Sveučilište u Rijeci – Odjel za informatiku

Informacijski i komunikacijski sustavi

Sanja Jotić

Podatkovna analitika na primjeru hrvatskog turizma

Diplomski rad

Mentor: prof. dr. sc. Sanda Martinčić – Ipšić dipl. ing.

Rijeka, listopad 2019.

Sadržaj

ODJEL ZA INFORMATIKU
SVEUČILIŠTE U RIJECI
Radmile Matejčić 2, 51000 Rijeka, www.inf.uniri.hr

Rijeka, 6. 6. 2019.

Zadatak za diplomski rad

Pristupnica: Sanja Jotić

Naziv diplomskog rada: Podatkovne analitika na primjeru hrvatskog turizma

Naziv diplomskog rada na eng. jeziku: Data analytics for turism in Croatia

Sažetak teme / sadržaj zadatka diplomskog rada:

Analizirati granični promet putnika na cestovnim, željezničkim, pomorskim i zračnim graničnim prijelazima Republike Hrvatske kako bi se usporedio sa brojem turista odnosno kao pokazatelj razvoja turizma. Za granični promet proučava se broj putnika po vrstama vozila, na kojim je graničnim prijelazima prešlo najviše putnika (analiza graničnog prijelaza obzirom na državu sa kojom RH graniči te svakog pojedinog prijelaza). Izdvojiti granične prijelaze sa najviše putnika. U zračnom prometu prati se broj putnika u pojedinoj zračnoj luci po mjesecima (uočiti koja zračna luka je imala najviše putnika te kada je bio pojačan broj putnika u svim zračnim lukama). Prikazati broj putnika i broj dana provedenih na kružnom putovanju prema zastavi broda, broj putovanja, dana boravaka i broja putnika na brodu po mjesecima (imam i najposjećenije morske luke ali je kumulativno, odnosno gleda se za siječanj, pa za razdoblje siječanj-veljača itd).

Prikazati broj dolazaka i noćenja domaćih i stranih turista po mjesecima (omjer domaćih i stranih turista ukupno i po mjesecima), broj dolazaka i noćenja stranih turista prema zemlji prebivališta (iz koje zemlje dolazi najviše turista – ukupno za godinu i po mjesecima, godišnjim dobima), broj dolazaka i noćenja turista prema vrstama turističkih smještajnih objekata (prikazati gdje turisti najviše odsjedaju, omjer domaćih i stranih turista po smještajnim objektima), broj dolazaka i noćenja prema načinu dolaska – individualno/organizirano (prikazati omjer domaćih i stranih turista), broj dolazaka i noćenja turista na razini RH, u kontinentalnoj i jadranskoj Hrvatskoj, na razini županija, gradova i općina, broj dolazaka i noćenja prema dobnim skupinama (omjer dobnih skupina između domaćih i stranih turista).

Analiza posjećenosti Plitvičkih jezera, najvećeg i najstarijeg nacionalnog parka. Prikazati omjer dolazaka i noćenja domaćih i stranih turista po vrstama smještajnih kapaciteta (uz naziv objekta), broj posjetitelja po mjesecima i predikcija broja za narednu godinu, omjer dolazaka i noćenja posjetitelja prema zemlji prebivališta i vrsti smještajnih kapaciteta (popisan je broj turista po zemljama za svaki smještajni kapacitet te se analizira koji objekt preferira određena nacija), broj dolazaka i noćenja posjetitelja prema zemlji prebivališta (kumulativna tablica, analizirati koje su tri nacije najviše posjećivale Plitvička jezera), omjer broja kreveta i noćenja po naseljima (kumulativno; istražiti da li veći broj kreveta privlači i veći broj turista ili im je draži manji smještaj zbog veće privatnosti i mira te gdje se to naselje nalazi – prikazati na karti – ručno ću izvući koordinate i prikazati sa kružićima gdje se nalazi), broj noćenja po naseljima po mjesecima (izvući 10 najposjećenijih mjesta i prikazati kako raste/pada broj noćenja kroz mjesece), broj raspoloživog kapaciteta privatnog smještaja po naseljima (prikazati po tipu privatnog smještaja – kao tortni grafikon za npr. apartmane broj smještaja po naseljima).

ODJEL ZA INFORMATIKU
SVEUČILIŠTE U RIJECI
Radmile Matejčić 2, 51000 Rijeka, www.inf.uniri.hr

Podaci posjećenosti Plitvičkih jezera će se proširiti s analizom mišljenja posjetitelja (pozitivnih i negativnih) te mišljenjem o 5 smještajnih jedinca s najvećim brojem noćenja (izlučiti 5 smještaja iz podataka o broju noćenja iz tablica).

Planirani alati: Pripremit će se python skripte za upis podatka u CSV format, te za automatsko prikupljanje i klasificiranje komentara o smještajnim jedinicama. Vizualizacija će se izvesti u Tableau alatu.

