Kontinuirano oblikovanje, evaluacija i prilagođavanje modela vremenskih nizova

Šajina, Romeo

SVEUČILIŠTE JURJA DOBRILE U PULI
FAKULTET INFORMATIKE

DIPLOMSKI RAD

Kontinuirano oblikovanje, evaluacija i prilagođavanje modela vremenskih nizova

Romeo Šajina

Pula, srpanj 2019.

SVEUČILIŠTE JURJA DOBRILE U PULI
FAKULTET INFORMATIKE

DIPLOMSKI RAD

Kontinuirano oblikovanje, evaluacija i prilagođavanje modela vremenskih nizova

Romeo Šajina

JMBAG: 0303054696, redovni student
Studijski smjer: Informatika

Predmet: Umjetna inteligencija
Znanstveno područje: Društvene znanosti
Znanstveno polje: Informacijske i komunikacijske znanosti
Znanstvena grana: Informacijski sustavi i informatologija

Mentor: doc.dr.sc. Darko Etinger
Komentor: dr.sc. Nikola Tanković

Pula, srpanj 2019.

SVEUČILIŠTE JURJA DOBRILE U PULI
FAKULTET INFORMATIKE

Pula, 21. ožujka 2019.

DIPLOMSKI ZADATAK

Pristupnik:	**Šajina Romeo (JMBAG: 0303054696)**
Studij:	Sveučilišni diplomski studij Informatike
Naslov (hrv.):	**Kontinuirano oblikovanje, evaluacija i prilagođavanje modela vremenskih nizova**
Naslov (eng.):	Continuous building, monitoring and adjusting time series forecast models
Opis zadatka:	Cilj ovog diplomskog rada je proučiti postupak izrade, praćenja i prilagodbe modela vremenskih nizova. Rad će pojasniti postupak oblikovanja podataka kao vremenskih nizova i iznijeti nekoliko metoda za izradu modela vremenskih nizova. Na temelju evaluacije metoda, nad odabranim skupom podataka, usporedit će se karakteristike pojedinih metoda. Svaka metoda bit će popraćena primjerom korištenja pomoću programskog jezika *Python* koristeći već dostupne knjižnice poput *Scikit-learna* i *Tensorflowa*. Modelirano ponašanje odabranog sustava podložno je promjenama, te ukoliko dođe do takve promjene model će ostvarivati lošije rezultate. Ta je pojava poznata kao odstupanje modela (engl. *concept drift*). Kontinuirana evaluacija modela i prepoznavanje trenutka u kojem je došlo do odstupanja važni su čimbenici uspješnih produkcijskih modela. U radu će se iznijeti postupci za evaluaciju i prilagodbu modela novom ponašanju modeliranih koncepata.

Mentor: doc.dr.sc. Darko Etinger

Komentor: dr.sc. Nikola Tanković

Kolegij: Umjetna inteligencija

IZJAVA O AKADEMSKOJ ČESTITOSTI

Ja, dolje potpisani Romeo Šajina, kandidat za magistra informatike ovime izjavljujem da je ovaj Diplomski rad rezultat isključivo mojega vlastitog rada, da se temelji na mojim istraživanjima te da se oslanja na objavljenu literaturu kao što to pokazuju korištene bilješke i bibliografija. Izjavljujem da niti jedan dio Diplomskog rada nije napisan na nedozvoljen način, odnosno da je prepisan iz kojega necitiranog rada, te da ikoji dio rada krši bilo čija autorska prava. Izjavljujem, također, da nijedan dio rada nije iskorišten za koji drugi rad pri bilo kojoj drugoj visokoškolskoj, znanstvenoj ili radnoj ustanovi.

Student

_____

U Puli, srpanj, 2019. godine

IZJAVA

o korištenju autorskog djela

Ja, Romeo Šajina dajem odobrenje Sveučilištu Jurja Dobrile u Puli, kao nositelju prava iskorištavanja, da moj diplomski rad pod nazivom „Kontinuirano oblikovanje, evaluacija i prilagođavanje modela vremenskih nizova“ koristi na način da gore navedeno autorsko djelo, kao cjeloviti tekst trajno objavi u javnoj internetskoj bazi Sveučilišne knjižnice Sveučilišta Jurja Dobrile u Puli te kopira u javnu internetsku bazu završnih radova Nacionalne i sveučilišne knjižnice (stavljanje na raspolaganje javnosti), sve u skladu s Zakonom o autorskom pravu i drugim srodnim pravima i dobrom akademskom praksom, a radi promicanja otvorenoga, slobodnoga pristupa znanstvenim informacijama.

Za korištenje autorskog djela na gore navedeni način ne potražujem naknadu.

U Puli, srpanj, 2019.

Potpis

_____

Zahvaljujem se svima koji su mi pomogli pri izradi ovog rada svojim savjetima i preporukama, a posebno mom komentoru dr.sc. Nikoli Tankoviću. Zahvaljujem se svojoj obitelji na strpljenju, moralnoj podršci i povjerenju koje su mi ukazali tijekom studija.

SADRŽAJ

1. Uvod

Vremenski nizovi su široko rasprostranjeni kroz razna područja u IT svijetu. Zbog svoje jednostavne strukture moguće je gotovo iz svakog procesa izvući podatke u obliku vremenskog niza koji će taj proces predstavljati. Vremenski nizovi se uglavnom analiziraju da bi se prepoznale njihove karakteristike u svrhu bolje predviđanja budućeg kretanja. Predviđanje potrošnje električne energije, predviđanje cijena dionica i predviđanje potrošnje proizvoda samo su neki od primjera predviđanja vremenskih nizova. Točno predviđanje vremenskog niza je vrlo korisno jer omogućava poduzimanje pravovremenih mjera kako bi proces kojeg vremenski niz predstavlja bio što efikasniji. Proizvođači električne energije žele što bolje predvidjeti koliko će se u nekom razdoblju potrošiti struje, gdje je bitno da se količina ne podcijeni jer će se tada prodati manje energije, također je bitno da se količina ne precijeni jer će se tada višak energije morati skladištiti, što izaziva dodatne troškove. Kod rezervacija hotelskih soba je cilj prodati sobe po najvišim cijena ovisno o tržištu, odnosno ponudi i potražnji. Ako su cijene niske, hotel će rasprodati sve sobe, ali neće ostvariti najveću dobit. Međutim ako su cijene visoke, hotel vjerojatno neće rasprodati sve sobe što znači da postoji neiskorišteni kapacitet. Rezervacije soba ovise i o prilikama na tržištu, odnosno postoji li velika ili mala potražnja za hotelskim sobama. Za ostvarivanje najboljeg odnosa cijene i broja prodanih soba potrebno je kontinuirano predviđati potencijalni broj rezervacija i u ovisnosti o popunjenosti napraviti korekcije cijena.

Postoji mnogo različitih metoda modeliranja vremenskih nizova, a odabir prave metode nije jednostavan. Često se kod analize vremenskih nizova testira više različitih modela i promatra njihovo ponašanje, pa se takav pristup prati kroz ovaj rad. Prema tome je cilj ovog rada prikazati proces analize vremenskog niza, od analize osnovnih karakteristika vremenskog niza do izrade i analize performansi različitih modela. U sljedećem poglavlju će biti prikazani i objašnjeni podaci koji su korišteni kroz rad za pojašnjavanje pojmova i modela. U trećem poglavlju će biti detaljnije pojašnjeni vremenski nizovi i ostali pojmovi koji se usko vezuju za njih. Nakon toga će biti prikazane mjere točnosti koje su korištene za evaluaciju svih modela kroz rad. U petom poglavlju su opisane metode predviđanja koje će modeli koristiti. U daljnjih tri poglavlja prikazani su razni modeli vremenskih nizova, uključujući pojašnjenje

rada modela i prikaz rezultata predviđanja modela. U pretposljednjem poglavlju su prikazani potencijalni problemi na koje model može naići u produkcijskom okruženju, zajedno sa strategijama savladavanja tih problema. U posljednjem poglavlju je napravljen osvrt na rezultate modela i metoda otkrivanja *concept drift*-a opisanih kroz rad.

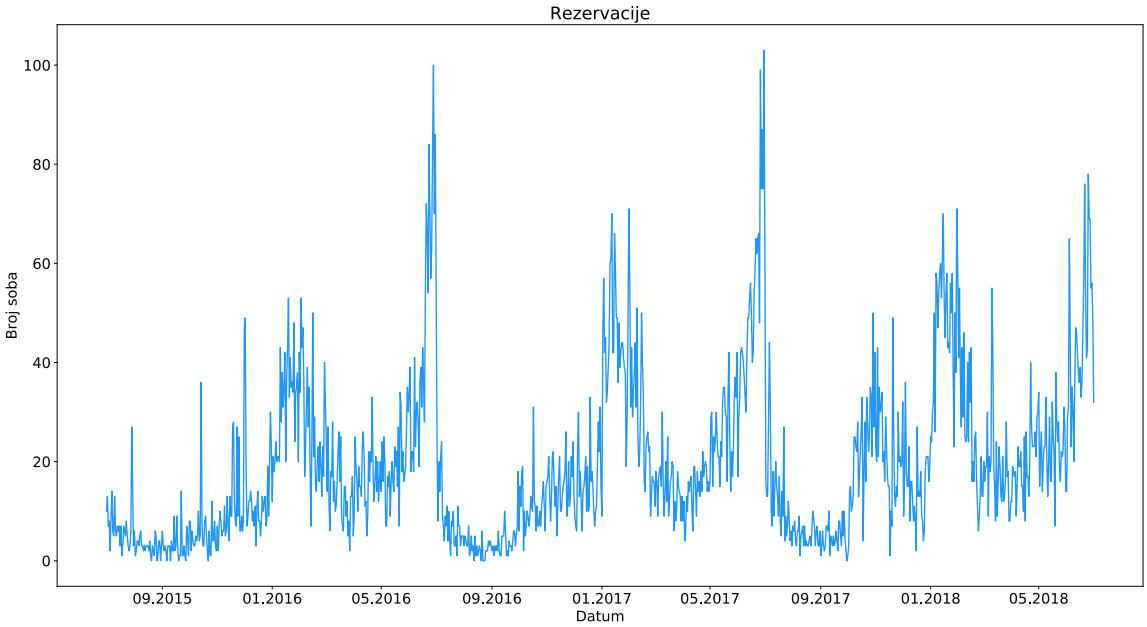
Svi modeli i potrebni podaci se nalaze u projektu koji se može pronaći na githubu: https://github.com/RomeoSajina/TSExploratory

2. Podaci korišteni u radu

U ovom će poglavlju biti opisani podaci korišteni kroz rad, koji će služiti za objašnjenje pojmova i izradu modela kod rada sa vremenskim nizovima. Podaci predstavljaju rezervacije soba za X hotela vodeće Hrvatske tvrtke kroz tri godine. Standardni prikaz takvih podataka bi prikazao koliko je na određeni dan bilo rezervacija, neovisno o datumu za koji su rezervacije napravljene. Međutim, moguće je podatke organizirati tako da se rezervacije promatraju samo za neki određeni dan (npr. 1.7.). Zapravo, da bi se rezervacija uključila u promatrani skup, boravišno vrijeme gosta mora uključivati promatrani datum. Bitno je napomenuti da jedna rezervacija može uključivati više soba, što je zapravo podatak koji se promatra.

Primjer: Ako gost 1.5. kreira rezervaciju sa 3 sobe i želi boraviti u hotelu od 1.7. do 7.7., ta će se rezervacija uključiti u promatrani skup za sve datume između 1.7. i 7.7. tako da se za dan 1.5. pribroje 3 sobe koje je gost rezervirao.

Takav pristup pri oblikovanju podataka je prikladan jer se otvara mogućnost, pomoću odgovarajućeg modela, predvidjeti kretanje rezervacija za određeni dan, a time i mogućnost korekcije cijena. Konkretan zahtjev tvrtke jest previđanje rezervacija 60, 30 i 7 dana unatrag od promatranog datuma, što znači da ako promatramo datum 1.7., predviđanja će se raditi na datume 2.5., 1.6. i 24.7. sa dotada prikupljenim podacima.
Kao glavni primjer promatrat će se rezervacije napravljene za 1.7. kroz tri godine, što je prikazano na slici 2.1



Slika 2.1: Vremenski niz rezervacija

3. Vremenski nizovi

Najjednostavnija definicija vremenskog niza jest da je to kronološki niz vrijednosti. Zapravo sve što se može promatrati sekvencijalno tijekom vremena je vremenski niz [13]. U svijetu postoji ogroman broj vremenskih nizova. Najpoznatiji primjeri vremenskih nizova su: cijena dionica, količina proizvedene električne energije, broj turista u zemlji ili nekoj drugoj jedinici, potrošnja određenih proizvoda, vrijednost BDP-a u zemlji i sl. Vremenski nizovi su posebna vrsta podataka jer se iz jednostavnog vizualnog prikaza može jako puno saznati o njima. Tako se može promatrati kretanje vrijednosti kroz vrijeme, može se usporediti trenutne trendove sa onim u prošlostima, ako postoji ciklično ponašanje unutar vremenskog niza može pomoći u shvaćanju poslovnog ciklusa organizacije, može se promatrati u kojim vremenskim razdobljima vremenski niz ima najveće vrijednosti i sl. Sve prethodno navedeno je vrlo važno kod predviđanja budućih kretanja vrijednosti vremenskog niza. U analizi vremenskih nizova bitno je prepoznati određena ponašanja koja se odvijaju u vremenskom nizu, kako bi metoda predviđanja budućih vrijednosti što bolje odgovarala vremenskom nizu. Mnogo ljudi pogrešno pretpostavlja da predviđanja nisu moguća u promjenjivom okruženju. Svako se okruženje mijenja, a dobar model obuhvaća način na koji se okruženje mijenja. Predviđanja rijetko pretpostavljaju da je okolina nepromijenjena. Ono što se obično pretpostavlja jest da će se način na koji se okolina mijenja nastaviti u budućnost [13]. Kroz sljedećih nekoliko poglavlja će biti objašnjena razna ponašanja koje iskušavaju vremenski nizovi, nakon čega slijede modeli vremenskih nizova.

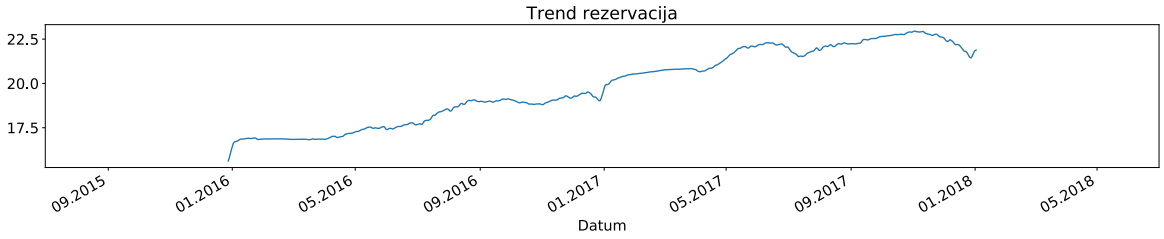
3.1. Trend, sezonalnost i cikličnost

Kod rada sa vremenskim nizovima učestalo se pojavljuju pojmovi trend, sezonalnost i cikličnost kojima se opisuju osnovne karakteristike vremenskih nizova. U ovom će se poglavlju spomenuti pojmovi pobliže pojasniti.

3.1.1. Trend

Trend se može prepoznati kada postoji dugoročno povećanje ili smanjenje u podacima. Trend ne mora biti linearan, već može poprimiti različite oblike pa je moguće da trend mijenja

smjerove, odnosno trend povećanja se može pretvoriti u trend opadanje i obrnuto [13]. Trend je u većini slučajeva jasno uočljiv, ali kada nije odmah uočljiv, to ne daje naznaku da trend ne postoji. Gledajući u vremenski niz rezervacija nije moguće odmah uočiti postoji li trend i u kojem smjeru se kreće. U tu svrhu možemo koristiti metodu iz knjižnice *statsmodels* [24] koja će između ostalog prikazati i trend u podacima. Na slici 3.1 je izdvojen samo prikaz trenda u podacima.



Slika 3.1: Trend vremenskog niza rezervacija

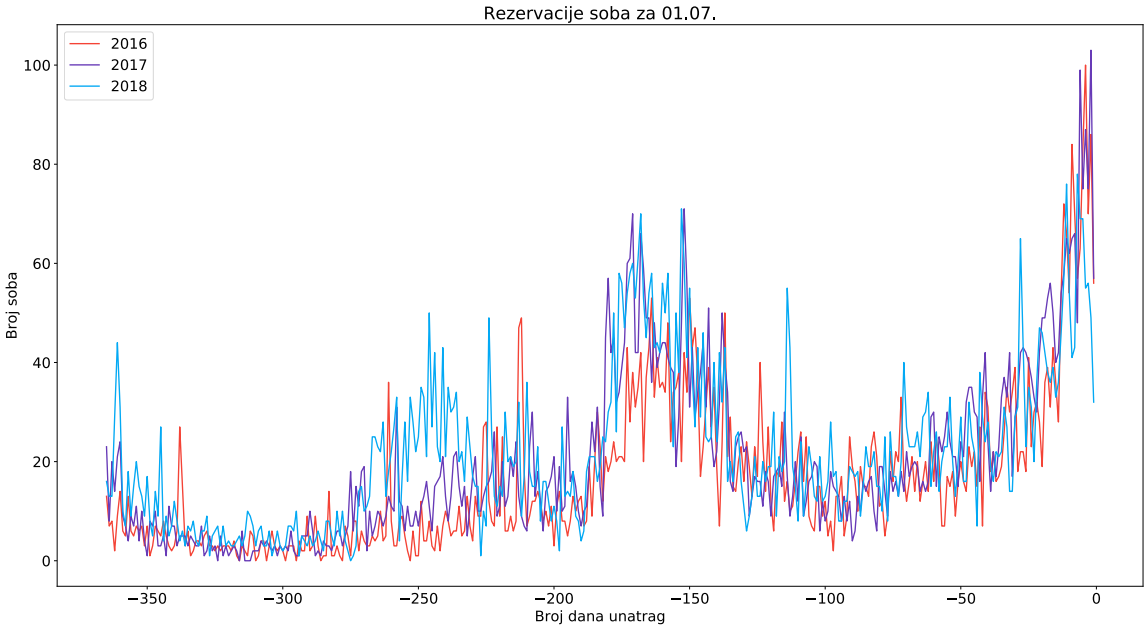
Iz slike 3.1 se može primijetiti da je do listopada 2017. godine trend bio rastući, nakon čega je okrenuo smjer kretanja u trend opadanja. Trend opadanja u ovom slučaju nije velik, ali može pomoći u objašnjenju lošijeg rezultata u 2018. godini.

3.1.2. Sezonalnost

Sezonalni uzorak nastaje kada na vremenski niz utječu sezonalni čimbenici kao što su doba godine ili dan u tjednu. Sezonalnost uvijek ima fiksnu ili poznatu učestalost, odnosno frekvenciju [13]. U primjeru vremenskog niza rezervacija jasno se može vidjeti da postoji godišnja sezonalnost, koja je posljedica specifičnog vremenskog razdoblja rada hotela i vremenskog doba, što se može vidjeti na slici 3.2.

3.1.3. Cikličnost

Ciklus se pojavljuje kada se u podacima pojavljuju porast ili pad koji nisu fiksne učestalosti, odnosno frekvencije. Takve su promjene obično posljedica ekonomskih uvjeta i često su povezana sa „poslovnim ciklusom“. Trajanje tih promjena je obično najmanje dvije godine [13]. Cikličnost je slična sezonalnosti u aspektu periodičnog ponavljanja uzorka, dok je različita u aspektu frekvencije, odnosno sezonalnost ima fiksnu frekvenciju dok cikličnost nema. U primjeru vremenskog niza rezervacija ne postoji dokaz o ikakvom cikličnom ponašanju, pa se može zaključiti da vremenski niz ne pokazuje ciklično ponašanje.



Slika 3.2: Sezonalnost vremenskog niza rezervacija

4. Mjere točnosti

Kroz rad će se, prilikom opisa pojedinog modela, prikazati performanse na primjeru vremenskog niza rezervacija, pa je potrebno definirati mjere točnosti modela. Najčešće mjere točnosti modela kod rada sa vremenskim nizovima su: Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE) itd. [3] Kroz ovaj rad koristit će se mjera Mean Absolute Error (MAE). Ona se definira kao:

$$MAE = \frac{1}{n}\sum_{t=1}^{n}|e_t|$$

Druga mjera koja će se koristiti kroz rad je točnost ukupnog predviđanja, poznata kao FA (engl. Forecasting Attainment). U primjeru vremenskog niza rezervacija, mjera će pokazati razliku stvarnog i predviđenog ukupnog broja rezerviranih soba u promatranom razdoblju. Mjera se može zapisati kao:

$$FA = \frac{\sum_{t=1}^{n} y'_t}{\sum_{t=1}^{n} y_t}$$

gdje je y'_t predviđena, a y_t stvarna vrijednost. Najbolji rezultat koji se kroz mjeru može ostvariti je 1.0 što u primjeru vremenskog niza rezervacija znači da je previđeni broj jednak stvarnom broju rezerviranih soba. Predstavljene mjere će se zajedno koristiti kroz ovaj rad za analizu performansi modela.

5. Metode predviđanja

U sljedećim poglavljima bit će predstavljeni razni modeli vremenskih nizova koji će se koristiti za predviđanje budućih vrijednosti, pa je važno objasniti na koji način će se predviđanja izvršavati. Postoje dvije glavne metode predviđanja: jednokoračno (engl. one-step) i višekoračno (engl. multi-step) predviđanje. Razlika između dviju metoda je značajnija nego što zvuči.

5.1. Jednokoračno predviđanje

Metoda jednokoračnog predviđanja uzima određene vrijednosti vremenskog niza i predviđa y_t. Vrijednosti koje se koriste sa metodom su uglavnom samo stvarne vrijednosti iz vremenskog niza. Ako se jednokoračnom metodom želi napraviti predviđanje nad podacima za trening $y_0, \ldots, y_n$, gdje je n broj dostupnih opservacija, onda će se za predviđanje y_t koristiti sve ili dio vrijednosti iz trening podataka $y_0, \ldots, y_n$.

Primjer: danim modelom je potrebno jednokoračnom metodom napraviti predviđanje sljedeće vrijednosti. Dani model mora imati dostupno minimalno n' opservacija kako bi mogao napraviti predviđanje (vrijednost n' je specifična za pojedini model), što znači da koristeći $y_0, \ldots, y_{n'}$ se predviđa prva moguća vrijednost $\hat{y}_{n'+1}$. Predviđena vrijednost se sprema u rezultat, ali se ne koristi za predviđanje vrijednosti $\hat{y}_{n'+2}$, već će se za njezino predviđanje koristiti stvarne vrijednosti $y_0, \ldots, y_{n'+1}$ iz podataka za trening (ukoliko postoje). Dakle iz ovog primjera je bitno izvući informaciju da se kod jednokoračnog predviđanja ne koriste predviđene vrijednosti za buduća predviđanja.

