R Shiny web aplikacija za praćenje podataka o Covid-19 pandemiji i kratkoročno predviđanje kretanja pandemije

Bardić, Lucija

Sveučilište J. J. Strossmayera u Osijeku
Odjel za matematiku
Sveučilišni preddiplomski studij matematike i računarstva

Lucija Bardić

R Shiny web aplikacija za praćenje podataka o Covid-19 pandemiji i kratkoročno predviđanje kretanja pandemije

Završni rad

Osijek, 2022.

Sveučilište J. J. Strossmayera u Osijeku
Odjel za matematiku
Sveučilišni preddiplomski studij matematike i računarstva

Lucija Bardić

R Shiny web aplikacija za praćenje podataka o Covid-19 pandemiji i kratkoročno predviđanje kretanja pandemije

Završni rad

Mentor: izv. prof. dr. sc. Danijel Grahovac

Osijek, 2022.

Sažetak

U ovom radu bit će objašnjeno korištenje paketa Shiny iz programskog jezika R koji služi za izradu interaktivnih web aplikacija. Kao primjer bit će objašnjena web aplikacija koja automatski svakodnevno ažurira podatke vezane uz pandemiju koronavirusa (broj novozaraženih, izliječenih, umrlih i sl.) na osnovu javno dostupnih izvora. Također, u drugom dijelu rada bit će predstavljen model koji služi za kratkoročno predviđanje kretanja zaraze.

Ključne riječi

Covid-19, Shiny, model

R Shiny web application for tracking Covid-19 pandemic data and short-term prediction of progression of the pandemic

Abstract

This paper will explain usage of Shiny, a package from programming language R, which helps in making interactive web applications. As an example, the paper will present a web application that makes automatic daily updates of data regarding Covid-19 pandemic (number of new cases, recovered, deaths etc.) using open source databases. Also, the second part of the paper will present a model used for short-term prediction of progression of the pandemic.

Keywords

Covid-19, Shiny, model

Sadržaj

1 Uvod

1.1 Opis problema

Bolest koronavirus (COVID-19), koja je prvi put otkrivena krajem 2019. u Kini, brzo se proširila na Europu i Ameriku, a zatim i na ostatak svijeta. Zbog toga je Svjetska zdravstvena organizacija u ožujku 2020. godine proglasila pandemiju na globalnoj razini. Najčešći simptomi su blago povišena tjelesna temperatura, suhi kašalj, osjećaj umora, bol u mišićima i gubitak osjeta mirisa i okusa. Kako je do ovog problema došlo u ovo suvremeno doba, kada su tehnološka dostignuća na vrhuncu, postoje mnogi izvori, podaci i informacije vezane za pandemiju što uvelike olakšava njezino proučavanje. U nastavku ovog rada bit će objašnjena aplikacija koja je nastala na temelju proučavanja navedenih podataka; aplikacija prati broj novozaraženih, umrlih, izliječenih i cijepljenih kako u Hrvatskoj tako i u svijetu, daje općenite informacije u virusu te definira model kratkoročnog predviđanja kretanja pandemije. Cilj aplikacije je praćenje promjene u kretanju pandemije i mogućnost donošenja zaključaka na temelju iste.
Link za web aplikaciju: http://tesla.mathos.hr:3838/covid-19/
Pripadni kod: https://github.com/akoturic/COVID-19-RShiny-App

2 Shiny

R je programski jezik koji se prvenstveno koristi za statističko računanje i grafiku. Poznato je da je jezik prilično nekonvencionalan u usporedbi s popularnim jezicima za razvoj softvera, poput C ++ ili Java. Ono što R izdvaja od većine drugih jezika jest to što djeluje kao interaktivno statističko okruženje. Ono što R čini posebno pogodnim za rad je:

- mogućnost za učinkovitim upravljanjem podacima,
- velika i integrirana zbirka programskih alata za analizu podataka,
- grafičke mogućnosti za prikazivanje podataka.

Sve funkcije i podaci u R-u su spremljeni u paketima. Kada je paket učitan njegov sadržaj postaje dostupan. Shiny je paket u R-u koji omogućava izradu interaktivnih web aplikacija koje mogu izvršavati R kod u pozadini. Ovaj paket omogućava izradu web aplikacije koje se mogu pokrenuti na računalu, postaviti na vlastiti server ili koristiti RStudio cloud.

```
install.packages("shiny")
library(shiny)
```

Primjer 1: Instalacija i učitavanje Shiny paketa

Shiny aplikacija sprema se u skriptu app.R u kojoj je opisano kako web aplikacija izgleda i kako se ponaša. Pokretanje aplikacije može se izvršiti na nekoliko načina:

- klikom na **Run App** unutar RStudia
- korištenjem prečaca na tiplovnici **Cmd/Ctrl+Shift+Enter**
- ukoliko se ne koristi RStudio, pozivanjem naredbe **shiny::RunApp()**

```r
#ucitavanje Shiny biblioteke
library(shiny)
ui <- fluidPage(
  "Hello, world!"
)
server <- function(input, output, session) {
}
#pokretanje Shiny aplikacije
shinyApp(ui, server)
```

Primjer 2: Jednostavna Shiny aplikacija

Osnovna Shiny aplikacija sastoji se od nekoliko dijelova:

- ui: ugniježđena R funkcija koja stvara HTML korisničko sučelje za aplikaciju,
- server: funkcija s instrukcijama o prikazivanju i brisanju R objekata prikazanih na korisničkom sučelju,
- shinyApp: poziv koji kombinira ui i server u funkcionalnu aplikaciju.

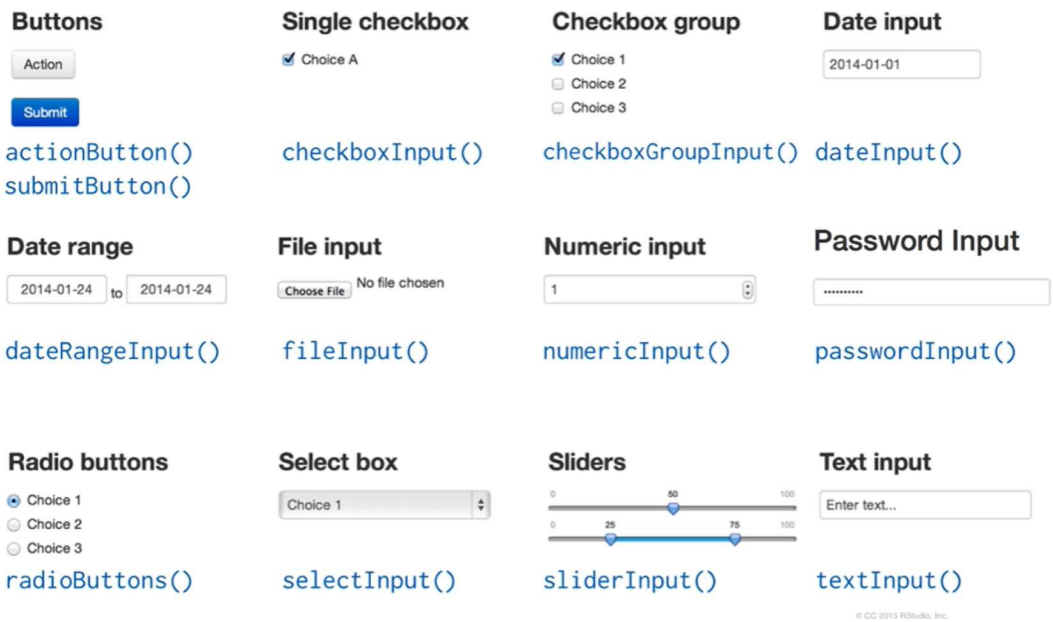
2.1 Funkcije ui i server

U svome najjednostavnijem obliku, Shiny aplikacije imaju dvije komponente, funkciju ui koja definira kako će aplikacija izgledati i funkciju server koja opisuje na koji način radi aplikacija. Kako bi se izgradila prava, funkcionalna aplikacija, unutar funkcije ui pozivaju se već ugrađene Shiny funkcije ili funkcije koje je definirao sam korisnik. Sav kod Shiny pretvara u HTML kod koji je potreban kako bi se aplikacija mogla prikazati na webu. Osim korištenja gotovih R funkcija za slaganje izgleda korisničkog sučelja, unutar R-a moguće je koristiti i već poznate HTML elemente. Također, CSS i Javascript datoteke mogu se uključiti u R skriptu kako bismo dobili ljepši i pregledniji dizajn korisničkog sučelja.

