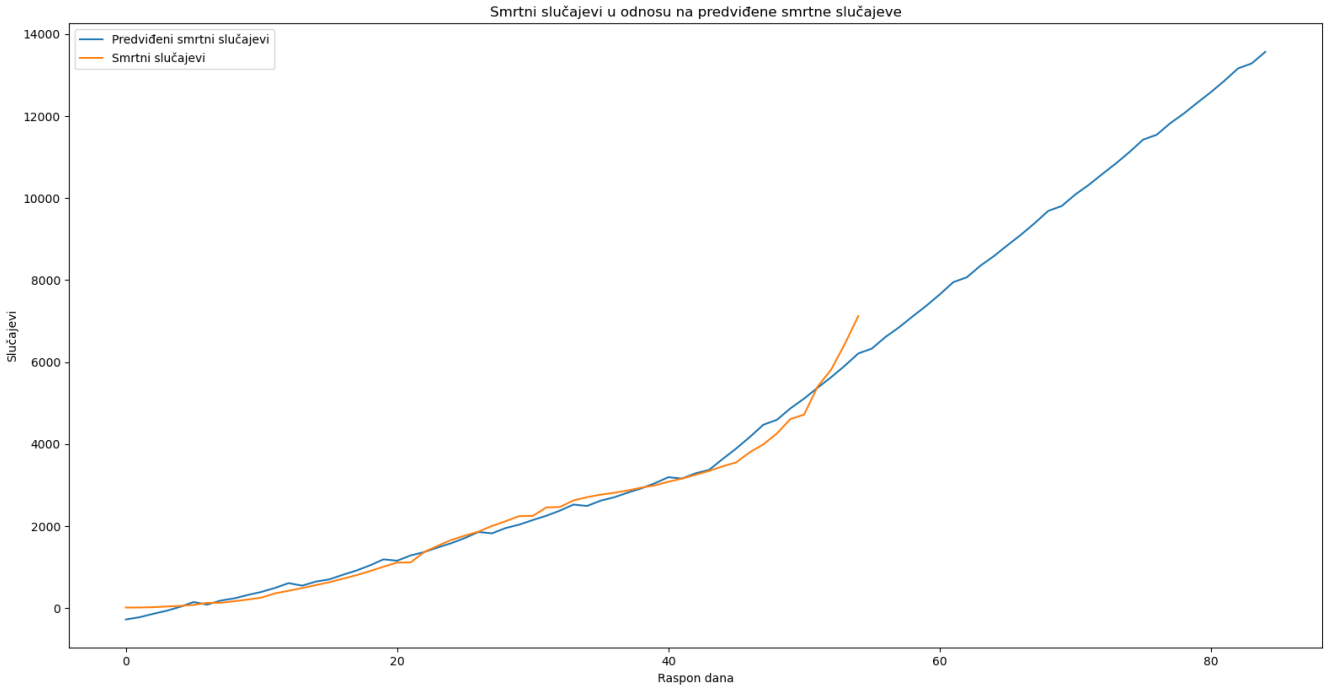


Slika 6. Potvrđeni slučajevi u odnosu na predviđene slučajeve

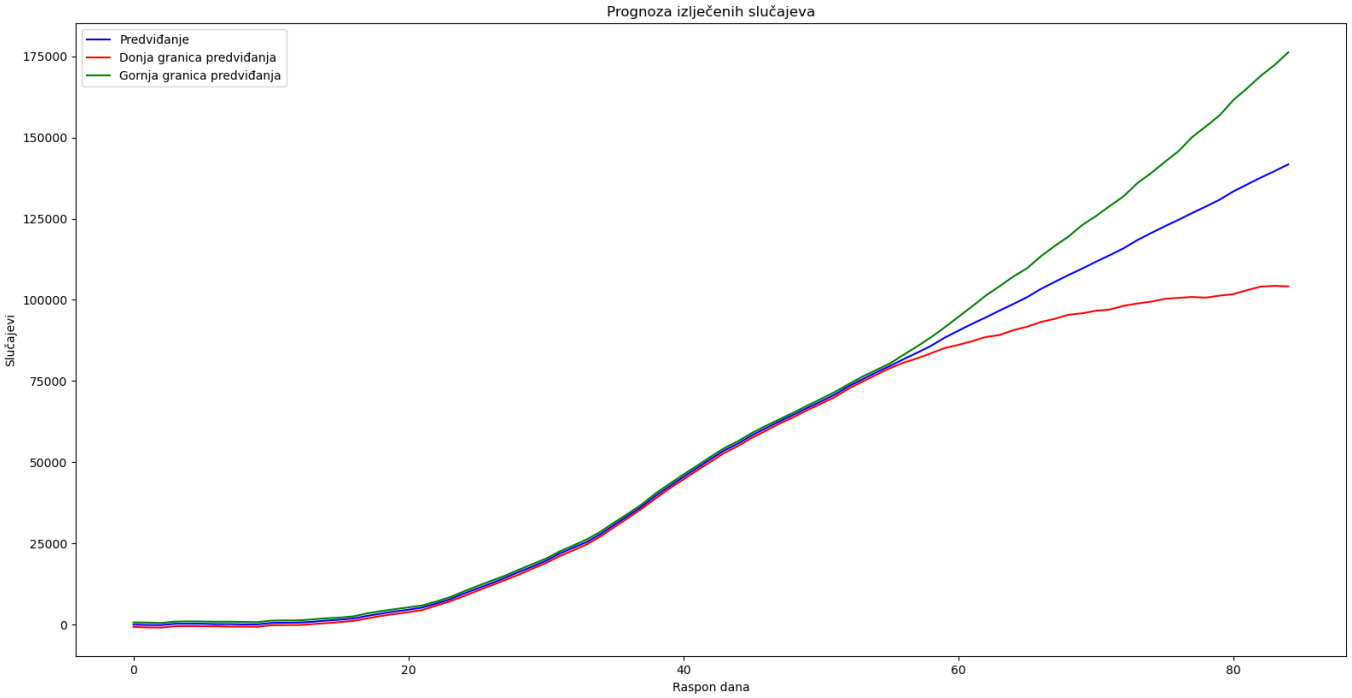
Kao što vidimo iz slike 6. predviđanje potvrđenih slučajeva je skoro pa linearnog rasta dok stvarni potvrđeni slučajevi počinju biti eksopnencijalnog rasta. Ovim modelom imamo velika odstupanja između predviđene i stvarne vrijednosti potvrđenih slučajeva koje su prikazane u tablici 1. Daljnje vrijednosti poput smrtnih i izlječenih slučajeva prikazat ćemo grafički u sljedećim slikama.



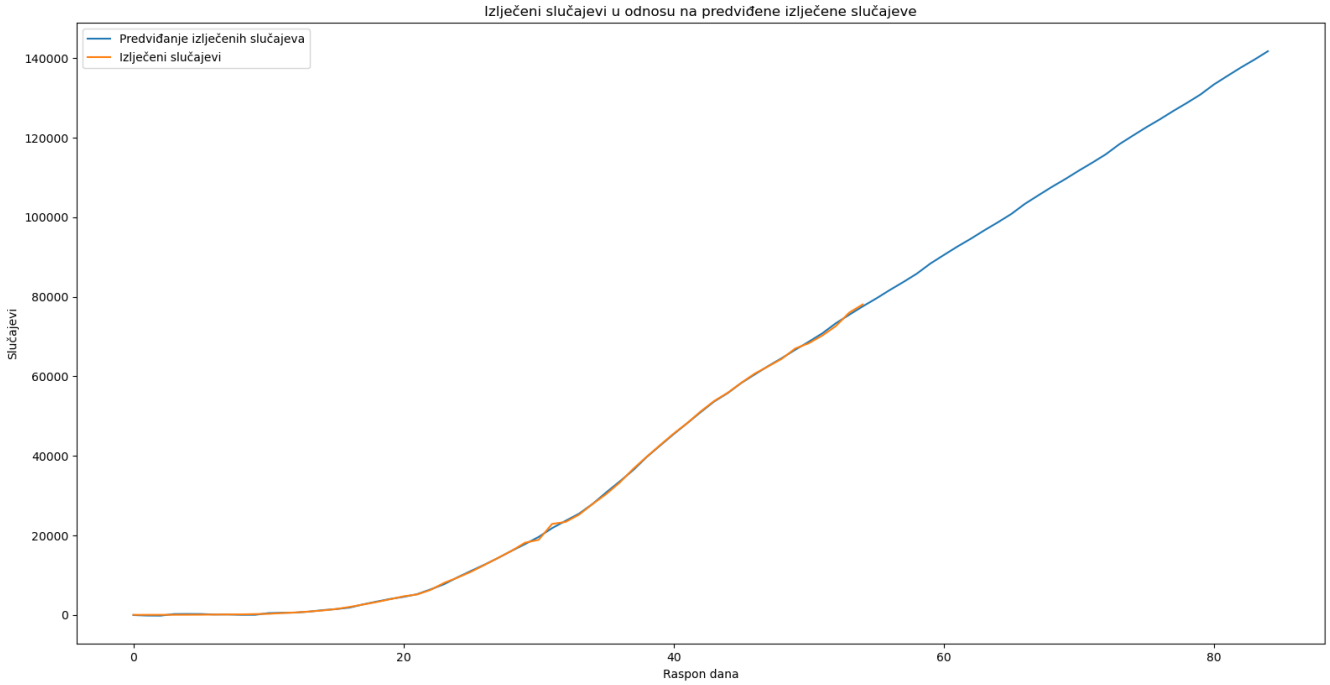
Slika 7. Prognoza smrtnih slučajeva



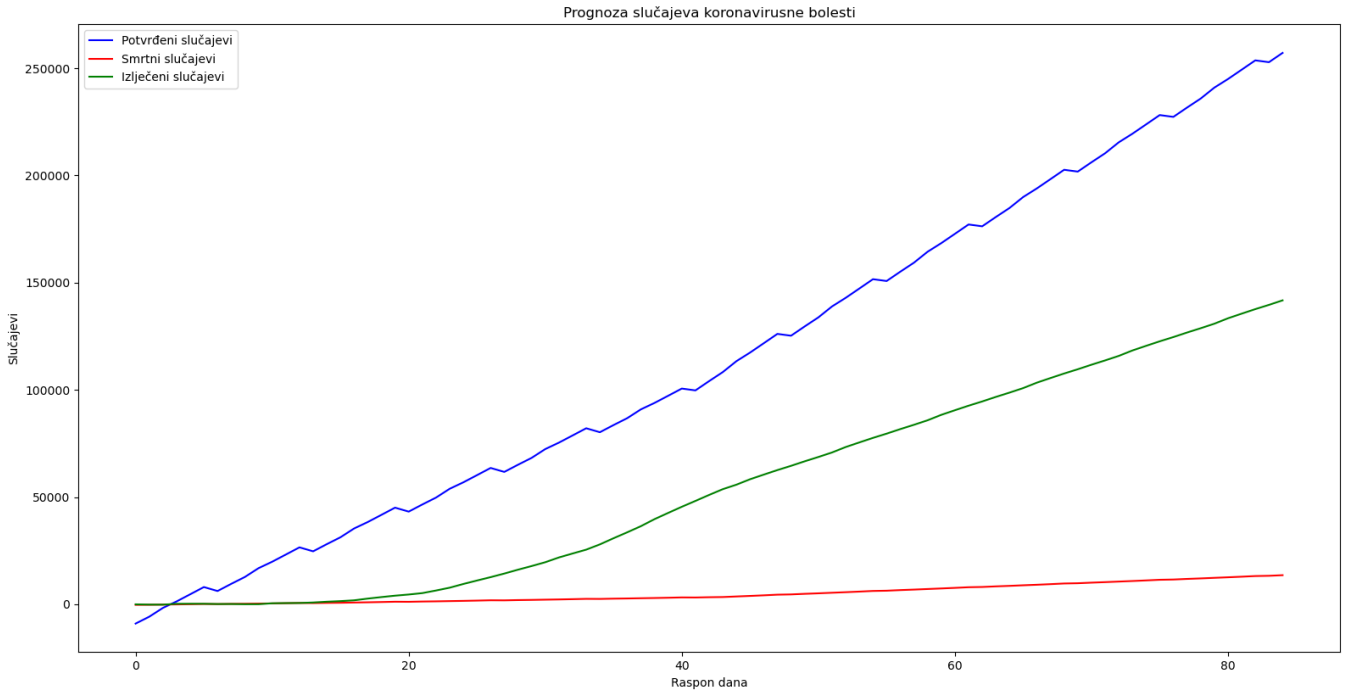
Slika 8. Potvrđeni smrtni slučajevi u odnosu na predviđene smrtne slučajeve



Slika 9. Prognoza izlječenih slučajeva



Slika 10. Potvrđeni izlječeni slučajevi u odnosu na predviđene izlječene slučajeve



Slika 11. Prognoza slučajeva koronavirusne bolesti (fbprophet model)

5.2. Predviđanje trenda širenja koronavirusne bolesti (COVID-19) u Python-u pomoću biblioteke Statsmodels

U teoriji vjerojatnosti, stohastički proces je matematički objekt koji se obično definira kao familija nasumično odabranih varijabli. Budući da je naš problem problem vremenskih serija, koristit ćemo stohastičke procese koji nam mogu pomoći u određivanju, tj. predviđanju ishoda u takvim situacijama [18]. U ovom dijelu zadatka koristit ćemo biblioteke:

- Pandas
- NumPy
- SciPy
- Matplotlib
- Statsmodels

gdje najviše do izražaja dolazi biblioteka statsmodels uključujući praktičnu primjenu njezinih modela AR, ARIMA i SARIMA.