5.2. Višekoračno predviđanje

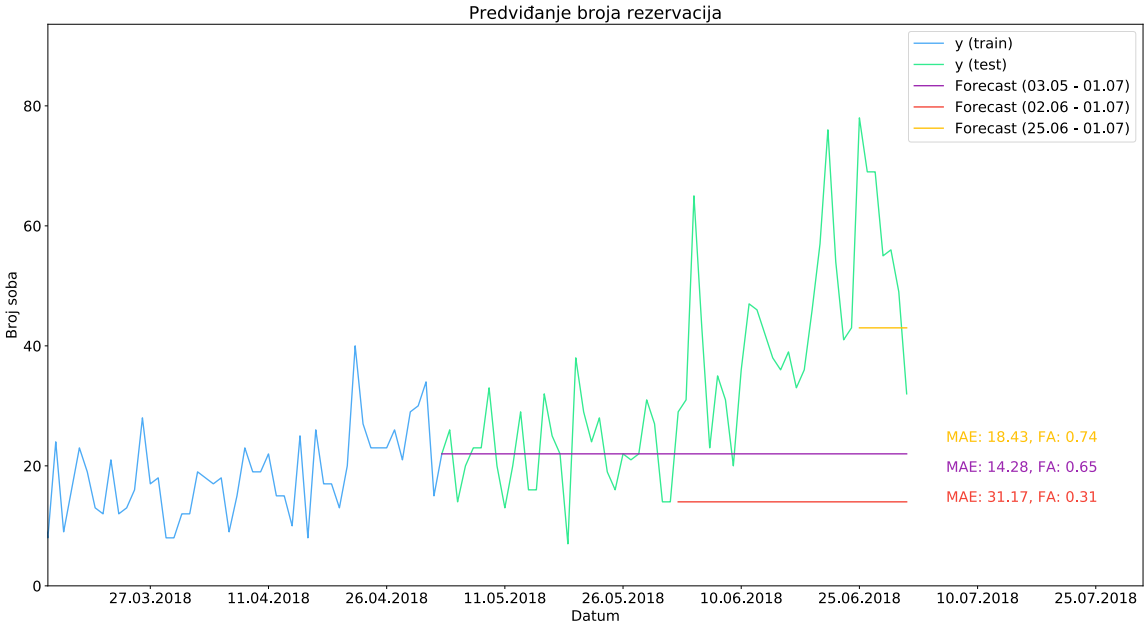
Predviđanje jednokoračnom metodom definira kako se može predvidjeti jedna sljedeća vrijednost, međutim problem nastaje kada želimo previdjeti više od jedne vrijednosti (npr. $y_{n+1}, y_{n+2}, y_{n+3}$). U tom trenutku se ne mogu koristiti samo podaci za trening jer ne postoje vrijednosti koje bi omogućile predviđanje k-te vrijednosti. Jedna od poznatijih metoda rješavanja ovog problema je iterativno korištenje jednokoračne metode gdje će se predviđene vrijednosti koristiti u predviđanju sljedećih vrijednosti [22, 15]. Ova metoda je izazovnija jer ima dobro poznat problem širenja ranih pogrešaka u buduća predviđanja, odnosno predviđene vrijednosti $\hat{y}_{n+1}$ i $\hat{y}_{n+2}$ će imati utjecaj na buduće vrijednosti. Drugim riječima, loše predviđanje u ranoj fazi može imati štetan utjecaj na buduća predviđanja [15]. Kroz rad će svi modeli koristiti ovu metodu za predviđanje rezervacija nad testnim skupom.

6. Naivni modeli

Kod rada sa vremenskim nizovima uvijek je dobra praksa prije početka analize složenijih modela izraditi jednostavne modele koji će biti osnova za validaciju ostalih modela. Osnovni modeli uobičajeno ne uključuju nikakvu obradu vremenskog niza, već za predviđanje uzimaju postojeće vrijednosti vremenskog niza prema definiranom vremenskom pomaku, prema čemu i naziv "naivni" (engl. Naive ili ponekad Persistent) modeli.

6.1. Naivni model

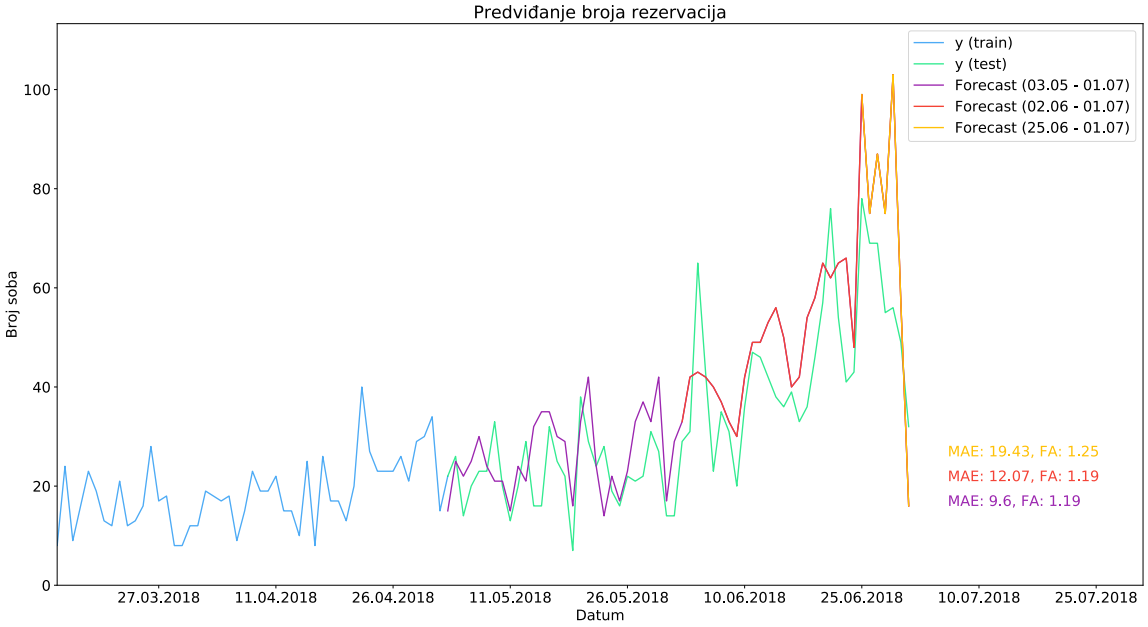
Kod najjednostavnije izvedbe naivnog modela se za predviđanje uzima posljednja dostupna vrijednost. Ova metoda radi izuzetno dobro za mnogo ekonomskih i financijskih vremenskih nizova kada vremenski niz poprima karakteristike bijelog šuma [13]. Formalno se može zapisati kao: $y_t = y_{t-1}$



Slika 6.1: Naivni model

6.2. Sezonalni naivni model

Kada vremenski niz sadrži sezonalnu komponentu može se izraditi "pametniji" naivni model. Za predviđanje se može iskoristiti vrijednost iz prošle sezone u istom trenutku [13]. Npr. za predviđanje vrijednosti za dan 1.7.2019. može se koristiti vrijednost na dan 1.7.2018. Formalno se može zapisati kao: $y_t = y_{t-m}$ gdje je m trajanje sezonalnog razdoblja (u primjeru vremenskog niza rezervacija m iznosi 365).



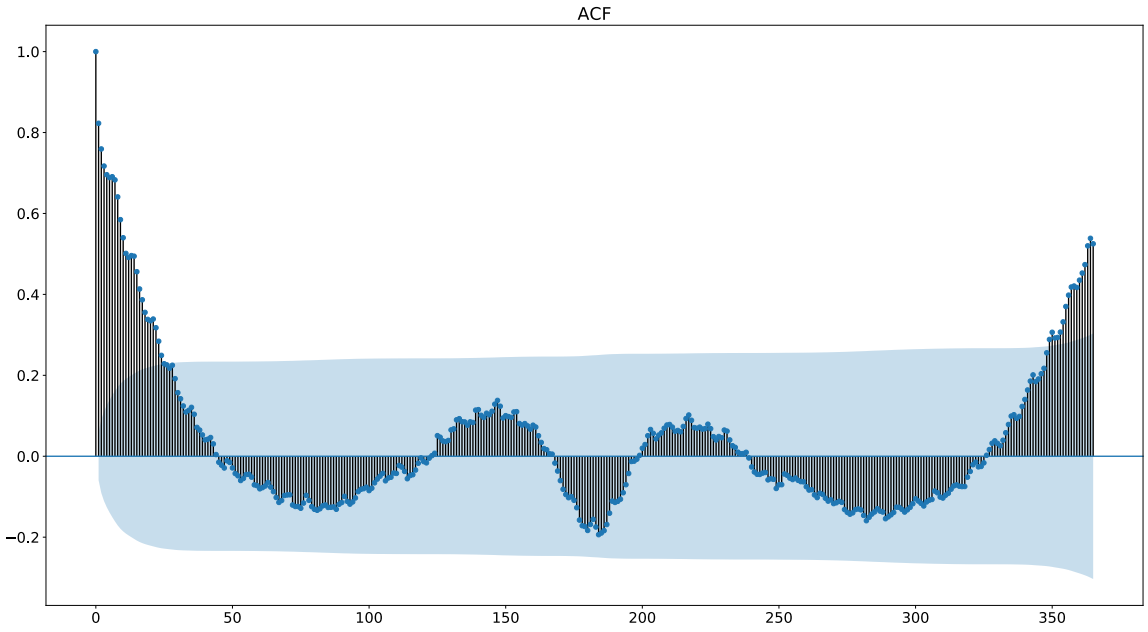
Slika 6.2: Sezonalni naivni model

7. Statistički modeli

7.1. Autokorelacija

Autokorelacija je blisko vezana sa korelacijom. Kao što korelacija mjeri opseg linearnog odnosa između dvije varijable, tako autokorelacija mjeri linearni odnos između prethodnih vrijednosti vremenskog niza [13]. Korelacija za vremenske nizove se može izračunati prema prethodnim vrijednostima koje se zovu vremenski pomaci (engl. lags). Kako se korelacija izračunava na vrijednostima istog vremenskog niza u različitim vremenima, ona se zove serijska korelacija (engl. serial correlation) ili autokorelacija.

Prikaz autokorelacije vremenskog niza prema vremenskom pomaku zove se AutoCorrelation Function (ACF). Kako bi se prikazala autokorelacija za primjer vremenskog niza rezervacija, koristit će se metoda iz knjižnice *statsmodels* [24]. Zadano se za izračun autokorelacije koristi Pearsonov koeficijent korelacije; broj između -1 i 1 koji opisuje negativnu ili pozitivnu korelaciju. Vrijednost 0 označava da ne postoji korelacija.

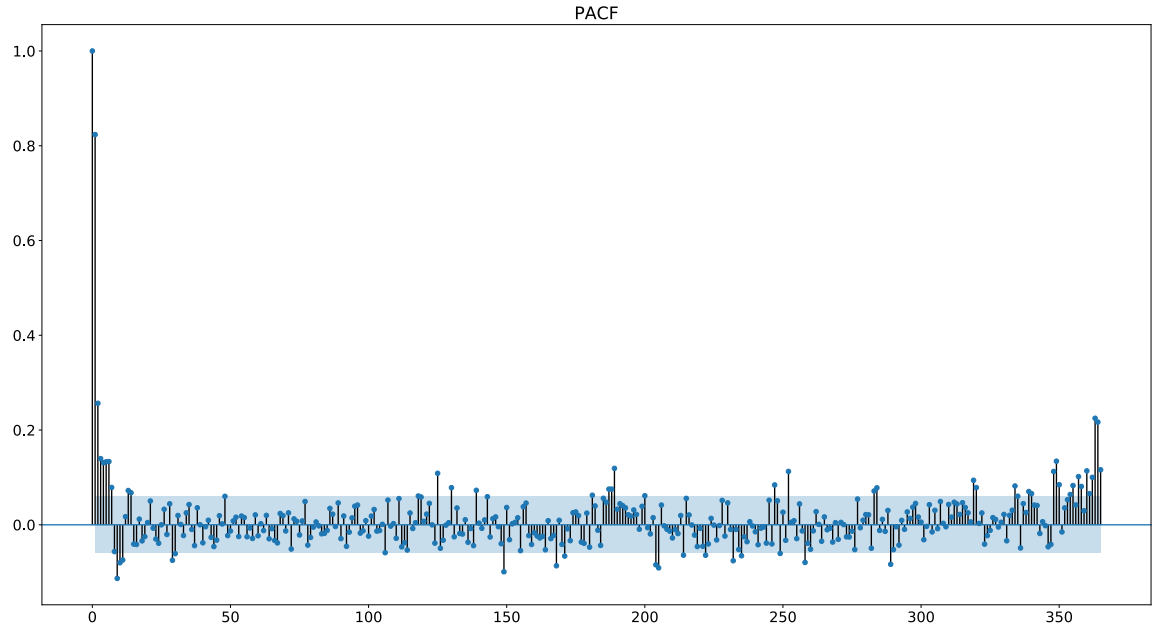


Slika 7.1: ACF prikaz

Na slici 7.1 se može vidjeti da postoje definirane granice do kojih je autokorelacija značajna, odnosno ako je vrijednost unutar granica onda ona nije u značajnoj korelaciji sa promatranom vrijednošću. Može se primijetiti da postoji značajna autokorelacija sa vremenskim pomakom do 25 dana.

7.2. Parcijalna autokorelacija

Autokorelacija za promatranu vrijednost i vrijednost u nekom prethodnom koraku se sastoji od direktne korelacije i indirektne korelacije. Indirektna korelacija se kroz primjer može prikazati sljedeći način: y_{t-3} ima indirektnu korelaciju sa y_{t-2} koji ima indirektnu korelaciju sa korakom y_{t-1} koji ima direktnu korelaciju sa y_t [13]. Parcijalna autokorelacija želi ukloniti indirektne korelacije kako bi se mogla izračunati samo direktna korelacija između y_t i y_{t-k} gdje k vremenski pomak. Prikaz parcijalne autokorelacije vremenskog niza prema vremenskom pomaku zove se Partial AutoCorrelation Function (PACF). Kako bi se prikazala parcijalna autokorelacija za primjer vremenskog niza rezervacija, koristit će se metoda iz knjižnice *statsmodels* [24].



Slika 7.2: PACF prikaz

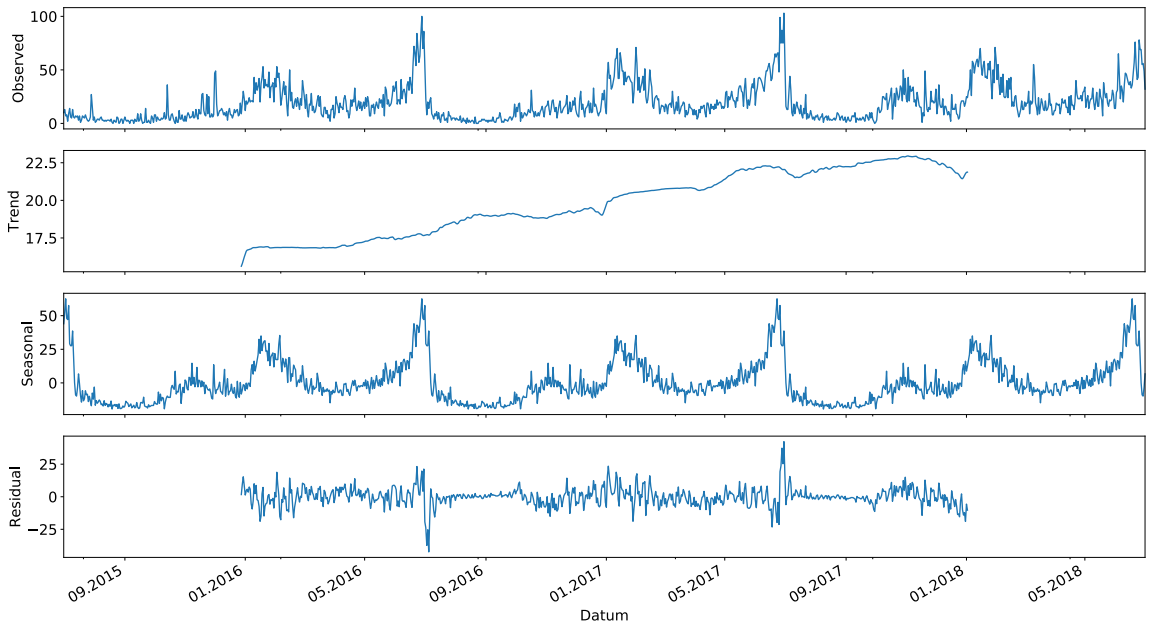
7.3. Bijeli šum

Bijeli šum (engl. White noise) je vremenski niz koji nema autokorelaciju. Kada je vremenski niz bijeli šum, autokorelacija će biti blizu 0 i 95% vrijednosti autokorelacije će biti unutar

granica ACF prikaza. Ako je više od 5% vrijednosti autokorelacije izvan granica ACF prikaza, vremenski niz vjerojatno nije bijeli šum [13]. Ispitivanje poprima li vremenski niz karakteristike bijelog šuma je ključno prije izrade autoregresivnih modela jer se oni oslanjaju na autokorelaciju. Za primjer vremenskog niza rezervacija može se primijetiti da nije bijeli šum jer postoji značajna autokorelacija do vremenskog pomaka 25, odnosno više od 5% vrijednosti je izvan granice ACF prikaza.

7.4. Dekompozicija vremenskog niza

Podaci vremenskog niza mogu pokazivati različite uzorke, a često je korisno podijeliti vremenski niz u nekoliko komponenti od kojih svaka predstavlja osnovni uzorak kategorije. Kada se napravi dekompozicija vremenskog niza, obično se kombinira trend i cikličnost u jednu komponentu trend-cikličnost (ponekad zvana trend zbog jednostavnosti). Prema tome, vremenski niz se sastoji od tri komponente: komponenta trend-cikličnost, komponenta sezonalnosti i komponenta preostalog (sadrži ostatak vremenskog niza) [13]. Ova metoda se većinom koristi zbog boljeg razumijevanja vremenskog niza i izgradnje boljeg predikcijskog modela. Za prikaz dekompozicije primjera vremenskog niza rezervacija koristi će se metoda iz knjižnice *statsmodels* [24].



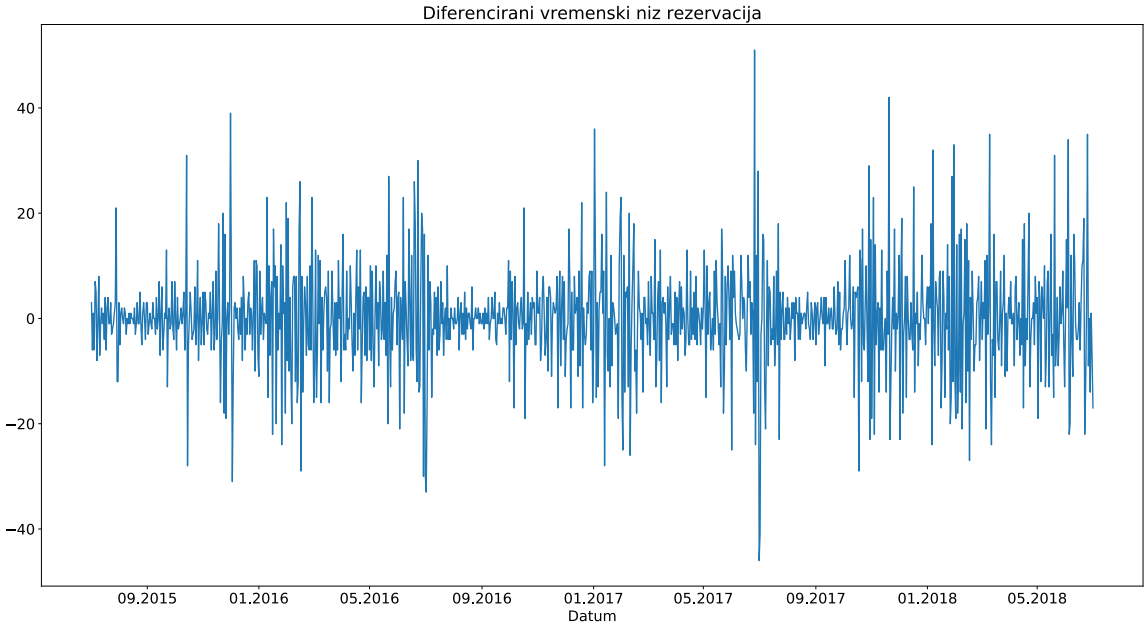
Slika 7.3: Dekompozicija vremenskog niza rezervacija

Iz slike 7.3 možemo primijetiti, iako iz podataka nije uočljivo, da postoji blago rastući trend koji je promijenio smjer u 2018. godini kao što je već prethodno spomenuto. U komponenti sezonalnosti može se primijetiti da postoji jaka sezonalnost koja se godišnje ponavlja. Kom-

ponenta preostalog prikazuje ostatak oduzimanja komponenti sezonalnosti i trend-cikličnosti od podataka.

7.5. Stacionarnost

Stacionarni vremenski niz je onaj čija svojstva ne ovise o vremenu u kojem se vremenski niz promatra. Dakle, vremenski niz sa trendovima ili sezonalnosti uglavnom nije stacionaran jer će trend i sezonalnost različito utjecati na vrijednosti vremenskog niza u različitim vremenima [13]. Stacionarnost vremenskog niza je nužna kod izgradnje statističkog predikcijskog modela, zato je potrebno izvršiti provjeru stacionarnosti vremenskog niza i napraviti potrebne transformacije ako je to potrebno. Jedna od poznatijih metoda za pretvaranje ne-stacionarnog vremenskog niza u stacionarni jest diferenciranje. Diferenciranje izvršava transformaciju podataka izračunavajući razlike između uzastopnih opservacija. Diferenciranje može pomoći stabilizirati srednju vrijednost vremenskog niza uklanjanjem promjene u razini vremenskog niza, te time eliminira (ili smanjuje) trend i sezonalnost.



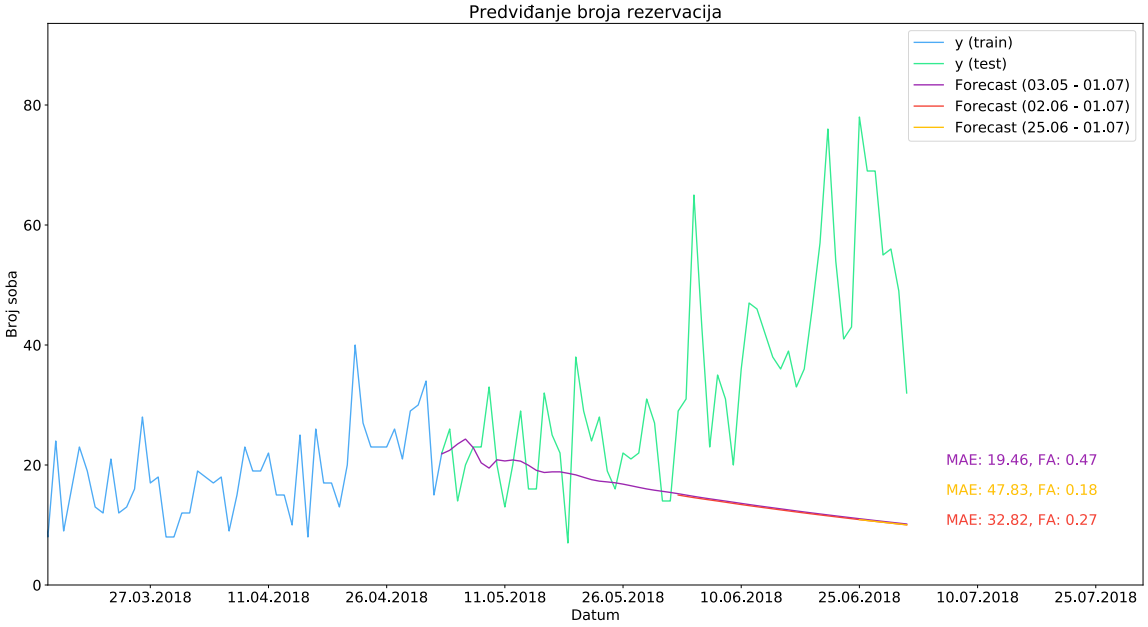
Slika 7.4: Diferencirani vremenski niz rezervacija

Ponekad stacionarnost, odnosno ne-stacionarnost, nije uočljiva samo iz vizualne inspekcije vremenskog niza. Postoji nekoliko metoda ispitivanja stacionarnosti vremenskog niza, a uglavnom se koristi matematička Dickey and Fuller [6] metoda. Početna hipoteza koja se želi opovrgnuti nalaže da vremenski niz nije stacionaran. Za ispitivanje stacionarnosti primjera vremenskog niza rezervacija koristi će se nadopunjena verzija Dickey-Fuller testa nazvana Augmented Dickey and Fuller (ADF) test. Nakon izračuna testa za vremenski niz rezer-

vacija, rezultat p-vrijednosti je 0.0013 što je manje od granične vrijednosti 0.05, odnosno odbacuje se početna hipoteza da vremenski niz nije stacionaran i prihvaća se hipoteza da je vremenski niz stacionaran.