tags$a	tags$data	tags$h6	tags$nav	tags$span
tags$abbr	tags$datalist	tags$head	tags$noscript	tags$strong
tags$address	tags$dd	tags$header	tags$object	tags$style
tags$area	tags$del	tags$hgroup	tags$ol	tags$sub
tags$article	tags$details	tags$hr	tags$optgroup	tags$summary
tags$aside	tags$dfn	tags$HTML	tags$option	tags$sup
tags$audio	tags$div	tags$i	tags$output	tags$table
tags$b	tags$dl	tags$iframe	tags$p	tags$tbody
tags$base	tags$dt	tags$img	tags$param	tags$td
tags$bdi	tags$em	tags$input	tags$pre	tags$textarea
tags$bdo	tags$embed	tags$ins	tags$progress	tags$tfoot
tags$blockquote	tags$eventsource	tags$kbd	tags$q	tags$th
tags$body	tags$fieldset	tags$keygen	tags$ruby	tags$thead
tags$br	tags$figcaption	tags$label	tags$rp	tags$time
tags$button	tags$figure	tags$legend	tags$rt	tags$title
tags$canvas	tags$footer	tags$li	tags$s	tags$tr
tags$caption	tags$form	tags$link	tags$samp	tags$track
tags$cite	tags$h1	tags$mark	tags$script	tags$u
tags$code	tags$h2	tags$map	tags$section	tags$ul
tags$col	tags$h3	tags$menu	tags$select	tags$var
tags$colgroup	tags$h4	tags$meta	tags$small	tags$video
		tags$meter	tags$source	tags$wbr

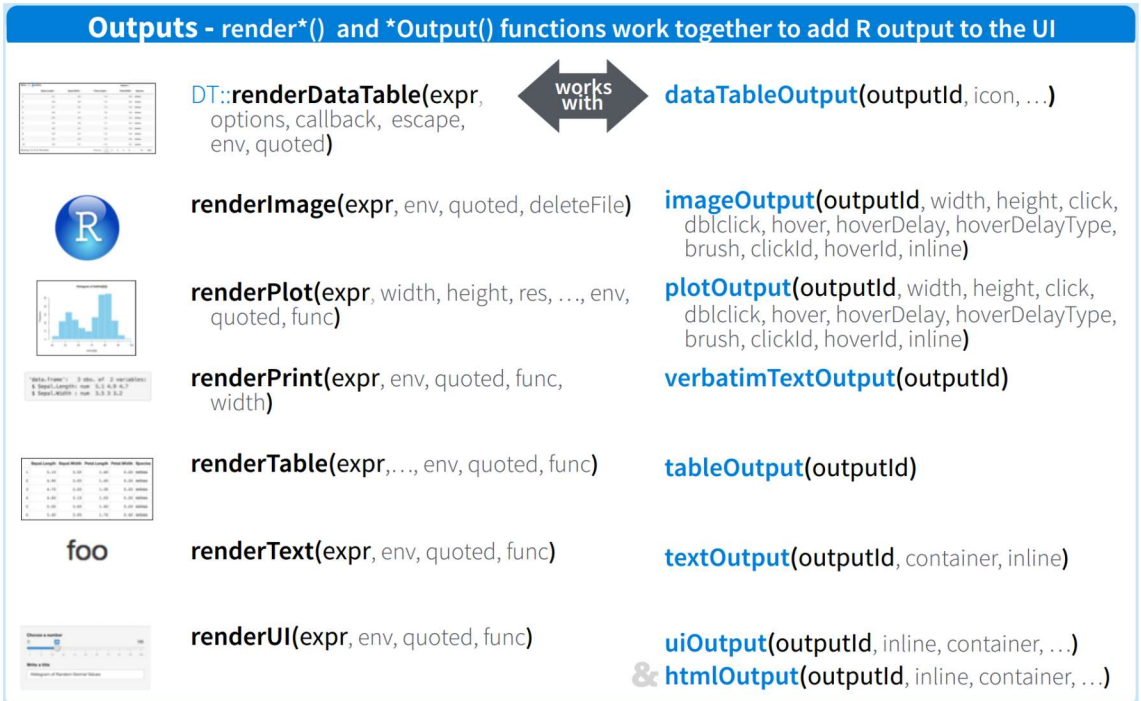
Slika 1: HTML elementi koje je moguće koristiti

Ui i server unutar Shiny-a komuniciraju putem ulaznih i izlaznih vrijednosti. Ulazne vrijednosti unutar Shiny-a možemo još nazivati i widgeti, a to su web elementi s kojima korisnik može ostvariti interakciju. Oni pružaju put kojim će korisnici slati poruku Shiny aplikaciji te prikupljaju vrijednosti od korisnika.



Slika 2: Widgeti koje je moguće koristiti kao ulazne vrijednosti

Izlazne vrijednosti ovise o definiciji server funkcije koja traži promjene u ulaznim vrijednostima kako bi procijenila treba li se izlazna vrijednost koja je prikazana na ekranu ažurirati ili ne.



Slika 3: Izlazne vrijednosti koje služe za prikaz podataka

2.2 Reaktivnost

Također, još jedna bitna stavka koja Shiny čini pogodnim za korištenje je činjenica da se ovdje koristi tzv. reaktivno programiranje. Reaktivno programiranje je moćan programski alat čija je glavna ideja ovisnost izlaznih varijabli o ulaznima, tj. želimo odrediti ovisnost tako da kada dođe do promjene ulazne vrijednosti, sve povezane izlazne vrijednosti se također mijenjaju. Upravo reaktivnost pomaže kako bi aplikacije dobivene pomoću Shiny-a bile interaktivne i pregledne.

```
ui <- fluidPage(
  textInput("name", "What's your name?"),
  textOutput("greeting")
)

server <- function(input, output, session) {
  output$greeting <- renderText({
    paste0("Hello ", input$name, "!")
  })
}
shinyApp(ui, server)
```

Primjer 3: Primjer reaktivne aplikacije - kod

Slika 4: Dobiveni rezultat primjera 3

3 Web stranica

3.1 Tehnologije

Budući da je paket Shiny pogodan za rad s velikim brojem podataka i prikladnu vizualizaciju istih, web aplikacija za praćenje novih podataka vezanih uz pandemiju napravljena je upravo koristeći njega. Kako aplikacija koristi otvorenu bazu podataka, ona se svaki dan automatski ažurira kako se u bazu dodaju novi podaci o pandemiji. Osim Shiny-a, unutar R-a korišteni su sljedeći paketi i biblioteke:

- *shinythemes* - sadrži nekoliko Bootstrap tema za korisničko sučelje
- *ggplot2* - jednostavan i detaljan način za prikaz grafova
- *plotly* - izrada interaktivnih grafova
- *jsonlite* - JSON parser
- *covid19.analytics* - sadrži podatke vezane uz koronavirus

- *lubridate* - mijenjanje formata datuma
- *dplyr* - manipulacija podacima i gramatikom

Također, osim R-a korišteni su HTML i CSS kako bi poboljšali izgled stranice.