5.2.1. Autoregresivni (AR) model

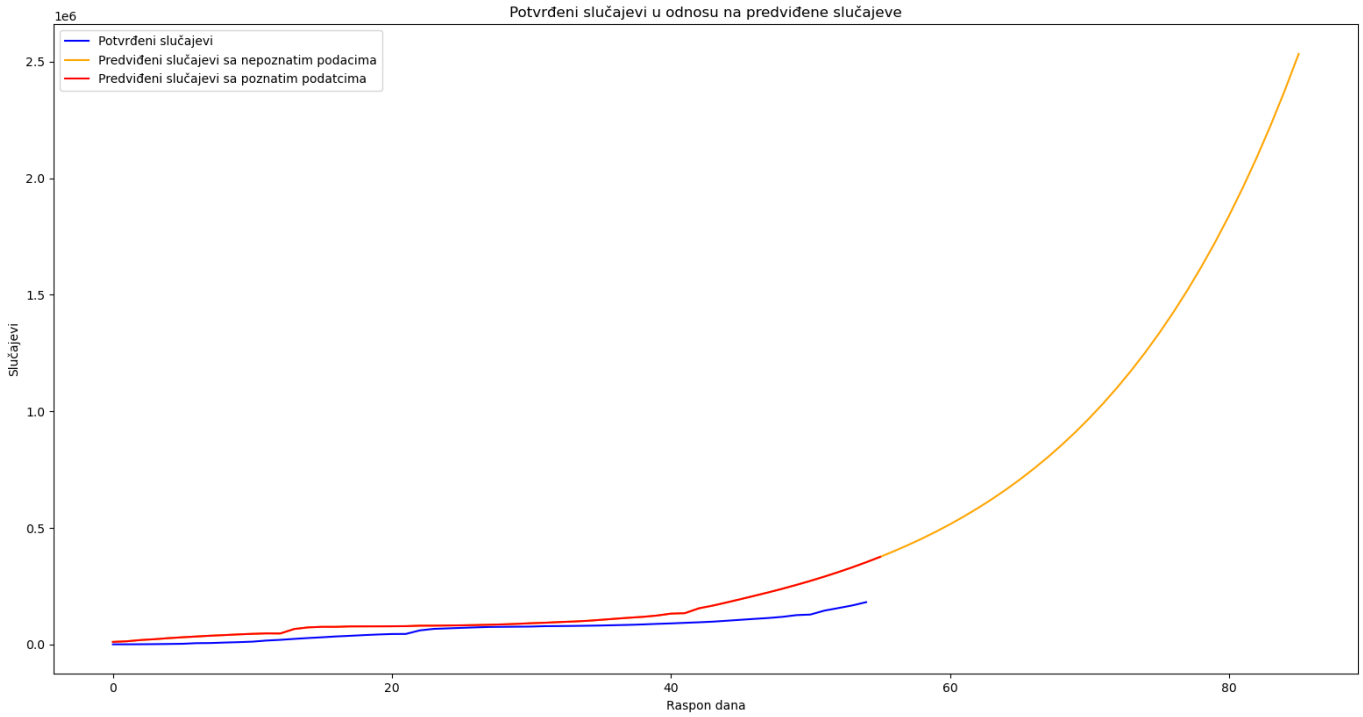
Kao i u prvom dijelu zadatka, prvo moramo uvesti potrebne biblioteke i njihove klase u Python kao i učitavanje potrebnih podataka i imenovanje varijabli preko kojih vršimo predviđanja.

```
datum_potvrđeni = df_datum[['datum', 'potvrđeni']]
datum_smrtni = df_datum[['datum', 'smrtni']]
datum_izliječeni = df_datum[['datum', 'izliječeni']]
print(datum_smrtni)
for index, row in datum_potvrđeni.iterrows():
    if row['potvrđeni'] is None:
        row['potvrđeni'] = 0.0

for index, row in datum_smrtni.iterrows():
    if row['smrtni'] is None:
        row['smrtni'] = 0.0

for index, row in datum_izliječeni.iterrows():
    if row['izliječeni'] is None:
        row['izliječeni'] = 0.0
```

U statistikama autoregresivni (AR) model je prikaz vrste slučajnog procesa. Koristi se za opisivanje određenog procesa koji varira u vremenu. AR model određuje da izlazna varijabla linearno ovisi o vlastitim prethodnim vrijednostima i o stohastičkom pojmu [18].

```
model_ar_potvrđeni = AR(np.asanyarray(datum_potvrđeni['potvrđeni']))
model_fit_ar_potvrđeni = model_ar_potvrđeni.fit()
predvidi_ar_potvrđeni = model_fit_ar_potvrđeni.predict(10,
len(datum_potvrđeni) + 40)
print(predvidi_ar_potvrđeni)

model_ar_smrtni = AR(np.asanyarray(datum_smrtni['smrtni']))
model_fit_ar_smrtni = model_ar_smrtni.fit()
predvidi_ar_smrtni = model_fit_ar_smrtni.predict(10, len(datum_smrtni) +
40)

model_ar_izliječeni = AR(np.asanyarray(datum_izliječeni['izliječeni']))
model_fit_ar_izliječeni = model_ar_izliječeni.fit()
predvidi_ar_izliječeni = model_fit_ar_izliječeni.predict(10,
len(datum_izliječeni) + 40)
```

Priloženim kodom prognoziramo ishod za potvrđene, smrtne i izliječene slučajeve, te grafičkim prikazom ucrtavamo svaku njihovu vrijednost s odgovarajućom prognozom kako bismo ih usporedili. Kao i u prijašnjem dijelu zadatka, vršit ćemo predviđanje u razdoblju od sljedećih 30 dana te usporediti rezultate sa stvarnim podatcima u tom vremenskom periodu.



Slika 12. Potvrđeni slučajevi u odnosu na predviđene slučajeve (AR model)

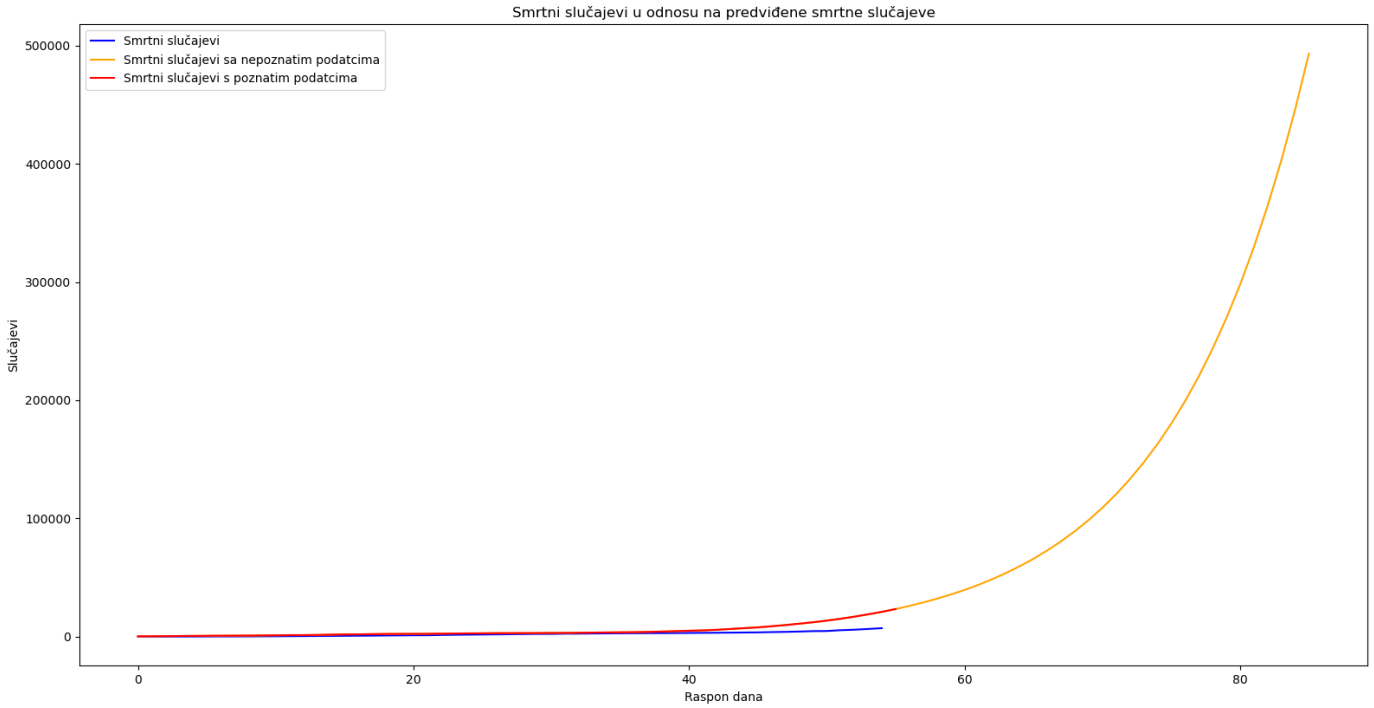
Iz priloženog vidimo progresivno veći rast predviđenih slučajeva u odnosu na prošlu metodu koja čak prognozira i veći broj zaraženih nego što je stvaran broj. Kao što je prikazano, kada predviđa za razdoblja od 22. siječnja do 16. ožujka, obje funkcije (plava i crvena) bliže su jedna drugoj. Žuta linija označuje prognozu koja izgleda puno točnije nego u prvom dijelu zadatka. U tablici 2. ubacujemo vrijednosti predviđenih slučajeva koristeći AR model i stvarne vrijednosti, te koristimo formule (1), (2) i (3) za izračunavanje relativne razlike.

Tablica 2. Predviđena i stvarna vrijednost potvrđenih slučajeva u rasponu dana od 11.4. do 15.4.2020. pomoću AR modela

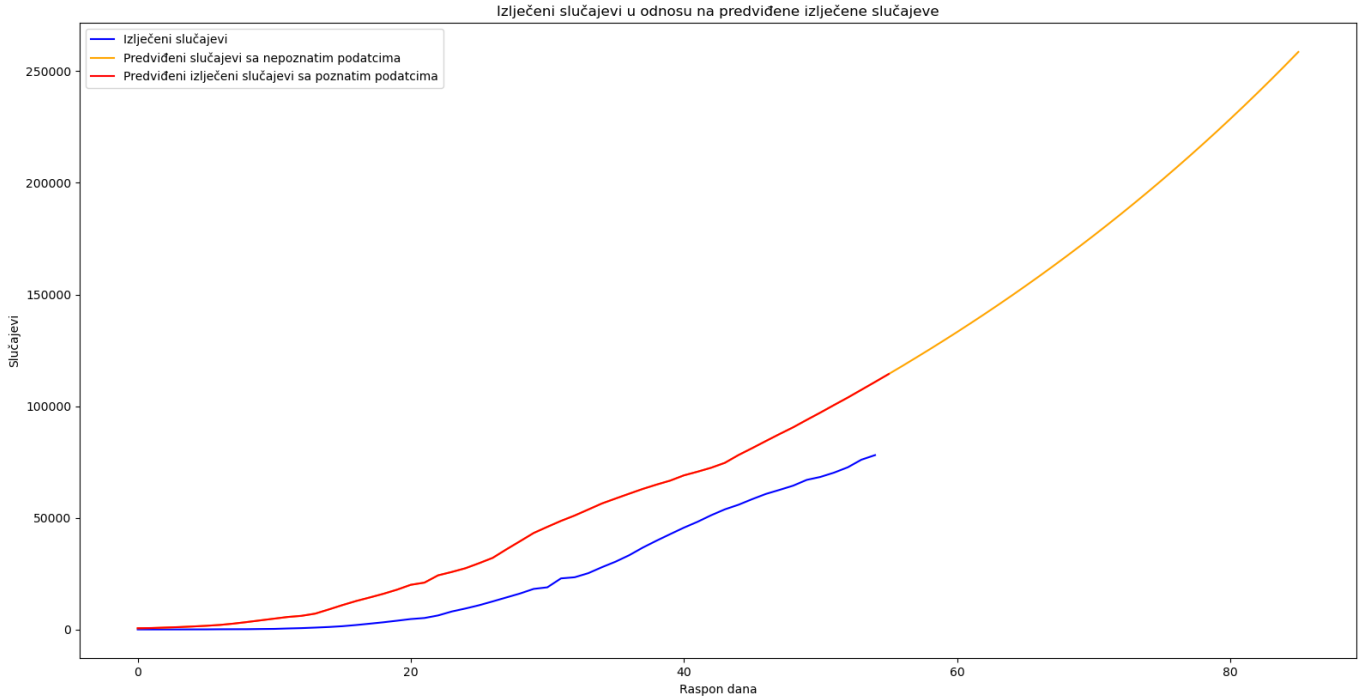
Datum	Predviđena vrijednost slučajeva	Stvarna vrijednost slučajeva (preuzeto iz *John Hopkins University* baze podataka)	Relativna razlika između stvarne i predviđene vrijednosti
11.4.2020.	1433640	1804228	22,89%
12.4.2020.	1517598	1872735	20,95%
13.4.2020.	1617826	1942828	18,26%

14.4.2020.	1724740	2016325	15,59%
15.4.2020.	1838790	2096106	13,08%

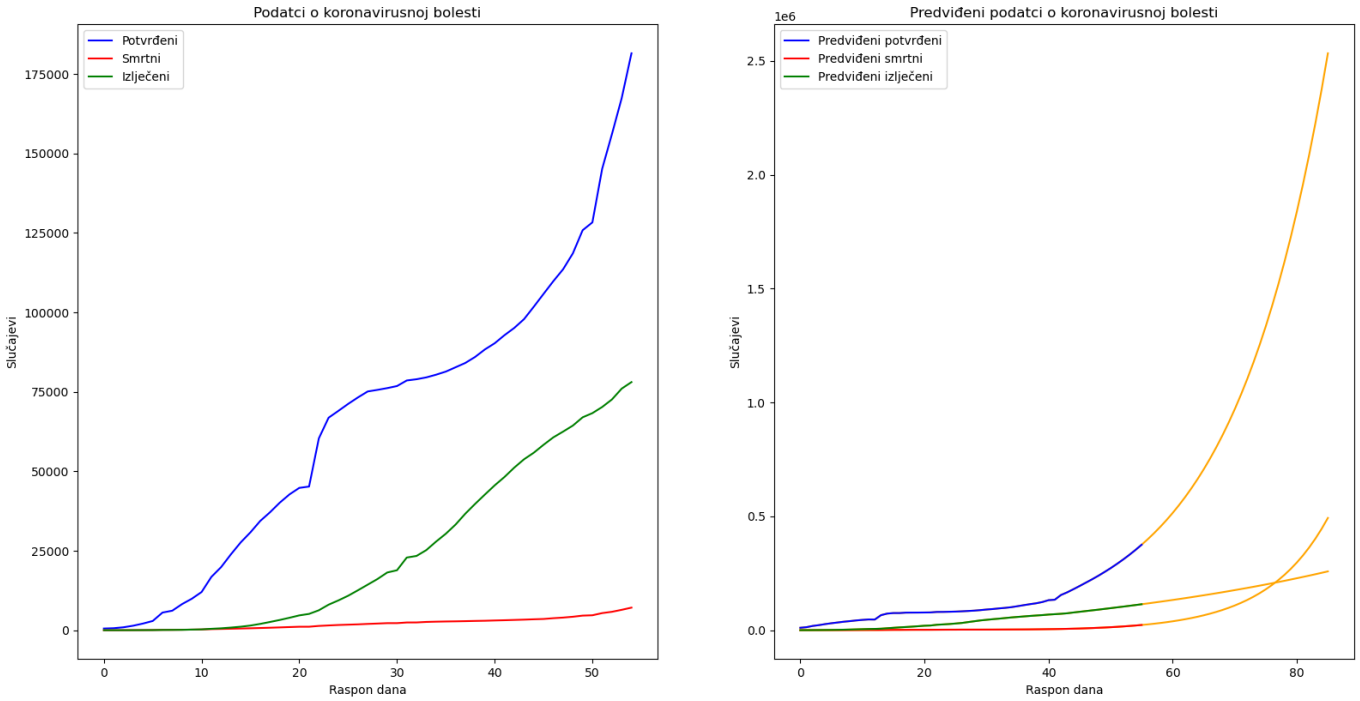
Iz tablice 2. vidimo kako nam je relativna razlika puno manja pri korištenju ovog model za predviđanje trenda širenja koronavirusne bolesti u odnosu na Fbprophet modela iz 4.1. odjeljka. Vidimo i kako relativna razlika postaje sve manja svakim sljedećim danom što nam govori da je ovaj model pouzdaniji. Da smo uzeli veći raspon dana za prognozu, vjerojatno bi i relativna razlika bila manja. Osim potvrđenih slučajeva, grafički ćemo prikazati smrtne i izlječene slučajeve koji će pokazivati vrijednosti na isti način kao i za potvrđene slučajeve.