7.6. Autoregresivni model

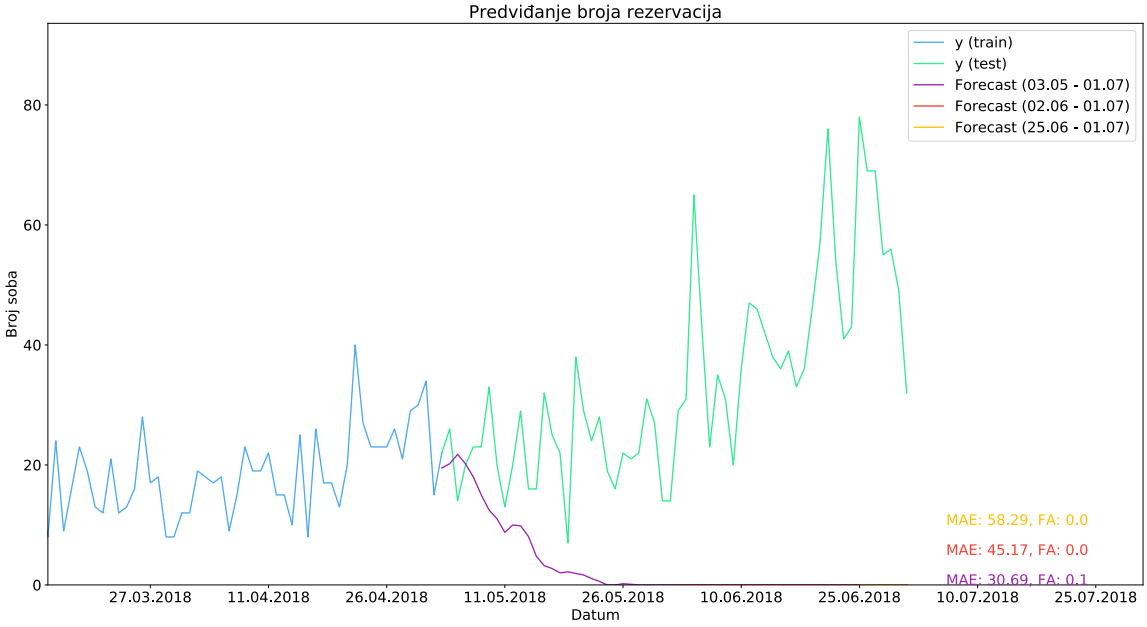
U autoregresivnim modelima predviđamo željenu varijablu koristeći linearnu kombinaciju njenih prethodnih vrijednosti. Pojam autoregresija ukazuje da je to regresija varijable sa samom sobom [13]. Autoregresijski model razine **p** se može zapisati kao: $y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \ldots + \phi_p y_{t-p} + \epsilon_t$, gdje je c konstanta, $\phi_1, \ldots, \phi_p$ su parametri vremenskih pomaka $y_{t-1}, \ldots . y_{t-p}$, a ϵ_t opisuje bijeli šum. Formula definira **AR(p)** model razine **p**. Za određivanje razine **p** može se iščitati vremenski pomak posljednje vrijednosti u PACF prikazu koja je iznad granice značajnosti. Za primjer vremenskog niza rezervacija će to biti vremenski pomak 9, pa će model biti **AR(9)**.



Slika 7.5: Rezultat AR modela na vremenskom nizu rezervacija

7.7. Model pomičnih prosjeka

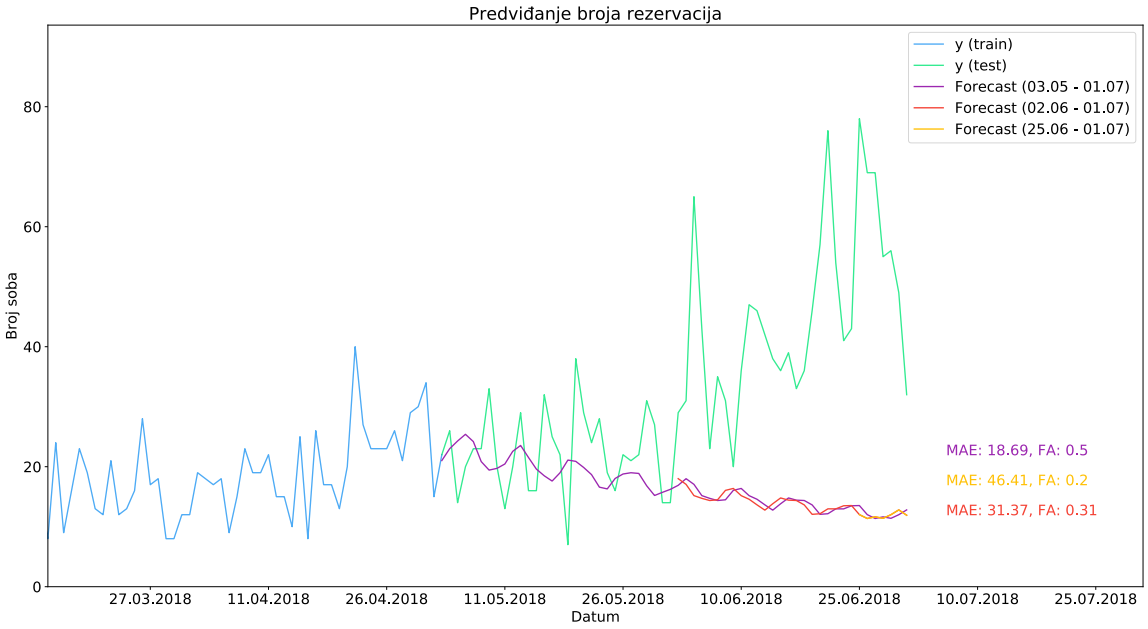
Za razliku od autoregresijskih modela, modeli pomičnih prosjeka ne koriste prošle vrijednosti varijable koju se predviđa, već koriste greške dobivene previđanjem varijable u prošlosti, time dobivajući model sličan regresijskom modelu [13]. Model pomičnih prosjeka razine **q** može se dati formulom: $y_t = c + \epsilon_t + \theta_1\epsilon_{t-1} + \theta_2\epsilon_{t-2} + \ldots + \theta_q\epsilon_{t-q}$, gdje je c konstanta, ϵ_t opisuje bijeli šum, $\theta 1, \ldots, \theta_q$ su parametri grešaka $\epsilon_{t-1}, \ldots, \epsilon_{t-q}$ predviđanja vremenskih pomaka. Formula definira **MA(q)** model razine **q**. Za određivanje razine **q** može se iščitati vremenski pomak posljednje vrijednosti u ACF prikazu koja je iznad granice značajnosti. Za primjer vremenskog niza rezervacija će to biti vremenski pomak 25, pa će model biti **MA(25)**.



Slika 7.6: Rezultat MA modela na vremenskom nizu rezervacija

7.8. ARIMA model

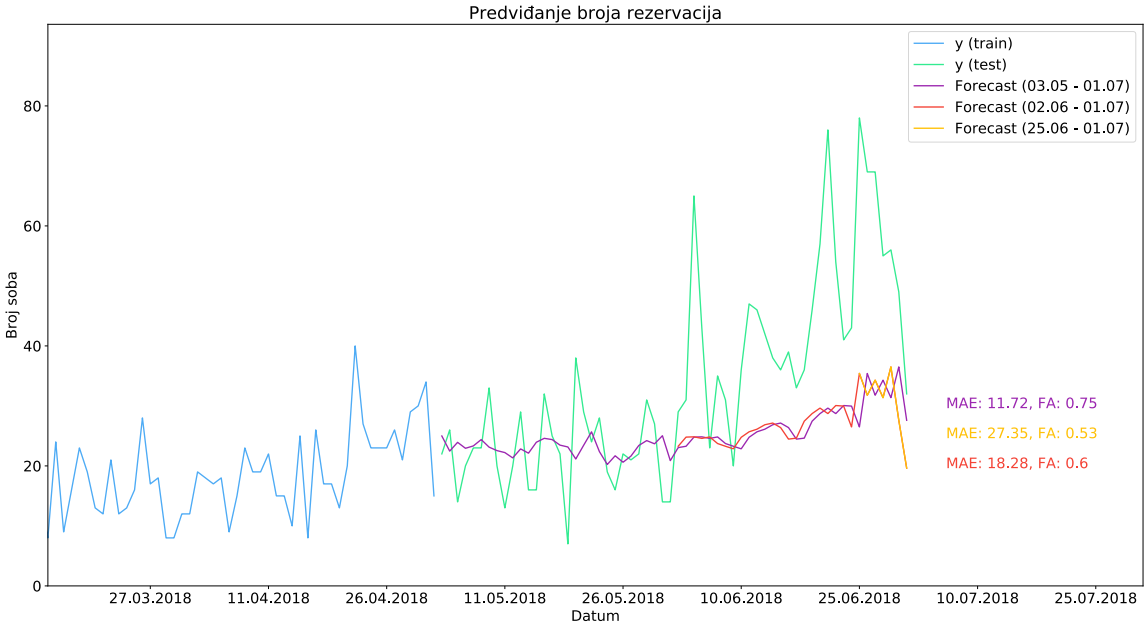
ARIMA (engl. AutoRegressive Integrated Moving Average) modeli su kombinacija diferenciranja, autoregresijskih modela i modela pomičnih prosjeka. ARIMA model se može dati formulom: $y'_t = c + \phi_1 y'_{t-1} + \ldots + \phi_p y'_{t-p} + \theta_1 \epsilon_{t-1} + \ldots + \theta_q \epsilon_{t-q} + \epsilon_t$, gdje je y'_t diferencirani niz koji može biti diferenciran više puta [13]. Formula definira **ARIMA(p, d, q)** model autoregresivnog dijela razine **p**, dijela pomičnih prosjeka razine **q** i stupnja diferenciranja **d**. Za primjer vremenskog niza rezervacija su vrijednosti **p** i **q** već prethodno definirane, dok će vrijednost **d** iznositi 0 jer je ADF-test pokazao da nije potrebno diferencirati vremenski niz da bi se postigla stacionarnost.



Slika 7.7: Rezultat ARIMA modela na vremenskom nizu rezervacija

7.9. SARIMA model

Sezonalni ARIMA (SARIMA) model je generalizirani ARIMA model koji može raditi sa sezonalnim vremenskim nizovima. SARIMA model je formiran uključivanjem dodatnih sezonskih izraza u ARIMA model. SARIMA model se može prikazati kao **ARIMA(p,d,q,)(P,D,Q)m** gdje je **m** broj opservacija godišnje. Sezonalni dio modela se sastoji od izraza sličnom ne sezonalnom dijelu, ali uključuje vremenske pomake sezonalne komponente [13]. U primjeru vremenskog niza rezervacija vrijednost **m** iznosi 365 zbog dnevnih podataka. Ako uzmemo da su **p** i **q** vrijednosti 1 dobiva se model $\mathbf{ARIMA(1,0,1)(1,0,1)_{365}}$ koji se može dati formulom: $(1-\phi_1 B)(1-\Phi_1 B^{365})(1-B)(1-B^{365})y_t = (1+\theta_1 B)(1+\Theta_1 B^{365})\epsilon_t$. Sezonalni i nesezonalni parametri **p** i **q** će za primjer vremenskog niza iznositi 1 zbog jednostavnosti modela i smanjivanja potrebe za ogromnim resursima prilikom treniranja.

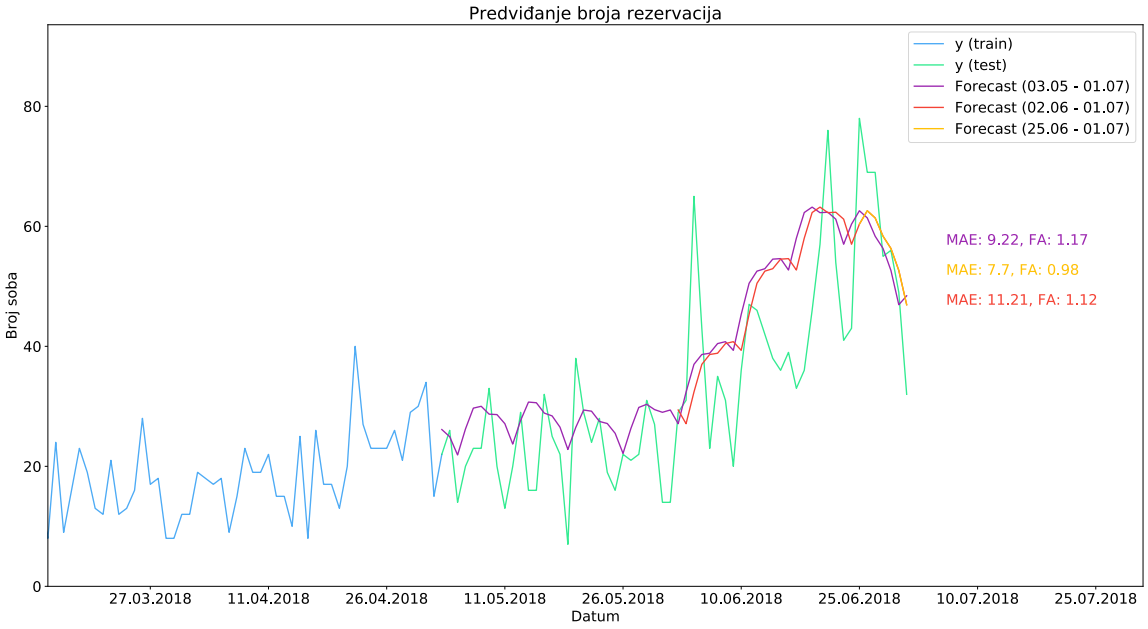


Slika 7.8: Rezultat SARIMA modela na vremenskom nizu rezervacija

7.10. Prophet model

Prophet model je razvijen od strane Facebook-a u svrhu predviđanja kretanja različitih vremenskih nizova, od kojih je jedan primjer: predviđanje broja izrađenih događaja na Facebook-u grupirano po danima u tjednu. Prophet je osmišljen da bude skalabilan i pristupačniji za korištenje kako bi ga mogli koristiti i početnici u radu sa vremenskim nizovima. Prophet koristi dekompozicijski model vremenskih nizova sa tri glavne komponente: trend, sezonalnost i praznici. Formalnije Prophet se u najjednostavnijem obliku može dati kao: $y(t) = g(t) + s(t) + h(t) + \epsilon_t$, gdje je $g(t)$ funkcija trenda koja modelira ne-periodične promjene u vremenskom nizu, $s(t)$ funkcija koja modelira periodične promjene, odnosno sezonalnost i $h(t)$ funkcija koja modelira posljedice praznika koji se odvijaju u potencijalno nepravilnom rasporedu kroz jedan ili više dana. Pojam pogreške ϵ_t predstavlja bilo koje specifične promjene koje nisu zahvaćene modelom [26].

Jedna od značajnijih prednosti kod Prophet modela jest da može pratiti više trendova tako da prati određeni trend do neke točke u vremenu, nakon koje onda prati drugi trend itd. Kao što je i prethodno spomenuto, model zapaža sezonalnost u podacima koja može biti definirana ili automatski otkrivena. Prophet uzima u obzir praznike i događaje, kao što su Božić ili Europsko prvenstvo u nogometu, koji inače u vremenske nizove mogu donijeti određenu razinu poremećaja, odnosno u većini slučajeva ne prate periodično ponašanje vremenskog niza. Praznici i događaji mogu biti definirani ili učitani iz postojećih definicija. Prophet model se može kreirati pomoću metode iz knjižnice *fbprophet*.



Slika 7.9: Rezultat Prophet modela na vremenskom nizu rezervacija

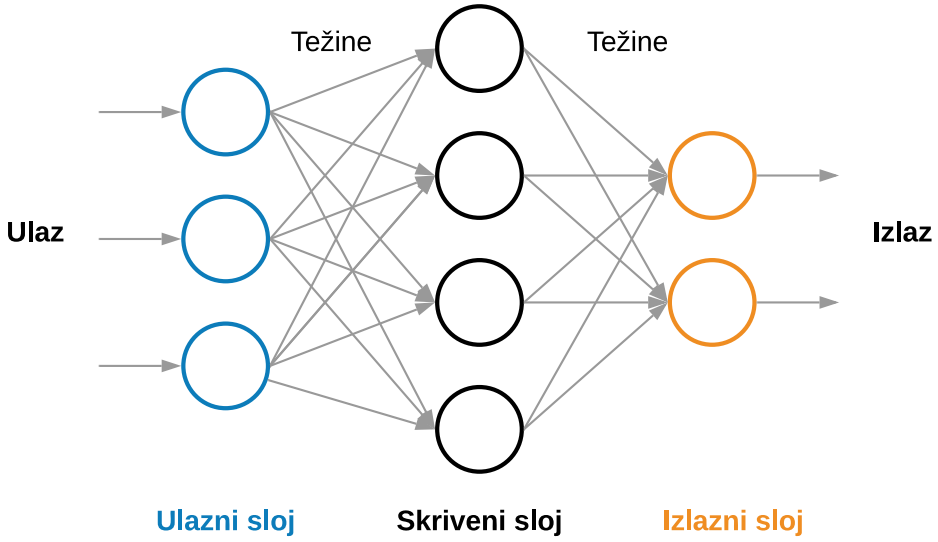
8. Neuronske mreže

Neuronske mreže (engl. Neural Network - NN) su trenutno možda i najpoznatija tehnika strojnog učenja. Nastale su kao imitacija ljudskog mozga koji se sastoji od živčanih stanica i sinapsi koje ih povezuju. Neuronske mreže su već uspješno primijenjene na probleme prepoznavanja slika, prepoznavanja govora, pretvaranja teksta u govor, a u novije vrijeme neuronske mreže su sposobne i generirati novi sadržaj (slika, tekst, zvuk). Neuronske mreže se mogu koristiti i za modele vremenskih nizova što će biti i prikazano dalje kroz ovaj rad, no prije toga slijedi pojašnjenje principa rada neuronskih mreža.

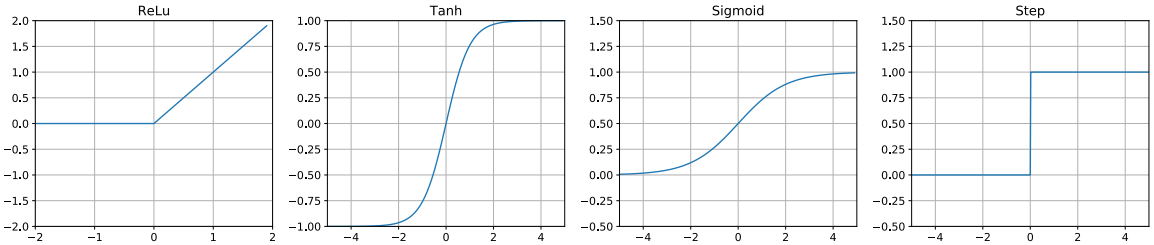
Cilj neuronske mreže je da aproksimira neku funkciju f^*, odnosno mreža uči parametre θ da što bolje aproksimiraju funkciju $y = f(x;\theta)$ gdje je x ulazni podatak, a y izlazni podatak [10]. Neuronske mreže se sastoje od tri vrste slojeva: ulazni sloj, skriveni sloj i izlazni sloj. Svaka neuronska mreža ima jedan ulazni i jedan izlazni sloj, dok skrivenih slojeva može biti jedan ili više. Konceptualni prikaz neuronske mreže i njezinih slojeva može se vidjeti na slici 8.1. Svaki sloj se sastoji od jednog ili više neurona koji su spojeni sa neuronima u sljedećem sloju. Unutar svakog neurona nalaze se težine za svaki od ulaza, gdje je ulaz veza koja dolazi iz prethodnog neurona. Da bi neuron mogao dati izlaznu vrijednost, što konačno može biti ulaz u drugi neuron, mora je prvo izračunati. Izlazna vrijednost je suma umnožaka težina i ulaza koja se zatim provlači kroz aktivacijsku funkciju. Zadaća aktivacijske funkcije je da normalizira izlaznu vrijednost. Neke od najčešćih aktivacijskih funkcija su *ReLu*, *Sigmoid*, *Tanh*, *Step* koje su prikazane na slici 8.2. Ako uzmemo za primjer aktivacijsku funkciju *ReLu*, onda će izlazna vrijednost biti 0 ako je suma umnožaka težina i ulaza manja ili jednaka 0, u suprotnom će vrijednost funkcije biti pozitivna. Opisani proces se ponavlja redoslijedno za svaki sloj i svaki neuron. Kada proces dođe do posljednjeg sloja i posljednjeg neurona, mreža će dati određeni izlaz koji zatim možemo usporediti sa stvarnom vrijednošću i odrediti pogrešku. Jednom kada je greška izračunana može se pokrenuti proces propagacije pogreške unatrag (engl. Backpropagation), gdje se greška propagira unatrag kroz slojeve. Kako bi se odredila mjera koliko je potrebno promijeniti težine pojedinog neurona da bi se greška smanjila, izračunava se gradijent greške. Kod računanja gradijenta prvo se određuju greške posljednjeg sloja, zatim se ukupna greška propagira na prethodni sloj. Tada se za taj sloj

(pretposljednji) određuje kako je svaki neuron utjecao na grešku. Nakon toga se određuje gradijent i ažurira vrijednost težine u ovisnosti o izračunatom gradijentu. Opisano se ponavlja za svaki sloj, a završava kada dostigne prvi (ulazni) sloj i tada se zaključuje jedna epoha treniranja modela. Proces treniranja mreže je uvijek isti pa tako postoji nekoliko poznatih knjižnica koje već pružaju sve potrebne funkcionalnosti za kreiranje i treniranje neuronskih mreža. Jedna od najpoznatijih knjižnica je *Tensorflow* [1], koja će se dalje koristiti kroz rad za izgradnju svih modela neuronskih mreža.

Prije izrade samih modela potrebno je formulirati problem koji će odgovarati potrebama za treniranje neuronskih mreža. Postoji više načina formulacije problema, a u ovom radu će biti opisane neuronske mreže sa unaprijednom propagacijom funkcijskog signala (engl. Feed Forward Neural Network – FNN), neuronske mreže sa vremenskim pomakom (engl. Time Lagged Neural Networks – TLNN) i sezonalne neuronske mreže (engl. Seasonal Artificial Neural Networks - SANN).



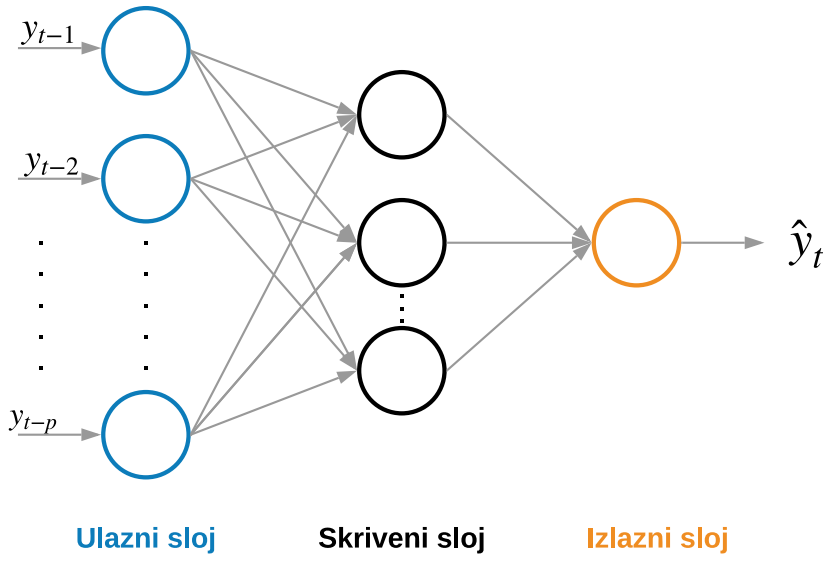
Slika 8.1: Pojednostavljeni prikaz neuronske mreže



Slika 8.2: Aktivacijske funkcije

8.1. Feed Forward Neural Network

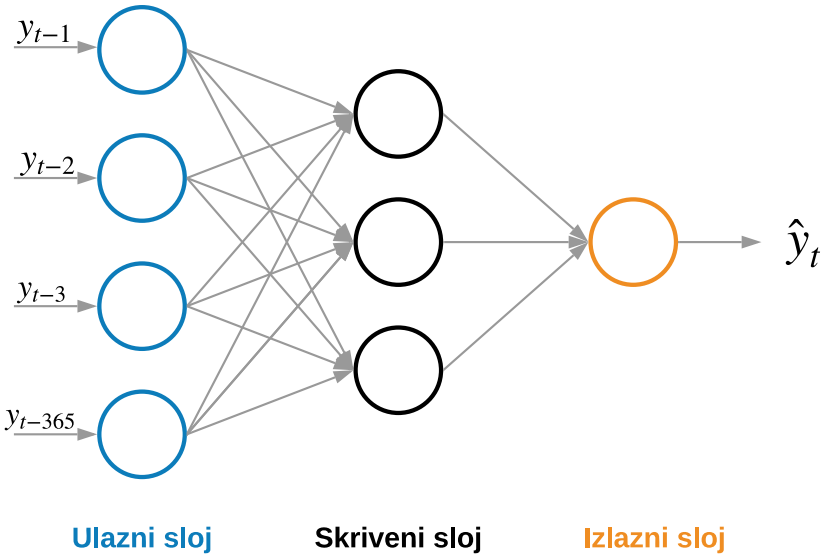
FNN pristup formulacije je na neki način sličan pristupu kao kod ARIMA modela. Za formulaciju problema uzima se sekvencijski niz sa vremenskim pomakom p, što znači da je $y_t = y_{t-1}, \ldots y_{t-p}$. Detaljnije, $y_{t-1}, \ldots y_{t-p}$ su ulazne vrijednosti u model, dok je y_t izlazna vrijednost modela [3]. Prema izrazu bi se moglo zaključiti da će model povećanjem p biti točniji, što je često točno u praksi. Međutim maksimalni p je ponajviše ograničen količinom dostupnih podataka, odnosno p mora biti manji od broja opservacija da bi postojao barem jedan uzorak za treniranje modela. Dakle, povećanjem p-a; smanjuje se broj mogućih uzoraka za treniranje (što može smanjiti generalizacijsku moć modela), povećava se veličina modela i usporava se proces treniranja modela. Izgled FNN modela je prikazan na slici 8.3.