3.2 Podaci

U svrhu izrade web aplikacije korišteni su otvoreni, strojno čitljivi, podaci koje ustupa Hrvatski zavod za javno zdravstvo te Ministarstvo zdravstva Republike Hrvatske. Podaci su zapisani u JSON formatu i nalaze se na sljedećim linkovima:

- https://www.koronavirus.hr/json/?action=podaci
- https://www.koronavirus.hr/json/?action=po_osobama

Osim podataka dostupnih putem HZJZ-a, korišteni su podaci iz paketa covid19.analytics koji se većinom dobivaju od strane Johns Hopkins University Center for Systems Science and Engineering te podaci od Our World in Data. Treba naglasiti da postoji određeno odstupanje u podacima HZJZ u odnosu na druge izvore.
Prije korištenja podataka, morali smo napraviti nekoliko izmjena danih podataka. Na početku smo kreirali funkciju koja, koristeći ugrađenje R funkcije, pretvara JSON datoteku u CSV datoteku na sljedeći način:

```
#pretvara json file u csv file
json_to_csv <- function(url){
  my.JSON <- fromJSON(url, flatten = TRUE)
  data_frame <- as.data.frame(my.JSON)
  return(data_frame)
}
```

Primjer 4: Funkcija koja pretvara JSON u CSV datoteku

Budući da nam nije odgovaralo kako su dani podaci izgledali, korištenjem nekoliko jednostavnih, ugrađenih funkcija smo ih prilagodili. Također, iz baze podataka smo izvukli željene podatke, poput dnevnog broja zaraženih, umrlih, oporavljenih i cijepljenih, ukupnog broja zaraženih, umrlih, oporavljenih i cijepljenih, broja zaraženih osoba po spolu, županiji ili gradištu od kojih su napravljeni grafovi i krivulje.

```
#obrada podataka
podaci_hrv <- json_to_csv(url =
    "https://www.koronavirus.hr/json/?action=podaci")
podaci_hrv[,11] <- round_date(as.Date(podaci_hrv[,11],
    origin='01-01-1997'), "day")
podaci_hrv <- podaci_hrv %>% mutate_if(is.character,as.numeric)
```

Primjer 5: Prilagođavanje podataka

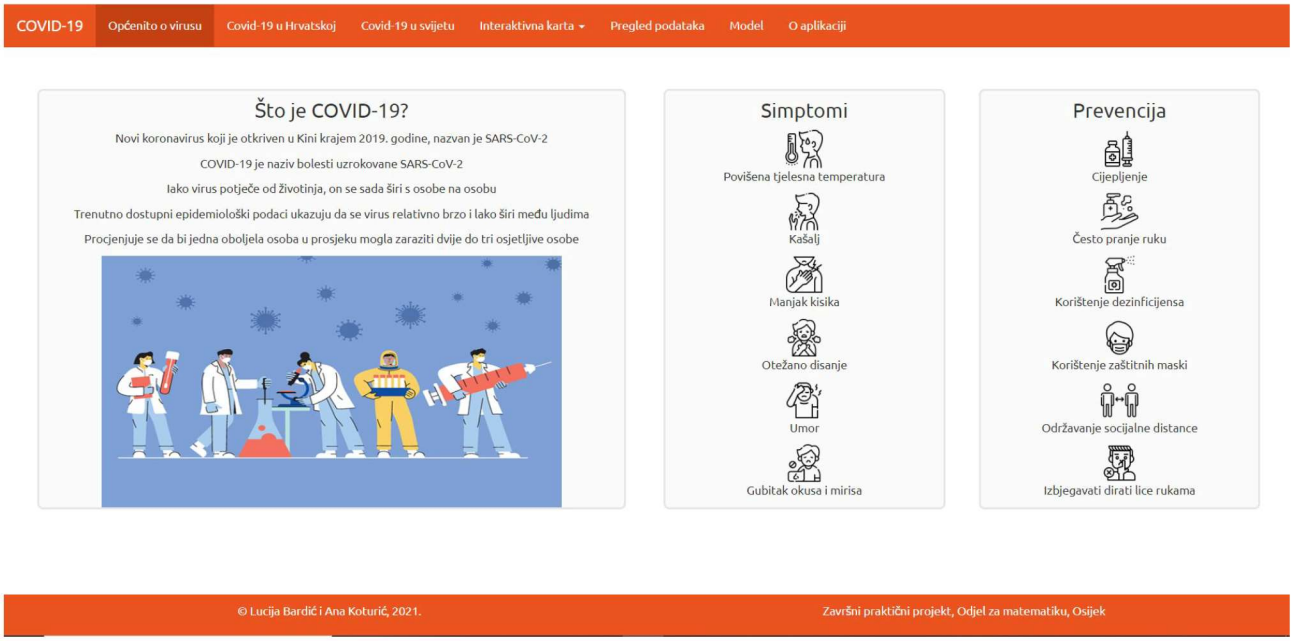
Napokon, nakon svih izmjena dobivamo tablice s podacima koje možemo koristiti za daljnje proučavanje podataka i jedna od njih izgleda ovako:

	SlucajeviSvijet	SlucajeviHrvatska	UmrliSvijet	UmrliHrvatska	IzlijeceniSvijet	IzlijeceniHrvatska
1	211619700	369392	4429425	8298	189367262	358430
2	210874026	368887	4417858	8295	188810002	358083
3	210211494	368419	4407211	8294	188306369	357770
4	209446938	367933	4396137	8291	187727095	357518
5	208736825	367409	4384159	8288	187123664	357318
6	208003536	367068	4375020	8285	186475601	357065
7	207584240	367022	4368519	8283	186071552	356836
8	207044460	366724	4359705	8282	185649218	356660
9	206263033	366357	4348337	8280	185111814	356437
10	205564844	366049	4337809	8278	184558988	356219
11	204814630	365716	4327361	8275	183940837	356043
12	204166927	365335	4316541	8275	183358160	355883

Slika 5: Primjer dobivenih podataka

3.3 Funkcije ui i server

Korisničko sučelje kreirano je korištenjem funkcije *bootstrapPage()* koja omogućava da se stranica automatski prilagodi veličini zaslona kojeg korisnik koristi. Sve elemente koji će se nalaziti na stranici ubacujemo unutar ove funkcije.
Kako bismo dobili estetski lijepu navigacijsku traku, korišten je paket *shinythemes* i tema "untitled" te pomoću funkcije *navbarPage* istu kreiramo. Svaka od kartica na navigacijskoj traci kreirana je s funkcijom *tabPanel*. Kao i u Bootstrapu, i ovdje se koristi *grid system* s 12 kontejnera za slaganje elemenata.
Sve funkcije koje u nazivu imaju "box" služe za kreiranje kartica s informacijama vidljivim na stranici. Kako bi web stranica imala lijep izgled, za estetsko uređivanje nekih elemenata unutar *UI-a* koristili smo CSS i u tu svrhu smo tim elementima dodijelili *ID* vrijednosti te uz pomoć njih u datoteci *style.css* definirali izgled tih elemenata.