Slika 13. Smrtni slučajevi u odnosu na predviđene smrtne slučajeve (AR model)



Slika 14. Izlječeni slučajevi u odnosu na predviđene izlječene slučajeve (AR model)



Slika 15. Usporedba grafova (AR model)

5.2.2. ARIMA model

ARIMA (*Autoregressive Integrated Moving Average*) je jedan od najčešće korištenih modela za predviđanje vremenskih serija koje se diferencijacijom mogu učiniti stacionarnima. ARIMA modeli primjenjuju se u slučajevima kada podaci pokazuju nestacionarnost, gdje se početni korak diferenciranja (koji odgovara "integriranom" dijelu modela) može primijeniti jednom ili više puta kako bi se eliminirala nestacionarnost. ARIMA model može se promatrati kao "filtar" koji pokušava odvojiti signal od šuma, a signal se zatim ekstrapolira u budućnost za dobivanje prognoza. Iako ARIMA model može obrađivati podatke s trendom, ona ne podržava vremenske serije sa sezonskom komponentom. Proširenje ARIMA-e koje podržava izravno modeliranje sezonske komponente serije naziva se SARIMA (*Seasonal Autoregressive Integrated Moving Average*) koju ćemo detaljnije obraditi u sljedećem ulomku[18, 19].

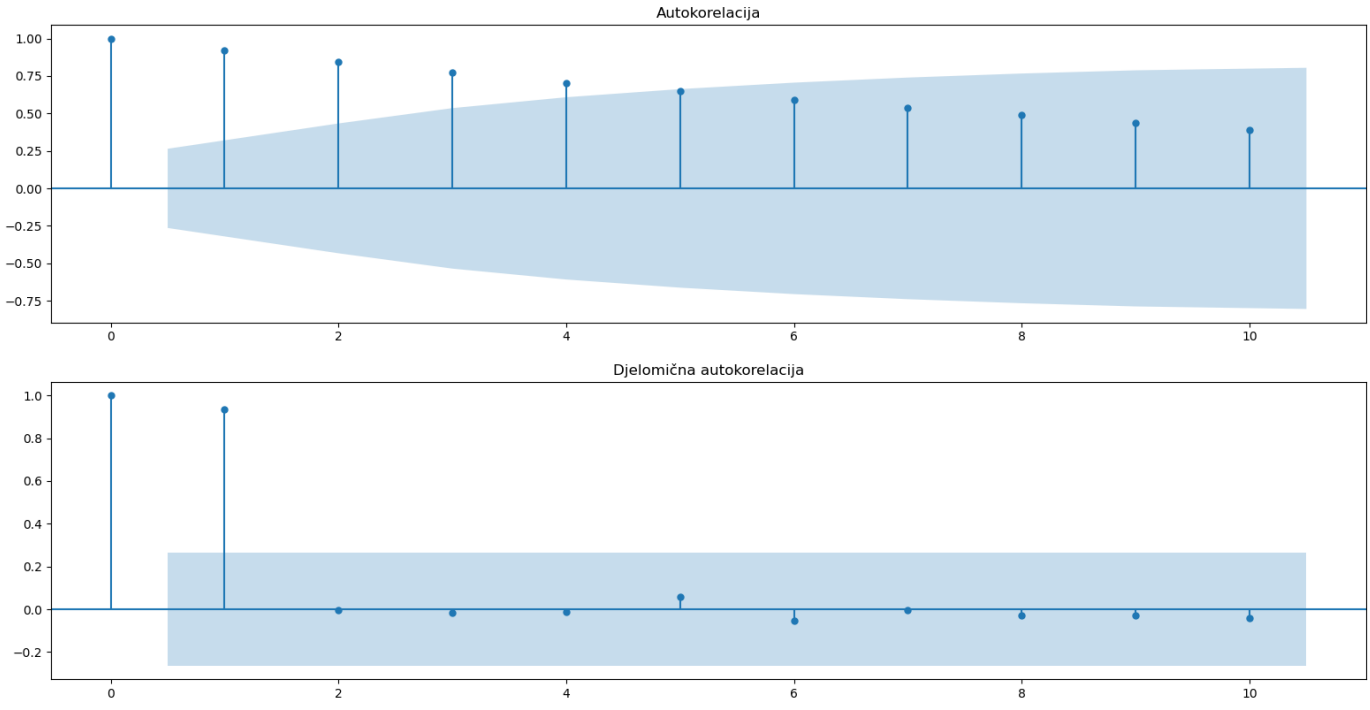
S prikazanim kodom grafički ćemo prikazati autokorelacijske (eng. *Autocorrelation function*; skraćeno *ACF*) dijagrame i dijagrame djelomične autokorelacije (eng. *Partial autocorrelation function*; skraćeno *PACF*) potvrđenih slučajeva kako bismo si pomogli u određivanju vrijednosti ARIMA i SARIMA modela. Analogno je i za druge slučajeve.

```
fig = plt.figure()
ax1 = fig.add_subplot(211)
fig = sm.graphics.tsa.plot_acf(datum_potvrđeni['potvrđeni'], lags=10,
ax=ax1) #
ax2 = fig.add_subplot(212)
fig = sm.graphics.tsa.plot_pacf(datum_potvrđeni['potvrđeni'], lags=10,
ax=ax2)#
plt.show()
```

ACF dijagram je trakasti dijagram koeficijenata korelacije između vremenske serije i samog zaostajanja. PACF dijagram je dijagram djelomičnih koeficijenata korelacije između vremenske serije i svojih zaostataka. Općenito, "djelomična" korelacija između dvije varijable je količina korelacije između njih koja se ne objašnjava njihovim međusobnim korelacijama s određenim skupom ostalih varijabli. Gledajući te dijagrame, možemo okvirno identificirati potrebne vrijednosti ARIMA modela [19].

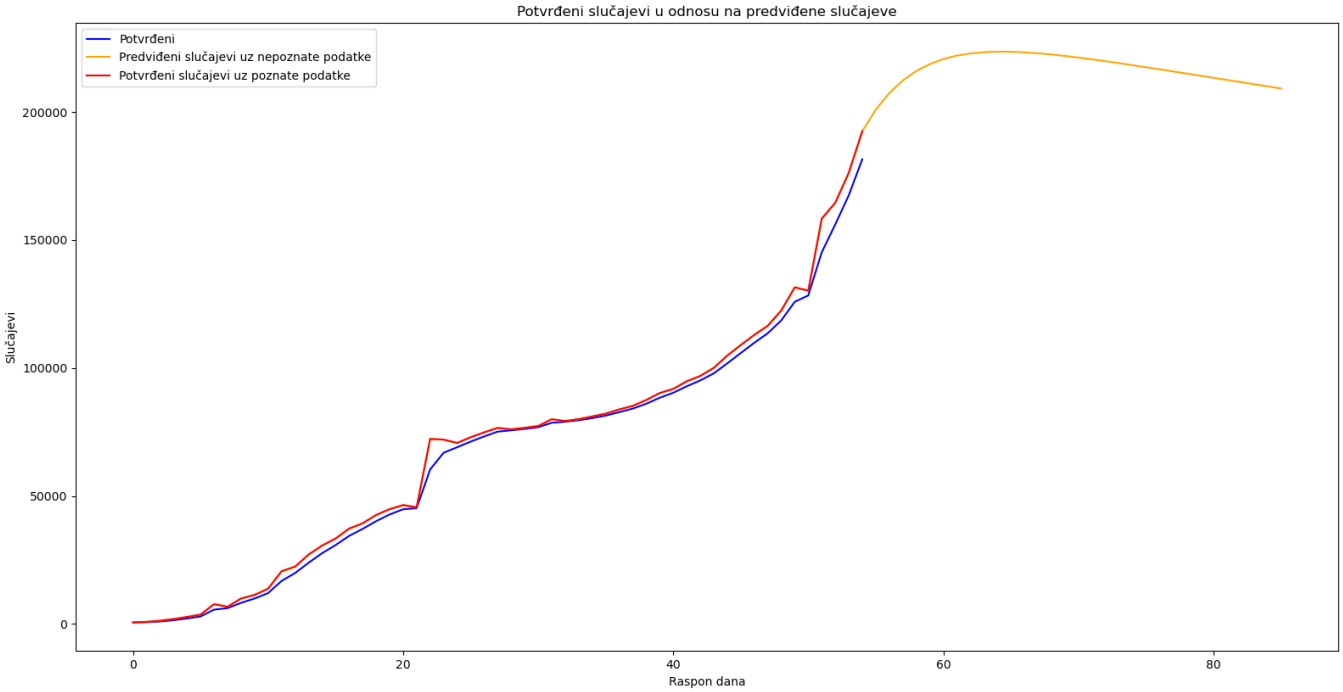
```
from statsmodels.tsa.arima_model import ARIMA
model_ma_potvrđeni = ARIMA(np.asanyarray(datum_potvrđeni['potvrđeni']),
order=(2,0,0))
model_fit_ma_potvrđeni = model_ma_potvrđeni.fit(disp=False)
predvidi_ma_potvrđeni = model_fit_ma_potvrđeni.predict(1,
len(datum_potvrđeni)+31)
print(predvidi_ma_potvrđeni)
plt.xlabel('datumi')
plt.ylabel('slucajevi')
plt.plot(datum_potvrđeni['potvrđeni'], label='potvrđeni', color='blue')
plt.plot(predvidi_ma_potvrđeni, label='predviđeni nepoznati podatci',
color='orange')
plt.plot(predvidi_ma_potvrđeni[:len(predvidi_ma_potvrđeni)-31],
label='predviđeni poznati podatci', color='red')
plt.title('potvrđeni slucajevi vs predviđeni potvrđeni slucajevi')
plt.legend()
plt.show()
```

Sa gore navedenim kodom, mi predviđamo vrijednosti za potvrđene slučajeve pomoću ARIMA modela. ARIMA model generalizirani je oblik ARMA modela, koji je kombinacija AR (autoregresija) i MA (pokretni prosjek) modela. ARIMA model određen je s tri vrijednosti, *p* (broj AR-a), *d* (redoslijed integracije) i *q* (broj MA-a). Ti se brojevi mogu odrediti eksperimentiranjem ili promatranjem ACF i PACF dijagrama [19].



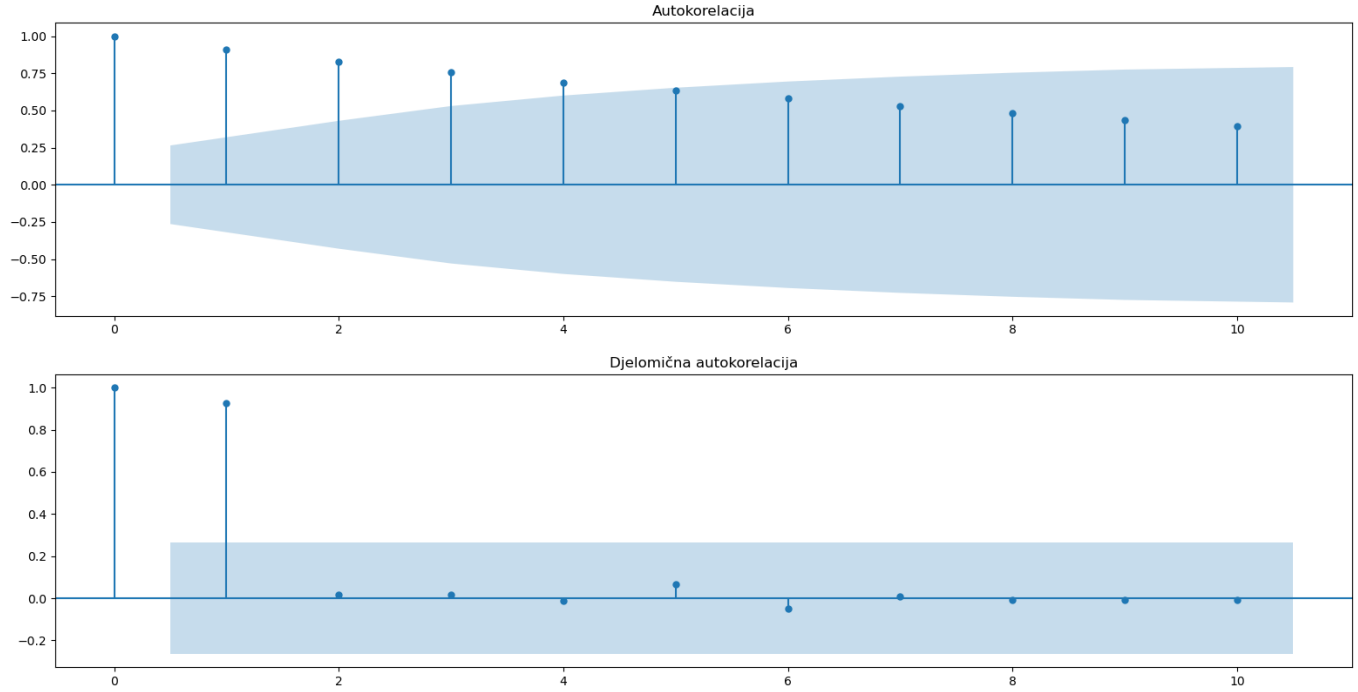
Slika 16. ACF i PACF dijagram za potvrđene slučajeve

Vrijednost za AR (*p*) najlakše je odrediti jer traži periode u PACF dijagramu, međutim, vidimo da PACF dijagram ima značajan skok u samo prve dvije periode, zato ćemo uzeti te dvije vrijednosti kao značajne. Na slici 17. možemo vidjet usporedbu između stvarne vrijednosti potvrđenih slučajeva, predviđanje potvrđenih slučajeva uz poznate podatke te predviđanje slučajeva za period od sljedećih 30 dana koji su označeni žutom linijom.