Slika 8.3: Feed Forward Neural Network – FNN

8.2. Time Lagged Neural Network

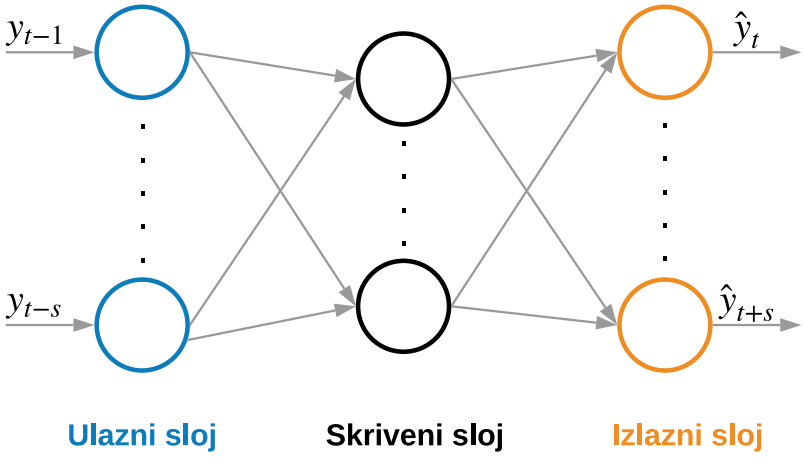
Za razliku od FNN-a, TLNN ne definira pravilo da ulazne vrijednosti za predviđanje y_t moraju biti sekvencijalni niz $y_{t-1}, \ldots, y_{t-p}$. U TLNN-u su ulazne vrijednosti odabrane prema određenim vremenskim pomacima [3, 9]. Na primjer: za predviđanje y_t kada vremenski niz ima dnevnu frekvenciju i sadrži sezonalnu komponentu se mogu odabrati $y_{t-1}, y_{t-2}, y_{t-365}$, što znači da se y_t može predvidjeti koristeći vrijednosti prethodna dva dana i istog dana prošle godine. Izgled TLNN modela je prikazan na slici 8.4.



Slika 8.4: Time Lagged Neural Networks - TLNN

8.3. Seasonal Artificial Neural Network

SANN metoda je osmišljena kako se poboljšale performanse na neuronskim mrežama kod rada sa sezonalnim vremenskim nizovima. Glavna razlika SANN-a u odnosu na FNN je u broju ulaznih i izlaznih vrijednosti koji su u SANN-u isti. SANN se formalno može zapisati kao $y_t, \ldots, y_{t+s} = y_{t-1}, \ldots, y_{t-s}$, gdje je s dužina sezonskog razdoblja [9, 11]. Detaljnije, $s = 365$ za dnevne podatke, $s = 12$ za mjesečne podatke, $s = 4$ za tromjesečne podatke itd. Izgled SANN modela je prikazan na slici 8.5.



Slika 8.5: Seasonal Artificial Neural Network – SANN

Problem ne mora nužno biti definiran u opisanim granicama. Problem se može definirati i kao ulaz sekvencijskog niza $y_{t-1}, \ldots, y_{t-p}$ i izlaz sekvencijskog niza $y_t, \ldots y_{t+p'}$ gdje je p' broj izlaznih vrijednosti. Nadalje, mogu se odabrati ulazne vrijednosti prema određenim

vremenskim pomacima kao kod TLNN-a, a izlaz može biti sekvencijalni niz $y_t, \ldots y_{t+p'}$ [9]. Za optimalnu formulaciju problema potrebno je testirati više varijanti i promatrati kako se ponaša točnost modela. Za formulaciju primjera vremenskog niza rezervacija je odabrana FNN metoda koja kao ulaz definira sekvencijalni niz za vremenski pomak p odnosno niz $y_{t-1}, \ldots, y_{t-p}$, a kao izlaz sekvencijalni niz dužine p' odnosno $y_t, \ldots y_{t+p'}$. Nadalje, kako bi se mogla promatrati različita ponašanja modela, problem je formuliran kroz 12 varijanti koje su dane u tablici 8.1.

Verzija	Broj ulaznih vrijednosti (p)	Broj izlaznih vrijednosti (p')
1	60	1
2	100	1
3	200	1
4	300	1
5	60	3
6	100	3
7	200	3
8	300	3
9	60	7
10	100	7
11	200	7
12	300	7

Tablica 8.1: Varijante formulacije problema

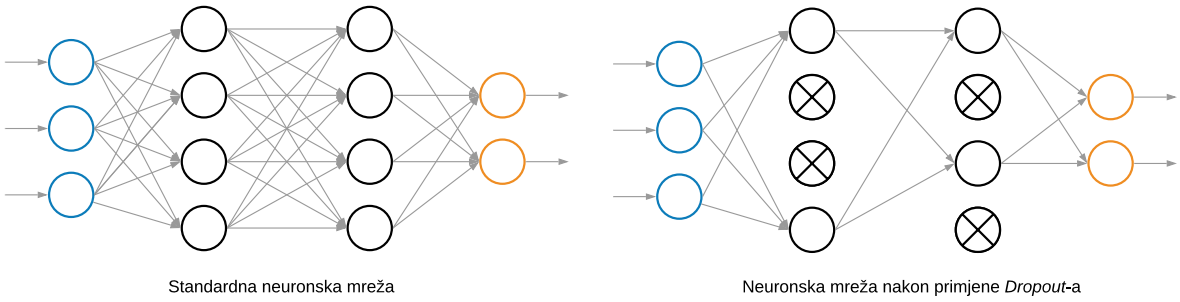
8.4. Optimizacija neuronskih mreža

Neuronske mreže mogu biti spore za treniranje i sklone preprilagođenosti trening podacima. Tako se u svrhu sprječavanja preprilagođenosti može koristiti metoda ispadanja (engl. Dropout) koja pospješuje generalizaciju mreže. Za ubrzavanje brzine treniranja mreže, uglavnom kod CNN-ova, se može koristiti metoda skupne normalizacije (engl. Batch Normalization). Metode će biti opisane prije modela neuronskih mreža jer su one korištene u svakom modelu prikazanom u daljnjim poglavljima.

8.4.1. Dropout

Metodu ispadanja (engl. Dropout) su izvorno predložili N. Srivastava i ostali [25] kao način regularizacije mreže. *Dropout* sprječava neuronske mreže od preprilagođenosti trening podacima i pruža način efikasnog kombiniranja mnogo različitih arhitektura neuronskih mreža.

Dropout radi na principu privremenog uklanjanja vidljivih i skrivenih neurona iz mreže, zajedno sa njihovim ulaznim i izlaznim vezama. Izbor neurona koji će biti uklonjeni je slučajan, a ovisi o fiksnoj vjerojatnosti p neovisnoj o ostalim neuronima. Uglavnom p može biti postavljen na 0.5 što znači da će biti nasumično odabrano 50% neurona za uklanjanje. *Dropout* radi samo prilikom treniranja, dok se kod testiranja *Dropout* ne koristi već se težine na izlaznim vezama neurona množe sa vjerojatnošću zadržavanja neurona prilikom treniranja. To osigurava da je izlaz bilo kojeg skrivenoga neurona jednak stvarnom izlazu prilikom testiranja.



Slika 8.6: Lijevo: Standardna neuronska mreža **Desno:** Neuronska mreža nakon primjene *Dropout*-a

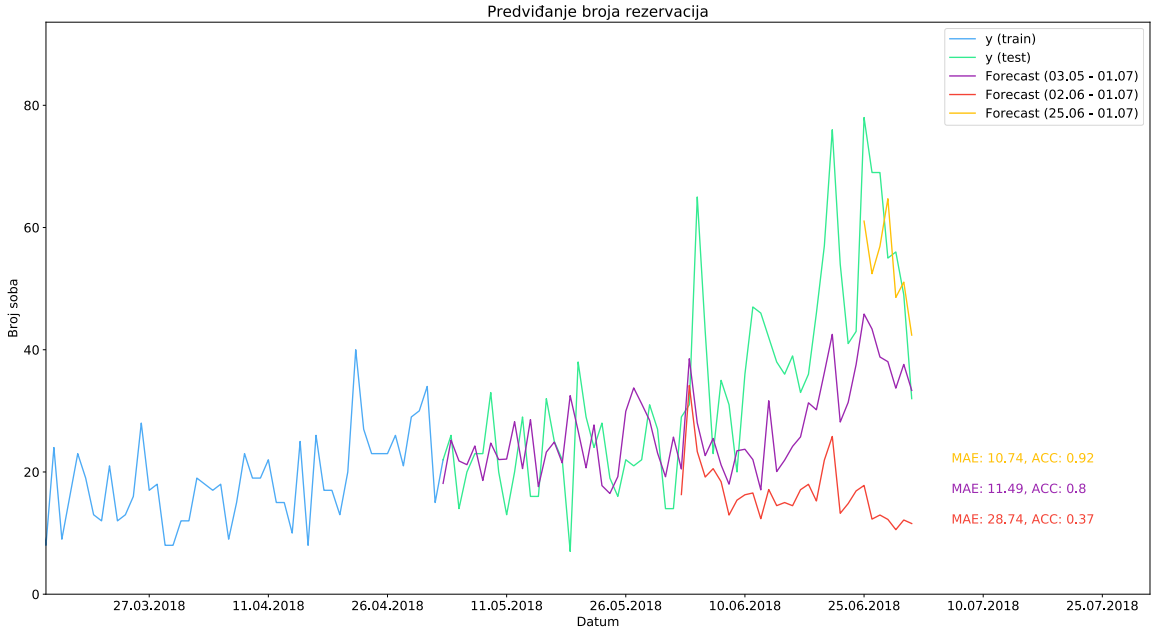
8.4.2. Batch Normalization

Skupna normalizacija (engl. Batch Normalization) izvorno predložena od S. Ioffe i C. Szegedy [14] kao rješenje unutrašnjeg kovarijantnog pomaka. Problem unutrašnjeg kovarijantnog pomaka nastaje prilikom ažuriranja parametara mreže tijekom treniranja mreža. On se manifestira kao promjena distribucije izlaznih vrijednosti sloja, što tada postaje ulaz u sljedeći sloj. Promjena u distribucijama ulaznih vrijednosti predstavlja problem jer se slojevi konstantno trebaju prilagođavati novim distribucijama što jako usporava treniranje mreže. Skupna normalizacija rješava taj problem normalizirajući srednju vrijednost izlaznog sloja blizu 0, a njezino standardno odstupanje blizu 1. To zapravo nije algoritam optimizacije, već je riječ o metodi adaptivnog parametriziranja.

8.5. Multilayer perceptron

Višeslojni perceptron (engl. Multilayer perceptron - MLP) odnosno višeslojna mreža sa unaprijednom propagacijom funkcijskog signala je najpoznatija vrsta neuronske mreže. U MLP-u su svi neuroni sloja povezani sa svim neuronima sljedećeg sloja (engl. fully connected). Ovo je jedna od jednostavnijih izvedbi neuronske mreže, ali ima tendenciju pružati

dobre rezultate. Iako ne sadrži nikakav sklop za praćenje vremenske komponente u vremenskim nizovima, MLP se pokazao kao dobra arhitektura za predviđanje vremenskih nizova [2, 16]. Zbog svoje jednostavnosti, trajanje treniranja MLP-a je vrlo kratko. Na slici 8.7 prikazan je rezultat modela kod predviđanja vremenskog niza rezervacija.

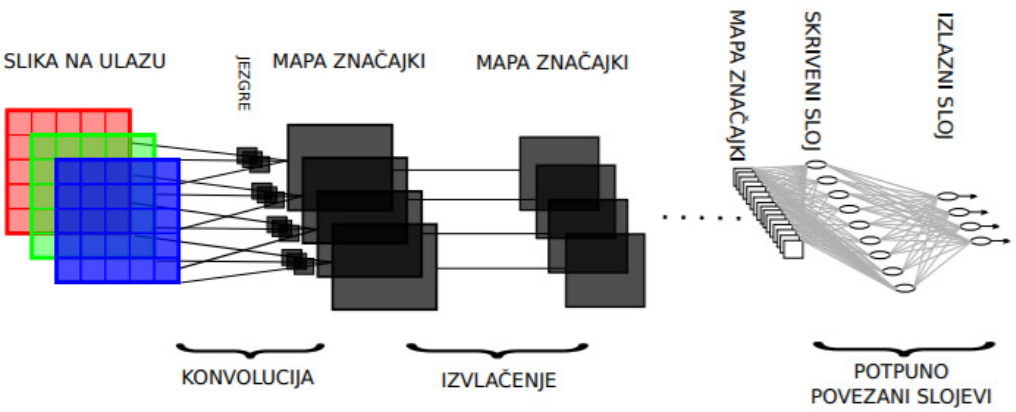


Slika 8.7: Rezultat MLP modela na vremenskom nizu rezervacija

8.6. Convolutional neural networks

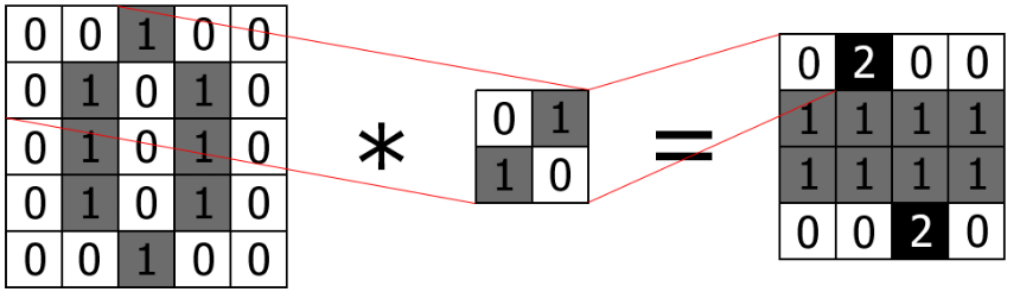
Konvolucijske neuronske mreže (engl. Convolutional neural networks - CNN) je izvorno predložio Y. LeCun kao arhitekturu mreže koja bi mogla klasificirati slike bolje nego dotadašnja rješenja. CNN su mreže specijalizirane za obradu podataka oblika rešetke. Slike su idealan primjer takve strukture podataka, dok se tu mogu uvrstiti i vremenski nizovi koji se mogu promatrati kao jednodimenzionalni podaci [10].

CNN u ulaznom sloju prihvaća neobrađene (originalne) podatke, u slučaju RGB slike je to matrica dimenzija $3x$širina$x$dužina. U ulaznom sloju se ne događaju nikakve transformacije nad podacima već se one odvijaju na sljedećem sloju nazvanom konvolucijski sloj. Konvolucijski sloj sadrži filtere pomoću kojih se, metodom pomičnih okvira, izračunava konvolucija dijelova slike. Kao rezultat tog procesa dobivaju se mape značajki (engl. feature maps). Svaki filter konačno rezultira u mapi značajki, koje se zatim slažu na stog, što su ujedno i izlazni podaci sloja. Treniranjem mreže se uči filtere prepoznavati rubove i uzorke oblika, a u dubljim mrežama i kompleksnije oblike. Moguće je graditi dublje mreže jer ulaz u ko-



Slika 8.8: Convolutional neural networks - CNN (preuzeto iz [4])

nvolucijski sloj mogu biti podaci iz ulaznog sloja ili izlazni podaci iz konvolucijskog sloja [20].

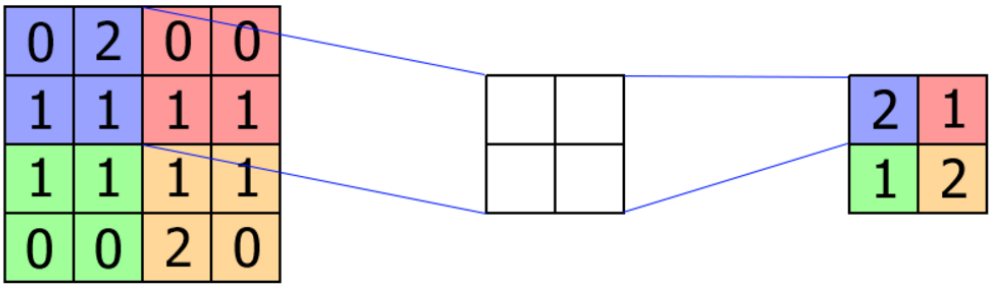


Slika 8.9: Vizualni prikaz konvolucije (preuzeto iz [4])

Sloj sažimanja (engl. pooling layer) služi za smanjivanje dimenzija podataka. Ideja je izvršiti sažimanje na definiranom okviru za cijeli ulazni podatak i izvući jednu vrijednost koja reprezentira taj okvir. Sažimanje je moguće izvršiti na dva načina:

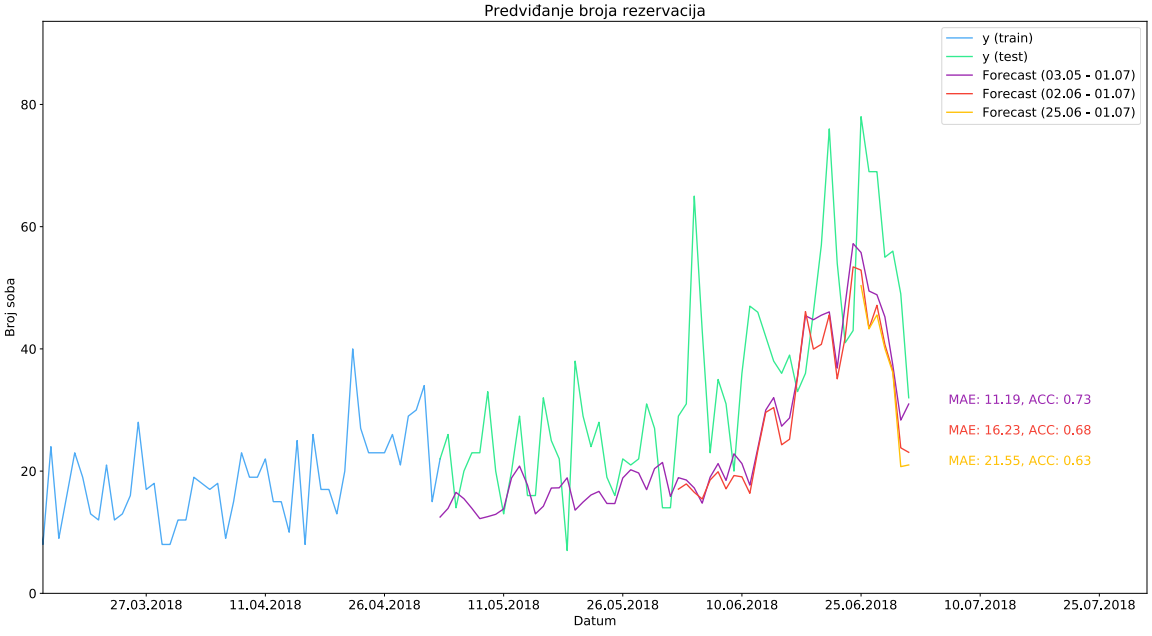
- **Sažimanje usrednjavanjem (engl. mean pooling)**: uzima aritmetičku sredinu vrijednosti koje se nalaze unutar okvira sažimanja.
- **Sažimanje maksimalnom vrijednošću (engl. max pooling)**: uzima maksimalnu vrijednost unutar okvira sažimanja. Ova verzija se najčešće koristi u praksi.

Parametar koji kontrolira vrijednost pomaka okvira naziva korak (engl. stride). Česta veličina okvira je $2x2$ sa korakom 2 što će smanjiti podatke na 25% originalne veličine [20].
CNN mreža tipično na kraju ima jedan ili više potpuno povezanih slojeva koji će izlazne vrijednosti iz konvolucijskih slojeva pripremiti za izlaz iz mreže. Izlaz tako može biti jedna od mogućih klasa u problemu klasifikacije ili vrijednost y_t u problemu vremenskog niza.

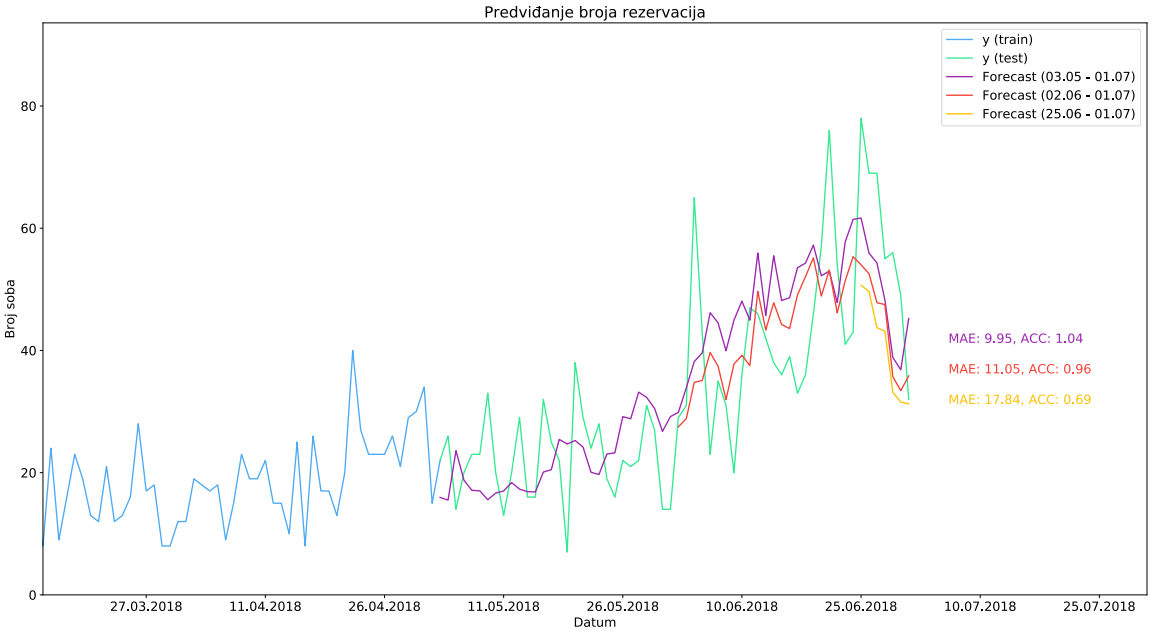


Slika 8.10: Vizualni prikaz sažimanja (preuzeto iz [4])

Glavna prednosti CNN-a je mogućnost izvlačenja glavnih značajki kroz konvolucijske slojeve bez prethodne analize i pripreme podataka, već će one biti naučene kroz treniranje. Motivacija za korištenje CNN-a za predviđanje vremenskih nizova jest sposobnost mreže da nauči filtere prepoznavati ponovljene obrasce u vremenskom nizu, koje dalje može koristiti za predviđanje budućih vrijednosti. Neki od primjera uspješnosti CNN-a u predviđanju vremenskih nizova uključuju: predviđanje električne energije [16] i predviđanje cijene dionica [7]. Na slici 8.11 su rezultati modela sa jednim konvolucijskim slojem, a na slici 8.12 su rezultati modela sa tri konvolucijska sloja.