Mentorica:

Prof. dr. sc. Sanda Martinčić-Ipšić

Voditeljica za diplomske radove:

Izv. prof. dr. sc. Ana Meštrović

Zadatak preuzet:

Sanja Jotić

(potpis pristupnika)

Sažetak

U radu je provedena analitika hrvatskog turizma na podacima prikupljenim sa stranica Državnog zavoda za statistiku i Turističke zajednice Općine Plitvička jezera. Za preuzimanje podataka izrađene su Python skripte čija funkcionalnost je detaljno opisana u radu. Uz navedene podatke preuzete su i recenzije za četiri najposjećenija hotela u Plitvičkim jezerima sa stranice TripAdvisor.

Analiziran je granični promet u kojem sudjeluju domaći i strani turisti, kružna putovanja stranih brodova koja postaju sve popularnija te posebno zračni i cestovni promet kao dva najkorištenija načina posjeta Hrvatskoj. Analiza prikupljenih podataka otkriva u koji dio Hrvatske dolazi najviše turista, koji strani turisti dolaze te koliko dugo odsjedaju, u kojoj vrsti smještaja te koje su dobi i spola.

Posebno se analizira turizam Plitvičkih jezera u koje dolazi sve veći broj posjetitelja te se prikazuje u kojim naseljima i smještajima odsjedaju. Pojedine smještaje favoriziraju određene nacije te je i to prikazano uz grafikon o dolascima i noćenjima po mjesecima. Izdvojena su četiri hotela sa najvećim brojem dolazaka i noćenja te su se za njih analizirale recenzije prikupljene sa stranice TripAdvisor.

Ključne riječi: hrvatski turizam, turisti, smještaj, Plitvička jezera, Python, TripAdvisor, analiza recenzija

Abstract

The paper analyzes the Croatian tourism on the data collected from the pages of the Central Bureau of Statistics and The Official Plitvice Lakes Tourist Board. Python scripts have been created to download the data, the functionality of which is described more detail in the paper. In addition to the above data, the reviews for the four most visited Plitvice Lakes hotels have been scraped from TripAdvisor.

The border traffic has been analyzed in which domestic and foreign tourists participate, cruise ships of foreign ships which are becoming more popular, and especially air and road traffic as two of the most used ways of visiting Croatia. It is revealed which part of Croatia most tourists come, which foreign tourists come and how long they stay, in what type of accommodation and what are their age and gender.

The tourism of Plitvice Lakes is being analyzed in particular, with an increasing number of visitors coming and showing in which settlements and accommodations they are staying. Some accommodations are favored by certain nations and this is also shown with a chart of arrivals and nights by month. Four hotels with the highest number of arrivals and overnights were identified and reviews for that hotels were collected from TripAdvisor analyzed.

Keywords: Croatian tourism, tourists, accommodation, Plitvice Lakes, Python, TripAdvisor, review analysis

Uvod

U ovom radu provedena je analiza hrvatskog turizma 2017. i 2018. godine. Analizirali su se podaci prikupljeni sa stranice Državnog zavoda za statistiku te podaci o turizmu Plitvičkih jezera prikupljeni sa stranice Turističke zajednice Općine Plitvička jezera i recenzije napisane na stranici TripAdvisor. Za preuzimanje podataka sa navedenih stranica napisane su Python skripte te je korišten alat Tabula da bi se izlučili podaci o turizmu Plitvičkih jezera sadržani u dokumentima PDF formata. Sve vizualizacije podataka izrađene su u alatu Tableau. Integracijom podataka iz službenih i društvenih izvora pokazuje se mogućnosti obogaćivanja standardnih podatka za poslovnu analitiku podacima iz vanjskih izvora.

U **prvom poglavlju** opisan je postupak prikupljanja podataka sa navedenih izvora. Opisan je postupak rada Python skripti koje služe za preuzimanje podataka sa Državnog zavoda za statistiku te izlučivanje podataka iz datoteka PDF formata koje sadrže podatke o turizmu Plitvičkih jezera. Kako se izlučivala jedna po jedna tablica jer su većinom objavljivani podaci po mjesecima za svaku godinu, radi jednostavnije izrade vizualizacija, sve tablice iste domene spojene su u jednu veliku tablicu. Za potrebe spajanja napisana je Python skripta.

U **drugom poglavlju** izrađene su vizualizacije pomoću tablica koje su nastale u prethodnom poglavlju. Analiziran je turizam na više razina – granični promet, vrste graničnog prometa, vrste putnika, vrste prijevoznih sredstava koje su putnici koristili te način na koji su došli u Hrvatsku (organizirano ili individualno). Posebno se analizirao cestovni i zračni promet jer se pokazao kao najviše korišten među putnicima. Popularnost kružnih putovanja sve više raste te se analizira iz koje države ima najviše putnika, koliko dugo ostaju te koliko ima putnika na brodovima. Uspoređen je broj dolazaka i noćenja domaćih i stranih turista na godišnjoj razini, mjesečnoj, prema vrsti turističkog smještaja, na razini Hrvatske, županija te općina za koje su izdvojene županije i općine u tim županijama koje imaju najveći broj posjeta domaćih i stranih turista. Također se uspoređuju dolasci i noćenja prema spolu. Za turizam Plitvičkih jezera analiziraju se dolasci i noćenja domaćih i stranih turista po ugostiteljskim objektima te su izdvojeni najposjećeniji objekti i analiziraju se po zemlji prebivališta. Prikazan je broj posjeta turista Nacionalnom parku Plitvička jezera te su izdvojeni mjeseci sa najvećim brojem posjeta. Također se analizira broj kreveta i noćenja po naseljima i na karti su prikazana tri naselja sa najvećim brojem dolazaka i noćenja te je na kraju prikazan udio broja apartmana i domaćinstava po naseljima kao najbrojnijih smještaja u privatnom posjedu.