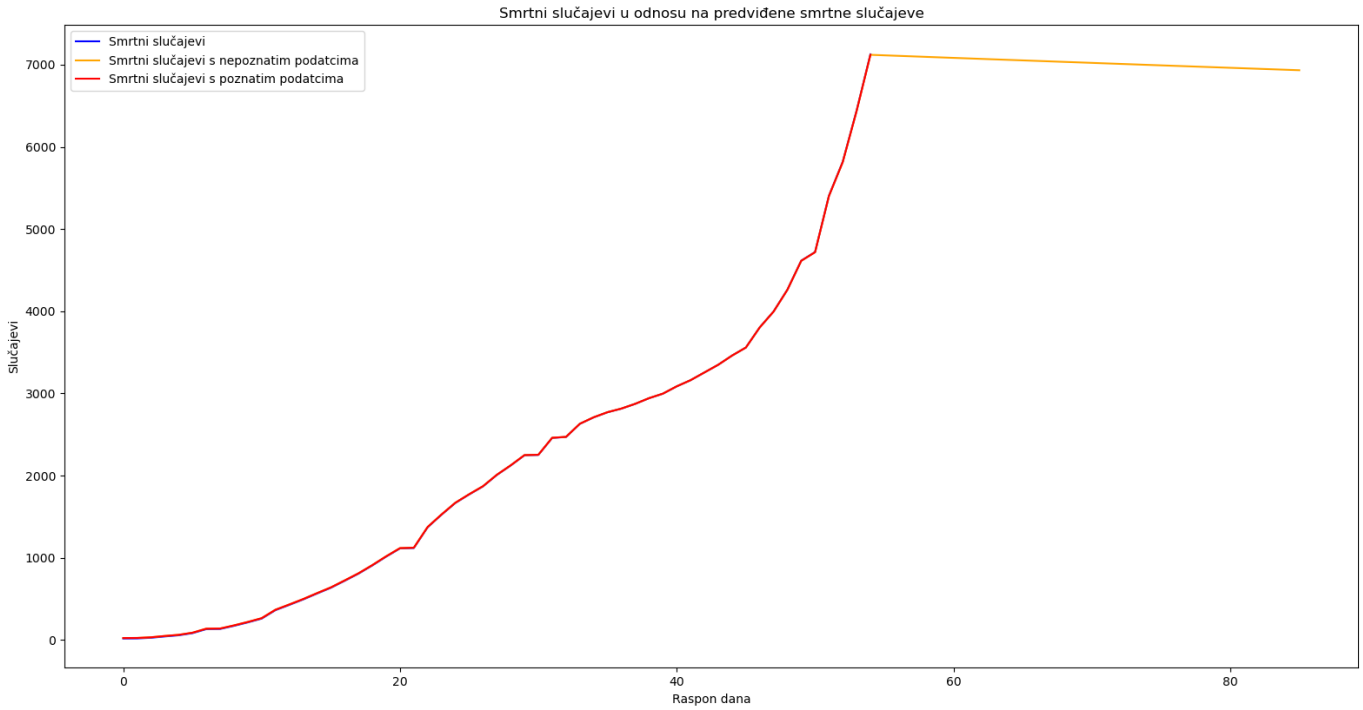


Slika 17. Dijagram potvrđenih slučajeva u odnosu na predviđene slučajeve

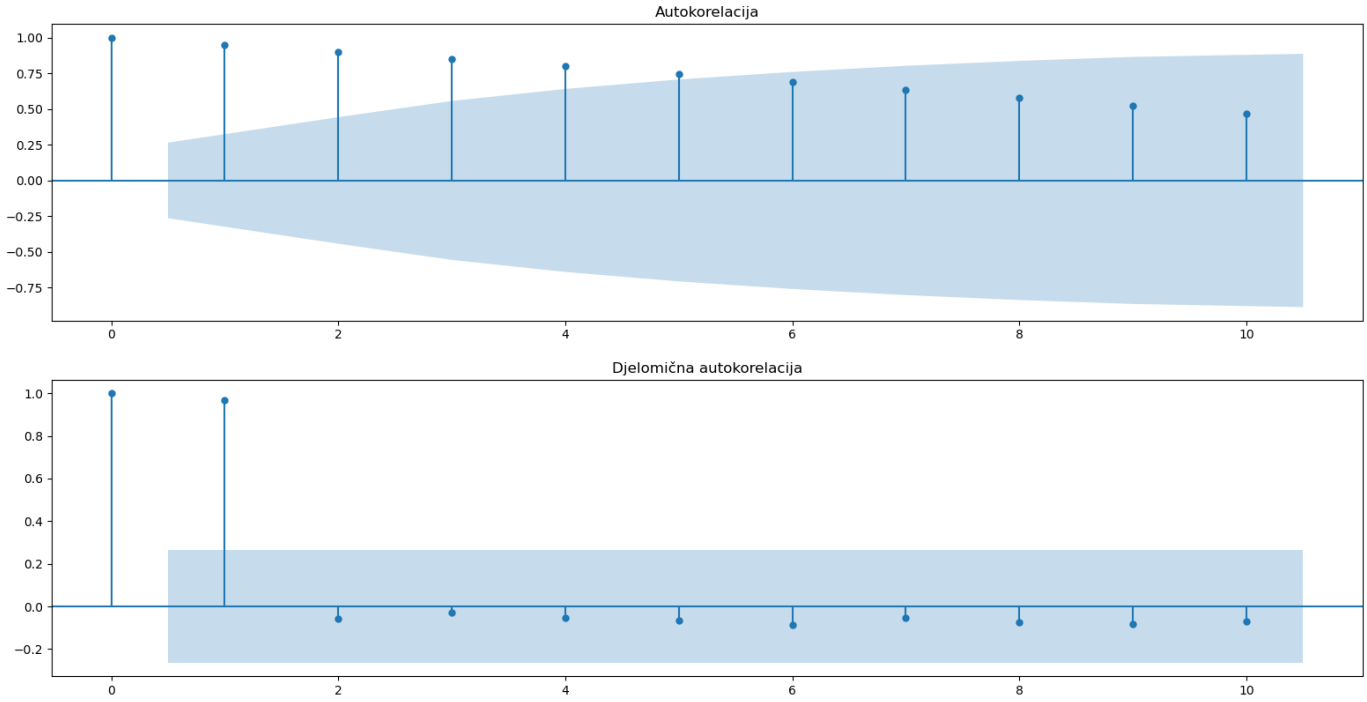
Analogno potvrđenim slučajevima, grafički ćemo prikazati ACF i PACF dijagrame kao i dijagrame smrtnih i izlječenih slučajeva u odnosu na predviđene smrtne i izlječene slučajeve ARIMA modela.



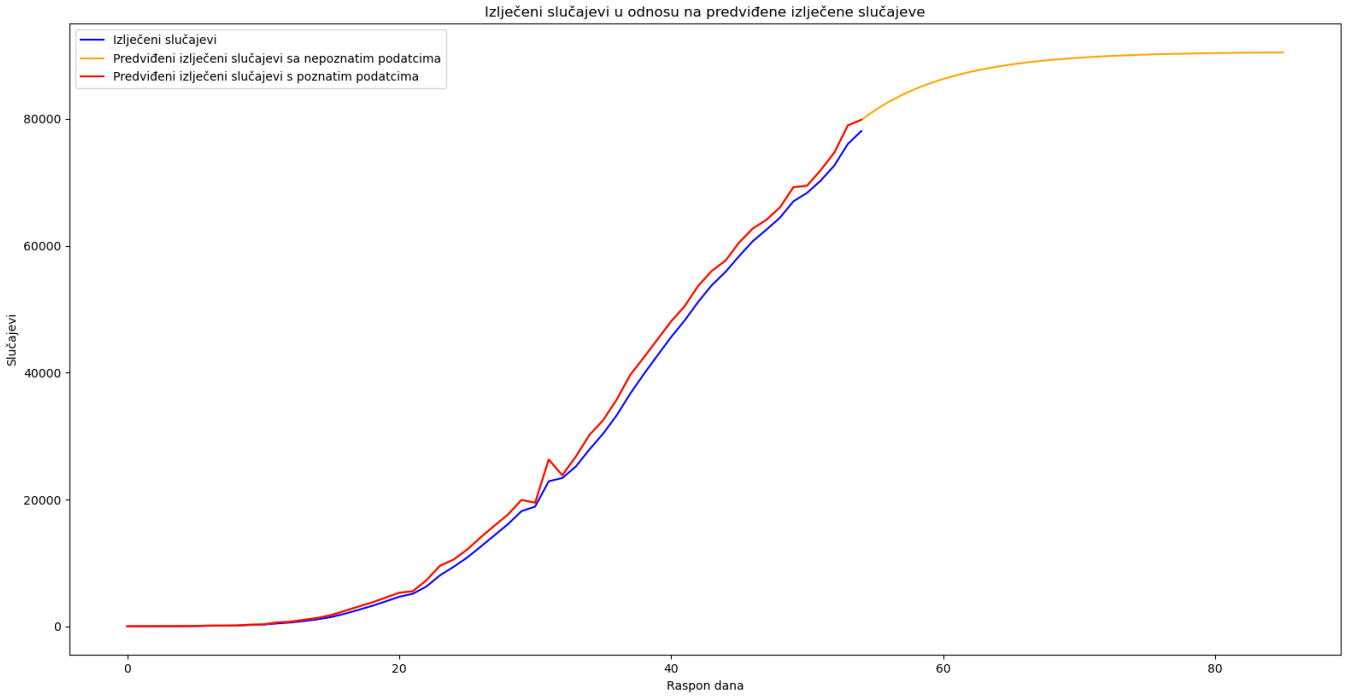
Slika 18. ACF i PACF dijagrami za smrtne slučajeve



Slika 19. Dijagram smrtnih slučajeva u odnosu na predviđene smrtne slučajeve (ARIMA model)

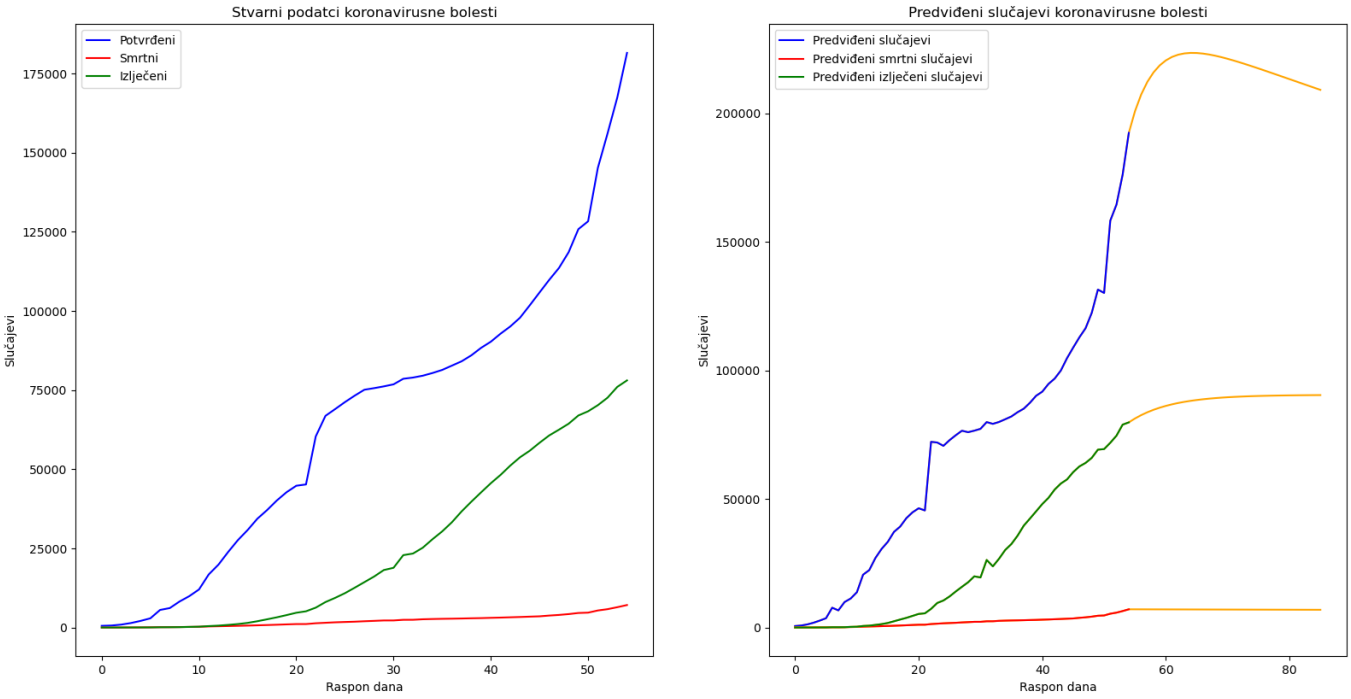


Slika 20. ACF i PACF dijagrami za izlječene slučajeve



Slika 21. Dijagram izlječenih slučajeva u odnosu na predviđene izlječene slučajeve (ARIMA model)

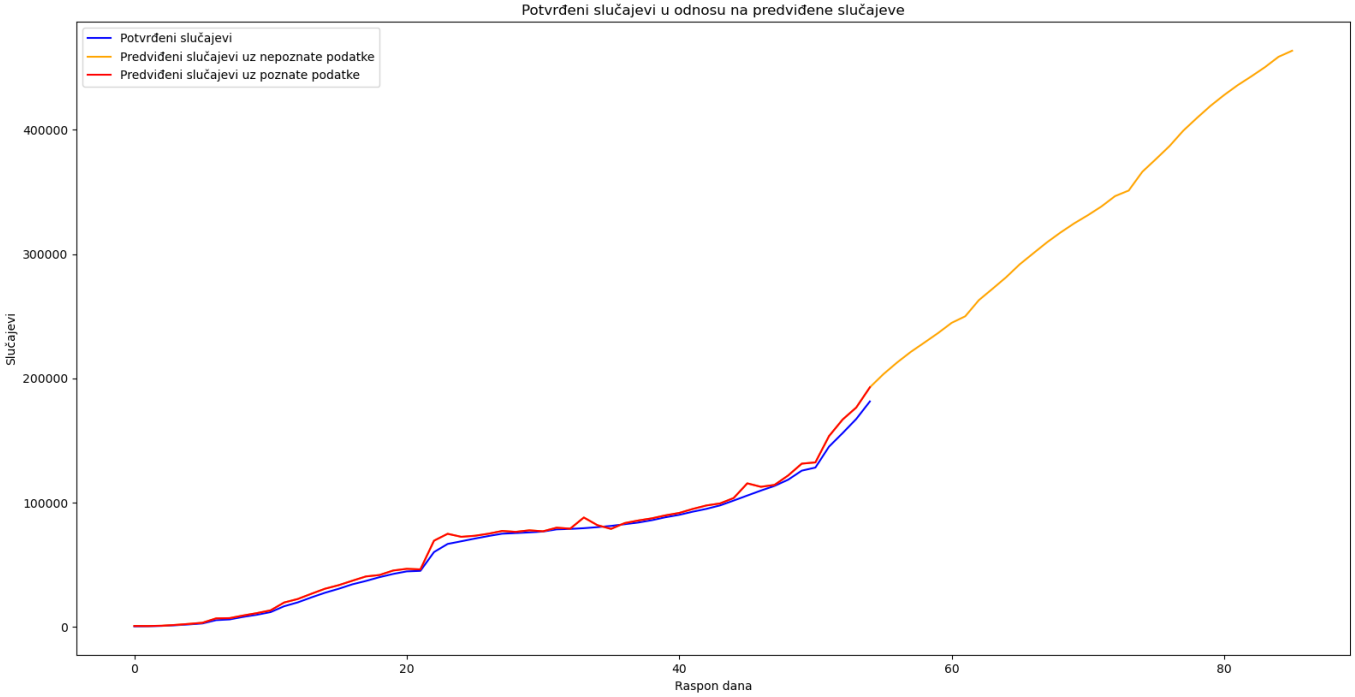
Na slici 22. imamo usporedbu grafova ARIMA modela radi jednostavnijeg prikaza.