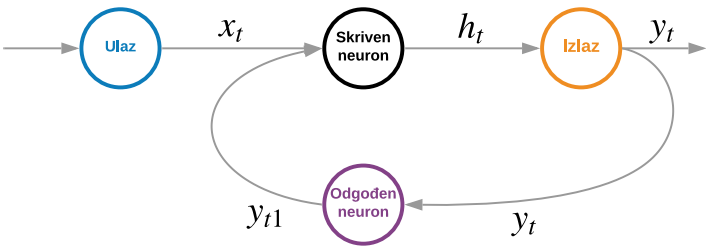
Slika 8.11: Rezultat CNN modela na vremenskom nizu rezervacija



Slika 8.12: Rezultat MultiCNN modela na vremenskom nizu rezervacija

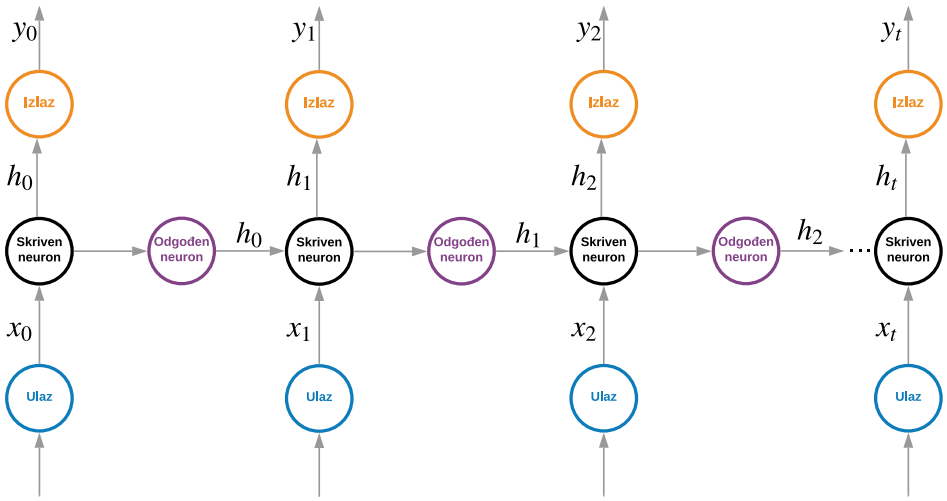
8.7. Recurrent Neural Network

Neuronska mreža sa povratnom vezom (Recurrent Neural Network - RNN) je vrsta neuronske mreže koja je specijalizirana za obradu sekvencijskih nizova $x^{(1)}, \ldots x^{(t)}$. RNN mreže omogućavaju obradu puno dužih nizova nego što bi to bilo moguće sa neuronskim mrežama koje nisu specijalizirane za obradu sekvencijskih nizova [10]. RNN sadrži sakrivena stanja koja su distribuirana kroz vrijeme, što omogućava spremanje mnogo informacija o prošlosti. RNN je primjer dinamičkih neuronskih mreža što znači da izlaz ovisi o trenutnom ulazu, prethodnim ulazima, prethodnim izlazima i/ili skrivenim stanjima unutar mreže [18]. Mogućnost RNN-a da kombinira podatke iz prošlosti sa trenutnim ulazom ostvarena je kroz povratne veze koje sadrži svaki neutron. U svakom vremenskom koraku mreža obrađuje ulazni vektor x_t, zatim ažurira svoja skrivena stanja putem aktivacijske funkcije h_t, koju koristi za predviđanje izlaza y_t. Svaki neuron u skrivenom sloju prima ulaze iz prethodnog sloja i izlaze trenutnog sloja iz prethodnog vremenskog koraka. Vrijednost koja je sadržana u odgođenom neuronu se koristi kao dodatni ulaz u skriveni neuron [18].



Slika 8.13: Pojednostavljen prikaz neurona RNN mreže

RNN radi sa sličnim algoritmom treniranja kao i ostale mreže nazvan propagacija pogreške unatrag kroz vrijeme (engl. Backpropagation Through Time). Osnovna ideja je odmotati RNN kroz vrijeme, pri se čemu otkriva neuronska mreža sa unaprijednom propagacijom koja se može trenirati standardnim algoritmom. Kada se greška primjeni na mrežu sa unaprijednom propagacijom, ona se zamota natrag u originalno stanje kao RNN.

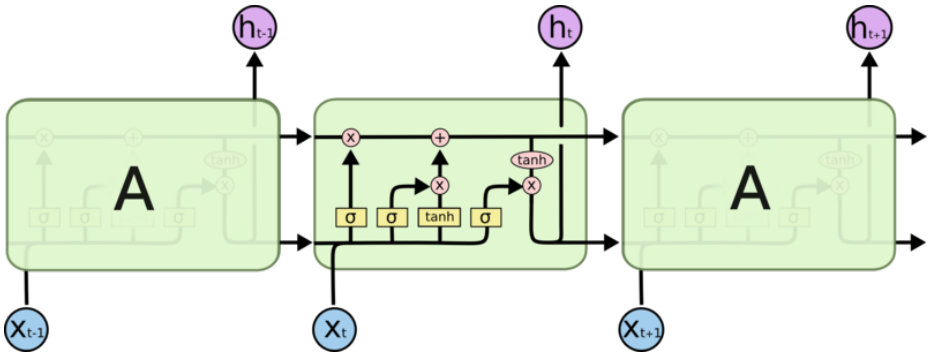


Slika 8.14: Odmotana RNN mreža

Kod treniranja RNN mreže kroz mnogo faza postoji mogućnost javljanja problema nestajućeg ili eksplodirajućeg gradijenta. Problem se javlja zbog učenja dugoročnih ovisnosti gdje će gradijent biti jako mali, time sprječavajući daljnje učenje mreže, ili će gradijent eksplodirati čime će napraviti veliku štetu procesu optimizacije. Jedno od mogućih rješenja na ovaj problem pružaju mreže koje se temelje na propusno povratnim ćelijama (engl. gated recurrent unit). Najpoznatije i trenutno najefikasnije su LSTM i GRU koje su predstavljene u sljedećim poglavljima.

8.7.1. Long Short-Term Memory

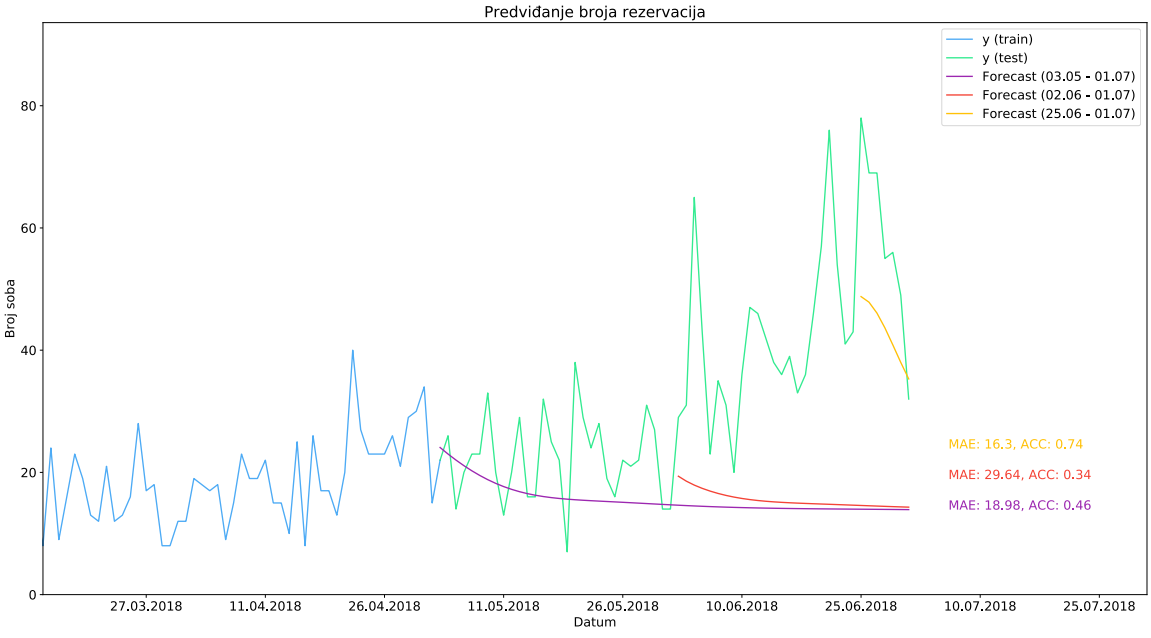
Mreže duge kratkoročne memorije (engl. Long Short-Term Memory - LSTM) su 90-ih godina predložili S. Hochreiter i J. Schmidhuber [12] kao rješenje problema dugoročnog pamćenja. U LSTM-u jedan blok sadrži četiri različite ćelije koje sadrže određena stanja. Vrijednosti stanja tih ćelija se mijenjaju kroz vrata (engl. gate) koja određuju koje će informacije prolaziti i ažurirati stanje. Razlikujemo tri vrste vratiju: ulazna, izlazna i vrata za zaboravljanje. Ona se sastoje od jednostavnih slojeva neurona sa određenom aktivacijskom funkcijom, uobičajeno sigmoidnom.



Slika 8.15: LSTM blok

Vrata za zaboravljanje su uvedena kako bi se mreža mogla resetirati, odnosno kontroliraju koliko informacija iz ulaza x_t i prethodnog izlaza y_{t-1} treba zadržati. Ulazna vrata definiraju koje informacije je potrebno spremiti u stanje ćelije, time sprječavajući spremanja nepotrebnih informacija. Izlazna vrata odlučuju koliko će informacija iz memorije poslati na izlaz iz bloka.

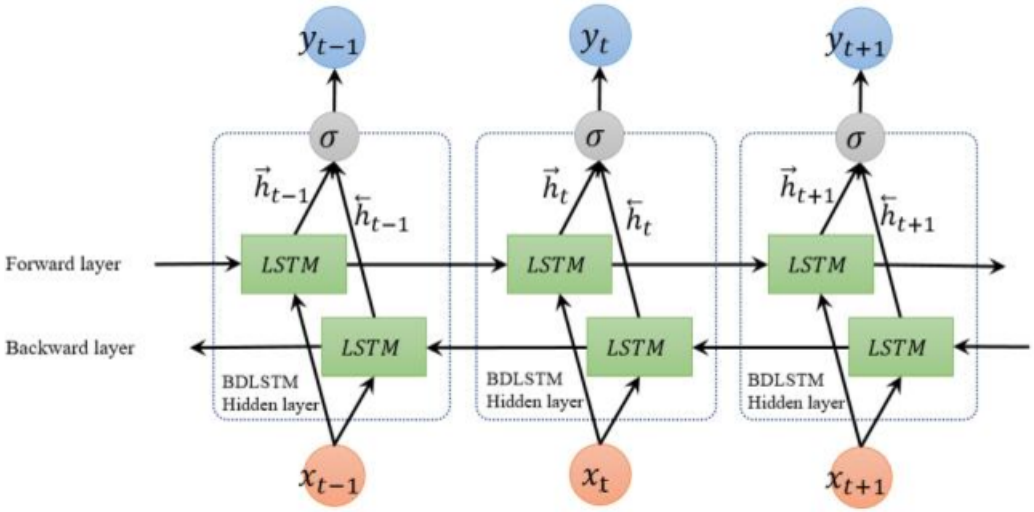
Zbog svoje mogućnosti dugoročnog pamćenja LSTM bi trebao odgovarati problemu predviđanja vremenskih nizova. U praksi to uglavnom nije slučaj, već će običan MLP ostvariti bolje performanse što je dokazano i u [16, 2]. U malo drugačijoj arhitekturi mreže, LSTM ipak može dati dobre rezultate [28]. LSTM ipak najbolje radi u kombinaciji sa nekim drugim vrstama mreže. Na slici 8.16 prikazan je rezultat modela kod predviđanja vremenskog niza rezervacija.



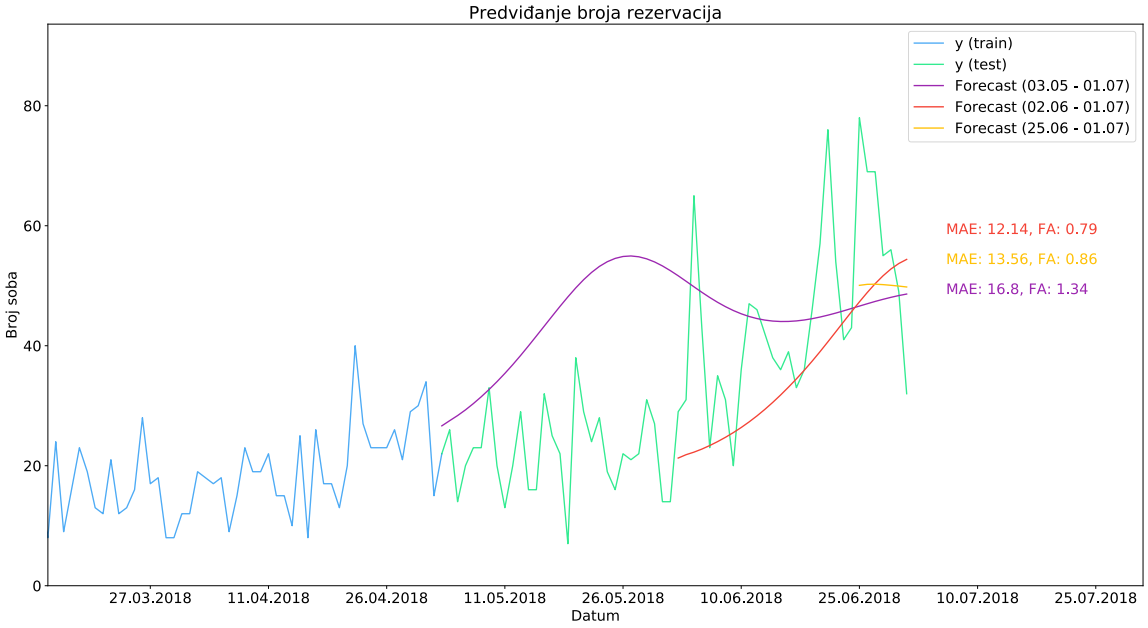
Slika 8.16: Rezultat LSTM modela na vremenskom nizu rezervacija

8.7.2. Bidirectional RNN

Dvosmjerno povratnu neuronsku mrežu (engl. Bidirectional Recurrent Neural Network - BRNN) su izvorno predložili M. Schuster i K. Paliwal [23] kao nadogradnju postojećih RNN mreža. Kod standardnih RNN mreža stanje u vremenu t zapaža samo informacije iz prošlosti $x_1, \ldots, x_{x-1}$ i trenutni ulaz x_t, dok u BRNN-u predviđanje za y_t ovisi o cijelom ulaznom nizu. BRNN kombinira RNN koji se kreće unaprijed kroz vrijeme, krećući od početka niza, zajedno sa dodatnom RNN mrežom koja se kreće unatrag kroz vrijeme krećući se od kraja niza. Izlaz iz mreže ovisi o zajedničkom izlazu dviju pod-mreža koje će svoje izlazne vrijednosti izračunati s obzirom na prošlost i budućnost, a vrijednosti će biti najosjetljivije oko vremena t [10]. Na slici 8.18 prikazan je rezultat modela kod predviđanja vremenskog niza rezervacija.



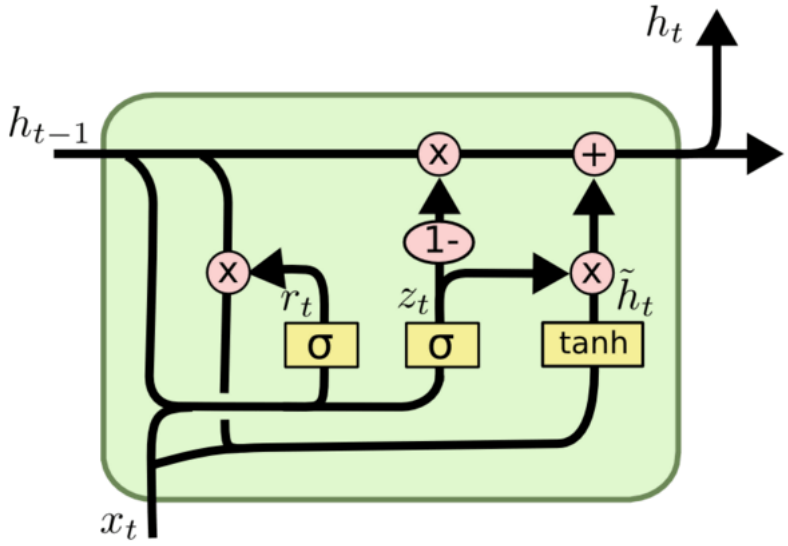
Slika 8.17: Bidirectional LSTM blok



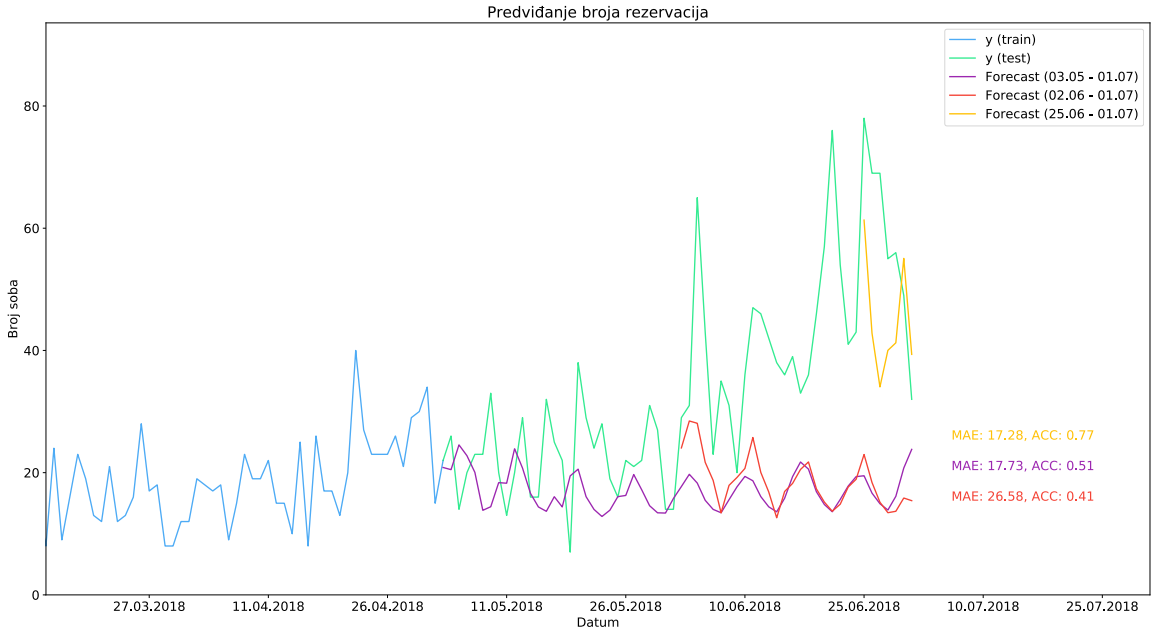
Slika 8.18: Rezultat BidirectionalLSTM modela na vremenskom nizu rezervacija

8.7.3. Gated Recurrent Unit

Mreže propusno povratnih ćelija (engl. Gated Recurrent Unit - GRU) su izvorno predložili K. Cho i ostali [5] kao pojednostavljenu inačicu LSTM-a. Glavna razlika u odnosu na LSTM je u tome što jedna vrata istodobno kontroliraju vrata za zaboravljanje i odluku o ažuriranju stanja ćelije [10]. Unutrašnje stanje GRU ćelije je uvijek vidljivo u izlazu zbog manjka kontrolnih sklopova, za razliku od izlaznih vratiju LSTM-a. Na slici 8.20 prikazan je rezultat modela kod predviđanja vremenskog niza rezervacija.



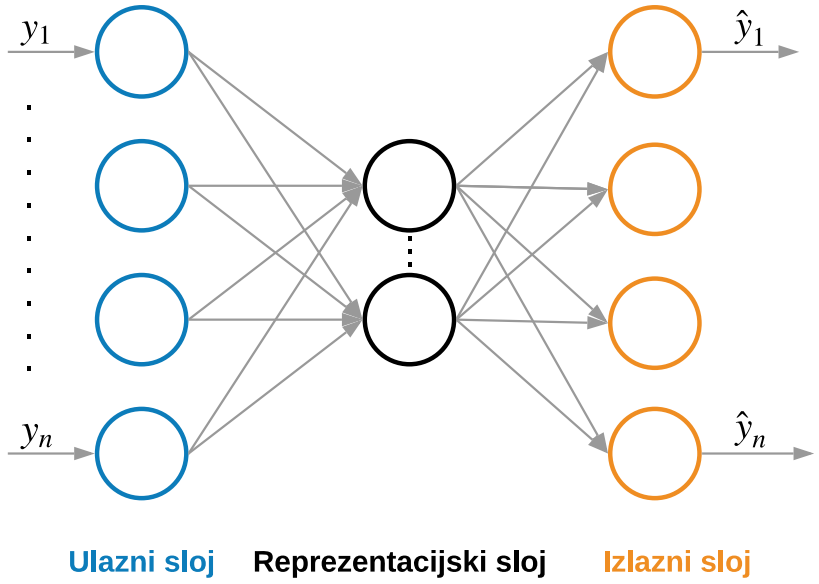
Slika 8.19: GRU blok



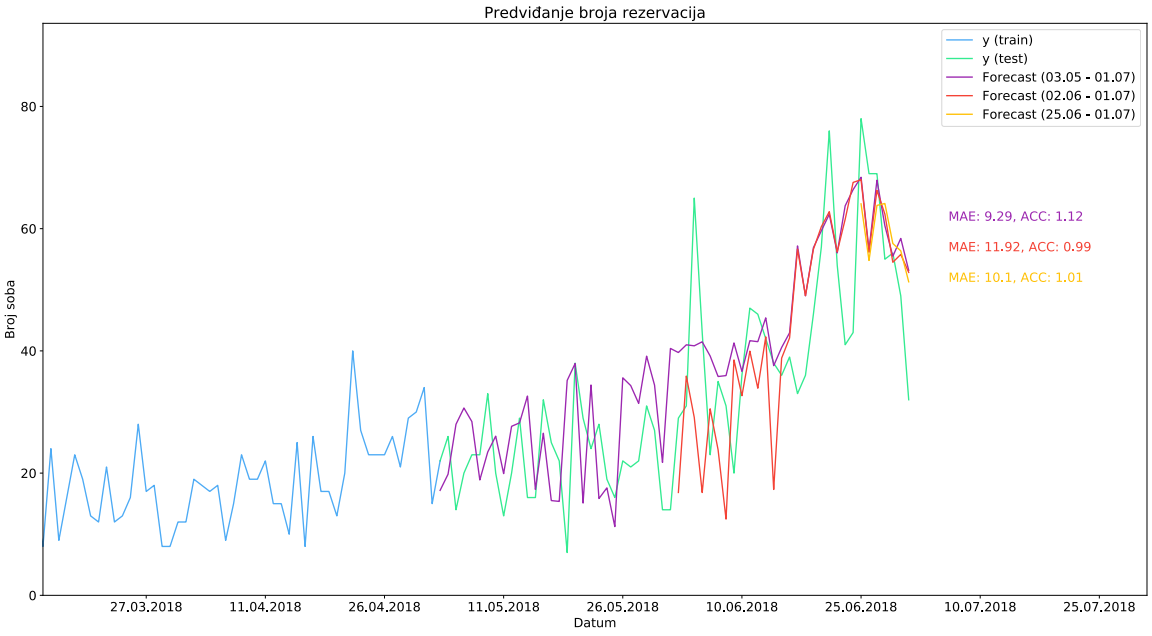
Slika 8.20: Rezultat GRU modela na vremenskom nizu rezervacija

8.8. Autoencoder

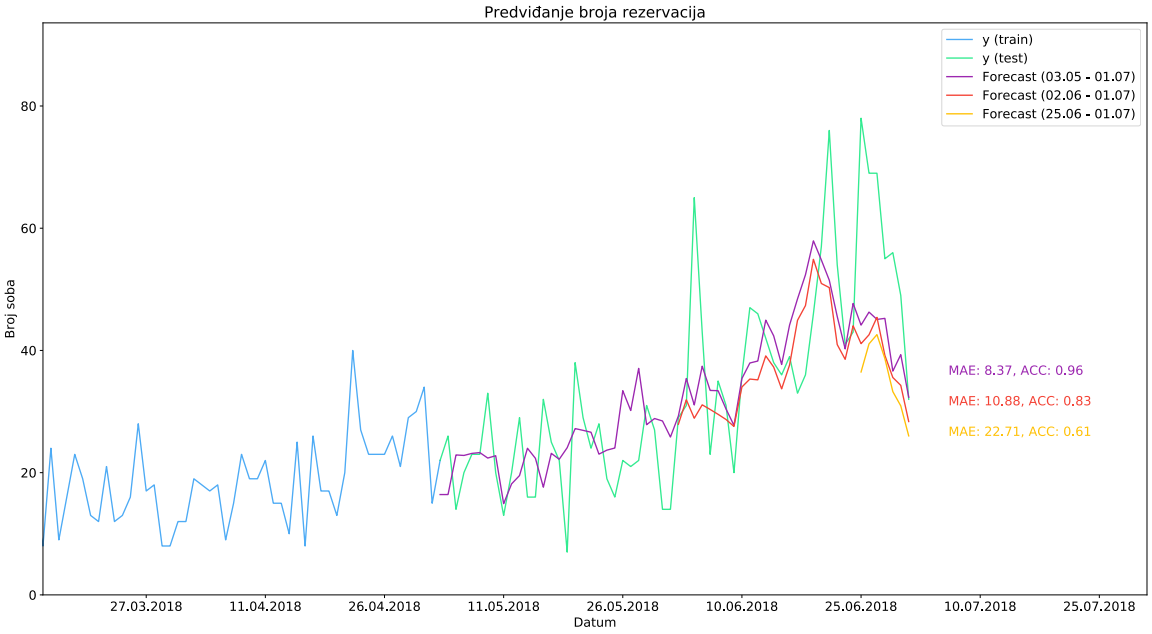
Autoencoder je neuronska mreža koja se trenira da kopira ulaz u izlaz. Mreža se sastoji od dvije komponente: *encoder* koji enkodira podatke $h = f(x)$ i *decoder* koji rekonstruira podatke $r = g(h)$. Da bi se spriječilo *autoencoder* da nauči samo mapirati funkciju $g(f(x)) = x$, ugrađuju se određena ograničenja. Najpoznatiji način izvedbe *autoencoder*-a je mreža sa slojem između dvije komponente (*encoder* i *decoder*) koji ima manje dimenzije nego ulazni podaci. U takvoj arhitekturi mreža je, zbog smanjenja dimenzionalnosti podataka, natjerana izvući najistaknutija obilježja trening podataka. *Autoencoder* sa nelinearnim *encoder*-om f i nelinearnim *decoder*-om g tako može naučiti snažniju nelinearnu generalizaciju PCA [10]. PCA (engl. Principal Component Analysis) je metoda smanjivanja dimenzionalnosti podataka zadržavajući većinu informacija iz originalnog seta podataka. *Autoencoder* je zbog svoje strukture usko vezan sa PCA jer se iz ulaznog podatka izvlače najistaknutija obilježja nakon čega se ulazni podaci rekonstruiraju na izlazu [8]. *Autoencoder*-i jako dobro rade kod otkrivanja anomalija u podacima, ali su se pokazali dobri i kod rada sa vremenskim nizovima [28, 19].