U **trećem** ujedno i zadnjem poglavlju opisan je postupak preuzimanja recenzija za četiri najposjećenija smještaja u Plitvičkim jezerima sa stranice TripAdvisor. Analiziran je polaritet recenzija i prikazane su najčešće korištene riječi te koje su bili najpozitivnije, najnegativnije i neutralne recenzije. Na kraju je izrađen klasifikator od prikupljenih riječi i izračunata njegova točnost.

1 Ekstrakcija i čišćenje podataka za vizualizaciju

Podaci koji se u radu koriste za vizualizaciju preuzeti su sa stranice Državnog zavoda za statistiku [1] te sa stranice Turističke zajednice općine Plitvička jezera [2]. Za preuzimanje podataka sa stranica Državnog zavoda za statistiku bilo je potrebno napisati *Python* skripte u kojima se pomoću *web scraping-a* preuzelo podatke iz tablica objavljenih u publikacijama. ***Web scraping*** je tehnika za izlučivanje velikih količina podataka sa web stranica pomoću koje se podaci ekstrahiraju i spremaju u datoteke na računalo ili u bazu podataka u obliku tablica [3]. Na većini web stranica podaci se mogu samo pregledati, ne postoji opcija za spremanje ili kopiranje podataka kako bi ih mogli dalje koristiti. U tom slučaju možemo ručno kopirati i zalijepiti podatke u datoteke, ali taj posao je mukotrpan i može trajati danima. Da bi se taj proces automatizirao koristi se *web scraping* za koji možemo napisati vlastiti kod/program ili koristiti neki od postojećih softvera.

Podaci na stranici Turističke zajednice općine Plitvička jezera objavljeni su u dokumentima u PDF formatu te su ti dokumenti preuzeti na računalo, a podaci u tablicama izlučeni pomoću alata Tabula [4]. **Tabula** je alat za izlučivanje tablica iz PDF datoteka, a omogućava spremanje izlučenih podataka u .csv ili .tsv formatu.

Za potrebe diplomskog rada svi podaci spremljeni su u .csv formatu.

1.1 Ekstrakcija podataka sa Državnog zavoda za statistiku

Za analizu turizma u Hrvatskoj sa stranica Državnog zavoda za statistiku preuzete su publikacije koje sadrže podatke o graničnom prometu, prometu u zračnim lukama, dolascima i noćenju turista u komercijalnom smještaju te podaci o kružnom putovanju stranih brodova.

Publikacija o graničnom prometu sadrži tri tablice sljedećih naziva:

- Granični promet putnika na cestovnim, željezničkim, pomorskim i zračnim graničnim, prijelazima te graničnim prijelazima na unutarnjim vodnim putovima,
- Granični promet putničkih cestovnih motornih vozila po vrstama vozila,
- Granični promet putničkih cestovnih motornih vozila i putnika po smjerovima kretanja.

Podaci u tablicama normalizirani su i prikazani su u tisućama, a ukoliko je vrijednost „–„ znači da nema pojave, a ako je „0“ podatak je manji od 0,5 upotrebljene mjerne jedinice [1].

Skripta za preuzimanje podataka iz ove i ostalih publikacija napisana je tako da se od korisnika traži unos u obliku linka web stranice. Pristupi se željenoj web stranici i preuzme njen sadržaj u HTML obliku, izluče se podaci iz tablica koje uredimo na željeni način, ekstrahira se naziv tablice te se on koristi kao naziv .csv datoteke u koju spremamo preuzete podatke. Svaka publikacija sadrži podatke po mjesecima za dvije godine (trenutnu i prethodnu). U radu će se analizirati podaci za 2017. i 2018. godinu koji su sadržani u publikacijama objavljenim 2017. i 2018. godine.

Skripta *cross-border_scraper.py* sadrži kôd za preuzimanje podataka iz publikacija o graničnom prometu. Kod pokretanja programa unosi se link stranice, a da bi se omogućio unos od strane korisnika koristi se funkcija *input()* koja na ekranu ispisuje „Unesite link:“ te se nakon unosa vrijednost sprema u varijablu *url*. Za dohvaćanje podataka potrebno je koristiti biblioteku *requests* odnosno njenu metodu *get()*. Sa *get()* metodom dohvaćamo podatke sa određenog izvora te se kao parametar metode prosljeđuje link stranice spremljen u varijabli *url* [5]. Rezultate poslanog zahtjeva spremamo u varijablu *response*. Obzirom da se na stranici nalaze dijakritički znakovi, kodiranje je u *meta* tagu postavljeno na Windows-1250 koje obuhvaća hrvatsku abecedu. HTTP zahtjev nije prepoznao da je stranica kodirana tim kodiranjem, pa je potrebno to ručno postaviti koristeći *propery encoding*. Da bi parsirali stranicu i dobili HTML koristi se funkcija *BeautifulSoup* iz biblioteke *bs4*. Ta funkcija prima dva argumenta: prvi je HTML *string* koji će se parsirati, a dobijemo ga putem *text property*-ja varijable *response*, a drugi je ime parsera [6]. Funkcija vraća *bs4.BeautifulSoup* objekt koji spremamo u *html_soup* varijablu i koristi se za izlučivanje podataka unutar HTML tagova:

```
url = input("Unesite link: ")
response = get(url)
response.encoding = 'windows-1250'
html_soup = BeautifulSoup(response.text, 'html.parser')
```