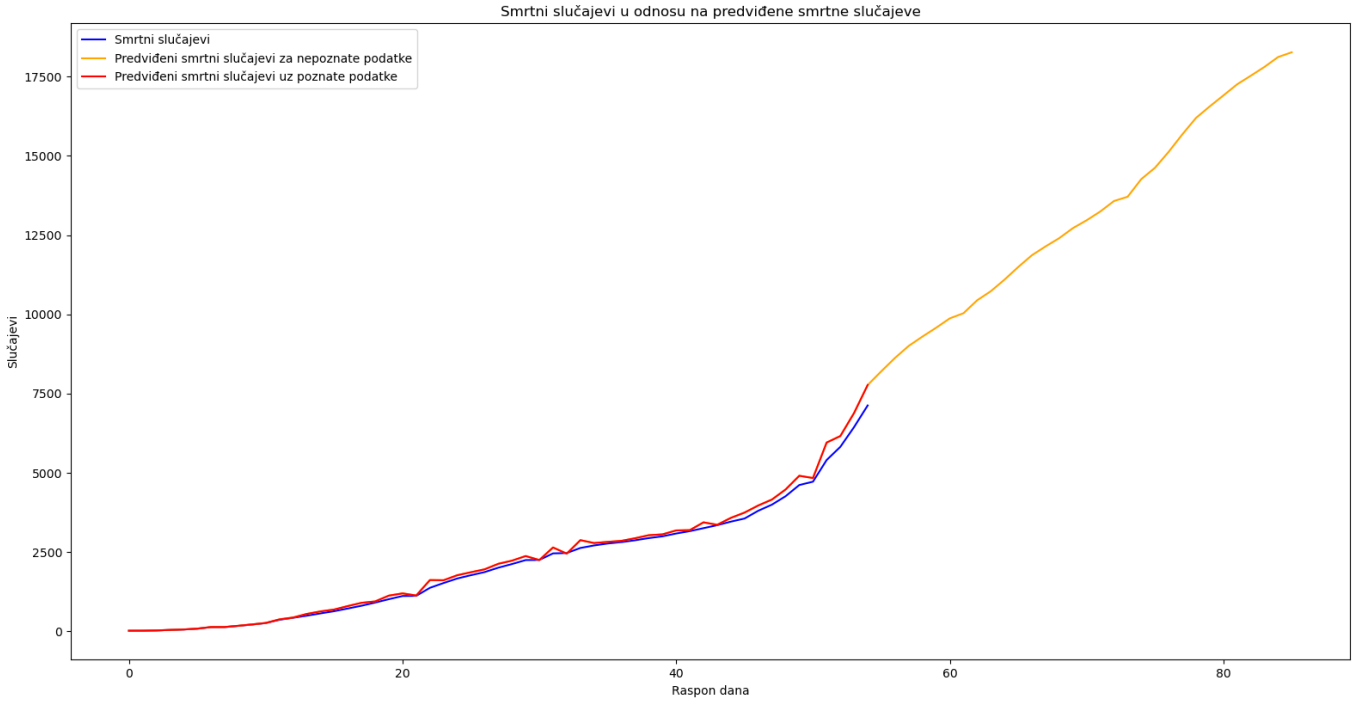
Slika 22. Usporedba stvarnih podataka koronavirusne bolesti sa predviđenim podacima ARIMA modela

5.2.3. SARIMA model

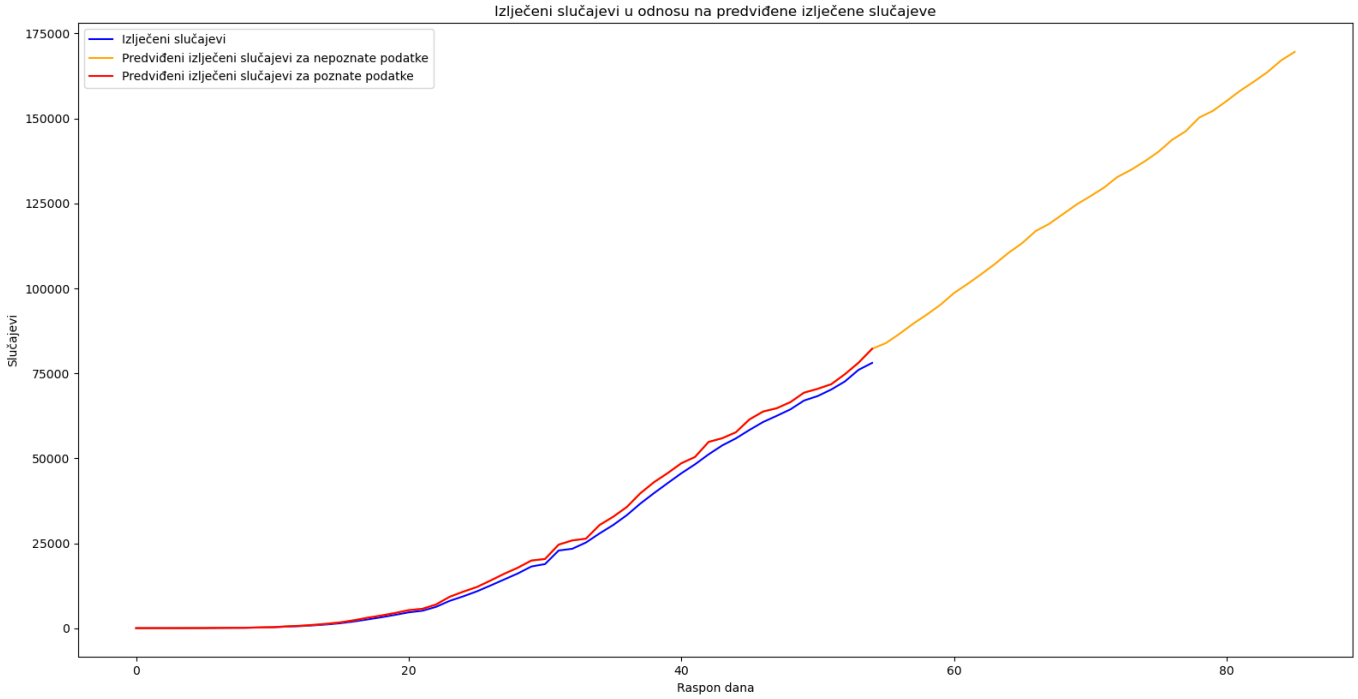
SARIMA (*Seasonal Autoregressive Integrated Moving Average*) model je proširenje ARIMA-e koje podržava izravno modeliranje sezonske komponente serije. Dodaje tri nova hiperparametara za određivanje autoregresije (AR), diferenciranja i pomičnog prosjeka (MA) za sezonsku komponentu serije, kao i dodatni parametar za razdoblje sezonalnosti [19].



Slika 23. Dijagram potvrđenih slučajeva u odnosu na predviđene slučajeve (SARIMA model)

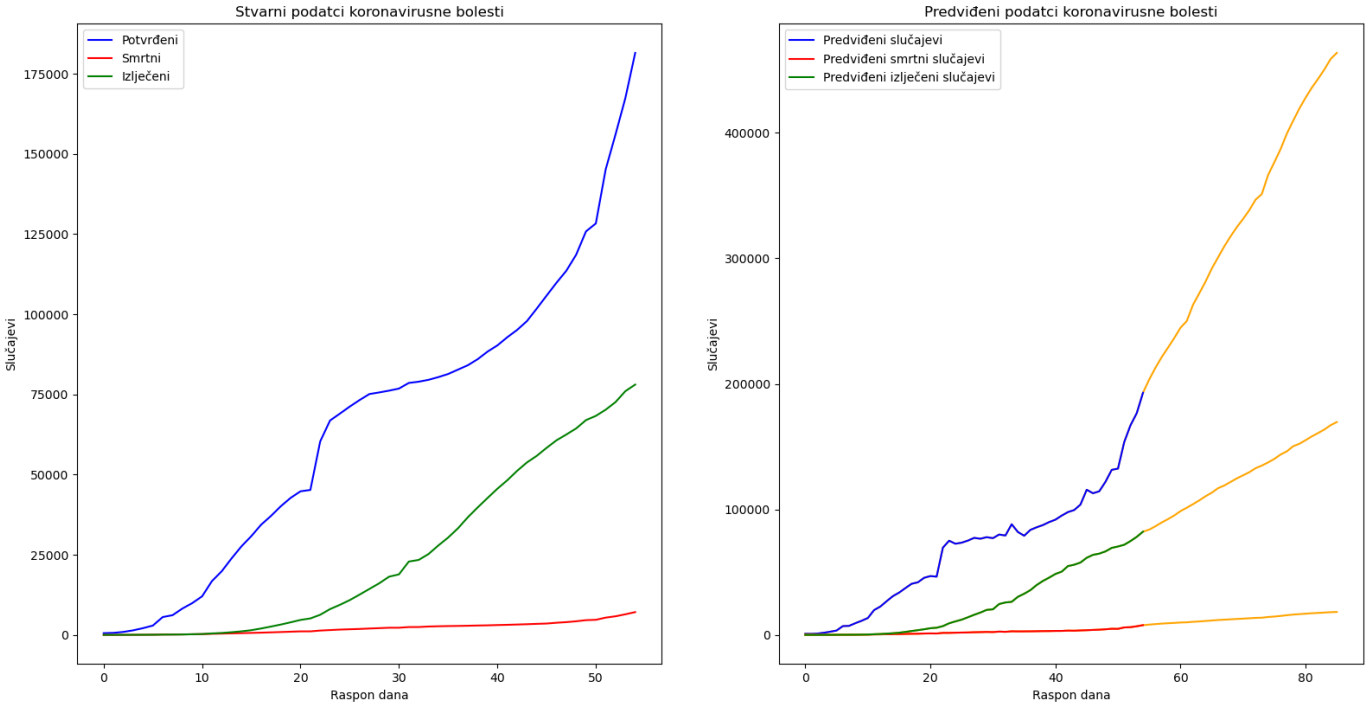


Slika 24. Dijagram smrtnih slučajeva u odnosu na predviđene smrtne slučajeve (SARIMA model)



Slika 25. Dijagram izlječenih slučajeva u odnosu na predviđene izlječene slučajeve (SARIMA model)

I radi adekvatnijeg prikaza, na slici 26. napravit ćemo usporedbu između stvarnih i predviđenih podataka koronavirusne bolesti korištenjem SARIMA modela



Slika 26. Usporedba stvarnih i predviđenih podataka (SARIMA model)

5.2.4. Usporedba AR, ARIMA i SARIMA modela koristeći biblioteku SciPy

Da bismo vidjeli koji je od naših modela najbolje predvidio podatke, upotrijebit ćemo *Spearman* korelaciju. Korelacija će biti blizu vrijednosti 1 jer su to već poznati podaci, ali koristit ćemo je kako bismo prikazali opću svrhu usporedbe podataka pomoću ove korelacije.

Spearman-ova korelacija između dvije varijable jednaka je Pearson-ovoj korelaciji između vrijednosti ranga te dvije varijable; dok Pearson-ova korelacija procjenjuje linearne odnose, Spearman-ova korelacija ocjenjuje monotone odnose (bili oni linearni ili ne).

Ako nema ponovljenih vrijednosti podataka, javlja se savršena Spearman-ova korelacija (+1 ili -1) ako je svaka od varijabli savršena monotona funkcija druge. Spearman-ova korelacija između dvije varijable bit će visoka kada promatranja imaju sličan (ili identičan za korelaciju 1) rang između dvije varijable, a niska kada opažanja imaju različit (ili potpuno suprotstavljen korelaciji -1) rang između dvije varijable [18].

```
#potvrđeni
spearman_ar_potvrđeni = stats.spearmanr(datum_potvrđeni['potvrđeni'],
predvidi_ar_potvrđeni[:len(predvidi_ar_potvrđeni)-31])[0]
spearman_arima_potvrđeni = stats.spearmanr(datum_potvrđeni['potvrđeni'],
predvidi_ma_potvrđeni[:len(predvidi_ma_potvrđeni)-31])[0]
spearman_sarima_potvrđeni = stats.spearmanr(datum_potvrđeni['potvrđeni'],
predvidi_sarima_potvrđeni[:len(predvidi_sarima_potvrđeni)-31])[0]
print()
print("SPEARMAN potvrđeni AR: ", spearman_ar_potvrđeni)
print("SPEARMAN potvrđeni ARIMA: ", spearman_arima_potvrđeni)
print("SPEARMAN potvrđeni SARIMA: ", spearman_sarima_potvrđeni)

#smrtni
spearman_ar_smrtni = stats.spearmanr(datum_smrtni['smrtni'],
predvidi_ar_smrtni[:len(predvidi_ar_smrtni)-31])[0]
spearman_arima_smrtni = stats.spearmanr(datum_smrtni['smrtni'],
predvidi_ma_smrtni[:len(predvidi_ma_smrtni)-31])[0]
spearman_sarima_smrtni = stats.spearmanr(datum_smrtni['smrtni'],
predvidi_sarima_smrtni[:len(predvidi_sarima_smrtni)-31])[0]
print()
print("SPEARMAN smrtni AR: ", spearman_ar_smrtni)
print("SPEARMAN smrtni ARIMA: ", spearman_arima_smrtni)
print("SPEARMAN smrtni SARIMA: ", spearman_sarima_smrtni)
#izliječeni

spearman_ar_izliječeni = stats.spearmanr(datum_izliječeni['izliječeni'],
predvidi_ar_izliječeni[:len(predvidi_ar_izliječeni)-31])[0]
spearman_arima_izliječeni = stats.spearmanr(datum_izliječeni['izliječeni'],
predvidi_ma_izliječeni[:len(predvidi_ma_izliječeni)-31])[0]
spearman_sarima_izliječeni = stats.spearmanr(datum_izliječeni['izliječeni'],
predvidi_sarima_izliječeni[:len(predvidi_sarima_izliječeni)-31])[0]
print()
```

```
print("SPEARMAN izlječeni AR: ", spearman_ar_izlječeni)
print("SPEARMAN izlječeni ARIMA: ", spearman_arima_izlječeni)
print("SPEARMAN izlječeni SARIMA: ", spearman_sarima_izlječeni)
```

Kod navednog koda dobijamo sljedeće rezultate:

SPEARMAN CONFIRMED AR: 0.9998556998557001

SPEARMAN CONFIRMED ARIMA: 0.9993506493506497

SPEARMAN CONFIRMED SARIMA: 0.9973304473304475

SPEARMAN DEATH AR: 0.9999278499278502

SPEARMAN DEATH ARIMA: 1.0

SPEARMAN DEATH SARIMA: 0.9991341991341994

SPEARMAN RECOVERED AR: 1.0

SPEARMAN RECOVERED ARIMA: 0.9997835497835499

SPEARMAN RECOVERED SARIMA: 1.0

Kao što smo spomenuli, vrijednosti će biti visoke zbog već poznatih nam podataka od prije, ali poanta je pokazati kako se to može koristiti. Iz navedenih rezultata, najbolje predviđanja za potvrđene slučajeve napravio je autoregresivni model (AR), za smrtne slučajeve ARIMA model, a za oporavljene slučajeve AR i SARIMA model.

5.3. Predviđanje trenda širenja koronavirusne bolesti (COVID-19) u Python-u pomoću biblioteke Keras

U ovom dijelu stvorit ćemo model dubokog učenja koji će izvesti predviđanje trenda širenja koronavirusne bolesti koristeći biblioteke

- Pandas
- Matplotlib
- Keras

gdje se od najvećeg značaja ističe biblioteka Keras. Keras je biblioteka primjenjena kod dubokog učenja neuronskih mreža u Python-u koja radi na visokoj razini. Sposobna je vršiti izračune poput množenje tenzora, konvolucija i drugih operacija[14, 20]. Ova knjižnica ima

mnoge prednosti, vrlo je jednostavna za korištenje te omogućava izgradnju modela neuronskih mreža u svega nekoliko redaka koda.