Slika 8.21: *Autoencoder* arhitektura neuronske mreže



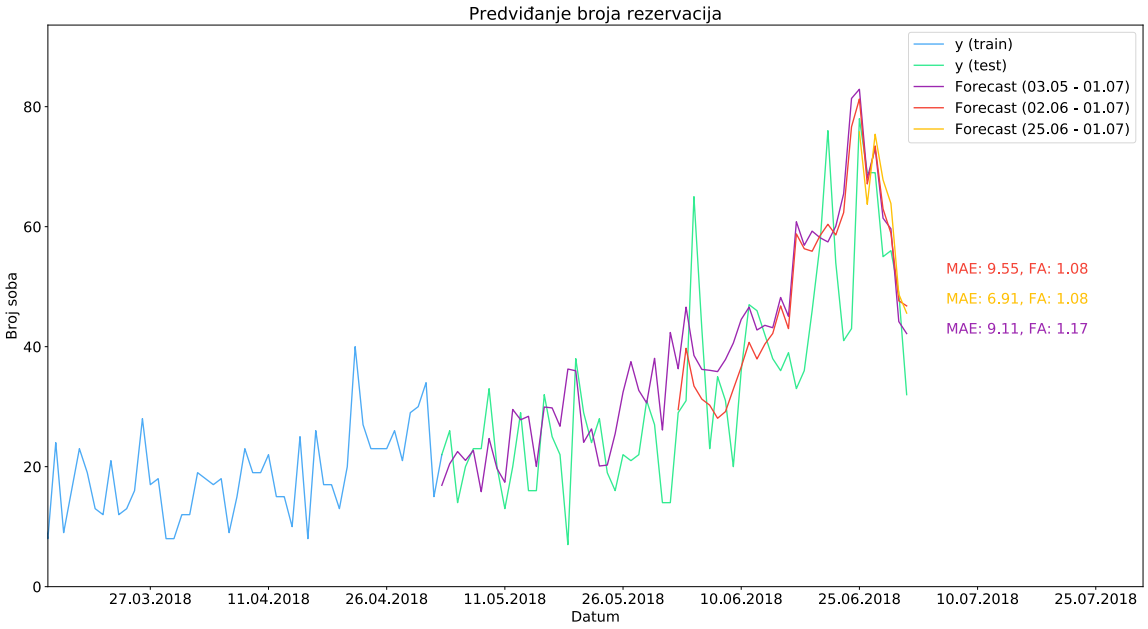
Slika 8.22: Rezultat AutoencoderMLP modela na vremenskom nizu rezervacija



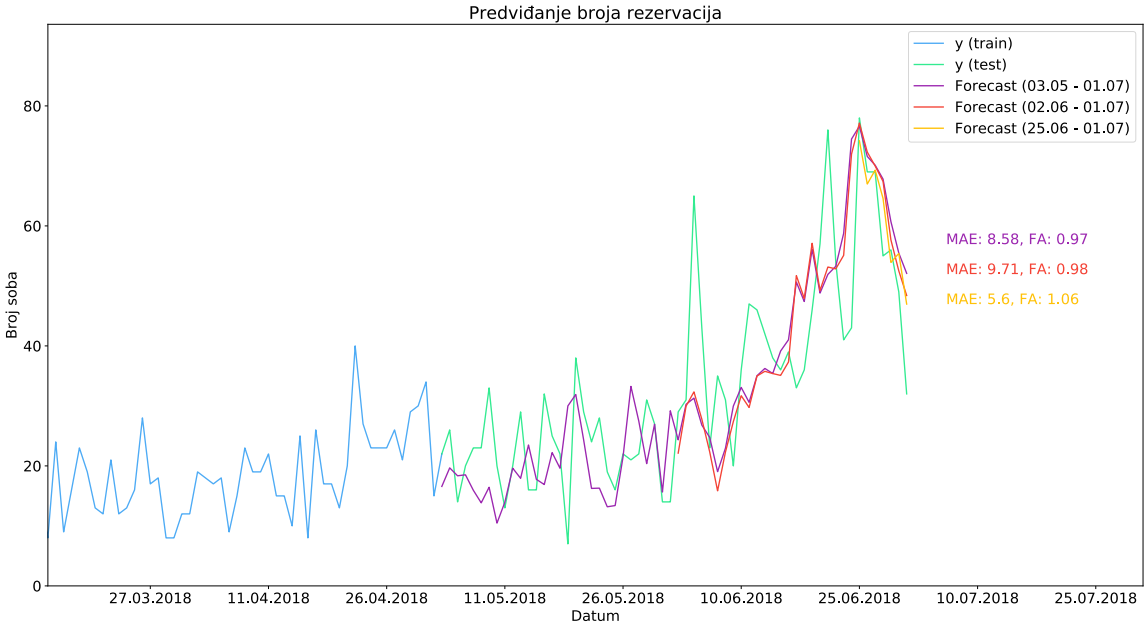
Slika 8.23: Rezultat AutoencoderMultiCNN modela na vremenskom nizu rezervacija

8.8.1. Autoencoder i RNN

RNN mreže nisu pokazale dobre rezultate na primjeru vremenskog niza rezervacija. Razlog tome možda leži u činjenici da vremenski niz konstantno iskušava manje nagle promjene, koje nisu nužno iste u svakom sezonskom razdoblju. Dalo bi se zapitati može li RNN mreža postići bolje rezultate u kombinaciji sa nekom drugom arhitekturom. Tako je osmišljena arhitektura mreže gdje će kao osnova biti model koji je ostvario najbolje rezultate, a to je AutoencoderMLP. Ideja je dodati RNN sloj u *autoencoder* arhitekturu odmah nakon sloja sa najmanjim dimenzijama. RNN sloj će u tom slučaju pratiti kretanje glavnih značajki vremenskog niza koje je sloj sa najmanjim dimenzijama prepoznao i biti će istih dimenzija kao prethodni sloj. U tu svrhu su izrađena dva modela: AutoencoderMLPLSTM sa LSTM slojem i AutoencoderMLPGRU sa GRU slojem. Rezultati modela su prikazani na slikama 8.24 i 8.25.



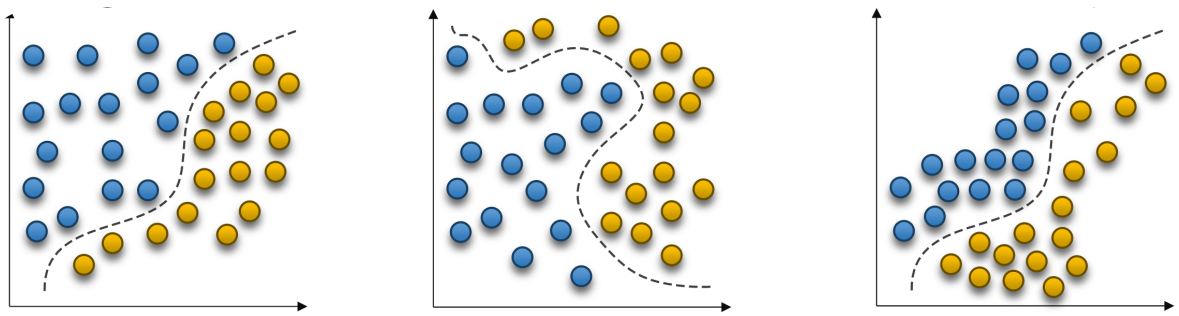
Slika 8.24: Rezultat AutoencoderMLPLSTM modela na vremenskom nizu rezervacija



Slika 8.25: Rezultat AutoencoderMLPGRU modela na vremenskom nizu rezervacija

9. Concept drift

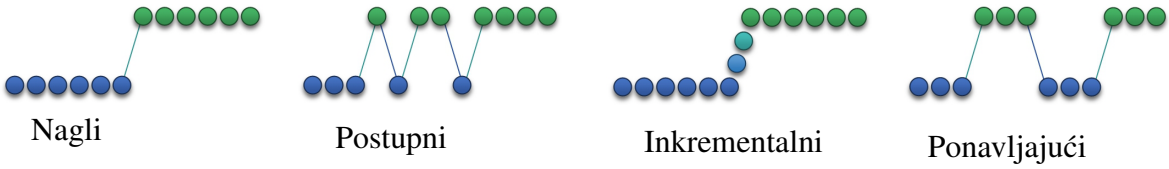
U prethodnim poglavljima bili su prikazani razni modeli vremenskih nizova. Nakon odabira najboljeg modela istog se može upotrijebiti u produkcijskom okruženju. Model će tada raditi sa novo dostupnim podacima koje nije nikad vidio, a nije ih vidjela niti osoba koja je model dizajnirala. Tu može doći do problema zbog konstantne i brzo mijenjajuće okoline u kojoj živimo. Može se pretpostaviti da podaci evoluiraju kroz vrijeme i moraju biti analizirani gotovo u stvarnom vremenu. Model ubrzo zastari ako ne primijeti novonastale promjene i/ili se ne prilagodi novoj okolini. Taj fenomen je poznat kao pomak koncepta (engl. concept drift). Najučestaliji primjer važnosti prepoznavanja *concept drift*-a je filtriranje e-pošte, odnosno prepoznavanje neželjene pošte. Pošiljatelji neželjene pošte konstantno traže načine kako će nadmudriti filtere e-pošte, a filteri se moraju konstantno prilagođavati novim načinima prijevare. Kao što je već spomenuto, tradicionalno nadzirano učenje pretpostavlja da će distribucija ulaznih podataka biti ista distribuciji podataka u produkcijskom okruženju. U takvom okruženju se predviđanja najčešće odvijaju u stvarnom vremenu gdje se može očekivati da će se distribucija ulaznih podataka promijeniti kroz vrijeme. Promjena u distribuciji ulaznih podataka izaziva virtualni pomak koncepta (engl. virtual concept drift), dok ostale promjene izazivaju stvarni pomak koncepta (engl. real concept drift) kao što je prikazano na slici 9.1.



Slika 9.1: Realni i virtualni *concept drift* (preuzeto iz [21])

Concept drift se može javljati u različitim oblicima kao što je prikazano na slici 9.2. Nagli *concept drift* je posljedica nagle promjene u distribuciji podataka. Postupni *concept drift* je

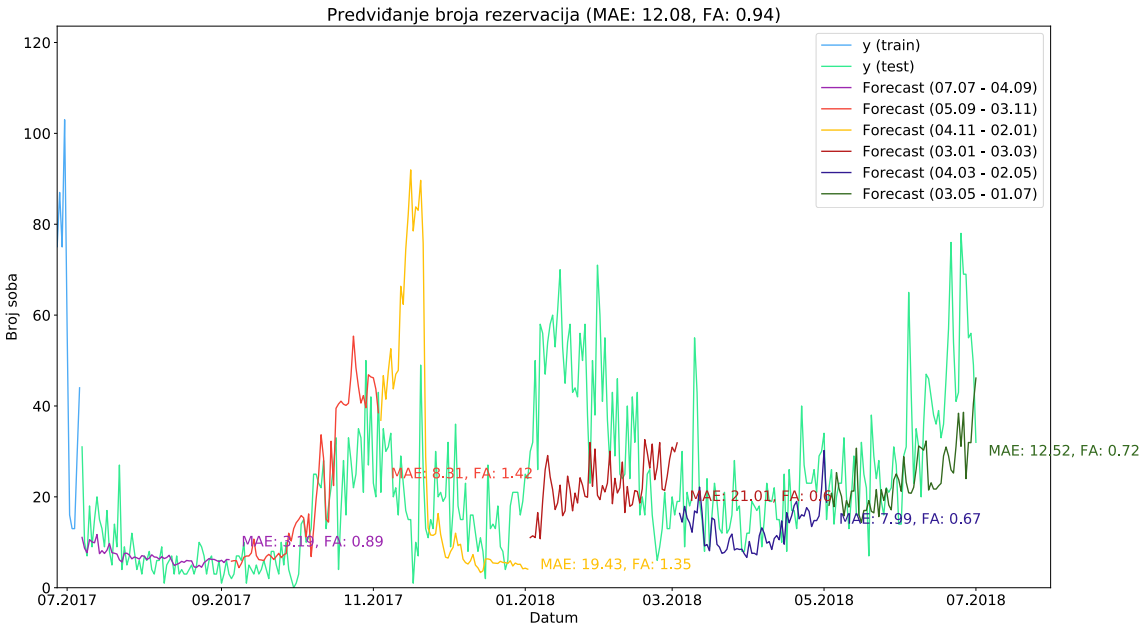
posljedica spore tranzicije podataka u novu distribuciju. Inkrementalni *concept drift* je posljedica pojavljivanja više različitih distribucija tijekom tranzicije. U ponavljajućem *concept drift*-u se nakon nekog vremena ponovno pojavljuje prethodni koncept.



Slika 9.2: Oblici *concept drift*-a (preuzeto iz [21])

Uglavnom nije bitno koji se pomak i na koji način se odvija, jer u svakom slučaju treba ažurirati model ako se želi održati određena razina točnosti. Jedan od najvećih problema rješavanja *concept drift*-a je prepoznavanje razlike između šuma u podacima i pravog pomaka, pa tako postoje različite metode za rješavanje *concept drift*-a, one se mogu grupirati u dvije glavne strategije. Prva strategija je kontinuirano ažuriranje modela kroz određena vremenska razdoblja, gdje se ne prate promjene u podacima već se model periodično ažurira. Druga strategija je konstantno praćenje ulaznih i izlaznih vrijednosti, uglavnom korištenjem statističkih testova. U slučaju da se otkrije promjena u podacima javlja se obavijest o pomaku, te je tada potrebno poduzeti mjere adaptacije modela. Premda većina radova koji spominju *concept drift* opisuje samo metode koje prate klasifikacijske modele, postoji i nekoliko načina praćenja modela vremenskih nizova. Neki od njih, odabrani za ovaj rad, su: Kruskal–Wallis test za ispitivanje potječu li uzorci iz iste distribucije, CUSUM metoda za praćenje promjene signala greške i BFAST metoda za praćenje promjena u vremenskom nizu.

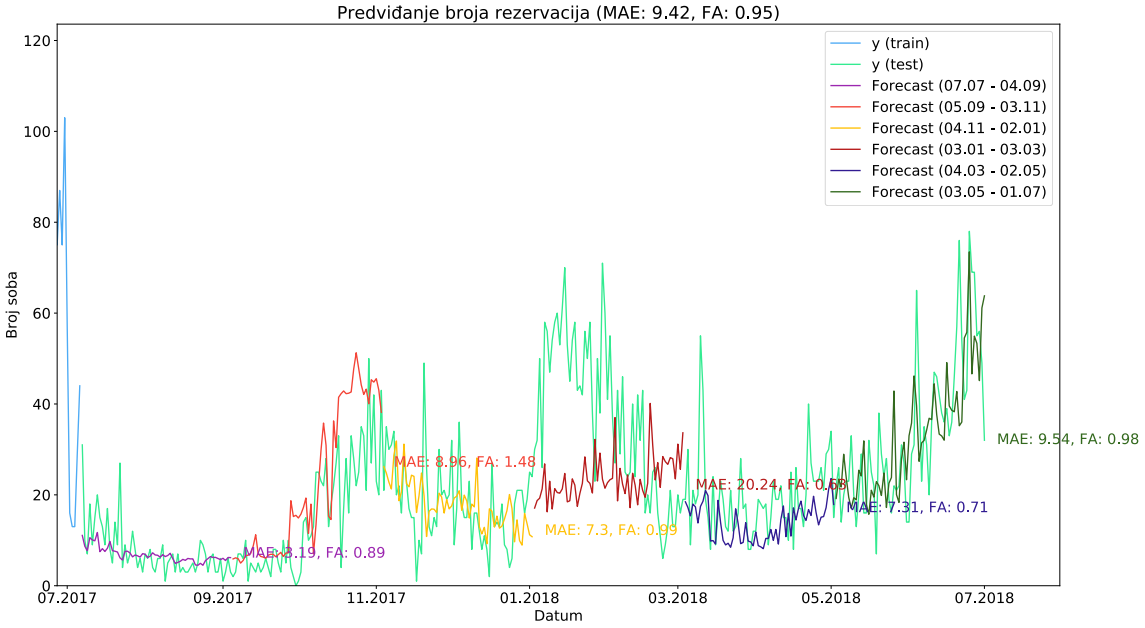
U sljedećim poglavljima će metode biti pojašnjene zajedno sa načinom korištenja na primjeru i prikazom performansi na odabranom modelu. Kako bi se omogućio dovoljan prostor za mjerenje performansi metoda, model će se trenirati na prve dvije godine i predviđati treću godinu po razdobljima od 60 dana. Prije nastavka na metode važno je napomenuti da se u stvarnom svijetu predviđanja očekuju gotovo u realnom vremenu što znači da je cilj smanjiti vrijeme ažuriranja modela, odnosno pokretati ažuriranje modela samo kada je to nužno. Kao osnovnu za uspoređivanje performansi metoda koristit će se adaptivna metoda koja će kod svakog novog dostupnog podatka pokretati ažuriranje modela bez ikakve provjere *concept drift*-a. Da bi se poboljšanje predviđanja mogla jasnije vidjeti odabran je model čije su performanse malo lošije (AutoencoderMLP: verzija = 2), što se vidi i na slici 9.3 gdje su predviđanja napravljena bez ikakvih metoda detekcija *concept drift*-a.



Slika 9.3: Rezultat modela nad vremenskim nizom rezervacija bez metode otkrivanja *concept drift*-a

9.0.1. Adaptivna metoda

Adaptivna metoda ne prati ulazne niti izlazne podatke, već ažurira model svaki put kada dobije novi podatak. Može se pretpostaviti da će ovakav model imati najbolje performanse jer se konstantno usklađuje sa novim podacima. Međutim mora se napomenuti da će za svako predviđanje biti potrebno određeno sačekati određeno vrijeme dok se model ažurira. Na slici 9.4 može se vidjeti da se model prilagodio novim podacima i ostvario bolji rezultat nego model bez metode otkrivanja *concept drift*-a.



Slika 9.4: Rezultat modela nad vremenskim nizom rezervacija sa adaptivnom metodom otkrivanja *concept drift*-a

9.0.2. Kruskal-Wallis test

Kruskal-Wallis test (nazvan po William Kruskal i W. Allen Wallis [17]) je neparametarska metoda za ispitivanje potječu li uzorci iz iste distribucije. Test se koristi za usporedbu dvaju ili više neovisnih uzoraka jednakih ili različitih veličina. Kako je metoda neparametarska, ne podrazumijeva normalnu distribuciju uzorka. Metoda radi na principu rangiranja vrijednosti iz svih grupa i dana je formulom:

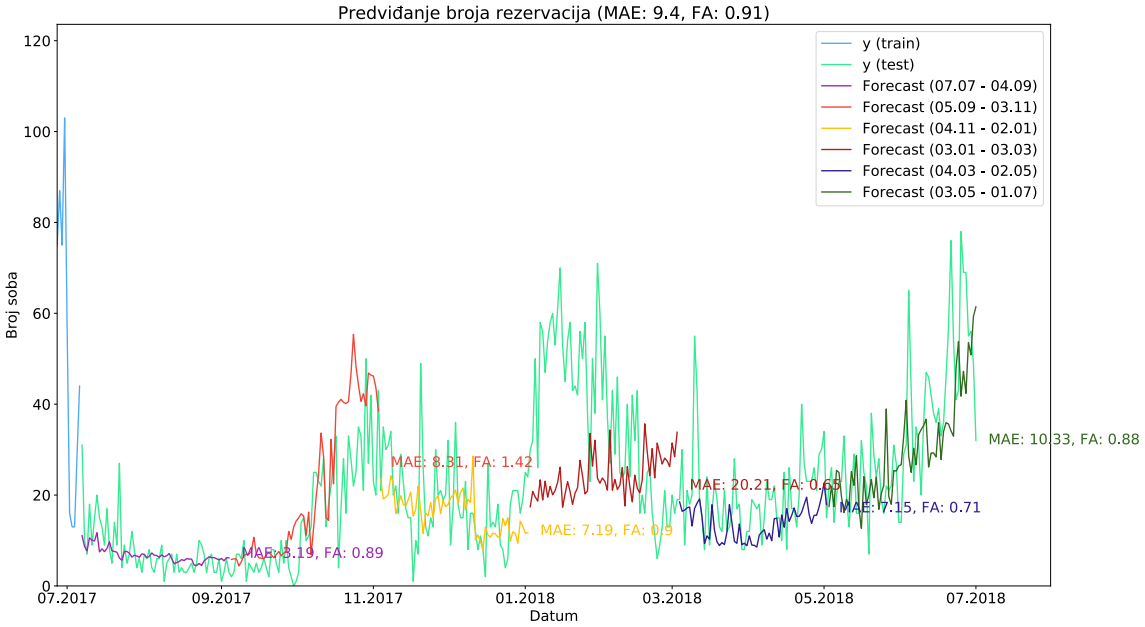
$$H = \frac{12}{N(N+1)} \sum_{i=1}^{g} n_i \bar{r}_{i\cdot}^2 - \ 3(N+1)$$

gdje je n_i broj opservacija grupe i, r_{ij} rang (između svih vrijednosti) opservacije j iz grupe i, N sveukupni broj opservacija u svim grupama, $\bar{r}_{i\cdot}$ = prosječni rang svih opservacija u grupi i i $\bar{r}$ je prosjek svih r_{ij}. U sljedećim poglavljima će metoda biti primijenjena na ulazne i izlaze podatke.