Navedeni dio koda nalazi se unutar *main* metode, a u nastavku su radi preglednosti napisane funkcije za pojedine funkcionalnosti.

Varijabla *tables* je lista u kojoj su elementi liste od kojih svaka predstavlja jedan redak tablice u publikaciji. Ta lista je rezultat koji vraća funkcija *get_tables()*. U toj funkciji imamo deklarirane dvije liste, *all_data* i *tr_data*, te tri brojača *oneAdd, secondIdx* i *addTd*. Funkciji *get_tables()* prosljeđuje se *html_soup* varijabla. Unutar funkcije u varijablu *tables* spremamo

rezultat funkcije *find_all()* pomoću koje u *html_soup* tražimo sve *table* tagove klase *MsoNormalTable*. Rezultat je lista objekata te uzimamo samo elemente od 3. do 6. indeksa jer su tu spremljene tri tablice koje želimo izlučiti iz publikacija:

```
tables = html_soup.find_all('table', attrs={'class':'MsoNormalTable'})
tables = tables[3:6]
```

U for petlji pristupamo svakom elementu koji predstavlja jednu tablicu te pronalazimo sve *tr* tagove i spremamo u *table_rows* varijablu. Zatim se iterira *table_rows* varijabla te se pronalazi *td* tag odnosno ćelije u tablici i sprema u *table_data* varijablu. Iteriramo i tu varijablu te sa funkcijom *find()* pronalazimo *span* tag i sa *text property-jem* dobivamo tekst odnosno podatak sadržan u ćeliji tablice. Taj podatak čistimo tako sa *strip()* funkcijom uklonimo prazna mjesta, zamijenimo znak \n koji predstavlja prelazak u novi red za prazno te ako imamo dva prazna mjesta (dva puta *space*) zamijenimo ih za prazno odnosno uklonimo. Tako očišćen podatak ćelije spremljen je u varijablu *data*:

```
for table in tables:
    table_rows = table.find_all('tr')
    for tr in table_rows:
        table_data = tr.find_all('td')
        for td in table_data:
            data = td.find('span').text.strip().replace("\n","").replace("  ","")
```

Kako broj ćelija u zaglavlju tablice nije konzistentan, primjerice u prvoj tablici u prvom retku imamo četiri ćelije, a u drugom šest, potrebno je više puta zapisati isti podatak da bi imali jednak broj ćelija u svakom retku. Tako provjeravamo ako je u *data* spremljena riječ „Ulazak“ ili riječ „Izlazak“, onda se u listu *all_data* ta riječ sprema tri puta:

```
if(data == "Ulazak" or data == "Izlazak"):
    for i in range(3):
        all_data.append(data)
    continue
```

Ako je u *data* spremljeno "promet, tis.", "Promet, tis.", "vozila, tis." ili "promet" provjeravamo da li je to prvo pojavljivanje, te ako je dodamo prazan string u *all_data* i povećamo brojač *oneAdd* za jedan. Zatim tu vrijednost dodamo dva puta u *all_data* i na kraju provjerimo da li je brojač došao do vrijednosti 2 i ako je postavimo ga na 0:

```
if(data == "promet, tis." or data == "Promet, tis." or data == "vozila, tis." or data == "promet"):
    if(oneAdd < 1):
        all_data.append("")

    oneAdd += 1
    for i in range(2):
        all_data.append(data)

    if(oneAdd == 2):
        oneAdd = 0
    continue
```

Ako je u *data* spremljeno "indeksi", brojač *secondIdx* poveća se za jedan i u *all_data* dodamo vrijednost iz *data*. Ako je brojač došao do vrijednosti 2 odnosno našli smo riječ „indeksi“ po drugi puta tada u *all_data* dodajemo prazan string i brojač postavljamo na 0:

```
if(data == "indeksi"):
    secondIdx += 1
    all_data.append(data)

    if(secondIdx == 2):
        all_data.append("")
        secondIdx = 0
        break

    continue
```

Ako u *data* nađemo datum u formatu „mjesec (rimskim brojem) godina“ za 2017. godinu, u *all_data* dodajemo najprije prazan string pa vrijednost spremljenu u *data*:

```
if(data in ["I. 2017.", "II. 2017.", "III. 2017.", "IV. 2017.", "V. 2017.",
            "VI. 2017.", "VII. 2017.", "VIII. 2017.","IX. 2017.", "X. 2017.",
            "XI. 2017.", "XII. 2017."]):
    all_data.append("")
    all_data.append(data)
    continue
```