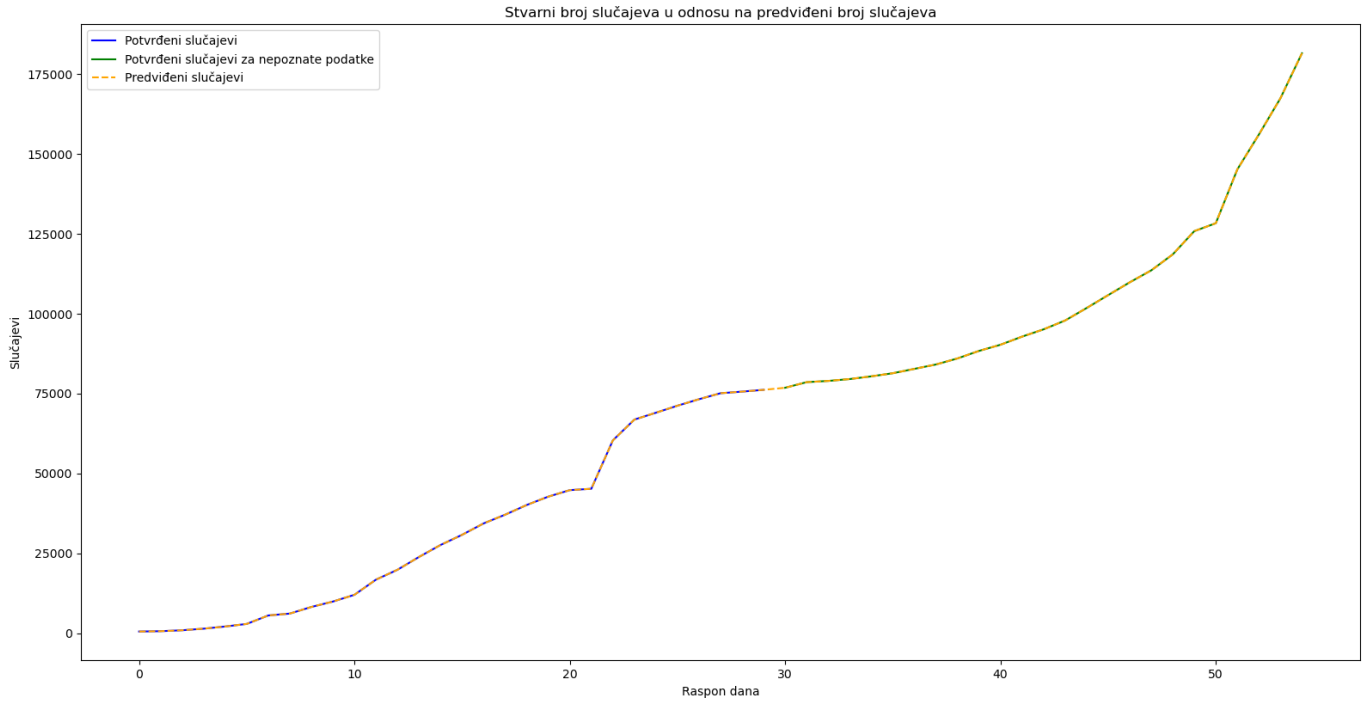
Kao i u ostalim djelovima zadataka, prvo ćemo učitati sve potrebne podatke za kreiranje modela predviđanja uz Pandas biblioteku dok će nam biblioteka matplotlib služiti za crtanje grafova kako bismo imali vizualni prikaz rezultata. Što se tiće keras biblioteke, učitat ćemo sekvencijalni model (eng. *Sequential model*) koji je prikladan za linearne slojeve gdje svaki sloj ima točno jedan ulazni i izlazni tenzor i tip sloja (eng. *layer*) koji će kod nas biti gusti sloj (eng. *dense layer*) [21].

```
model = Sequential()
model.add(Dense(32, activation='relu', input_dim=1))
model.add(Dense(32, activation='relu'))
model.add(Dense(64, activation='relu'))
model.add(Dense(64, activation='relu'))
model.add(Dense(32, activation='relu'))
model.add(Dense(1, activation='softmax'))
model.compile(optimizer='adam', loss='binary_crossentropy',
metrics=['accuracy'])
```

S navedenim kodom stvaramo model koji ćemo koristiti za izvršavanje zadatka. Ovaj model ima 6 slojeva. Prva dva imaju 32 perceptora (algoritam za nadzirano učenje binarnih klasifikatora), druga dva imaju 64 perceptrona, a posljednja 1 perceptron jer nam je potreban samo jedan broj kao izlaz. Svi oni, osim izlaznog sloja, koriste ReLU (eng. *Rectified Linear Unit*) funkciju aktivacije. Nakon što smo dodali slojeve trebamo sastaviti model. Koristimo Adam-ov optimizator (eng. *Adam optimizer*) koji se primjenjuje u tehnikama dubokog učenja. To je algoritam za optimizaciju koji se može koristiti umjesto klasičnog postupka stohastičkog gradijentnog spuštanja za ažuriranje mrežnih iteracija podataka. Koristit ćemo ga za mjerenje funkcije gubitaka koja je binarna unakrsna entropija, a za mjerene podatke ćemo izmjeriti točnost [21].

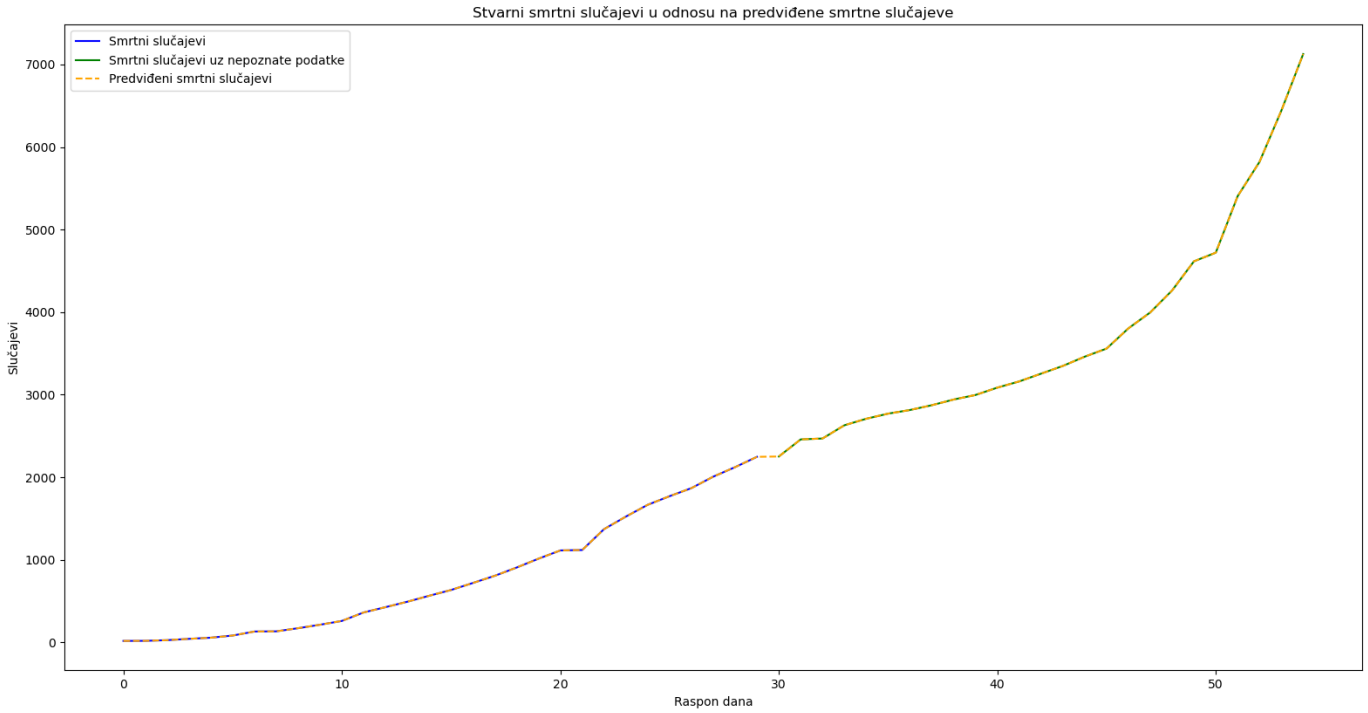
```
model.fit(datum_potvrđeni["potvrđeni"][:30],datum_potvrđeni["potvrđeni"][:3
0],epochs=20,)
predviđanje_potvrđeni = model.predict(datum_potvrđeni["potvrđeni"])
konačno_predviđanje_potvrđeni = []
for i in range(0,len(predviđanje_potvrđeni)):

konačno_predviđanje_potvrđeni.append(predviđanje_potvrđeni[i]*datum_potvrđe
ni["potvrđeni"][i])
```

U ovom dijelu mi podešavamo model i učimo ga prvih 30 dana, a onda koristimo svih 55 dana za predikciju. Jedina je razlika ta što su vrijednosti zadnjih 25 dana nepoznati modelu. Kao što vidimo u gornjem dijelu koda, množimo predviđene vrijednosti sa stvarnim vrijednostima. To je zato što je predviđanje uzorak vjerojatnosti o tome koliko je točna vrijednost predviđanja. Množeći ga sa stvarnom vrijednošću, kao rezultat dobijamo približnu vrijednost.

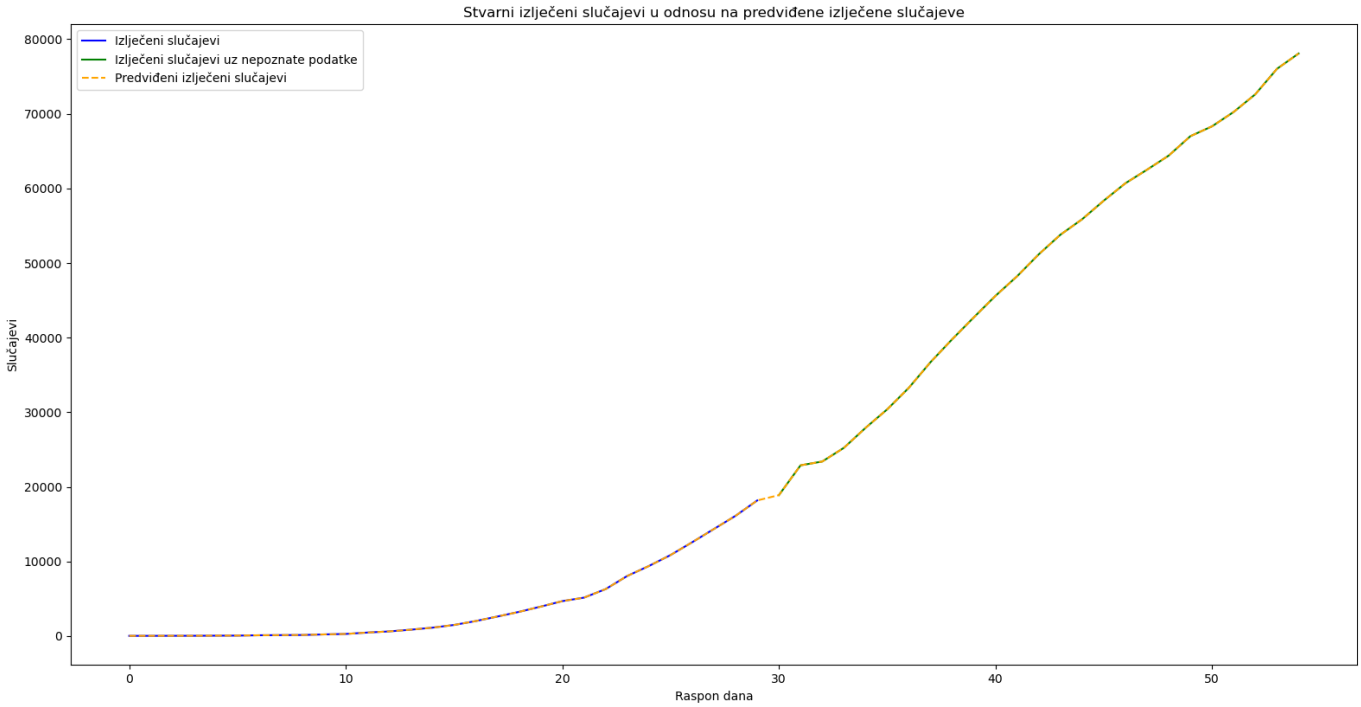


Slika 27. Dijagram stvarnog broja slučajeva u odnosu na predviđeni broj slučajeva

Na slici 27. možemo vidjeti dijagram gdje uspoređujemo predviđanja i stvarne vrijednosti potvrđenih slučajeva koronavirusne bolesti. S plavom bojom označeni su već poznate vrijednosti, sa zelenom bojom imamo nepoznate vrijednosti koje model koristi za predviđanje, a sa žutom isprekidanom linijom označena je funkcija predviđenih vrijednosti. Kao što vidimo, obje funkcije, žuta i plava/zelena (plava i zelena se broje kao jedna funkcija) su iste, što znači da smo napravili poprilično dobro predviđanje. Jedan od razloga je taj što imamo mali skup podataka. Kod za predviđanje smrtnih i izlječenih slučajeva je isti, pa je i analogija kod tih kodova ista.



Slika 28. Dijagram za stvarne smrtne slučajeve u odnosu na predviđene smrtne slučajeve



Slika 29. Dijagram za stvarne izlječene slučajeve u odnosu na predviđene izlječene slučajeve

6. ZAKLJUČAK

Brzo širenje COVID-19-a širom svijeta i sve veći broj smrtnih slučajeva zahtjeva hitne akcije svih mogućih sektora. Predviđanje trenda širenja koronavirusne bolesti može omogućiti učinkovito i efikasno djelovanje vlasti države kako bi smanjili broj potencijalnih zaraza. Nadalje, potrebno je pratiti broj zaraženih ljudi obavljanjem redovitih pregleda, smještanjem zaraženih u karantenu i poštivati mjere samoizolacije za dobrobit svih oko nas. Uz to, treba se ponašati odgovorno i poštivati mjere sigurnosti dane od nadležnih stožera civilne zaštite, znanstvenika i vlasti države kako bi se suzbilo širenje COVID-19-a. Analizom vremenskih serija koje služe za procjenu podataka pohranjenih u vremenskom redoslijedu koristimo za izdvajanje značajnih vrijednosti ili njihovu evaluaciju u određenim dijelovima stohastičkih procesa. Tehnikama dubokog učenja koji se temelje na konvulacijskim neuronskim mrežama pokušavamo predvidjeti širenje trenda pomoću raznih modela i klasa u Python programskom jeziku. Koristeći razne modele predviđanja širenja trenda došli smo do zaključka da modeli Python biblioteke Statsmodels kao i modeli biblioteke Keras pružaju najvjerodostojniju predikciju trenda širenja COVID-19-a, od potvrđenih slučajeva do smrtnih ili izlječenih slučajeva dok Fbprophet modeli trenda širenja poprilično su linearnog rasta što u ovom slučaju čini velika odstupanja jer se bolest eksponencijalno širi diljem svijeta. Koronavirusna bolest vrlo je ozbiljna i svuda je oko nas. Ako je ne prihvatimo kao takvu, može utjecati na naše živote još dugi niz godina kao i na sraz svih gospodarskih grana, ne samo u Hrvatskoj, već i u drugim zemljama. Zato bi naša misija trebala biti da svi djelujemo kao jedno te odgovornim ponašanjem pokušamo suzbiti daljnje širenje virusa.

LITERATURA

[1] COVID-19: https://hr.wikipedia.org/wiki/COVID-19 Pristupljeno: 5.9.2020.

[2] Utjecaj COVID-19-a u svijetu: https://hr.wikipedia.org/wiki/Pandemija_koronavirusa_2019./20. Pristupljeno 5.9.2020.

[3] Pandemija koronavirusa u Hrvatskoj 2020.: https://hr.wikipedia.org/wiki/Pandemija_koronavirusa_u_Hrvatskoj_2020. Pristupljeno: 5.9.2020.

[4] Utjecaj COVID-19-a na Hrvatsko gospodarstvo: https://www.poslovni.hr/hrvatska/hrvatska-medu-tri-najpogodenija-gospodarstva-u-eu-prve-dvije-poharala-je-korona-4240483 Pristupljeno: 5.9.2020.

[5] Industrija 4.0: https://www.happtory.hr/post/industrija-4-0 Pristupljeno 13.9.2020.

[6] HGK Industrija 4.0: https://www.hgk.hr/documents/hgk-industrija-4058d8c59722f1e.pdf Pristupljeno 13.9.2020.