Slika 6: Korisničko sučelje web stranice

```
ui <- bootstrapPage(
 includeCSS("www/style.css"),
 theme = shinytheme("united"),
 navbarPage(title = "COVID-19", collapsible = TRUE, windowTitle = "COVID-19",
          tabPanel(title = "Openito o virusu",
                  column(width = 6, align = "center", box_opcenito()),
                  column(width = 3, align = "center", box_simptomi()),
                  column(width = 3, align = "center", box_prevencija())),
          tabPanel("Covid-19 u Hrvatskoj",
                  column(width = 3, align = "center", box_zarazeni()),
                  column(width = 3, align = "center", box_oporavljeni()),
                  column(width = 3, align = "center", box_umrli()),
                  column(width = 3, align = "center", box_cijepljeni()),
                  fluidRow(id = "graf_1", column(6, graf1()),
                                          column(6,graf2() )),
                  fluidRow(id = "graf_3", graf3()),
                  fluidRow(id = "graf_3", graf4())),
          tabPanel("Covid-19 u svijetu",
                  column(width = 3, align = "center", box_zarazeni2()),
                  column(width = 3, align = "center", box_oporavljeni2()),
                  column(width = 3, align = "center", box_umrli2()),
                  column(width = 3, align = "center", box_cijepljeni2()),
                  fluidRow(id = "graf_1", column(6, graf11()),
                                          column(6, graf22())),
                  fluidRow(id = "graf_3", graf33()),
                  fluidRow(id = "graf_3", graf44())),
          navbarMenu("Interaktivna karta",
                    tabPanel(id ="karta", "Ukupno
                       sluajeva",plotlyOutput("karta", height="800px")),
                    tabPanel(id ="karta", "Ukupno umrlih",
```

```
                    plotlyOutput("karta2", height="800px"))),
            tabPanel(title = "Pregled podataka",
                    column(6, align = "center", graf5()),
                    column(6, align = "center", graf6()),
                    column(6, align = "center", graf7()),
                    column(6, align = "center", graf1D(100000))),
            tabPanel(title = "Model",
                    fluidRow(id = "model_box", box_model()),
                    fluidRow(id = "range_input", column(3, uiOutput("pocetak")),
                                                 column(3, uiOutput("kraj"))),
                    fluidRow(id = "graf_3", plotlyOutput("graf_predvidanje"))),
            tabPanel(title = "O aplikaciji",
                    fluidRow(column(6, align="center", box_aplikacija()),
                            column(6, align="center", box_aplikacija1()))
            )
  ),
  footer()
)
```

Primjer 6: Kod za kreiranje korisničkog sučelja web stranice

Drugi dio web stranice nalazi se u funkciji server(), koja se bavi logikom web stranice. Ova funkcija gradi objekte pod nazivom *output* koji sadrže sav kod koji je potreban da bi se neki R objekti ažurirali.
Output objekti spremljeni su kao varijable s nazivom output$outputId, a outputId nam je potreban kako bismo mogli pozvati te objekte u UI-u kroz output funkcije kao što su plotlyOutput(*outputId*) i uiOutput(*outputId*).

```
server <- function(input, output, session) {
    output$karta <- renderPlotly({
    karta()
  })
  output$karta2 <- renderPlotly({
    karta2()
  })
  output$pocetak <- renderUI({
    dateInput('pocetak',
              label = "Poetak predikcije:",
              value = Sys.Date()-1,
              min = as.Date("2020-10-01"),
              max = Sys.Date()-1)
  })
  output$kraj <- renderUI({
    dateInput('kraj',
              label = "Kraj predikcije:",
              value = Sys.Date()+7,
              min = input$pocetak + 5,
    )
  })
  output$graf_predvidanje <- renderPlotly({
    req(input$pocetak)
    req(input$kraj)
    model(input$pocetak[1], input$kraj[1])
  })
}
```

Primjer 7: Kreiranje server funkcije

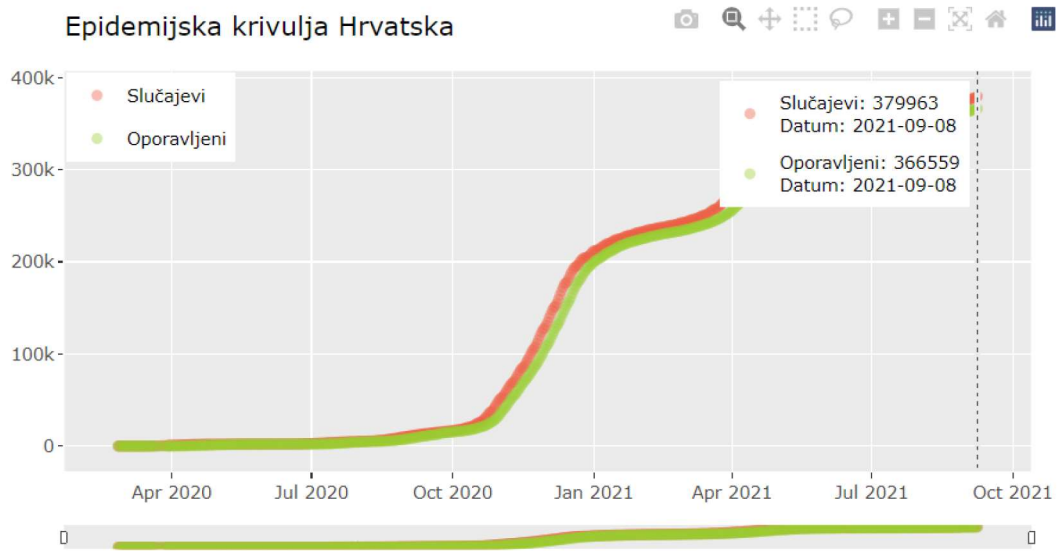
3.4 Vizualizacija podataka

Sljedeći grafovi bit će prikazani za Hrvatsku, ali na web stranici napravljeni su i grafovi za svijet na identičan način pa zbog toga neće biti prikazani u ovom radu.
Kako bi podaci i sami grafovi bili što pregledniji, dodali smo *pop up* prozorčiće, pomoću kojih se jasno mogu vidjeti informacije pridružene vrijednostima na x-osi. Također, dodali smo *rangeslider* kako bi se mogao izabrati interval u kojem želimo promatrati određene podatke.

3.4.1 Epidemijske krivulje

Za prikaz i uređivanje krivulja korišteno je nekoliko gotovih funkcija:

- *ggplot()* koja inicijalizira ggplot objekt.
- *geom_point()* koja se koristi za prikaz odnosa između dvije neprekidne varijable (npr. broj slučajeva i datum)
- *ggplotly()* pomoću koje graf postaje interaktivan
- *layout()* pomaže pri uređivanju izgleda kod npr. *pop up* prozorčića i legende



Slika 7: Primjer krivulje koja prikazuje ukupni broj zaraženih i broj oporavljenih u Hrvatskoj

- *labs()* služi za uređivanje osi, legende i oznaka

```
graf1 <- function(){
    colors <- c("Slucajevi" = "tomato2", "Oporavljeni" = "yellowgreen",
       encoding = 'UTF-8')
    krivulja_hrv <- ggplot(podaci_hrv, aes(x=Datum)) +
     geom_point(aes(y=SlucajeviHrvatska, color = "Sluajevi",
         text=sprintf("Slucajevi: %d", SlucajeviHrvatska)), stat="identity",
         alpha=0.4) +
     geom_point(aes(y=IzlijeceniHrvatska, color = "Oporavljeni",
         text=sprintf("Oporavljeni: %d", IzlijeceniHrvatska)),
         stat="identity", alpha=0.4) +
     labs(title = "Epidemijska krivulja Hrvatska", x = "", y = "", color =
         "") + scale_color_manual(values = colors)

    ggplotly(krivulja_hrv, dynamicTicks = TRUE, tooltip = c("Datum", "text"))
       %>%
       rangeslider(thickness = 0.05) %>%
       layout(hovermode = "x unified")%>%
       layout(legend = list(orientation = "v", x = 0, y =1))
}
```

Primjer 8: Pripadni kod slike 7

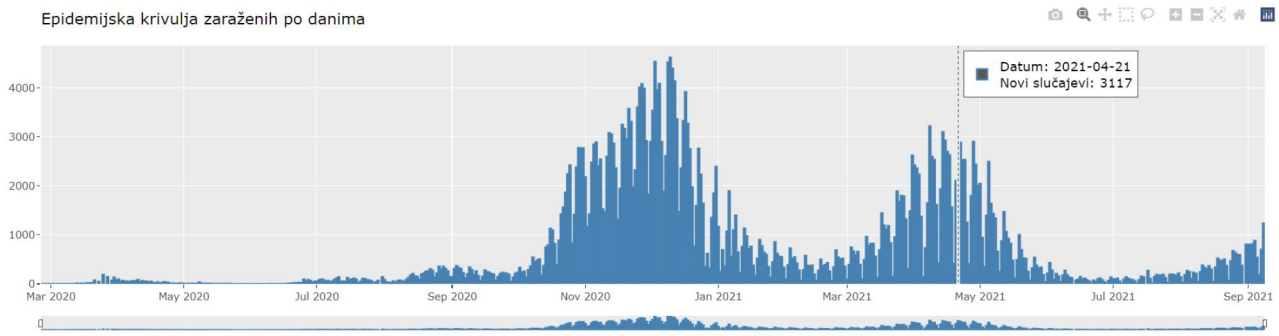
Osim krivulje za prikaz broja zaraženih i oporavljenih, napravljene su i krivulje za praćenje broja umrlih te broja cijepljenih jednom ili dvije doze.