Kada nađemo datum i istom formatu ali za 2018. godinu, tada najprije povećamo brojač *secondIdx* za jedan pa dodamo vrijednost u *all_data*. Kada vrijednost brojača bude jednaka 2, tada pomoću for petlje dva puta dodamo prazan string u *all_data* i postavimo vrijednost brojača na 0. U svim ostalim slučajevima samo spremimo vrijednost iz *data* u *all_data*:

```
if(data in ["I. 2018.", "II. 2018.", "III. 2018.", "IV. 2018.", "V. 2018.",
            "VI. 2018.", "VII. 2018.", "VIII. 2018.","IX. 2018.", "X. 2018.",
            "XI. 2018.", "XII. 2018."]):
    secondIdx += 1
    all_data.append(data)

    if(secondIdx == 2):
        for i in range(2):
            all_data.append("")
        secondIdx = 0
        break

else:
    all_data.append(data)
```

Svaka od tri tablice ima osam stupaca. Kako bi rasporedili koji podatak se nalazi u kojem retku koristimo brojač *addTd*. Ako je vrijednost brojača jednaka 0 tada u listu *tr_data* dodajemo novu listu sa vrijednosti *data* iz liste *all_data* i povećamo vrijednost brojača za jedan. Ako brojač ima neku drugu vrijednost u prethodnu listu dodajemo vrijednost *data*. Provjeravamo da li je vrijednost brojača 8 te ako je postavljamo ga na 0 kako bi mogao spremiti sljedećih 8 vrijednosti u novu listu:

```
for data in all_data:
    if(addTd == 0):
        tr_data.append([data])
        addTd += 1
    else:
        tr_data[-1].append(data)
        addTd += 1

        if(addTd == 8):
            addTd = 0
```

Lista *tr_data* vraća se kao rezultat funkcije *get_tables()* i izgleda ovako:

Index	Type	Size	Value
0	list	8	['', 'Ulazak', 'Ulazak', 'Ulazak', 'Izlazak', 'Izlazak', 'Izlazak', '' ...
1	list	8	['', 'promet, tis.', 'promet, tis.', 'indeksi', 'promet, tis.', 'prome ...
2	list	8	['', 'I. 2017.', 'I. 2018.', '', 'I. 2017.', 'I. 2018.', '', '']
3	list	8	['', '', '', '', '', '', '', '']
4	list	8	['Ukupan granični promet', '4 230', '4 307', '101,8', '4 412', '4 ...
5	list	8	['Domaći putnici', '1 652', '1 692', '102,4', '1 720', '1 707', ' ...
6	list	8	['Strani putnici', '2 578', '2 615', '101,4', '2 692', '2 687', ' ...
7	list	8	['', '', '', '', '', '', '', '']
8	list	8	['Cestovni granični promet', '4 133', '4 198', '101,6', '4 300', ' ...
9	list	8	['Domaći putnici', '1 612', '1 642', '101,8', '1 682', '1 660', ' ...
10	list	8	['Strani putnici', '2 521', '2 556', '101,4', '2 618', '2 609', ' ...

Podaci su izlučeni i sada treba ih treba urediti tako da se zaglavlje tablice objedini u jedan redak i da se prazni reci izbrišu. Taj dio odrađuje funkcija *edit_tables()* koja kao argument prima listu iz prethodne funkcije spremljenu u varijablu *tables* u glavnoj funkciji. Lista *tr_data* sadrži liste te se provjerava da li je element unutar liste riječ „Ulazak“. Ako je sa for petljom iteriramo elemente liste i element na poziciji *d* koji odgovara riječi „Ulazak“ spaja se sa elementom sljedeće dvije liste na istoj poziciji u novi string, a elementi u tim listama postaju prazan string:

```
for data in range(len(tr_data)):
    if("Ulazak" in tr_data[data]):
        for d in range(len(tr_data[data])):
            tr_data[data][d] += " " + tr_data[data+1][d] + " " + tr_data[data+2][d]
            tr_data[data+1][d] = ""
            tr_data[data+2][d] = ""
```

Sljedeći korak u funkciji je izbrisati prazne retke. Ukoliko lista unutar liste *tr_data* ne sadrži prazan string, nju dodajemo u *tmp* listu. Na taj način eliminiramo liste kojima su elementi prazni stringovi. Ovako uređenu listu spremamo u varijablu *edited_tables* u main funkciji programa:

```
tmp = []

for data in tr_data:
    if "" not in data:
        tmp.append(data)
tr_data = tmp
```

Sada je potrebno podatke spremiti u .csv datoteku. Funkciji *save_to_files()* u main funkciji prosljeđuje se uređena lista *edited_tables* i *html_soup* varijabla. U ovoj funkciji odredimo koji retci pripadaju kojoj tablici na način da iteriramo kroz *tables* (odnosno *edited_tables*) listu i provjeravamo da li je prvi element svake unutarnje liste dva *space*-a. Ako je tu listu spremimo u *tmp* listu, a ako nije, listu dodajemo u prethodnu listu. Tako dobijemo tri liste od kojih svaka za elemente ima listu koja predstavlja jedan redak tablice:

```
for tr in tables:
    if tr[0] == "  ":
        tmp.append([tr])
    else:
        tmp[-1].append(tr)
```