[7] Metode umjetne inteligencije: https://www.hup.hr/EasyEdit/UserFiles/Ivana%20Zlatari%C4%87/hup-ict-de-ai-potencijal-umjetne-inteligencije-za-hrvatsku.pdf Pristupljeno 13.9.2020.

[8] Python: https://www.w3schools.com/python/python_intro.asp Pristupljeno 7.9.2020.

[9] Python: https://en.wikipedia.org/wiki/Python_(programming_language) Pristupljeno 7.9.2020.

[10] Analiza vremenskih serija: https://hr.sciencewal.com/25031-a-quick-start-of-time-series-forecasting-with-a-practical-example-using-fb-prophet-31c4447a2274-52 Pristupljeno: 7.9.2020.

[11] Python libraries: https://www.edureka.co/blog/python-libraries/ Pristupljeno 7.9.2020.

[12] Prophet library: https://facebook.github.io/prophet/ Pristupljeno 7.9.2020.

[13] Statsmodels library: https://www.statsmodels.org/stable/index.html Pristupljeno 7.9.2020.

[14] Keras library: https://hr.sciencewal.com/84577-deep-learning-for-beginners-practical-guide-with-python-and-keras-d295bfca4487-18 Pristupljeno 7.9.2020.

[15] COVID-19 Dashboard by the Center for Systems Science and Engineering (CSSE) at Johns Hopkins University https://gisanddata.maps.arcgis.com/apps/opsdashboard/index.html#/bda7594740fd40299423467b48e9ecf6 Pristupljeno 8.9.2020.

[16] Predviđanje trenda širenja COVID-19-a pomoću biblioteke *fbprophet*

https://laconicml.com/coronavirus-prediction-covid-19/ Pristupljeno 8.9.2020.

[17] Shumway, R. H.; Stoffer, D.S.; Time Series Analysis and Its Applications, Springer-Verlag, New York, 2006.

[18] Predviđanje trenda širenja COVID-19-a pomoću ARIMA modela:

https://laconicml.com/predict-infected-people-coronavirus-python/ Pristupljeno 9.9.2020.

[19] ARIMA modeli za predviđanje vremenskih serija:
https://people.duke.edu/~rnau/411arim3.htm Pristupljeno: 9.9.2020.

[20] Ketkar, N.: Deep Learning with Python, Manning Publications, 2017.

[21] Predviđanje trenda širenja COVID-19-a koristeći tehnike dubokog učenja:
https://laconicml.com/predict-coronavirus-cases-deep-learning/ Pristupljeno: 9.9.2020.

PRILOZI

I. CD-R disc

DODACI

1. *Fbprophet* model

```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
from fbprophet import Prophet

df = pd.read_csv('datasets/covid_19_data.csv',parse_dates=['Last Update'])
df.rename(columns={'ObservationDate':'datum', 'Country/Region':'država'},
inplace=True)




df_datum = df.groupby(["datum"])[['potvrđeni', 'smrtni',
'izliječeni']].sum().reset_index()
df_država = df.groupby(["država"])[['potvrđeni', 'smrtni',
'izliječeni']].sum().reset_index()
#
# print(df_datum.tail())
# print(df_država.tail())
#
datum_x_ticks = [] #datum
država_x_ticks = [] #država
datum_potvrđeni=[] #potvrđenih slučajeva na dnevnoj bazi
datum_smrtni=[]
datum_izliječeni=[]
država_potvrđeni = [] #po državi potvrđenih slučajeva za jedan datum
država_smrtni = []
država_izliječeni = []

for index, row in df_datum.iterrows():

    datum_x_ticks.append(row['datum'])
    datum_potvrđeni.append(row['potvrđeni'])
    datum_smrtni.append(row['smrtni'])
    datum_izliječeni.append(row['izliječeni'])


for index, row in df_država.iterrows():

    država_x_ticks.append(row['država'])
    država_potvrđeni.append(row['potvrđeni'])
    država_smrtni.append(row['smrtni'])
    država_izliječeni.append(row['izliječeni'])

plt.xticks(np.arange(len(datum_x_ticks[:10])), datum_x_ticks[:10])
plt.xlabel('datumi')
plt.ylabel('slucajevi')
plt.plot(datum_potvrđeni[:10], label='potvrđeni', color='blue')
plt.plot(datum_smrtni[:10], label='smrtni', color='red')
plt.plot(datum_izliječeni[:10], label='izliječeni', color='green')
plt.title("koronavirus slucajevi u svijetu datuma")
plt.legend()
plt.show()


plt.xticks(np.arange(len(datum_x_ticks)), datum_x_ticks)
```

```
plt.xlabel('datumi')
plt.ylabel('slucajevi')
plt.plot(datum_potvrđeni, label='potvrđeni', color='blue')
plt.plot(datum_smrtni, label='smrtni', color='red')
plt.plot(datum_izliječeni, label='izliječeni', color='green')
plt.title("koronavirus slucajevi u svijetu datuma")
plt.legend()
plt.show()



datum_potvrđeni_prophet = df_datum[['datum', 'potvrđeni']]
datum_smrtni_prophet = df_datum[['datum', 'smrtni']]
datum_izliječeni_prophet = df_datum[['datum', 'izliječeni']]

datum_potvrđeni_prophet.columns = ['ds', 'y']
datum_smrtni_prophet.columns = ['ds', 'y']
datum_izliječeni_prophet.columns = ['ds', 'y']
#
model_potvrđeni = Prophet(interval_width=0.99)
model_potvrđeni.fit(datum_potvrđeni_prophet)
budućnost_potvrđeni = model_potvrđeni.make_future_dataframe(periods=30)
prognoza_potvrđeni = model_potvrđeni.predict(budućnost_potvrđeni)

print(prognoza_potvrđeni.tail())
#

prognoza_potvrđeni_yhat = []
prognoza_potvrđeni_yhat_u = []
prognoza_potvrđeni_yhat_l = []

for index, row in prognoza_potvrđeni.iterrows():

    prognoza_potvrđeni_yhat.append(row['yhat'])
    prognoza_potvrđeni_yhat_l.append(row['yhat_lower'])
    prognoza_potvrđeni_yhat_u.append(row['yhat_upper'])


plt.xlabel('datumi')
plt.ylabel('slucajevi')
plt.plot(prognoza_potvrđeni_yhat, label='predviđanje', color='blue')
plt.plot(prognoza_potvrđeni_yhat_l, label='predviđanje donje', color='red')
plt.plot(prognoza_potvrđeni_yhat_u, label='predviđanje gornje',
color='green')
plt.title("prognoza potvrđenih slucajeva ")
plt.legend()
plt.show()

#
model_smrtni = Prophet(interval_width=0.99)
model_smrtni.fit(datum_smrtni_prophet)
budućnost_smrtni = model_smrtni.make_future_dataframe(periods=30)
prognoza_smrtni = model_smrtni.predict(budućnost_smrtni)

datumi_prognoza_smrtni = []
prognoza_smrtni_yhat = []
prognoza_smrtni_yhat_u = []
prognoza_smrtni_yhat_l = []

for index, row in prognoza_smrtni.iterrows():
    datumi_prognoza_smrtni.append(row['ds'])
```

```
    prognoza_smrtni_yhat.append(row['yhat'])
    prognoza_smrtni_yhat_l.append(row['yhat_lower'])
    prognoza_smrtni_yhat_u.append(row['yhat_upper'])


plt.xlabel('datumi')
plt.ylabel('slucajevi')
plt.plot(prognoza_smrtni_yhat, label='predviđanje', color='blue')
plt.plot(prognoza_smrtni_yhat_l, label='predviđanje donje', color='red')
plt.plot(prognoza_smrtni_yhat_u, label='predviđanje gornje', color='green')
plt.title("prognoza  smrtnih slucajeva ")
plt.legend()
plt.show()

model_izliječeni = Prophet(interval_width=0.99)
model_izliječeni.fit(datum_izliječeni_prophet)
budućnost_izliječeni = model_izliječeni.make_future_dataframe(periods=30)
prognoza_izliječeni = model_izliječeni.predict(budućnost_izliječeni)

datumi_prognoza_izliječeni = []
prognoza_izliječeni_yhat = []
prognoza_izliječeni_yhat_u = []
prognoza_izliječeni_yhat_l = []

for index, row in prognoza_izliječeni.iterrows():
    datumi_prognoza_izliječeni.append(row['ds'])
    prognoza_izliječeni_yhat.append(row['yhat'])
    prognoza_izliječeni_yhat_l.append(row['yhat_lower'])
    prognoza_izliječeni_yhat_u.append(row['yhat_upper'])


plt.xlabel('datumi')
plt.ylabel('slucajevi')
plt.plot(prognoza_izliječeni_yhat, label='predviđanje', color='blue')
plt.plot(prognoza_izliječeni_yhat_l, label='predviđanje donje', color='red')
plt.plot(prognoza_izliječeni_yhat_u, label='predviđanje gornje',
color='green')
plt.title("prognoza izliječenih slucajeva")
plt.legend()
plt.show()


plt.xlabel('datumi')
plt.ylabel('slucajevi')
plt.plot(prognoza_potvrđeni_yhat, label='potvrđeni', color='blue')
plt.plot(prognoza_smrtni_yhat, label='smrtni', color='red')
plt.plot(prognoza_izliječeni_yhat, label='izliječeni', color='green')
plt.title("prognoza koronavirusa potvrđeni slucajevi")
plt.legend()
plt.show()

datum_potvrđeni_prophet_y = []
datum_izliječeni_prophet_y = []
datum_smrtni_prophet_y = []

for index, row in datum_potvrđeni_prophet.iterrows():
    datum_potvrđeni_prophet_y.append(row['y'])
for index, row in datum_smrtni_prophet.iterrows():
    datum_smrtni_prophet_y.append(row['y'])
for index, row in datum_izliječeni_prophet.iterrows():
    datum_izliječeni_prophet_y.append(row['y'])
```

```
plt.xlabel('datumi')
plt.ylabel('slucajevi')
plt.plot(prognoza_potvrđeni_yhat, label = 'potvrđeni prognoza')
plt.plot(datum_potvrđeni_prophet_y, label = 'potvrđeni')
plt.title("potvrđeni vs potvrđeni prognoza")
plt.legend()
plt.show()

plt.xlabel('datumi')
plt.ylabel('slucajevi')
plt.plot(prognoza_smrtni_yhat, label = 'smrtni prognoza')
plt.plot(datum_smrtni_prophet_y, label = 'smrtni')
plt.title("smrtni vs smrtni prognoza")
plt.legend()
plt.show()

plt.xlabel('datumi')
plt.ylabel('slucajevi')
plt.plot(prognoza_izliječeni_yhat, label = 'izliječeni prognoza')
plt.plot(datum_izliječeni_prophet_y, label = 'izliječeni')
plt.title("izliječeni vs izliječeni prognoza")
plt.legend()
plt.show()
```