3.4.2 Histogrami

Osim krivulja, za prikaz podataka na web stranici korišteni su i histogrami koji su u prikazu određenih podataka pregledniji i jednostavniji. Na ovaj način prikazan je dnevni broj novozaraženih osoba uz pomoć pop up prozorčića, koji ispisuje datum i broj zaraženih toga dana,

ukupan broj zaraženih osoba po županiji, ukupan broj zaraženih osoba po spolu te ukupan broj zaraženih po godištu.
Za prikaz i uređivanje histograma korištena je sljedeća gotova funkcija (uz one koje su već spomenute kod krivulja):

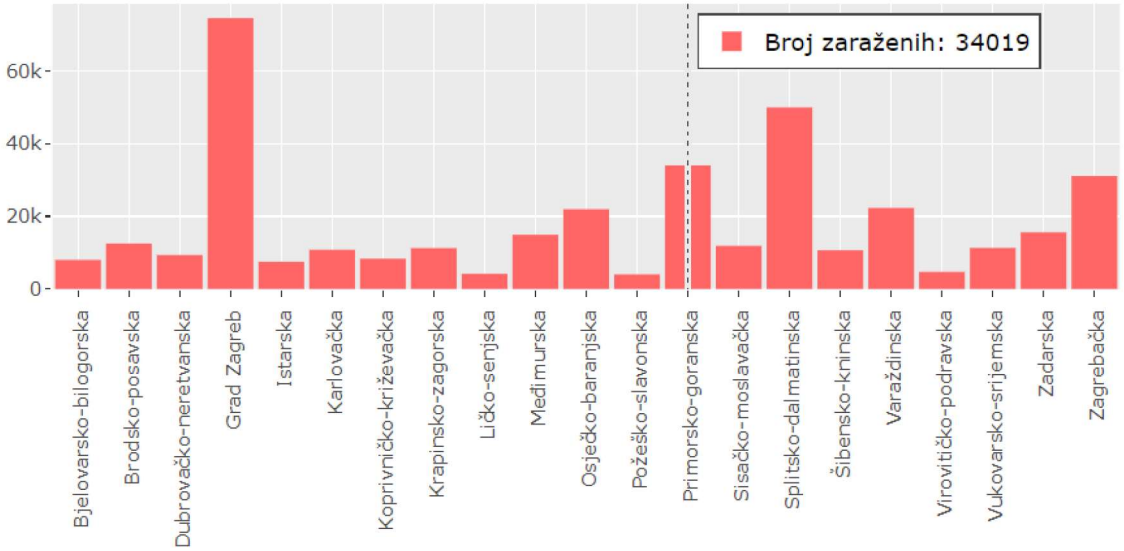
- *geom_bar()* kod koje visina predstavlja broj zaraženih u odnosu na neku varijablu



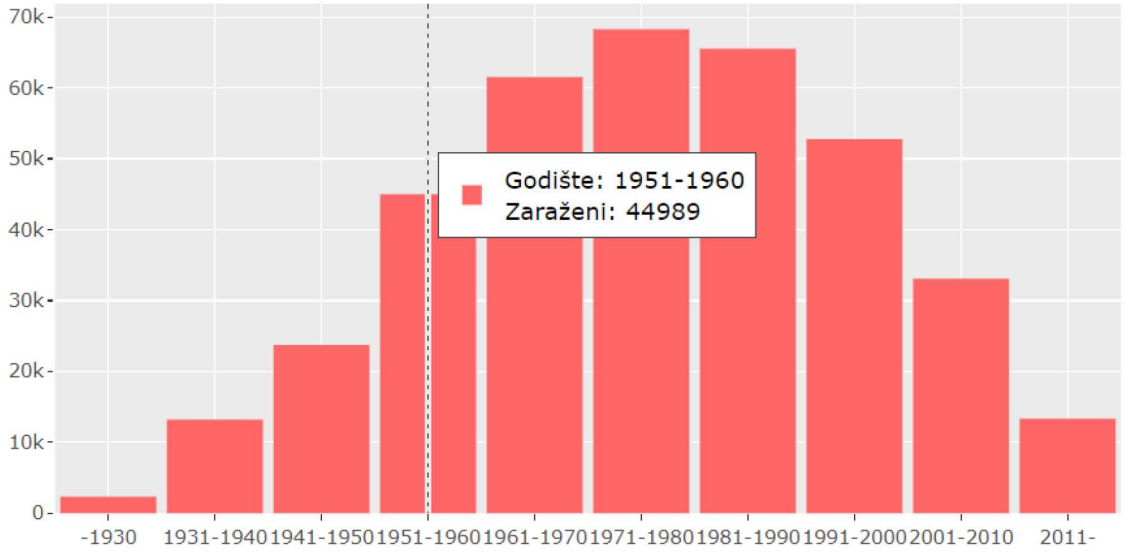
Slika 8: Primjer histograma koji prikazuje broj novozaraženih po danu

```
graf3 <- function(){
    histogram_hrv_podaci <-
    ggplot(podaci_hrv, aes(x=Datum, y=noviSlucajevi, text=sprintf("Novi
        sluajevi: %d", noviSlucajevi))) +
    geom_bar(stat = "identity", color='steelblue') +
    labs(title = "Epidemijska krivulja zaraenih po danima", x = "", y =
        "", color = "", encoding = 'UTF-8')

    ggplotly(histogram_hrv_podaci, dynamicTicks = TRUE, tooltip =
        c("Datum", "text")) %>%
        rangeslider(thickness = 0.05) %>%
        layout(hovermode = "x unified")
}
```