Prije nego spremimo podatke u datoteku potrebno je dohvatiti naslove svake tablice. Za to služi funkcija *get_title()* radi koje smo u *save_to_files()* funkciju proslijedili *html_soup* varijablu, a sad ju prosljeđujemo u *get_title()*. U toj funkciji najprije pretražujemo sve *bold* tagove i spremamo u varijablu *bolds* jer su svi naslovi podebljani. Zatim iteriramo listu *bolds* i ako u elementu liste možemo naći *span* tag sa atributom *lang = EN-US*, tekst tog elementa spremamo

u listu *spans*, te od te liste uzmemo samo elemente od 5 do 10 indeksa jer se u tim elementima nalaze naslovi naših tablica:

```
spans = []
temp = []

bolds = html_soup.find_all('b')
for bold in bolds:
    if bold.find('span', {'lang':'EN-US'}):
        span = bold.find('span', {'lang':'EN-US'})
        spans.append(span.text)

spans = spans[5:10]
```

Neki naslovi tablica zapisani su pomoću dva *span* tag-a te je potrebno ponovno ih spojiti. Iz tog razloga provjeravamo da i je prvi znak elementa u listi *spans* broj jer svaki naslov počinje sa rednim brojem, te ako je spajamo taj element sa narednim. Da bismo eliminirali dio naslova koji smo prethodno spojili i višak je, provjeravamo da li je prvi znak elementa liste broj, te ukoliko je spremamo ga u *temp* listu i njen sadržaj prepisujemo u sadržaj liste *spans*:

```
for i in range(0, len(spans)-1):
    if(spans[i][0].isdigit()):
        spans[i] += spans[i+1]

for i in range(0, len(spans)):
    if(spans[i][0].isdigit()):
        temp.append(spans[i])
spans = temp
```

Svaki naslov koji je element liste *spans* pročistimo od nepotrebnih znakova te tako uređenu listu vratimo kao rezultat funkcije i spremimo unutar funkcije *save_to_file()* u varijablu *titles*:

```
for span in range(len(spans)):
    if span == 0:
        spans[span] = spans[span].replace(u'\xa0',' ').replace("     "," ")
        .replace("  "," ").replace("\r","").replace("\n"," ").strip().lower()[4:]
    else:
        spans[span] = spans[span].replace(u'\xa0',' ').replace("  ","")
        .replace("\r","").replace("\n"," ").strip().lower()[3:]

return spans
```

Sada kada imamo naslove tablica iteriramo kroz listu *tmp* u kojoj se nalaze podaci. U *open()* funkciji navedemo putanju do mape u kojoj ćemo spremiti .csv datoteke i konkateniramo sa naslovom na način da brojač *i* koristimo i za iteraciju listom *tmp* i listom *titles* obzirom da obje imaju isti broj elemenata. Sa *'w'* naznačimo da ćemo pisati u datoteku, sa *newline=''* da nemamo praznog retka kada se prelazi u novi red odnosno sprema sljedeći element liste, te kodiranje datoteke postavimo na UTF-8. Sa *j* brojačem iteriramo kroz svaku unutarnju listu, u *writer()* funkciji navedemo datoteku koja se sprema i kao delimiter naznačimo zarez. *writerow()*

funkcija služi za upis lista u datoteku te pomoću nje upisujemo sve elemente jer nam je jedan redak u tablici predstavljen kao lista čiji su elementi podaci iz ćelija. Na ovaj način spremili smo sve preuzete tablice i njihove naslove iskoristili kao naziv za datoteku u koju se spremaju:

```
for i in range(len(tmp)):
    with open("C:\\Users\\Sanja\\Documents\\FAKS\\diplomski rad\\data\\granicni promet\\" + titles[i] + ".csv",
              'w', newline='', encoding='utf-8') as csv_file:
        for j in range(len(tmp[i])):
            table_writer = csv.writer(csv_file, delimiter=',')
            table_writer.writerow(tmp[i][j])
```

Sljedeća publikacija čije tablice preuzimamo je publikacija o prometu u zračnim lukama i sadrži tri tablice sljedećih naslova:

- Promet u zračnim lukama,
- Promet putnika u zračnim lukama,
- Deset zemalja s najvećim ostvarenim prometom putnika u hrvatskim zračnim lukama.

Nas interesira samo tablica o Prometu putnika u zračnim lukama te ćemo samo nju preuzeti i spremiti u .csv datoteku.

U skripti *airports_scraper.py* nalazi se identičan kôd kao i u prethodnoj skripti, funkcije su iste jedino su neki dijelovi koda promijenjeni.