2. *Statsmodels* model

```
df = pd.read_csv('datasets/covid_19_data.csv',parse_dates=['Last Update'])
df.rename(columns={'ObservationDate':'datum', 'Country/Region':'država'},
inplace=True)

df_datum = df.groupby(["datum"])[['potvrđeni', 'smrtni',
'izliječeni']].sum().reset_index()
df_država = df.groupby(["država"])[['potvrđeni', 'smrtni',
'izliječeni']].sum().reset_index()

datum_x_ticks = [] #datum
država_x_ticks = [] #država
datum_potvrđeni=[] #potvrđenih slučajeva na dnevnoj bazi
datum_smrtni=[]
datum_izliječeni=[]
država_potvrđeni = [] #po državi potvrđenih slučajeva za jedan datum
država_smrtni = []
država_izliječeni = []

from statsmodels.tsa.ar_model import AR

datum_potvrđeni = df_datum[['datum', 'potvrđeni']]
datum_smrtni = df_datum[['datum', 'smrtni']]
datum_izliječeni = df_datum[['datum', 'izliječeni']]
print(datum_smrtni)
for index, row in datum_potvrđeni.iterrows():
    if row['potvrđeni'] is None:
        row['potvrđeni'] = 0.0

for index, row in datum_smrtni.iterrows():
    if row['smrtni'] is None:
        row['smrtni'] = 0.0
```

```
for index, row in datum_izliječeni.iterrows():
    if row['izliječeni'] is None:
        row['izliječeni'] = 0.0


model_ar_potvrđeni = AR(np.asanyarray(datum_potvrđeni['potvrđeni']))
model_fit_ar_potvrđeni = model_ar_potvrđeni.fit()
predvidi_ar_potvrđeni = model_fit_ar_potvrđeni.predict(10,
len(datum_potvrđeni) + 40)
print(predvidi_ar_potvrđeni)
plt.xlabel('datumi')
plt.ylabel('slucajevi')
plt.plot(datum_potvrđeni['potvrđeni'], label='potvrđeni', color='blue')
plt.plot(predvidi_ar_potvrđeni, label='predviđeni nepoznati podatci',
color='orange')
plt.plot(predvidi_ar_potvrđeni[:len(predvidi_ar_potvrđeni)-30],
label='predviđeni poznati podatci', color='red')
plt.title('potvrđeni slucajevi vs predviđeni potvrđeni slucajevi')
plt.legend()
plt.show()

model_ar_smrtni = AR(np.asanyarray(datum_smrtni['smrtni']))
model_fit_ar_smrtni = model_ar_smrtni.fit()
predvidi_ar_smrtni = model_fit_ar_smrtni.predict(10, len(datum_smrtni) +
40)
print(predvidi_ar_smrtni)
plt.xlabel('datumi')
plt.ylabel('slucajevi')
plt.plot(datum_smrtni['smrtni'], label='smrtni', color='blue')
plt.plot(predvidi_ar_smrtni, label='predviđeni nepoznati podatci',
color='orange')
plt.plot(predvidi_ar_smrtni[:len(predvidi_ar_smrtni)-30], label='predviđeni
poznati podatci', color='red')
plt.title('smrtni slucajevi vs predviđeni smrtni slucajevi')
plt.legend()
plt.show()

model_ar_izliječeni = AR(np.asanyarray(datum_izliječeni['izliječeni']))
model_fit_ar_izliječeni = model_ar_izliječeni.fit()
predvidi_ar_izliječeni = model_fit_ar_izliječeni.predict(10,
len(datum_izliječeni) + 40)
print(predvidi_ar_izliječeni)
plt.xlabel('datumi')
plt.ylabel('slucajevi')
plt.plot(datum_izliječeni['izliječeni'], label='izliječeni', color='blue')
plt.plot(predvidi_ar_izliječeni, label='predviđeni nepoznati podatci',
color='orange')
plt.plot(predvidi_ar_izliječeni[:len(predvidi_ar_izliječeni)-30],
label='predviđeni poznati podatci', color='red')
plt.title('izliječeni slucajevi vs predviđeni izliječeni slucajevi')
plt.legend()
plt.show()

plt.subplot(121)
plt.title("podatci koronavirus")
plt.plot(datum_potvrđeni['potvrđeni'], label='potvrđeni', color='blue')
plt.plot(datum_smrtni['smrtni'], label='smrtni', color='red')
plt.plot(datum_izliječeni['izliječeni'], label='izliječeni', color='green')
plt.legend()
plt.subplot(122)
plt.title("podatci koronavirus predviđeni")
```

```
plt.plot(predvidi_ar_potvrđeni, color='orange')
plt.plot(predvidi_ar_potvrđeni[:len(predvidi_ar_potvrđeni)-30],
label='predviđeni potvrđeni ', color='blue')
plt.plot(predvidi_ar_smrtni, color = 'orange')
plt.plot(predvidi_ar_smrtni[:len(predvidi_ar_smrtni)-30], label='predviđeni
smrtni', color = 'red')
plt.plot(predvidi_ar_izlječeni,  color = 'orange')
plt.plot(predvidi_ar_izlječeni[:len(predvidi_ar_izlječeni)-30],
label='predviđeni izlječeni', color = 'green')
plt.legend()
plt.show()




import statsmodels.api as sm

fig = plt.figure()
ax1 = fig.add_subplot(211)
fig = sm.graphics.tsa.plot_acf(datum_potvrđeni['potvrđeni'], lags=10,
ax=ax1) #
ax2 = fig.add_subplot(212)
fig = sm.graphics.tsa.plot_pacf(datum_potvrđeni['potvrđeni'], lags=10,
ax=ax2)#
plt.show()

from statsmodels.tsa.arima_model import ARIMA
model_ma_potvrđeni = ARIMA(np.asanyarray(datum_potvrđeni['potvrđeni']),
order=(2,0,0))
model_fit_ma_potvrđeni = model_ma_potvrđeni.fit(disp=False)
predvidi_ma_potvrđeni = model_fit_ma_potvrđeni.predict(1,
len(datum_potvrđeni)+31)
print(predvidi_ma_potvrđeni)
plt.xlabel('datumi')
plt.ylabel('slucajevi')
plt.plot(datum_potvrđeni['potvrđeni'], label='potvrđeni', color='blue')
plt.plot(predvidi_ma_potvrđeni, label='predviđeni nepoznati podatci',
color='orange')
plt.plot(predvidi_ma_potvrđeni[:len(predvidi_ma_potvrđeni)-31],
label='predviđeni poznati podatci', color='red')
plt.title('potvrđeni slucajevi vs predviđeni potvrđeni slucajevi')
plt.legend()
plt.show()

fig = plt.figure()

ax1 = fig.add_subplot(211)
fig = sm.graphics.tsa.plot_acf(datum_smrtni['smrtni'], lags=10, ax=ax1) #
ax2 = fig.add_subplot(212)
fig = sm.graphics.tsa.plot_pacf(datum_smrtni['smrtni'], lags=10, ax=ax2)#
plt.show()


model_ma_smrtni = ARIMA(np.asanyarray(datum_smrtni['smrtni']),  order=(1,
0, 0))
model_fit_ma_smrtni = model_ma_smrtni.fit(disp=False)
predvidi_ma_smrtni = model_fit_ma_smrtni.predict(1, len(datum_smrtni) + 31)
print(predvidi_ma_smrtni)
plt.xlabel('datumi')
plt.ylabel('slucajevi')
```

```
plt.plot(datum_smrtni['smrtni'], label='smrtni', color='blue')
plt.plot(predvidi_ma_smrtni, label='predviđeni nepoznati podatci',
color='orange')
plt.plot(predvidi_ma_smrtni[:len(predvidi_ma_smrtni)-31], label='predviđeni
poznati podatci', color='red')
plt.title('smrtni slucajevi vs predviđeni smrtni slucajevi')
plt.legend()
plt.show()
fig = plt.figure()
ax1 = fig.add_subplot(211)
fig = sm.graphics.tsa.plot_acf(datum_izliječeni['izliječeni'], lags=10,
ax=ax1) #
ax2 = fig.add_subplot(212)
fig = sm.graphics.tsa.plot_pacf(datum_izliječeni['izliječeni'], lags=10,
ax=ax2)#
plt.show()

model_ma_izliječeni = ARIMA(np.asanyarray(datum_izliječeni['izliječeni']),
order=(2,0, 0))
model_fit_ma_izliječeni = model_ma_izliječeni.fit(disp=False)
predvidi_ma_izliječeni = model_fit_ma_izliječeni.predict(1,
len(datum_izliječeni) + 31)
print(predvidi_ma_izliječeni)
plt.xlabel('datumi')
plt.ylabel('slucajevi')
plt.plot(datum_izliječeni['izliječeni'], label='izliječeni', color='blue')
plt.plot(predvidi_ma_izliječeni, label='predviđeni nepoznati podatci',
color='orange')
plt.plot(predvidi_ma_izliječeni[:len(predvidi_ma_izliječeni)-31],
label='predviđeni poznati podatci', color='red')
plt.title('izliječeni slucajevi vs predviđeni izliječeni slucajevi')
plt.legend()
plt.show()


plt.subplot(121)
plt.title("podatci koronavirus")
plt.plot(datum_potvrđeni['potvrđeni'], label='potvrđeni', color='blue')
plt.plot(datum_smrtni['smrtni'], label='smrtni', color='red')
plt.plot(datum_izliječeni['izliječeni'], label='izliječeni', color='green')
plt.legend()
plt.subplot(122)
plt.title("podatci koronavirus predviđeni")
plt.plot(predvidi_ma_potvrđeni, color='orange')
plt.plot(predvidi_ma_potvrđeni[:len(predvidi_ma_potvrđeni)-31],
label='predviđeni potvrđeni ', color='blue')
plt.plot(predvidi_ma_smrtni, color = 'orange')
plt.plot(predvidi_ma_smrtni[:len(predvidi_ma_smrtni)-31], label='predviđeni
smrtni', color = 'red')
plt.plot(predvidi_ma_izliječeni,  color = 'orange')
plt.plot(predvidi_ma_izliječeni[:len(predvidi_ma_izliječeni)-31],
label='predviđeni izliječeni', color = 'green')
plt.legend()
plt.show()




from statsmodels.tsa.statespace.sarimax import SARIMAX
```

```
model_sarima_potvrđeni =
SARIMAX(np.asanyarray(datum_potvrđeni['potvrđeni']),  order=(2,1,0),
seasonal_order=(1,1,0,12))
model_fit_sarima_potvrđeni = model_sarima_potvrđeni.fit(disp=False,
enforce_stationarity=False)
predvidi_sarima_potvrđeni = model_fit_sarima_potvrđeni.predict(1,
len(datum_potvrđeni)+31)
print(predvidi_sarima_potvrđeni)
plt.xlabel('datumi')
plt.ylabel('slucajevi')
plt.plot(datum_potvrđeni['potvrđeni'], label='potvrđeni', color='blue')
plt.plot(predvidi_sarima_potvrđeni, label='predviđeni nepoznati podatci',
color='orange')
plt.plot(predvidi_sarima_potvrđeni[:len(predvidi_sarima_potvrđeni)-31],
label='predviđeni poznati podatci', color='red')
plt.title('potvrđeni slucajevi vs predviđeni potvrđeni slucajevi')
plt.legend()
plt.show()

model_sarima_smrtni = SARIMAX(np.asanyarray(datum_smrtni['smrtni']),
order=(1,1,0), seasonal_order=(1,1,0,12))
model_fit_sarima_smrtni = model_sarima_smrtni.fit(disp=False,
enforce_stationarity=False)
predvidi_sarima_smrtni = model_fit_sarima_smrtni.predict(1,
len(datum_smrtni)+31)
print(predvidi_sarima_smrtni)
plt.xlabel('datumi')
plt.ylabel('slucajevi')
plt.plot(datum_smrtni['smrtni'], label='smrtni', color='blue')
plt.plot(predvidi_sarima_smrtni, label='predviđeni nepoznati podatci',
color='orange')
plt.plot(predvidi_sarima_smrtni[:len(predvidi_sarima_smrtni)-31],
label='predviđeni poznati podatci', color='red')
plt.title('smrtni slucajevi vs predviđeni smrtni slucajevi')
plt.legend()
plt.show()

model_sarima_izliječeni =
SARIMAX(np.asanyarray(datum_izliječeni['izliječeni']),  order=(2,1,0),
seasonal_order=(1,1,0,12))
model_fit_sarima_izliječeni = model_sarima_izliječeni.fit(disp=False,
enforce_stationarity=False)
predvidi_sarima_izliječeni = model_fit_sarima_izliječeni.predict(1,
len(datum_izliječeni)+31)
print(predvidi_sarima_izliječeni)
plt.xlabel('datumi')
plt.ylabel('slucajevi')
plt.plot(datum_izliječeni['izliječeni'], label='izliječeni', color='blue')
plt.plot(predvidi_sarima_izliječeni, label='predviđeni nepoznati podatci',
color='orange')
plt.plot(predvidi_sarima_izliječeni[:len(predvidi_sarima_izliječeni)-31],
label='predviđeni poznati podatci', color='red')
plt.title('izliječeni slucajevi vs predviđeni izliječeni slucajevi')
plt.legend()
plt.show()

plt.subplot(121)
plt.title("podatci koronavirus")
plt.plot(datum_potvrđeni['potvrđeni'], label='potvrđeni', color='blue')
plt.plot(datum_smrtni['smrtni'], label='smrtnini', color='red')
plt.plot(datum_izliječeni['izliječeni'], label='izliječeni', color='green')
```

```
plt.legend()
plt.subplot(122)
plt.title("podatci koronavirus predviđeni")
plt.plot(predvidi_sarima_potvrđeni, color='orange')
plt.plot(predvidi_sarima_potvrđeni[:len(predvidi_sarima_potvrđeni)-31],
label='predviđeni potvrđeni ', color='blue')
plt.plot(predvidi_sarima_smrtni, color = 'orange')
plt.plot(predvidi_sarima_smrtni[:len(predvidi_sarima_smrtni)-31],
label='predviđeni smrtni', color = 'red')
plt.plot(predvidi_sarima_izliječeni,  color = 'orange')
plt.plot(predvidi_sarima_izliječeni[:len(predvidi_sarima_izliječeni)-31],
label='predviđeni izliječeni', color = 'green')
plt.legend()
plt.show()

import scipy.stats as stats

#potvrđeni
spearman_ar_potvrđeni = stats.spearmanr(datum_potvrđeni['potvrđeni'],
predvidi_ar_potvrđeni[:len(predvidi_ar_potvrđeni)-31])[0]
spearman_arima_potvrđeni = stats.spearmanr(datum_potvrđeni['potvrđeni'],
predvidi_ma_potvrđeni[:len(predvidi_ma_potvrđeni)-31])[0]
spearman_sarima_potvrđeni = stats.spearmanr(datum_potvrđeni['potvrđeni'],
predvidi_sarima_potvrđeni[:len(predvidi_sarima_potvrđeni)-31])[0]
print()
print("SPEARMAN potvrđeni AR: ", spearman_ar_potvrđeni)
print("SPEARMAN potvrđeni ARIMA: ", spearman_arima_potvrđeni)
print("SPEARMAN potvrđeni SARIMA: ", spearman_sarima_potvrđeni)

#smrtni
spearman_ar_smrtni = stats.spearmanr(datum_smrtni['smrtni'],
predvidi_ar_smrtni[:len(predvidi_ar_smrtni)-31])[0]
spearman_arima_smrtni = stats.spearmanr(datum_smrtni['smrtni'],
predvidi_ma_smrtni[:len(predvidi_ma_smrtni)-31])[0]
spearman_sarima_smrtni = stats.spearmanr(datum_smrtni['smrtni'],
predvidi_sarima_smrtni[:len(predvidi_sarima_smrtni)-31])[0]
print()
print("SPEARMAN smrtni AR: ", spearman_ar_smrtni)
print("SPEARMAN smrtni ARIMA: ", spearman_arima_smrtni)
print("SPEARMAN smrtni SARIMA: ", spearman_sarima_smrtni)
#izliječeni

spearman_ar_izliječeni = stats.spearmanr(datum_izliječeni['izliječeni'],
predvidi_ar_izliječeni[:len(predvidi_ar_izliječeni)-31])[0]
spearman_arima_izliječeni = stats.spearmanr(datum_izliječeni['izliječeni'],
predvidi_ma_izliječeni[:len(predvidi_ma_izliječeni)-31])[0]
spearman_sarima_izliječeni = stats.spearmanr(datum_izliječeni['izliječeni'],
predvidi_sarima_izliječeni[:len(predvidi_sarima_izliječeni)-31])[0]
print()
print("SPEARMAN izliječeni AR: ", spearman_ar_izliječeni)
print("SPEARMAN izliječeni ARIMA: ", spearman_arima_izliječeni)
print("SPEARMAN izliječeni SARIMA: ", spearman_sarima_izliječeni)
```

3. *Keras* model

```
import pandas as pd
from matplotlib import pyplot as plt
from keras.models import Sequential
from keras.layers import Dense

df = pd.read_csv('datasets/covid_19_data_original.csv',parse_dates=['Last
Update'])
df.rename(columns={'ObservationDate':'datum', 'Country/Region':'država'},
inplace=True)
df_datum = df.groupby(["datum"])[['potvrđeni', 'smrtni',
'izliječeni']].sum().reset_index()

datum_potvrđeni = df_datum[['datum', 'potvrđeni']]
datum_smrtni = df_datum[['datum', 'smrtni']]
datum_izliječeni = df_datum[['datum', 'izliječeni']]

for index, row in datum_potvrđeni.iterrows():
    if row['potvrđeni'] is None:
        row['potvrđeni'] = 0.0

for index, row in datum_smrtni.iterrows():
    if row['smrtni'] is None:
        row['smrtni'] = 0.0

for index, row in datum_izliječeni.iterrows():
    if row['izliječeni'] is None:
        row['izliječeni'] = 0.0


model = Sequential()
model.add(Dense(32, activation='relu', input_dim=1))
model.add(Dense(32, activation='relu'))
model.add(Dense(64, activation='relu'))
model.add(Dense(64, activation='relu'))
model.add(Dense(32, activation='relu'))
model.add(Dense(1, activation='softmax'))
model.compile(optimizer='adam', loss='binary_crossentropy',
metrics=['accuracy'])


model.fit(datum_potvrđeni["potvrđeni"][:30],datum_potvrđeni["potvrđeni"][:3
0],epochs=20,)
predviđanje_potvrđeni = model.predict(datum_potvrđeni["potvrđeni"])
konačno_predviđanje_potvrđeni = []
for i in range(0,len(predviđanje_potvrđeni)):

konačno_predviđanje_potvrđeni.append(predviđanje_potvrđeni[i]*datum_potvrđe
ni["potvrđeni"][i])

plt.title("stvarni vs predviđanje za potvrđene slučajeve")
plt.plot(datum_potvrđeni['potvrđeni'][:30], label='potvrđeni',
color='blue')
plt.plot(datum_potvrđeni['potvrđeni'][30:], label='potvrđeni poznati',
color='green')
plt.plot(konačno_predviđanje_potvrđeni, label='predviđanje',
linestyle='dashed', color='orange')
plt.legend()
```

```
plt.show()


model.fit(datum_smrtni["smrtni"][:30],datum_smrtni["smrtni"][:30],epochs=20
,)
predviđanje_smrtni = model.predict(datum_smrtni["smrtni"])
konačno_predviđanje_smrtni = []
for i in range(0,len(predviđanje_smrtni)):

konačno_predviđanje_smrtni.append(predviđanje_smrtni[i]*datum_smrtni["smrtn
i"][i])

print(konačno_predviđanje_smrtni)
plt.title("stvarni vs predviđanje za smrtne slučajeve")
plt.xlabel('datums')
plt.ylabel('slučajevi')
plt.plot(datum_smrtni['smrtni'][:30], label='smrtni', color='blue')
plt.plot(datum_smrtni['smrtni'][30:], label='smrtni poznati',
color='green')
plt.plot(konačno_predviđanje_smrtni, label='predviđanje',
linestyle='dashed', color='orange')
plt.legend()
```