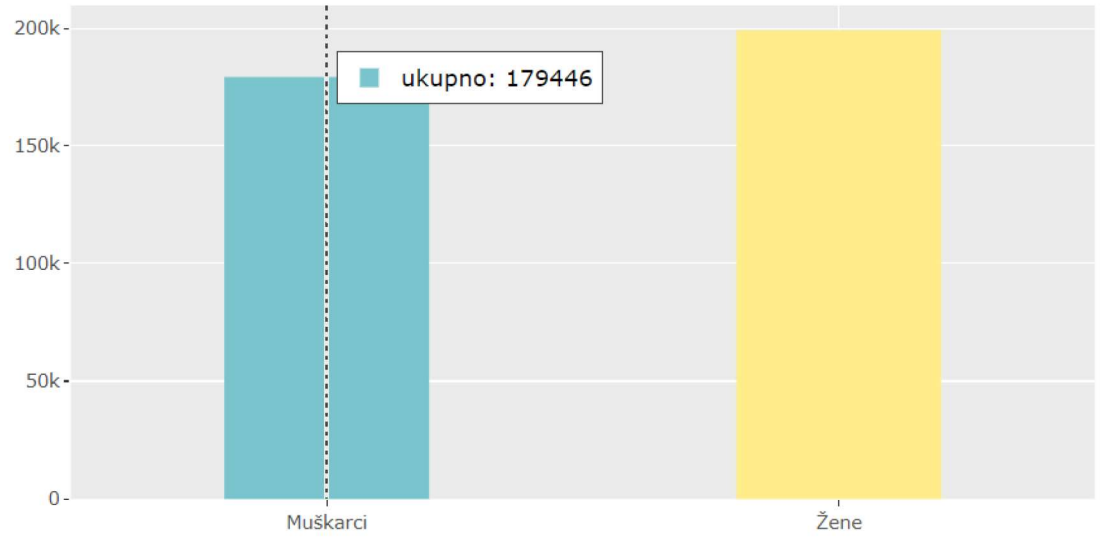
Primjer 9: Prpadni kod slike 8



Slika 9: Ukupan broj zaraženih po županijama u Hrvatskoj



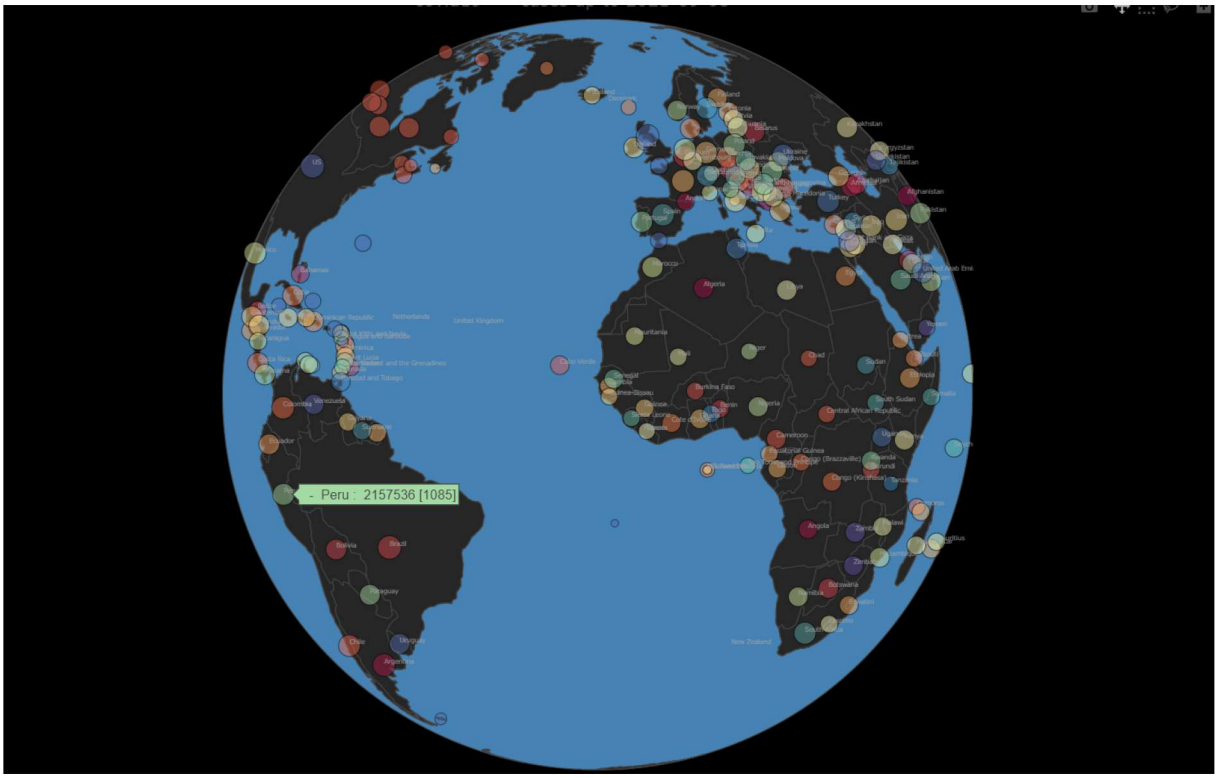
Slika 10: Ukupan broj zaraženih po godištima u Hrvatskoj



Slika 11: Ukupan broj zaraženih po spolu u Hrvatskoj

3.4.3 Karte

Kao vizualno ljepši i zanimljiviji prikaz podataka na web stranici korištene su interaktivne karte. Paket *covid19.analtytics*, kao dio R-a, sadrži sve potrebne podatke i funkcije za jednostavno kreiranje karte svijeta. Paket koristi baze podataka koje sadrže informacije o dnevnom broju zaraženih po državama te nekoliko analiza, vizualizacija te modela pomoću kojih se prikazuju podaci globalno ili za specifičnu geografsku lokaciju. Budući da web stranica već sadrži dnevni broj zaraženih ili umrlih u Hrvatkoj, na karti je prikazan broj novozaraženih na globalnoj razini, tj. u svakoj državi posebno.



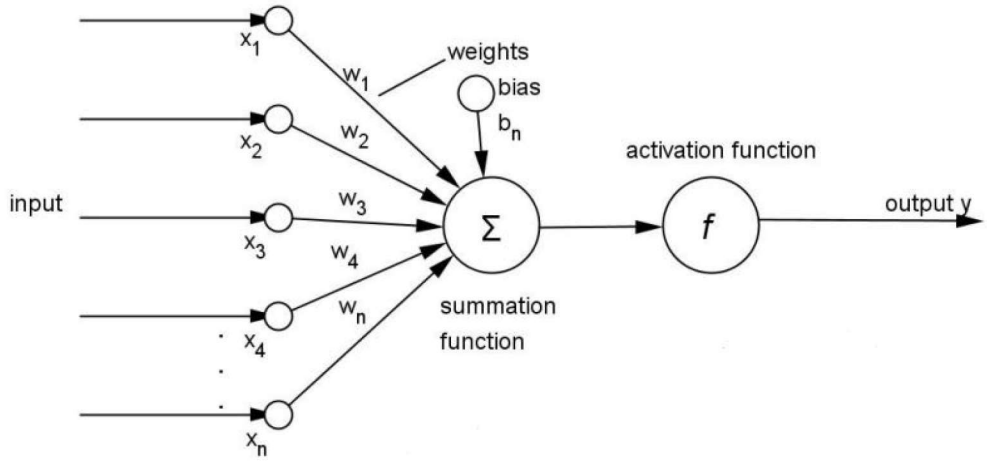
Slika 12: Prikaz svijeta i podataka za svaku državu posebno

```
karta <- function() {
    cases <- covid19.data("ts-confirmed")
    live.map(cases, select.projctn = FALSE, no.legend = TRUE, title = "",
        szRef = 0.9)
}
```

Primjer 10: Pripadni kod slike 12

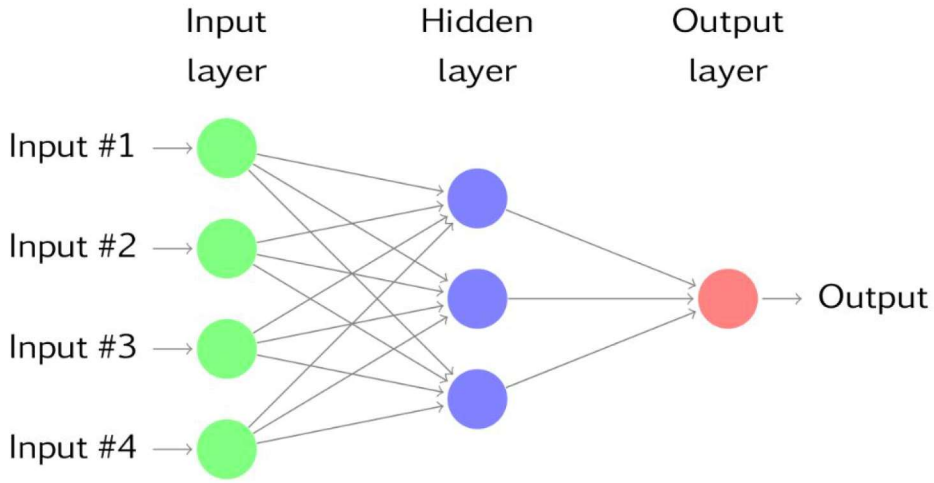
4 Model kratkoročnog predviđanja kretanja pandemije

Tehnološki napredak omogućio je spremanje velikog broja podataka koji se mogu proučavati i koristiti za statističko modeliranje i računanje. Kako bi se opisala dinamika i napravilo učinkovito predviđanje jako složenih podataka, koriste se se umjetne neuronske mreže. One omogućavaju složene nelinearne odnose između ulaznih i izlaznih varijabli. Također, služe za razumijevanje i rješavanje problema na području umjetne inteligencije te se koriste strukturom ljudskog mozga kako bi razvile primjerenu strategiju analize podataka.