U *get_tables()* funkciju prosljeđuju se dva argumenta – *html_soup* i *url*. Stranica je koncipirana tako da se prva tablica nalazi unutar *table tag*-a, a druge dvije su ugniježđene u jedan *table tag* i svaka je novi *table tag* odnosno imamo tablicu čiji su elementi dvije tablice. Iz tog razloga drugi element liste *tables* je tablica koja sadrži dvije tablice i da bi ih izlučili potrebno je sa *findAll()* funkcijom pronaći sve *table tag*-ove:

```
for i in range(len(tables)):
    if i == 1:
        tables[i] = tables[i].findAll('table')
```

Sada imamo listu u kojoj je drugi element opet dvije tablice ali se više ne nalaze unutar *table tag-a* koji je služio kao *container*. Svaku tablicu spremimo kao poseban element liste tako da prvi element liste spremimo u *tmp* listu, a drugi iteriramo i svaku od dvije tablice posebno spremimo u *tmp* listu:

```
for i in range(len(tables)):
    if i == 0:
        tmp.append(tables[i])
    else:
        for j in range(len(tables[i])):
            tmp.append(tables[i][j])
tables = tmp
```

Obzirom da tablice nemaju jednak broj stupaca, listu *all_data* inicijaliziramo na onoliko *None* vrijednosti koliko lista *tables* ima elemenata. Kao i u prethodnom kodu pronalazimo *tr* pa *td tag-ove* i u konačnici spremamo pročišćen podatak. Da bi znali na koje mjesto u listi spremiti provjeravamo da li je lista *all_data* prazna te ako je na to mjesto dodajemo novu listu sa podatkom, a ako nije dodajemo podatak u prethodno popunjenu listu. Za prolazak kroz *all_data* listu koristimo isti brojač kao i za *tables* listu obzirom da moraju imati isti broj elemenata (tablica):

```
all_data = [None] * len(tables)

for table in range(len(tables)):
    table_rows = tables[table].find_all('tr')
    for tr in table_rows:
        table_data = tr.find_all('td')
        for td in table_data:
            data = td.text.strip().replace("\n"," ").replace("    ","")

            if all_data[table] is None:
                all_data[table] = [data]
            else:
                all_data[table].append(data)
```

Kako smo do sada izlučili sve tablice, a treba nam samo druga, pomoću indeksa uzimamo samo drugu tablicu i provjeravamo ako je podatak u varijabli *data* jednak „2017“, „2018“ ili „Indeksi Indices“, te ako je u *tmp* listu spremamo taj podatak dva puta, inače samo spremimo podatak u *tmp* listu. Ovaj dio je potreban radi tablice u publikaciji za 12. mjesec koja ima veći broj stupaca i potrebno je dodati pojedine nazive da bi se izjednačio broj podataka u svakom retku tablice:

```
tmp = []

for d in range(len(all_data)):
    for data in all_data[d]:
        if d == 1:
            if data == "2017." or data == "2018." or data == "Indeksi Indices":
                for i in range(2):
                    tmp.append(data)
            else:
                tmp.append(data)
all_data = tmp
```

Da bi odredili koji podatak pripada kojem retku tablice u ovom slučaju koristimo isti kod kao u prethodnoj skripti ali sadrži sitne modifikacije. Provjerava se da li je u *url_date* varijabli u kojoj je izlučen podatak iz linka stranice o mjesecu za koji je objavljena publikacija jednak 12, te ako je spremat će se po 5 elemenata u listu, a ako nije onda će se spremati 3 jer tablica u prosincu ima pet stupaca, a sve ostale tri:

```
url_date = url[-11:-9]

for data in all_data:
    if(addTd == 0):
        tr_data.append([data])
        addTd += 1
    else:
        tr_data[-1].append(data)
        addTd += 1

        if(url_date == "12"):
            if(addTd == 5):
                addTd = 0
        else:
            if(addTd == 3):
                addTd = 0
```

Ponovno imamo listu čiji elementi su liste koje predstavljaju jedan redak tablice i potrebno je urediti podatke prije nego ih spremimo u datoteku. U funkciji *edit_tables()* iteriramo kroz listu *tr_data* i slično kao u prethodnoj listi provjeravamo da li se u nekoj od lista nalazi podatak „2017.“ ili „2018.“ te ako se nalazi na taj podatak dodamo element iz naredne liste na istoj poziciji, a element na toj poziciji zamijenimo za prazan string:

```
for data in range(len(tr_data)):
    if("2017." in tr_data[data] or "2018." in tr_data[data]):
        for d in range(len(tr_data[data])):
            tr_data[data][d] += " " + tr_data[data+1][d]
            tr_data[data+1][d] = ""
```

Kada smo spojili zaglavlje tablice u jedan redak obrišemo sve liste koje sadrže prazne elemente:

```
tmp = []

for d in range(len(tr_data)):
    if tr_data[d][1] != "":
        tmp.append(tr_data[d])
tr_data = tmp
```