Praćenje ulaza

U primjeru vremenskog niza rezervacija praćenje ulaznih vrijednosti odnosi se na stvaran broj rezerviranih soba. Kod usporedbe uzoraka promatrat će se svi podaci koji su nužni za predviđanje, odnosno ako model zahtjeva n koraka, onda će se kao prvi uzorak uzeti podaci $y_{t-n}, \ldots, y_t$. Kao drugi uzorak koristit će se isto vremensko razdoblje prošle godine, odnosno $y_{t-n-365}, \ldots, y_{t-365}$. Na primjeru vremenskog niza rezervacija metoda u prvom i

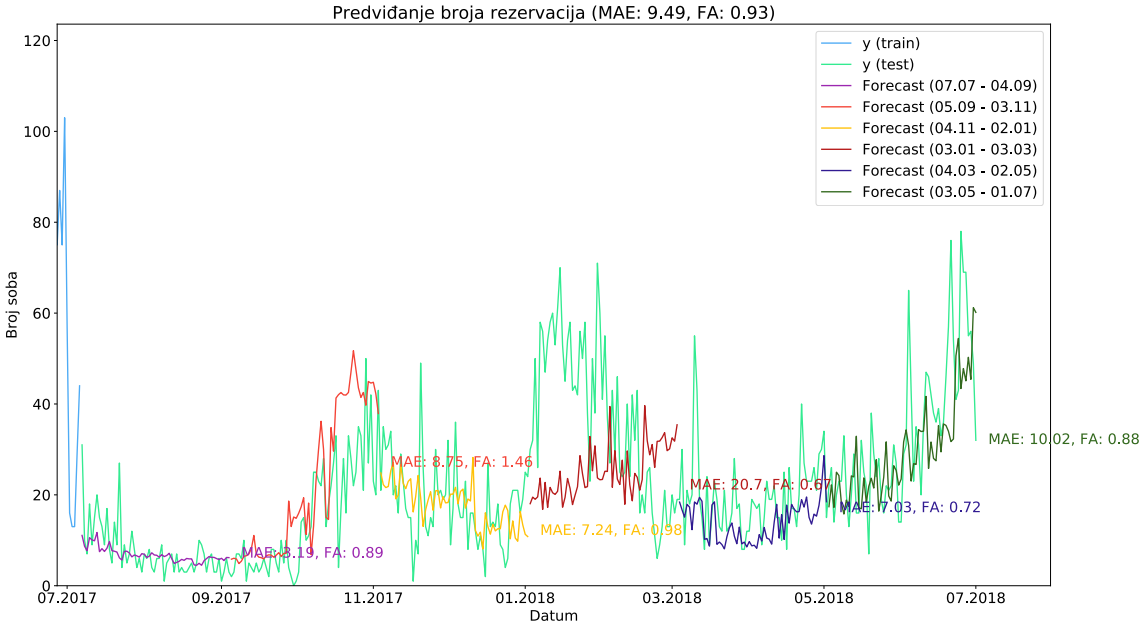
pretposljednjem razdoblju nije ukazala na postojanje značajne razlike novih podataka u odnosu na prošlogodišnje podatke, dok je za ostala tri središnja razdoblja ukazala na postojanje pomaka u podacima, pa je model tada ažuriran. Rezultati metode su prikazani na slici 9.5.



Slika 9.5: Rezultat modela nad vremenskim nizom rezervacija sa Kruskal-Wallis testom na ulaznim vrijednostima za otkrivanje *concept drift*-a

Praćenje izlaza

U primjeru vremenskog niza rezervacija, praćenje izlaznih vrijednosti se odnosi na vrijednosti predviđanja modela. Kod usporedbe uzoraka promatrat će se predviđanja modela i stvarne vrijednosti u istom razdoblju prošle godine, odnosno predviđanja $y'_t, \ldots, y'_{t+r}$ i vrijednosti prošle godine $y_{-365}, \ldots, y_{-365+r}$ gdje je r razdoblje predviđanja. Ovaj pristup želi provjeriti da uzorak predviđenih vrijednosti i uzorak stvarnih vrijednosti prošle godine u istom vremenskom razdoblju potječu iz iste distribucije. Ako uzorci ne potječu iz iste distribucije, onda se model treba ažurirati kako bi se prilagodio novim ulaznim podacima. Metoda je za sva razdoblja osim posljednjeg ukazala da postoji značajna razlika između vrijednosti prošle godine i predviđenih vrijednosti. Rezultati će onda biti slični onima kao kod adaptivnog modela, a prikazani su na slici 9.6.



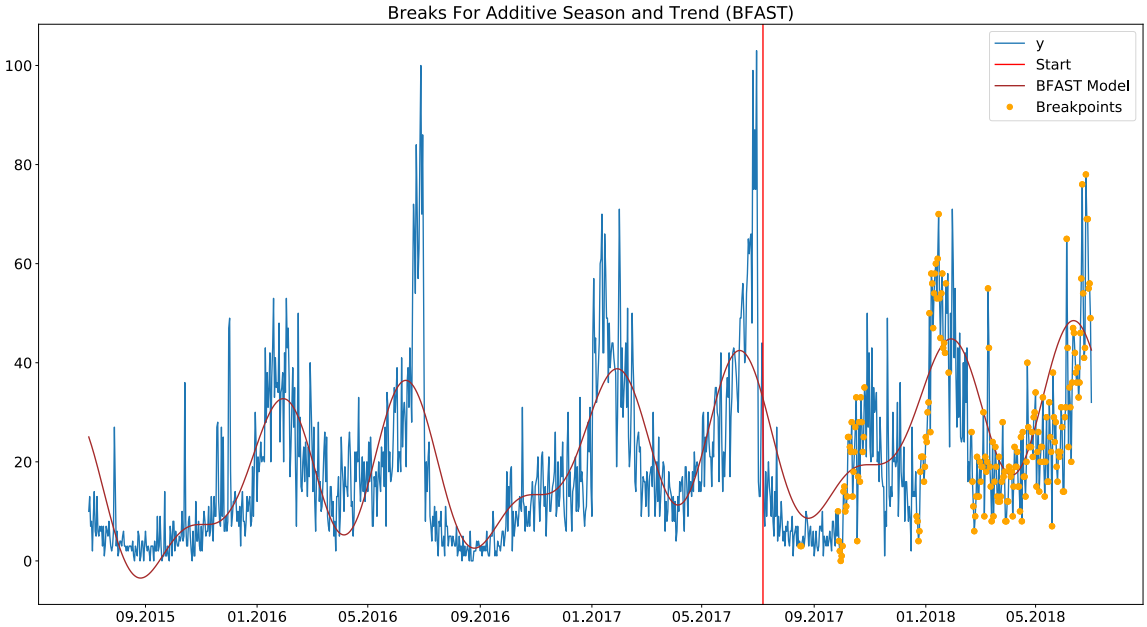
Slika 9.6: Rezultat modela nad vremenskim nizom rezervacija sa Kruskal-Wallis testom na izlaznim vrijednostima za otkrivanje *concept drift*-a

9.0.3. BFAST

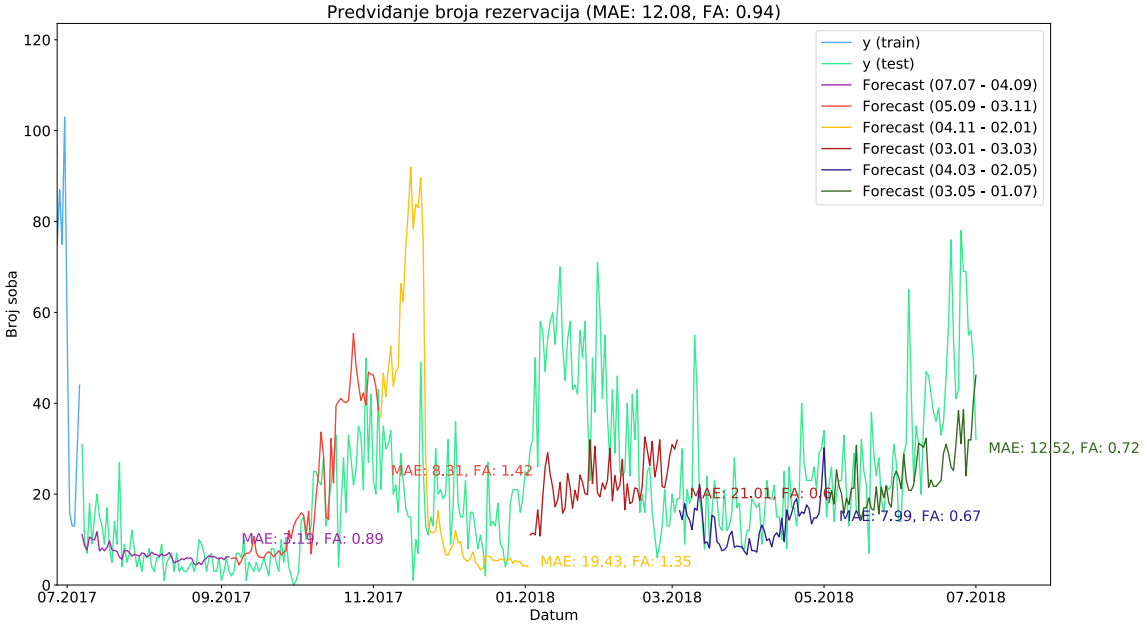
BFAST (engl. Breaks For Additive Season and Trend) su izvorno predložili J. Verbesselt i ostali [27] kao metodu prepoznavanja promjena u vremenskim nizu. BFAST radi na principu dekompozicije vremenskog niza u komponentu trenda, komponentu sezonalnosti i komponentu preostalog, kao što je opisano u poglavlju 7.4, sa metodama za otkrivanje i karakterizaciju promjene unutar vremenskog niza. Algoritam iterativno kreira linearni trend i sezonalni model, zatim traži eventualne trendove i sezonske promjene poznate kao prekidne točke (engl. breakpoints). BFAST će se za primjer vremenskog niza rezervacija koristi za prepoznavanja promjena u novim ulaznim podacima. Postojanje promjene će se promatrati u trenutku kada model dobije stvarne vrijednosti rezervacija i to tako da će BFAST na temelju prethodnih podataka izraditi model i analizirati postojanje promjene u novim podacima. Na slici 9.7 je metoda primijenjena na cijeli vremenski niz rezervacija gdje se promatra postojanje promjena u posljednjoj godini u odnosu na prethodne dvije godine. Iz prikaza se može vidjeti da je BFAST detektirao poveći broj promjena (breakpoints) u posljednjoj godini.

Primjenom BFAST-a kao metode otkrivanja *concept drift*-a dobiveni su rezultati prikazani na slici 9.8. BFAST metoda nije u niti jednom razdoblju primijetila pomak u vremenskom nizu pa će rezultat biti isti rezultatu bez metode otkrivanja *concept drift*-a. Moguće objašnjenje zašto metoda nije primijetila nikakve promjene može biti u smanjenom okviru, odnosno vremenskom razdoblju, na kojem se provjera izvršila. Na slici 9.7 je vremensko razdoblje

provjere bilo 365 dana gdje je BFAST primijetio promjene u vremenskom nizu, dok je kroz ovaj primjer svako razdoblje provjere bilo samo 60 dana.



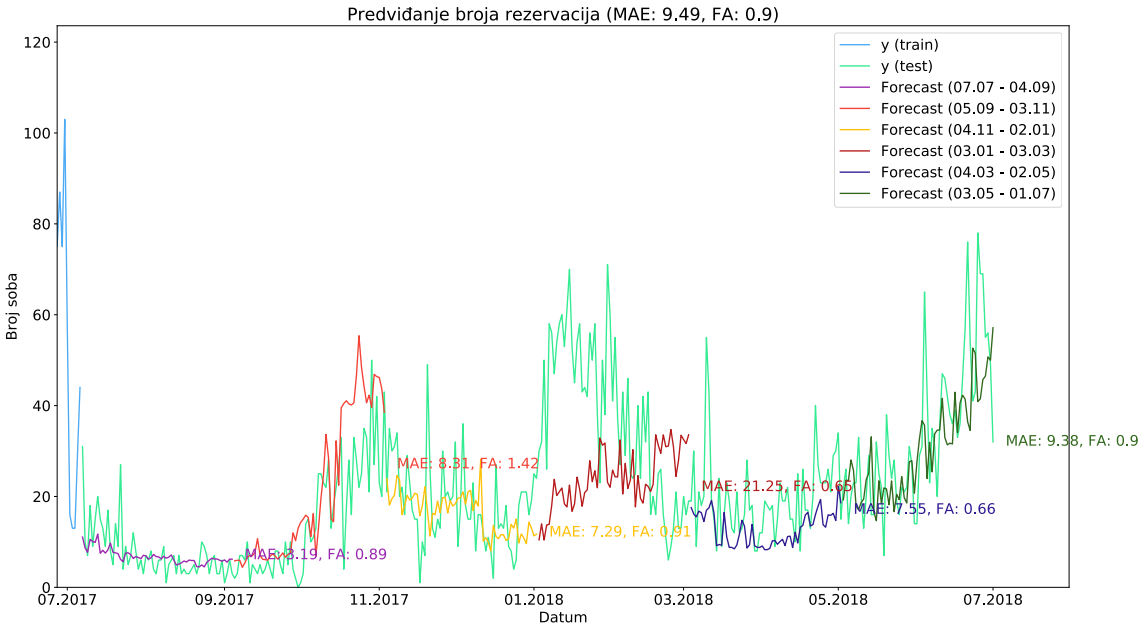
Slika 9.7: BFAST metoda za otkrivanje pomaka u vremenskom nizu



Slika 9.8: Rezultat modela nad vremenskim nizom rezervacija sa BFAST metodom za otkrivanje *concept drift*-a

9.0.4. CUSUM

CUSUM, skraćeno od kumulativna suma (engl. cumulative sum), je metoda otkrivanje promjena u signalu. Postoji više različitih metoda koje se temelje na CUSUM-u. Ona koja će se dalje promatrati uključuje računanje kumulativne sume pozitivnih i negativnih promjena u podacima i usporedbu sa pragom (engl. threshold). Kada se prag prekorači otkriva se promjena i kumulativna suma se poništi na 0. Da bi se izbjeglo otkrivanje promjene u nedostatku stvarne promjene ili sporog pomaka, ovaj algoritam ovisi i o parametru pomak (engl. drift) za korekciju pomaka. U primjeru vremenskog niza rezervacija CUSUM će se koristiti za praćenje promjene greške modela. Greške će se pratiti tako da će model, kada dobije nove stvarne vrijednosti broja rezervacija, napraviti predviđanje tih istih vrijednosti i vidjeti kolika je bila greška u odnosu na stvarne vrijednosti. Nakon toga će se greške predviđanja koristiti u CUSUM algoritmu za otkrivanje promjena. Ako algoritam primijeti promjene u greškama, pokrenuti će se ažuriranje modela. Na primjeru vremenskog niza rezervacija metoda nije ukazala na postojanje pomaka u prvom i trećem razdoblju, odnosno ukazala je na postojanje pomaka u drugom, četvrtom i petom razdoblju. Na slici 9.9 prikazani su rezultati koje je metoda ostvarila.



Slika 9.9: Rezultat modela nad vremenskim nizom rezervacija sa CUSUM metodom za otkrivanje *concept drift*-a

10. Rezultati

10.1. Modeli vremenskih nizova

Kao osnova za evaluaciju ostalih modela izrađeni su dva naivna modela čiji su rezultati prikazani u tablici A.1. Početni naivni model nije ostvario dobar rezultat kod predviđanja rezervacija sa prosječnom greškom $MAE = 21.29$ i prosječnom točnosti $FA = 0.57$. Značajno bolji rezultat je ostvario sezonalni naivni model koji je imao prosječnu grešku $MAE = 13.70$ i prosječnu točnost $FA = 1.21$. Sezonalni naivni model je za sva razdoblja predviđanja u prosjeku predvidio oko 21% više rezervacija, što ga čini prihvatljivim osnovnim modelom kojeg će ostali modeli morati nadmašiti. Kod naivnih modela ne postoji faza treniranja pa je tako trajanje treniranja instantno.

Gotovo svi statistički modeli nisu uspjeli ostvariti bolje rezultate nego osnovni model, što se može vidjeti u tablici A.2. Modeli AR, MA, ARIMA i SARIMA sa najmanjom prosječnom greškom $MAE = 19.12$ i najvećom prosječnom točnosti $FA = 0.63$ nisu uspjeli ostvariti bolje rezultate nego osnovni model. Prvi model koji je ostvario bolje rezultate od osnovnog modela je Prophet, sa prosječnom greškom $MAE = 9.38$ i prosječnom točnosti $FA = 1.09$. Trajanje treniranja ovih modela je u okviru nekoliko minuta, osim u slučaju SARIMA modela kojem je potrebno oko sat vremena.

Modeli neuronskih mreža su testirani u 12 varijanti kako je prikazano u tablici A.3. Pojedini modeli su najbolje rezultate ostvarili na pojedinim varijantama, te prema rezultatima nije moguće generalno odabrati najbolju varijantu. Prvi model je MLP, koji je najbolji rezultat ostvario na varijanti 8, postižući prosječnu grešku $MAE = 10.99$ i prosječnu točnost $FA = 0.90$. Trajanje treniranja MLP modela je u okviru nekoliko minuta, a rezultat modela ne odstupa značajno od Prophet modela.

CNN model je ostvario najbolje rezultate na varijanti 3, postižući prosječnu grešku $MAE = 9.55$ i prosječnu točnost $FA = 0.94$. Trajanje treniranja CNN modela je u okviru nekoliko minuta, a rezultat modela je u razini sa Prophet modelom i neznatno bolji od MLP mo-

dela. Višeslojni CNN (MultiCNN) nije ostvario bolje rezultate od CNN modela, postižući prosječnu grešku $MAE = 12.80$ i prosječnu točnost $FA = 0.84$ na varijanti 3.

RNN modeli nisu opravdali svoja obećanja specijalizacije u obradi sekvencijalnih nizova. LSTM model je postigao najbolji rezultat na varijanti 6 sa prosječnom greškom $MAE = 12.59$ i prosječnu točnost $FA = 0.88$. GRU model je najbolji rezultat ostvario na varijanti 11 sa prosječnom greškom $MAE = 13.80$ i prosječnom točnosti $FA = 0.99$. BidirectionalLSTM model je najbolji rezultat ostvario na varijanti 4 sa prosječnom greškom $MAE = 14.17$ i prosječnom točnosti $FA = 1.00$. Trajanje treniranja pojedinog RNN modela je u rasponu od tri do dvanaest sati, što im je velik nedostatak u odnosu na ostale modele. Nadalje, niti u jednom slučaju nije opravdano dugo trajanje faze treniranja RNN modela, jer je njihov konačni rezultat lošiji od jednostavnijih modela kao što su Prophet i CNN.

AutoencoderMLP model je ostvario najbolje rezultate na varijanti 12, postižući prosječnu grešku $MAE = 9.39$ i prosječnu točnost $FA = 1.08$, dok je AutoencoderMultiCNN model ostvario najbolje rezultate na varijanti 3, postižući prosječnu grešku $MAE = 11.63$ i prosječnu točnost $FA = 0.92$. *Autoencoder* modeli su ostvarili bolji rezultat nego modeli iste arhitekture u ne-*autoencoder* varijanti, gdje je AutoencoderMLP postigao rezultat u razini Prophet i CNN modela.

Modeli izrađeni kombinacijom MLP-a u *autoencoder* arhitekturi i RNN sloja nakon sloja sa najmanjim dimenzijama postigli su uvjerljivo najbolje rezultate. Model AutoencoderMLPLSTM je tako najbolji rezultat ostvario na varijanti 12, postižući prosječnu grešku $MAE = 8.52$ i prosječnu točnost $FA = 1.11$. Model AutoencoderMLPGRU je postigao najbolje rezultate od svih prethodno navedenih modela, postižući prosječnu grešku $MAE = 7.96$ i prosječnu točnost $FA = 1.00$.

10.2. Concept drift

Kao osnovna za evaluaciju metoda za otkrivanje *concept drift*-a uzeti će se rezultat predviđanja modela bez ikakve metode. Model je tako ostvario prosječnu grešku $MAE = 12.08$ i prosječnu točnost $FA = 0.94$. Adaptivna metoda je kod svakog razdoblja ažurirala model i time značajno poboljšala rezultat modela smanjujući prosječnu grešku na $MAE = 9.42$ i prosječnu točnost $FA = 0.95$. Metoda sa Kruskal-Wallis testom na ulaznim vrijednostima je otkrila *concept drift* u tri od pet razdoblja, pa je tako model postigao prosječnu grešku $MAE = 9.40$ i prosječnu točnost $FA = 0.91$, što je u razini sa adaptivnom metodom. Metoda sa Kruskal-Wallis testom na izlaznim vrijednostima je otkrila *concept drift* u četiri od

pet razdoblja, pa je tako model postigao prosječnu grešku $MAE = 9.49$ i prosječnu točnost $FA = 0.93$. BFAST metoda nije u niti jednom razdoblju otkrila postojanje *concept drift*-a, pa će njen rezultat biti isti osnovnom rezultatu. CUSUM metoda je otkrila postojanje *concept drift*-a u tri od pet razdoblja, pa je tako model postigao prosječnu grešku $MAE = 9.49$ i prosječnu točnost $FA = 0.90$.

Iz opisa principa rada metoda dalo bi se naslutiti da će adaptivna metoda imati značajno bolje rezultate. Tvrdnja bi mogla biti točna ako se promatra BFAST metoda, koja nije ostvarila poboljšanje u odnosu na osnovni rezultat. Međutim ostale metode, metode primjenom Kruskal-Wallis testa na ulaznim ili izlaznim vrijednostima i CUSUM metoda, postižu rezultate u razini adaptivne metode uz manji broj otkrivanja *concept drift*-a, samim time i ažuriranja modela.

11. Zaključak

U radu su prikazani različiti modeli vremenskih nizova i njihov rezultat na primjeru vremenskog niza rezervacija. Izrada modela bez prethodnog upoznavanja sa osnovnim pojmovima koji se vežu za vremenske nizove može rezultirati ne-optimalnim modelom ili modelom koji će davati kriva predviđanja. Razumijevanje pojmova kao što su sezonalnost, trend i cikličnost su nužni za izradu dobrog modela. Prilikom predviđanja i evaluacije performansi modela koriste se neke od standardnih mjera kao što je MAE, ali u obzir treba uzeti i način na koji su predviđanja izvršena. Predviđanje jednokoračnom metodom će gotovo uvijek dati bolja rješenja nego višekoračna metoda, prema tome je kod predviđanja i prikaza performansi modela nužno napomenuti o kakvoj se metodi predviđanja radi, što mnogi radovi ne navode.

Prije početka analize različitih modela dobra je praksa uspostaviti jednostavni bazni (naivni) model sa kojim će se ostali modeli uspoređivati. Ponekad nije moguće postići bolji rezultat od predviđanja da će sutrašnja vrijednosti biti jednaka današnjoj vrijednosti, kao što je to slučaj kod predviđanja nekih cijena dionica. Tradicionalni pristup za modeliranje vremenskih nizova uključuju korištenje statističkih modela kao što su ARIMA, SARIMA, Prophet i sl. Oni su uglavnom dobra polazna točka jer su brzi za treniranje i relativno je lako doći do objašnjenja kako je došlo do određenog predviđanja, ali je za kreiranje modela potrebna određena razina stručnosti. Za modeliranje vremenskih nizova mogu se koristiti i neuronske mreže. Savladavanje neuronskih mreža može zahtijevati određeno vrijeme, no ako je osoba upoznata sa njima može vrlo jednostavno formulirati vremenski niz za model neuronske mreže. Postoje različite arhitektura neuronskih mreža koje se mogu koristiti kod vremenskih nizova kao što su MLP, LSTM, GRU, CNN i sl. Svaka od tih arhitektura je drugačija i imat će drugačije performanse, no potrebno je naglasiti da je formulacija problema uvijek ista. Neuronske mreže uglavnom postižu dobre rezultate kod rada sa vremenskim nizovima, u mnogo slučajeva bolje nego statistički modeli. Gledajući rezultate različitih modela kroz ovaj rad može se zaključiti da su neuronske mreže i u ovom slučaju ostvarile bolji rezultat. Neuronske mreže imaju i svoje mane, kao što su: duže trajanje treniranja i otežana mogućnost objašnjavanja kako je došlo do određenog predviđanja.

Nakon analize modela, izabrat će se onaj koji pruža najbolje performanse uz određene kriterije jednostavnosti i brzine modela. Jednom kada se model postavi u produkcijsko okruženje, radit će sa novim podacima koje još nije "vidio". Nije realno pretpostaviti da će okolina ostati uvijek ista, već se može pretpostaviti da će se okolina kontinuirano mijenjati, što model mora pratiti ili će zastarjeti. Postoje različite tehnike rješavanja tog problema, poznatog kao *concept drift*, od kojih su neke prikazane u radu.