Slika 13: Shematski prikaz neurona (preuzeto iz [16])

Umjetna neuronska mreža predstavlja skup "neurona" koji su međusobno povezani i organizirani u slojeve. Neuronska mreža ima tri osnovne vrste slojeva: ulazni sloj, skriveni sloj i izlazni sloj. Ulazni sloj je prvi sloj mreže i on obrađuje ulazne podatke iz sustava. Izlazni sloj je posljednji sloj u kojemu su izlazi neurona zapravo predviđanja rješenja sustava. Svi ostali slojevi su skriveni. Pojedinačni neuroni međusobno su spojeni vezama kroz koje idu signali. Najjednostavnije mreže ne sadrže skrivene slojeve i za njihovo treniranje koriste se linearne regresije. Više o linearnoj regresiji može se naći na [15]. Predviđanja u neuronskim mrežama se dobivaju linearnom kombinacijom ulaznih podataka.



Slika 14: Jednostavna neuronska mreža (preuzeto iz [9])

Općeniti model neurona dan je na slici 13. Ulazni signali označeni su s x_1, x_2,..., x_n, težine s w_{1j}, w_{2j},..., w_{nj}, a izlazna funkcija s y_j iz čega dobivamo sljedeću formulu:

$$y_j = b_j + \sum_{i=1}^{n} w_{i,j} x_i$$

pri čemu b_j predstavlja težinski faktor tj. pomak. Na navedenu formulu se primjenjuje

aktivacijska funkcija te se izračunata vrijednost prosljeđuje do sljedećeg neurona ili je baš ta vrijednost traženo predviđanje.
Umjetne neuronske mreže imaju široku upotrebu na mnogim područjima kao što su marketing, financije, trgovina, fizika, medicina itd., a neki od zadataka koje obavljaju su:

- predviđanje cijena dionica i prognoziranje cijena na tržištima,
- obrada slike i govora,
- problemi optimizacije,
- određivanje tržišta pogodnog za određeni proizvod,
- segmentiranje potencijalnih kupaca u određene grupe,
- nelinearno upravljanje.

4.1 NNAR model

Autoregresivni ili AR model je model koji opisuje kako prethodne vrijednosti određene varijable utječu na njezinu trenutačnu vrijednost. Drugim riječima, AR model pokušava predvidjeti sljedeću vrijednost u nizu tako što koristi najnovije prošle vrijednosti i koristi ih kao ulazne podatke, temelji se na ideji da nam prošli događaji mogu pomoći u predviđanju budućih događaja. Kao što sam naziv govori, autoregresivni model predstavlja linearnu regresiju na samoga sebe. AR modeli se mogu koristiti za modeliranje svega što ima neki stupanj autokorelacije što znači da postoji korelacija između opažanja u susjednim vremenskim koracima. Najčešći slučaj korištenja za ovu vrstu modeliranja je cijena na burzi gdje je cijena danas u velikoj korelaciji s cijenom prije jednog dana. Jedan od autoregresivnih modela naziva se NNAR model, tj. autoregresivni model neuronske mreže. Ovaj model predstavlja *feed-forward* neuronsku mrežu s jednim skrivenim slojem i prethodnim vrijednostima kao ulaznim podacima te uključuje linearnu kombinaciju funkcije i aktivacijske funkcije. Više o feed-forward neuronskim mrežama može se pročitati na [6]. Model se označava kao NNAR(p,P,k)$_m$ i tada model za ulazne vrijednosti prima $(y_{t-1}, y_{t-1}, ..., y_{t-p}, y_{t-m}, y_{t-2m}, ..., y_{t-Pm})$ te sadrži k neurona u skrivenom sloju. Postupak predviđanja pomoću ovog modela je iterativan, predviđanje jedan korak unaprijed koristi povijesne ulaze, predviđanje dva koraka unaprijed koristi predviđanje jedan korak unaprijed i povijesne ulaze.
Kao primjer korištenja neuronskih mreža, na web stranici kreiran je NNAR model koji predviđa kretanje zaraze virusom SARS-CoV-2 za određeni vremenski interval i na osnovu dostupnih podataka. (Prema [7] i [9].)

4.2 Podaci za model

Za izradu modela korišten je *forecast* paket, koji sadrži više različitih metoda, funkcija i alata za prikazivanje i analiziranje predviđanja za određeni atribut, odnosno karakteristiku u nekom vremenskom razdoblju. Za ovaj model koristili smo podatke dostupne putem HZJZ-a, ali bilo ih je potrebno modificirati kako bi predviđanje bilo što realnije. S obzirom na to da se lako može uočiti da je broj novozaraženih osoba u dane vikenda i blagdana znatno manji nego ostalim danima, modificirali smo podatke na način da je broj novozaraženih osoba za određeni dan jednak aritmetičkoj sredini broja novo zaraženih osoba u posljednja tri dana i na taj način smo dobili 'zaglađeniju' krivulju broja novozaraženih osoba.

```
model_data <-data.frame(rev(podaci_hrv$noviSlucajevi), rev(podaci_hrv$Datum))
for(x in 3:length(podaci_hrv$noviSlucajevi)){
    model_data$rev.podaci_hrv.noviSlucajevi.[x] <-
        round(mean(c(rev(podaci_hrv$noviSlucajevi)[x],
        rev(podaci_hrv$noviSlucajevi)[x-1],
        rev(podaci_hrv$noviSlucajevi)[x-2])))
}
names(model_data)[1] <- "noviSlucajevi"
names(model_data)[2] <- "Datum"
```

Primjer 11: Modificiranje podataka za model

4.3 Kreiranje modela

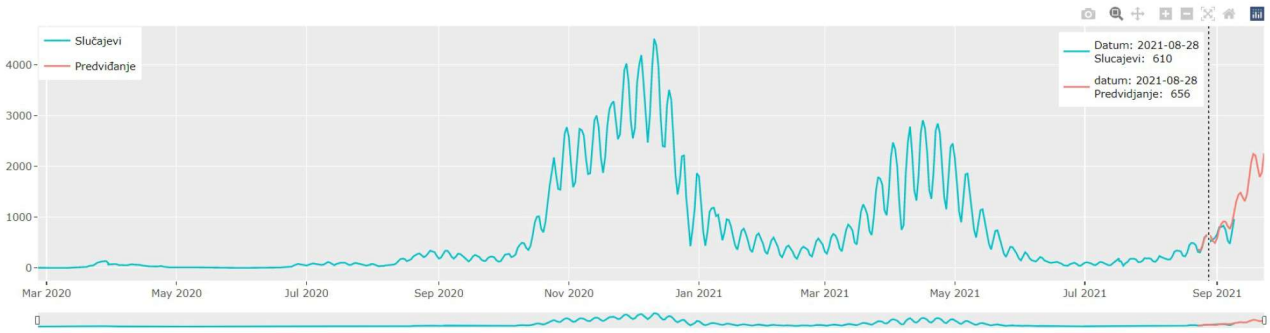
Za kreiranje modela koristili smo funkciju *nnetar()* iz već spomenutog paketa *forecast*. To je funkcija koja koristi neuronske mreže za predviđanje određenog vremenskog atributa u zadanom vremenskom intervalu. Nakon što smo kreirali model napravili smo predviđanje uz pomoć funkcije *forecast()*.

```
real_data <- model_data$noviSlucajevi
original_data <-
   model_data$noviSlucajevi[1:(which(model_data$Datum==pocetak)-1)]
rows <- NROW(original_data) #racuna broj redaka
training_data<-original_data[1:(rows)]
N_forecasting_days <- as.numeric(kraj-pocetak)+1

#NNAR Model
data_series<-ts(training_data)
model_NNAR<-nnetar(data_series, size = 5)

#predvidanje
forecasting_NNAR <- forecast(model_NNAR, h=N_forecasting_days)
```