Modeli prikazani kroz rad su koristili podatke rezervacija samo za 1.7., no što ako se želi predvidjeti kretanje rezervacija za 2.7. ili 15.7.? U budućem radu se treba istražiti kako naučiti model generalizirati, tako da može dobro raditi za sve dane. Također je potrebno istražiti kako u model dodati dodatne informacije kao što je podatak o vremenu (sunčano, kišno, oblačno), podatak o odvijanju nekog događaja (npr. nogometno prvenstvo) i sl. Sljedeće područje koje zahtjeva dublje proučavanje jest objašnjavanje predviđanja koja daje neuronska mreža, odnosno objašnjenje zbog čega je došlo do takvog predviđanja, koja su to ključna obilježja vremenskog niza. Područje budućih istraživanja se može proširiti i na metode otkrivanja *concept drift*-a, jer je većina dosadašnjih istraživanja usmjerena samo na klasifikacijske probleme, a vrlo malo na regresijske probleme.

Svi modeli i potrebni podaci se nalaze u projektu koji se može pronaći na githubu: https://github.com/RomeoSajina/TSExploratory

POPIS TABLICA

POPIS SLIKA

LITERATURA

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, i Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

[2] Roger Achkar, Fady Elias-Sleiman, Hasan Ezzidine, i Nourhane Haidar. Comparison of bpa-mlp and lstm-rnn for stocks prediction. U *2018 6th International Symposium on Computational and Business Intelligence (ISCBI)*, stranice 48–51. IEEE, 2018.

[3] Ratnadip Adhikari i Ramesh K Agrawal. An introductory study on time series modeling and forecasting. *arXiv preprint arXiv:1302.6613*, 2013.

[4] Dunja Božić-Štulić. Semantička segmentacija slika metodama dubokog učenja. Kvalifikacijski ispit za poslijediplomski doktorski studij, Sveučilište u Splitu fakultet elektrotehnike, strojarstva i brodogradnje, dec 2017.

[5] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, i Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[6] John H Cochrane. Time series for macroeconomics and finance. *Manuscript, University of Chicago*, 2005.

[7] Luca Di Persio i Oleksandr Honchar. Artificial neural networks architectures for stock price prediction: Comparisons and applications. *International Journal of Circuits, Systems and Signal Processing*, 10:403–413, 2016.

[8] Konstantinos I Diamantaras, Yu Hen Hu, i Jeng-Neng Hwang. Neural networks and principal component analysis. U *Handbook of Neural Network Signal Processing*, svezak 8. CRC Press, 2002.

[9] Julian Faraway i Chris Chatfield. Time series forecasting with neural networks: a comparative study using the air line data. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 47(2):231–250, 1998.

[10] Ian Goodfellow, Yoshua Bengio, i Aaron Courville. *Deep learning*. MIT press, 2016.

[11] Coşkun Hamzaçebi. Improving artificial neural networks' performance in seasonal time series forecasting. *Information Sciences*, 178(23):4550–4559, 2008.

[12] Sepp Hochreiter i Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[13] Rob J Hyndman i George Athanasopoulos. *Forecasting: principles and practice*. OTexts, 2018.

[14] Sergey Ioffe i Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[15] Douglas M Kline. Methods for multi-step time series forecasting neural networks. U *Neural networks in business forecasting*, stranice 226–250. IGI Global, 2004.

[16] Irena Koprinska, Dengsong Wu, i Zheng Wang. Convolutional neural networks for energy time series forecasting. U *2018 International Joint Conference on Neural Networks (IJCNN)*, stranice 1–8. IEEE, 2018.

[17] William H Kruskal i W Allen Wallis. Use of ranks in one-criterion variance analysis. *Journal of the American statistical Association*, 47(260):583–621, 1952.

[18] Dr ND Lewis. Deep time series forecasting with python: An intuitive introduction to deep learning for applied time series modeling, 2016.

[19] James NK Liu, Yanxing Hu, Jane Jia You, i Pak Wai Chan. Deep neural network based feature representation for weather forecasting. U *Proceedings on the International Conference on Artificial Intelligence (ICAI)*, stranica 1. The Steering Committee of The World Congress in Computer Science, Computer . . . , 2014.

[20] Sigurd Øyen. Forecasting multivariate time series data using neural networks. Magistarski rad, NTNU, 2018.

[21] Ali Pesaranghader, Herna L Viktor, i Eric Paquet. Mcdiarmid drift detection methods for evolving data streams. U *2018 International Joint Conference on Neural Networks (IJCNN)*, stranice 1–9. IEEE, 2018.

[22] Sandhya Samarasinghe. *Neural networks for applied sciences and engineering: from fundamentals to complex pattern recognition*. Auerbach publications, 2016.

[23] Mike Schuster i Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997.

[24] Skipper Seabold i Josef Perktold. Statsmodels: Econometric and statistical modeling with python. U *9th Python in Science Conference*, 2010.

[25] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, i Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[26] Sean J Taylor i Benjamin Letham. Forecasting at scale. *The American Statistician*, 72 (1):37–45, 2018.

[27] Jan Verbesselt, Rob Hyndman, Glenn Newnham, i Darius Culvenor. Detecting trend and seasonal changes in satellite image time series. *Remote sensing of Environment*, 114(1):106–115, 2010.

[28] Lingxue Zhu i Nikolay Laptev. Deep and confident prediction for time series at uber. U *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, stranice 103–110. IEEE, 2017.

Kontinuirano oblikovanje, evaluacija i prilagođavanje modela vremenskih nizova

Sažetak

Predviđanje budućeg ponašanja vremenskog niza je vrlo važno za napredovanje i poboljšanje procesa kojeg vremenski niz predstavlja. Točno predviđanje budućeg ponašanja vremenskog niza omogućava poduzimanje pravovremenih mjera kako bi proces kojeg vremenski niz predstavlja bio što efikasniji. Kod primjera rezervacija hotelskih soba, točno predviđanje budućih rezervacija će omogućiti bolju korekciju cijena ovisno o ponudi soba i potencijalnoj potražnji. Postoje različite metode modeliranja vremenskih nizova i ne može se tvrditi da će određena metoda biti najbolja u svim slučajevima. Ovaj rad pruža uvid u različite metode modeliranja vremenskih nizova sa popratnim primjerima nad vremenskim nizom rezervacija hotelskih soba. Kroz rad su prikazani razni tradicionalni modeli vremenskih nizova, ali i neki moderniji kao što su neuronske mreže, pri čemu su performanse modela evaluirane i međusobno uspoređene. Predviđanje je vrlo zahtjevan zadatak zbog konstantno mijenjajuće okoline što može imati negativne posljedice na model ako ih ne uspije pratiti. Model mora kontinuirano pratiti nove podatke i ispitivati postoji li u njima pomak, poznato kao *concept drift*, kada mora poduzeti određene mjere kako bi se prilagodio novim podacima. Tako su kroz ovaj rad prikazane neke od metoda otkrivanja *concept drift*-a, koje su evaluirane i međusobno uspoređene.

Ključne riječi: Vremenski niz, statistički modeli, neuronske mreže, pomak koncepta, predviđanje rezervacija hotelskih soba

Continuous building, monitoring and adjusting time series forecast models

Abstract

Forecasting of the time series future behavior is very important for advancing and improving their generating process. Accurate forecasting of the time series future behavior enables timely actions to make their generating process as efficient as possible. In an example of booking hotel rooms, an accurate forecast of future bookings will allow better price correction depending on the room availability and potential demand. There are different methods of modeling time series and it cannot be argued that certain method will be the best in all cases. This paper provides an insight into different methods of modeling time series with accompanying examples of time series of hotel room reservations. Various traditional time series models as well as some more modern ones, such as neural networks, are presented throughout the work, where the performance of the models is evaluated and compared to each other. Forecasting is a very demanding task due to a continuously changing environment which can have negative consequences on the model if it fails to track them. The model must continually monitor new data and examine whether there is a drift, known as concept drift, when it must take certain actions to adapt to new data. This paper presents some of the concept drift detection methods, which are evaluated and compared to each other.

Keywords: Time series, statistical models, neural networks, concept drift, forecasting hotel room reservations

Dodaci

A. Rezultati modela

Tablica A.1: Rezultati naivnih (Persistent) modela

Model	MAE(60)	FA(60)	MAE(30)	FA(30)	MAE(7)	FA(7)	Broj parametara
Persistent	14.28	0.65	31.17	0.31	18.43	0.74	0
SeasonalPersistent	9.6	1.19	12.07	1.19	19.43	1.25	0

Tablica A.2: Rezultati statističkih modela

Model	MAE(60)	FA(60)	MAE(30)	FA(30)	MAE(7)	FA(7)	Broj parametara
AR	19.46	0.47	32.82	0.27	47.83	0.18	9
MA	30.69	0.1	45.17	0	58.29	0	25
ARIMA	18.69	0.5	31.37	0.31	46.41	0.2	10
SARIMA	11.72	0.75	18.28	0.6	27.35	0.53	6
Prophet	9.22	1.17	11.21	1.12	7.7	0.98	2,219

Tablica A.3: Rezultati modela neuronskih mreža

Model	Verzija	MAE(60)	FA(60)	MAE(30)	FA(30)	MAE(7)	FA(7)	Broj parametara
MLP	1	16.59	0.55	25.87	0.43	16.47	1.09	$1.31 \cdot 10^5$
MLP	2	8.42	0.92	12.57	0.81	13.6	0.97	$1.51 \cdot 10^5$
MLP	3	11.68	1.23	18.67	1.23	14.38	1.11	$2.01 \cdot 10^5$
MLP	4	11.49	0.8	28.74	0.37	10.74	0.92	$2.51 \cdot 10^5$
MLP	5	11.28	1.14	25.01	0.45	14.18	0.85	$1.31 \cdot 10^5$
MLP	6	9.93	0.89	10.64	0.88	12.61	0.96	$1.51 \cdot 10^5$
MLP	7	12.36	1.27	19.25	1.32	8.79	1.15	$2.01 \cdot 10^5$
MLP	8	8.08	1	17.67	0.67	7.21	1.02	$2.51 \cdot 10^5$
MLP	9	8.33	1.01	17.22	0.65	13.76	0.96	$1.32 \cdot 10^5$
MLP	10	8.45	1	13.9	0.77	13.17	0.94	$1.52 \cdot 10^5$
MLP	11	10.76	1.22	15.21	1.2	9.63	1.07	$2.02 \cdot 10^5$
MLP	12	9.16	1.13	13.95	0.73	14.36	0.78	$2.52 \cdot 10^5$
CNN	1	10.85	1.07	15.85	0.66	18.29	0.71	$9.3 \cdot 10^5$
CNN	2	10.25	0.94	14.46	0.7	23.33	0.6	$1.57 \cdot 10^6$
CNN	3	8.83	1.01	10.92	0.95	8.89	0.85	$3.17 \cdot 10^6$
CNN	4	11.19	0.73	16.23	0.68	21.55	0.63	$4.77 \cdot 10^6$
CNN	5	11.15	1.02	18.64	0.59	17.1	0.76	$9.31 \cdot 10^5$
CNN	6	27.47	1.71	18.78	1.39	11.51	1.11	$1.57 \cdot 10^6$
CNN	7	7.81	0.98	11.65	0.83	19.34	0.67	$3.17 \cdot 10^6$
CNN	8	11.09	0.76	16.91	0.66	26.22	0.55	$4.77 \cdot 10^6$
CNN	9	14.12	0.84	18.44	0.59	18.34	0.69	$9.33 \cdot 10^5$
CNN	10	12.52	1.12	12.07	0.85	14.09	0.77	$1.57 \cdot 10^6$
CNN	11	10.32	1.15	10.03	0.97	13.11	0.78	$3.17 \cdot 10^6$
CNN	12	9.57	0.91	13.18	0.78	20.1	0.66	$4.77 \cdot 10^6$
MultiCNN	1	16.79	0.54	28.15	0.38	29.82	0.49	89,689
MultiCNN	2	10.16	0.92	15.72	0.69	27.6	0.53	$1.7 \cdot 10^5$
MultiCNN	3	8.64	0.99	10.67	0.85	19.1	0.67	$3.7 \cdot 10^5$
MultiCNN	4	9.95	1.04	11.05	0.96	17.84	0.69	$5.7 \cdot 10^5$
MultiCNN	5	20.53	0.43	29.67	0.34	39.01	0.33	90,691
MultiCNN	6	13.42	0.87	16.24	0.66	33.54	0.42	$1.71 \cdot 10^5$
MultiCNN	7	9.21	0.89	12.22	0.81	25.76	0.56	$3.71 \cdot 10^5$
MultiCNN	8	10.42	1.04	12.07	0.89	25.92	0.56	$5.71 \cdot 10^5$
MultiCNN	9	11.94	0.81	17.49	0.62	29.63	0.49	92,695
MultiCNN	10	14.92	0.89	12.88	0.75	28.09	0.52	$1.73 \cdot 10^5$
MultiCNN	11	9.13	0.95	11.53	0.9	20.72	0.65	$3.73 \cdot 10^5$
MultiCNN	12	9.67	0.94	10.93	0.89	23.43	0.6	$5.73 \cdot 10^5$
LSTM	1	18.66	1.23	19.61	0.58	10.35	0.88	$1.08 \cdot 10^6$
LSTM	2	20.43	1.33	28.97	0.36	6.94	1.05	$1.08 \cdot 10^6$
LSTM	3	16.79	1.3	18.59	0.6	13.47	0.86	$1.08 \cdot 10^6$
LSTM	4	18.98	0.46	29.64	0.34	16.3	0.74	$1.08 \cdot 10^6$

Model	Verzija	MAE(60)	FA(60)	MAE(30)	FA(30)	MAE(7)	FA(7)	Broj parametara
LSTM	5	10.07	0.91	22.25	0.51	12.61	0.87	$1.08 \cdot 10^6$
LSTM	6	10.7	1.06	18.12	0.61	8.94	0.96	$1.08 \cdot 10^6$
LSTM	7	17.83	0.58	28.68	0.37	13.1	0.92	$1.08 \cdot 10^6$
LSTM	8	59.67	1.38	45.17	0	58.29	0	$1.08 \cdot 10^6$
LSTM	9	14.71	1.3	22.72	0.5	10.96	0.91	$1.09 \cdot 10^6$
LSTM	10	18.38	1.31	12.96	0.78	11.39	1.12	$1.09 \cdot 10^6$
LSTM	11	69.32	3.06	57.83	2.28	44.71	1.77	$1.09 \cdot 10^6$
LSTM	12	40.33	0.36	45.17	0	58.29	0	$1.09 \cdot 10^6$
BidirectionalLSTM	1	26.27	1.17	27.27	0.4	14.01	1.03	$7.85 \cdot 10^5$
BidirectionalLSTM	2	12.63	1.21	27.26	0.4	12.57	1.05	$7.85 \cdot 10^5$
BidirectionalLSTM	3	13.14	1.09	14.09	1.06	16.71	1.16	$7.85 \cdot 10^5$
BidirectionalLSTM	4	16.8	1.34	12.14	0.79	13.56	0.86	$7.85 \cdot 10^5$
BidirectionalLSTM	5	21.74	1.32	19.34	0.57	16.64	1.09	$7.85 \cdot 10^5$
BidirectionalLSTM	6	12.96	1.11	15.83	0.67	15.51	0.79	$7.85 \cdot 10^5$
BidirectionalLSTM	7	27.62	0.98	12.8	1.02	13.87	1.13	$7.85 \cdot 10^5$
BidirectionalLSTM	8	14.72	1.2	12.07	0.78	14.34	0.82	$7.85 \cdot 10^5$
BidirectionalLSTM	9	21.05	1.14	22.42	0.51	14.18	1.01	$7.86 \cdot 10^5$
BidirectionalLSTM	10	13.45	1.26	20.13	0.56	12.27	1.05	$7.86 \cdot 10^5$
BidirectionalLSTM	11	10.88	1.06	13.26	1.1	19.23	1.3	$7.86 \cdot 10^5$
BidirectionalLSTM	12	20.2	1.5	23	0.49	12.72	1	$7.86 \cdot 10^5$
GRU	1	18	0.53	17.89	0.71	9.94	0.91	$8.13 \cdot 10^5$
GRU	2	15.86	0.59	26.57	0.41	18.31	0.72	$8.13 \cdot 10^5$
GRU	3	20.29	0.43	30.51	0.32	8.49	0.86	$8.13 \cdot 10^5$
GRU	4	17.73	0.51	26.58	0.41	17.28	0.77	$8.13 \cdot 10^5$
GRU	5	28.67	0.85	25.89	0.43	12.85	1.12	$8.14 \cdot 10^5$
GRU	6	9.6	1.06	20.71	0.54	14.33	0.8	$8.14 \cdot 10^5$
GRU	7	10.92	0.98	20.25	0.56	16.1	0.78	$8.14 \cdot 10^5$
GRU	8	14.76	0.65	25.05	0.45	14.7	0.78	$8.14 \cdot 10^5$
GRU	9	25.02	0.97	28.39	0.37	9.87	0.89	$8.15 \cdot 10^5$
GRU	10	26.68	0.95	26.66	0.41	8.6	1.01	$8.15 \cdot 10^5$
GRU	11	15.27	1.36	19.52	0.59	6.61	1.02	$8.15 \cdot 10^5$
GRU	12	16.57	0.56	27.27	0.4	8.98	0.88	$8.15 \cdot 10^5$
AutoencoderMLP	1	14.21	0.66	25.8	0.43	14.26	0.91	$3.92 \cdot 10^5$
AutoencoderMLP	2	13.21	0.72	11.02	0.94	13.87	1.05	$4.12 \cdot 10^5$
AutoencoderMLP	3	11.74	1.29	15.05	1.24	15.04	1.24	$4.62 \cdot 10^5$
AutoencoderMLP	4	9.29	1.12	11.92	0.99	10.1	1.01	$5.12 \cdot 10^5$
AutoencoderMLP	5	29.65	0.75	11.53	1	13.46	1.07	$3.93 \cdot 10^5$
AutoencoderMLP	6	10.64	0.83	13.86	0.82	14.21	1.12	$4.13 \cdot 10^5$
AutoencoderMLP	7	9.8	1.16	13.34	1.08	15.6	1.22	$4.63 \cdot 10^5$
AutoencoderMLP	8	8.55	1.16	10.87	1.18	10.25	1.06	$5.13 \cdot 10^5$
AutoencoderMLP	9	16.78	1.26	25.88	0.43	11.52	0.98	$3.95 \cdot 10^5$
AutoencoderMLP	10	13.89	1.25	10.12	0.92	16.11	1.05	$4.15 \cdot 10^5$
AutoencoderMLP	11	12.08	1.28	14.65	1.26	19.3	1.26	$4.65 \cdot 10^5$
AutoencoderMLP	12	10.79	1.2	11.01	1.03	6.36	1.02	$5.15 \cdot 10^5$
AutoencoderCNN	1	16.44	0.55	26.94	0.4	15.36	0.78	$2.37 \cdot 10^5$

Model	Verzija	MAE(60)	FA(60)	MAE(30)	FA(30)	MAE(7)	FA(7)	Broj parametara
AutoencoderCNN	2	11.93	0.99	14.26	0.71	19.26	0.67	$3.65 \cdot 10^5$
AutoencoderCNN	3	9.6	0.81	12.53	0.77	19.62	0.66	$6.85 \cdot 10^5$
AutoencoderCNN	4	13.15	0.65	17.4	0.63	29.03	0.5	$1.01 \cdot 10^6$
AutoencoderCNN	5	15.93	0.58	25.95	0.43	24.16	0.59	$2.38 \cdot 10^5$
AutoencoderCNN	6	16.29	0.56	26.01	0.42	22.02	0.62	$3.66 \cdot 10^5$
AutoencoderCNN	7	8.75	0.91	10.26	0.88	13.74	0.78	$6.86 \cdot 10^5$
AutoencoderCNN	8	11.85	0.72	15.92	0.69	26.68	0.54	$1.01 \cdot 10^6$
AutoencoderCNN	9	16.62	0.56	27.43	0.39	36.26	0.38	$2.4 \cdot 10^5$
AutoencoderCNN	10	15.13	0.82	15.84	0.67	19.68	0.66	$3.68 \cdot 10^5$
AutoencoderCNN	11	9.09	0.81	11.71	0.82	16.2	0.72	$6.88 \cdot 10^5$
AutoencoderCNN	12	15.43	0.55	21.26	0.53	37.26	0.36	$1.01 \cdot 10^6$
AutoencoderMultiCNN	1	16	0.82	20.4	0.55	18.75	0.7	75,789
AutoencoderMultiCNN	2	12.48	0.93	11.59	0.79	25.16	0.57	91,789
AutoencoderMultiCNN	3	9.04	1.06	9.38	0.94	16.48	0.75	$1.32 \cdot 10^5$
AutoencoderMultiCNN	4	8.37	0.96	10.88	0.83	22.71	0.61	$1.72 \cdot 10^5$
AutoencoderMultiCNN	5	15.62	0.59	28.22	0.38	32.69	0.44	76,791
AutoencoderMultiCNN	6	20.69	0.93	12.81	0.81	27.81	0.52	92,791
AutoencoderMultiCNN	7	9.45	0.92	11.36	0.83	22.9	0.61	$1.33 \cdot 10^5$
AutoencoderMultiCNN	8	8.67	0.96	10.61	0.9	18.8	0.68	$1.73 \cdot 10^5$
AutoencoderMultiCNN	9	18.74	0.77	31.04	0.31	30.65	0.47	78,795
AutoencoderMultiCNN	10	17.61	0.91	18	0.62	24.57	0.59	94,795
AutoencoderMultiCNN	11	9.13	1.03	10.03	0.94	19.92	0.66	$1.35 \cdot 10^5$
AutoencoderMultiCNN	12	10.1	0.86	11.98	0.8	25.17	0.57	$1.75 \cdot 10^5$
AutoencoderMLPLSTM	1	15.49	1.21	17.26	0.64	9.21	0.98	$4.73 \cdot 10^5$
AutoencoderMLPLSTM	2	15.77	0.59	23	0.5	17.16	0.72	$4.93 \cdot 10^5$
AutoencoderMLPLSTM	3	10.53	0.79	14.32	0.74	8.05	1.01	$5.43 \cdot 10^5$
AutoencoderMLPLSTM	4	8.9	1.07	10.05	1.03	9.08	1.06	$5.93 \cdot 10^5$
AutoencoderMLPLSTM	5	13.89	0.7	19.56	0.57	8.95	0.97	$5.33 \cdot 10^5$
AutoencoderMLPLSTM	6	9.54	0.94	21.26	0.53	14.24	0.78	$5.93 \cdot 10^5$
AutoencoderMLPLSTM	7	8.43	1.09	11.43	0.94	8.37	1.08	$7.43 \cdot 10^5$
AutoencoderMLPLSTM	8	8.8	0.98	11.58	0.9	9.19	1.09	$8.93 \cdot 10^5$
AutoencoderMLPLSTM	9	14.86	1.24	10.79	0.82	7.75	0.93	$6.53 \cdot 10^5$
AutoencoderMLPLSTM	10	13.32	1.14	19.72	0.57	14.81	0.78	$7.93 \cdot 10^5$
AutoencoderMLPLSTM	11	9.25	0.89	11.94	0.84	5.58	1.01	$1.14 \cdot 10^6$
AutoencoderMLPLSTM	12	9.11	1.17	9.55	1.08	6.91	1.08	$1.49 \cdot 10^6$
AutoencoderMLPGRU	1	10.26	1.03	25.24	0.45	12.06	0.83	$4.53 \cdot 10^5$
AutoencoderMLPGRU	2	11.44	1.03	14.83	0.69	13.46	0.77	$4.73 \cdot 10^5$
AutoencoderMLPGRU	3	8.54	0.91	10.32	0.92	5.44	1.04	$5.23 \cdot 10^5$
AutoencoderMLPGRU	4	9.19	0.91	9.99	0.96	5.86	1.01	$5.73 \cdot 10^5$
AutoencoderMLPGRU	5	12.82	1.12	22.22	0.51	9.08	0.9	$5.13 \cdot 10^5$
AutoencoderMLPGRU	6	12.04	1.06	16.78	0.64	9.74	0.9	$5.73 \cdot 10^5$
AutoencoderMLPGRU	7	8.58	0.97	9.71	0.98	5.6	1.06	$7.23 \cdot 10^5$
AutoencoderMLPGRU	8	9.33	0.88	10.83	0.89	6.63	0.99	$8.73 \cdot 10^5$
AutoencoderMLPGRU	9	12.32	1.1	19.72	0.56	8.76	0.87	$6.33 \cdot 10^5$
AutoencoderMLPGRU	10	12.36	1.01	19.44	0.57	13.98	0.76	$7.73 \cdot 10^5$
AutoencoderMLPGRU	11	8.66	0.9	9.75	0.9	5.9	0.95	$1.12 \cdot 10^6$
AutoencoderMLPGRU	12	8.81	1.03	9.04	0.98	7.81	1.01	$1.47 \cdot 10^6$