Primjer 12: Kreiranje modela i predikcija

4.4 Vizualizacija modela

Unutar paketa *forecast* postoje funkcije za automatsku vizualizaciju predikcije, no kako smo mi htjeli da naš prikaz bude interaktivan, vizualizaciju smo napravili sami.
Da bismo vizualizirali predviđanje, morali smo prvo izvući 'predviđene' podatke o broju novozaraženih osoba iz varijable *forecasting_NNAR* koja sadrži i neke dodatne podatke o predviđanju. Nakon što smo pristupili podacima o predviđanju prikazali smo ih uz pomoć paketa *ggplot2* i *plotly* (na isti način na koji smo vizualizirali ranije krivulje), te smo uz predviđene podatke, prikazali i stvarne podatke.



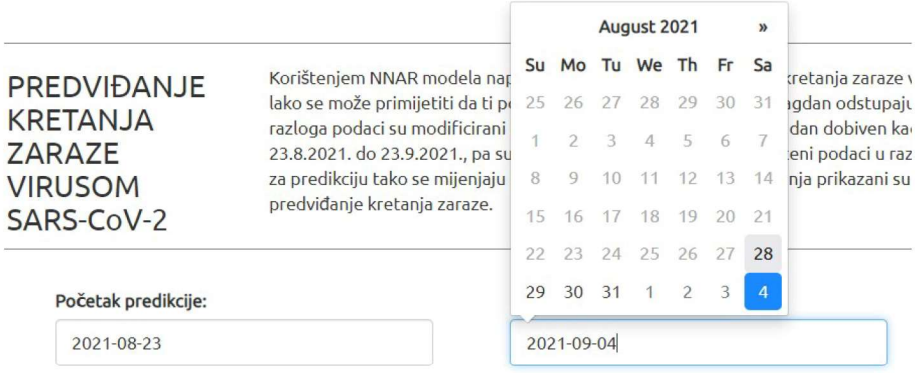
Slika 15: Graf predviđanja kretanja zaraze

```
#vizualizacija
Predvidjanje <- round(forecasting_NNAR$mean)
datum <- seq(pocetak, kraj, by="days")
Datum <- model_data$Datum
Slucajevi <- model_data$noviSlucajevi
colors <- c("Sluajevi" = "tomato2", "Predvianje" = "yellowgreen", encoding =
    'UTF-8')
krivulja_hrv <- ggplot() +
    geom_line(aes(x=Datum, y = Slucajevi, color = "Sluajevi")) +
    geom_line(aes(x=datum, y = Predvidjanje, color = "Predvianje")) +
    labs(title = "", x = "", y = "", color = "")

graf <- ggplotly(krivulja_hrv, dynamicTicks = TRUE, tooltip = c("Datum",
    "datum", "Slucajevi", "Predvidjanje")) %>%
    rangeslider(thickness = 0.05) %>%
    layout(hovermode = "x unified")%>%
    layout(legend = list(orientation = "v", x = 0, y =1))
```

Primjer 13: Pripadni kod slike 14

Uz pomoć funkcije *dateInput()* kreirali smo sučelje za unos intervala za predikciju, te smo postavili neke uvjete:

- prvi dan u intervalu može najviše prethodni dan
- zadnji dan u intervalu mora biti veći od prvog dana u intervalu
- interval mora biti najmanje razdoblje od 6 dana



Slika 16: Sučelje za unos intervala za predikciju

```
output$pocetak <- renderUI({
    dateInput('pocetak',
            label = "Poetak predikcije:",
            value = Sys.Date()-1,
            min = as.Date("2020-10-01"),
            max = Sys.Date()-1)
})

output$kraj <- renderUI({
    dateInput('kraj',
            label = "Kraj predikcije:",
            value = Sys.Date()+7,
            min = input$pocetak + 5,
    )
})
```

Primjer 14: Pripadni kod slike 15

Ono što se može na kraju zaključiti je da model dosta dobro predviđa kretanje pandemije kada se koriste mali vremenski intervali od nekoliko dana.

Literatura

[1] M. T. Anatolyevna i A. Mostafa, *Epidemic.TA System for Forecasting Covid-19 Cases Using Time Series and Neural Networks Models*, Department of Electrical Engineering and Computer Science South ural state university, Chelyabinsk, Russian Federation, URL: https://rpubs.com/abotalebmostafa/744347

[2] *Dokumentacija paketa covid19.analytics.* URL: https://cran.r-project.org/web/packages/covid19.analytics/covid19.analytics.pdf

[3] G. Benrhmach, K. Namir, A. Namir, J. Bouyaghroumni, *Nonlinear Autoregressive Neural Network and Extended Kalman Filters for Prediction of Financial Time Series*, Journal of Applied Mathematics (2020), URL: https://www.hindawi.com/journals/jam/2020/5057801/

[4] *Dokumentacija paketa Shiny*, URL: https://shiny.rstudio.com/

[5] *Dokumentacija programskog jezika R*, URL: https://www.rdocumentation.org/

[6] T. Gupta, *Deep Learning: Feedforward Neural Network*, (2017), URL: https://towardsdatascience.com/deep-learning-feedforward-neural-network-26a6705dbdc7

[7] A. Kumar, *Autoregressive (AR) models with Python examples*, (2022), URL: https://vitalflux.com/autoregressive-ar-models-with-python-examples/

[8] Mathieu, E., Ritchie, H., Ortiz-Ospina, E. *et al.* A global database of COVID-19 vaccination, *Nat Hum Behav* (2021), URL: https://github.com/owid/covid-19-data/tree/master/public/data/vaccinations

[9] OTexts, *Neural network models*, URL: https://otexts.com/fpp2/nnetar.html

[10] Središnja medicinska ustanova javnog zdravstva u Hrvatskoj, URL: https://www.hzjz.hr/

[11] *The R Graph Gallery*, https://www.r-graph-gallery.com/index.html

[12] W.N. Venables, D.M. Smith i osnovna grupa suradnika za razvoj R-a, *Uvod u korištenje R-a*, (2004), URL: https://cran.r-project.org/doc/contrib/Kasum+Legovic-UvodUr.pdf

[13] Vlada Republike Hrvatske, Službena stranica Vlade za pravodobne i točne informacije o koronavirusu, URL: https://www.koronavirus.hr/

[14] H. Wickham, *Mastering Shiny*, O'Reilly Media (2020), URL: https://mastering-shiny.org

[15] Yale University, Department of Statistics and Data Science, *Linear Regression*, Url: http://www.stat.yale.edu/Courses/1997-98/101/linreg.htm

[16] M. Zekić-Sušac, Ekonomski fakultet u Osijeku, *3. Neuronske mreže*, Url: http://www.efos.unios.hr/upravljanje-marketingom/wp-content/uploads/sites/192/2013/04/P3_Neuronske-mreze-2017.pdf