Sa funkcijom *save_to_files()* spremamo podatke ali prije toga moramo izlučiti naslov tablice sa *get_title()* funkcijom. Postupak pronalaženja tag-ova je isti, ali sada uzimamo manje elemenata iz liste *spans* jer je naslov naše tablice smješten u dva *span* tag-a. Elemente u listi *spans* spojimo i dobijemo *string* koji očistimo od nepotrebnih znakova i takav naslov vratimo kao rezultat funkcije i spremimo u varijablu *titles*:

```
spans = []

bolds = html_soup.find_all('b')
for bold in bolds:
    if bold.find('span', {'lang':'EN-US'}):
        span = bold.find('span', {'lang':'EN-US'})
        spans.append(span.text)
spans = spans[7:9]
spans = ''.join(spans)
spans = spans.replace(u'\xa0',' ').replace("  ","").replace("   "," ")
.replace("\r"," ").replace("\n"," ").lower().lstrip()

return spans
```

Kako u listi *tables* imamo listu listi, potrebno je samo otvoriti datoteku sa funkcijom *open()* i iterirati kroz listu te svaki element (listu) spremiti u datoteku:

```
titles = get_title(html_soup)

with open("C:\\Users\\Sanja\\Documents\\FAKS\\diplomski rad\\data\\zracne luke\\" + titles + ".csv",
          'w', newline='', encoding='utf-8') as csv_file:
    for i in range(len(tables)):
        table_writer = csv.writer(csv_file, delimiter=',')
        table_writer.writerow(tables[i])
```

Publikacije o dolasku i noćenju turista u komercijalnom smještaju sadrže šest tablica pomoću kojih je turizam analiziran iz više aspekata, a tablice su sljedećih naziva:

- Dolasci i noćenja turista,
- Dolasci i noćenja stranih turista prema zemlji prebivališta,
- Dolasci i noćenja turista prema NKD-u 2007., Odjeljak 55, i vrstama turističkih smještajnih objekata,
- Dolasci i noćenja turista prema načinu dolaska turista,

- Dolasci i noćenja turista, Republika Hrvatska, NKPJS 2012. – 2. razina, županije, gradovi, općine,
- Dolasci i noćenja turista prema dobnim skupinama.

Za potrebe preuzimanja ovih tablica izrađene su dvije skripte – *tourism_scraper.py* i *tourism_scraper_2017.py*. Razlog izrade dvije skripte je taj što zadnje dvije tablice ne sadrže podatke za dvije godine nego samo za tekuću te je bilo potrebno izraditi drugu skriptu koja preuzima podatke za navedene tablice.

U skripti *tourism_scraper.py* funkciji *get_tables()* dodan je još jedan parametar – url, koji sadrži link stranice sa koje se preuzima publikacija. U *url_date* varijabli spremi se datum i godina izlučeni iz linka te uređeni na način da se znak '_' zamijeni za razmak. Funkcija sadrži listu *all_data* koja ima onoliko *None* vrijednosti koliko je tablica u publikaciji. Na taj način odmah se odvoji svaku tablicu u posebnu listu, a to je potrebno jer je u protivnom kasnije teško odrediti koji reci pripadaju kojoj tablici. Kada se pronađe podatak i spremi u *data* varijablu provjerava se da li se u *all_data* listi nalazi *None* te ako se nalazi na to mjesto sprema se lista sa vrijednosti *data*. Ako već postoji spremljena vrijednost odnosno lista, tada se na tu poziciju u listu dodaje nova vrijednost:

```
all_data = [None] * len(tables)

for table in range(len(tables)):
    table_rows = tables[table].find_all('tr')
    for tr in table_rows:
        table_data = tr.find_all('td')
        for td in table_data:
            if td.find('span') is not None:
                data = td.find('span').text.strip().replace("\n","").replace("\r","")
                .replace("    ","")

                if all_data[table] is None:
                    all_data[table] = [data]
                else:
                    all_data[table].append(data)
```

Tablice je potrebno modificirati kako bi svaki redak imao jednaki broj stupaca. Da bi i dalje svaka tablica bila spremljena u posebnu listu uvodi se lista *tmp* koja se inicijalizira na onoliko *None* vrijednosti koliko je elemenata odnosno tablica u listi *all_data*.

Na nultom, prvom ili trećem elementu liste *all_data* provjerava se da li je u varijabli *data* spremljena vrijednost „Dolasci“ ili „dolasci“ te ako je dodaje se u listu *tmp* na poziciju *d*. Riječ „Dolasci“ se nalazi u prvoj i drugoj tablici, a riječ „dolasci“ u četvrtoj te ih je potrebno dodati

u listu tri puta kako bi se izjednačio broj stupaca u svakom retku. Isto se radi i za svaku sljedeću tablicu gdje indeks liste *all_data* predstavlja tablicu za koju se dodaju vrijednosti:

```
tmp = [None] * len(all_data)

for d in range(len(all_data)):
    for data in all_data[d]:
        if d in [0, 1, 3]:
            if data == "Dolasci" or data == "dolasci":
                for i in range(3):
                    tmp[d].append(data)
```

Ako se nalazimo na prvom ili trećem elementu liste *all_data*, te ako je u *data* spremljena vrijednost „Noćenja“ ili „noćenja“, ta se vrijednost dodaje tri puta u *tmp* listu:

```
if d in [1, 3]:
    if data == "Noćenja" or data == "noćenja":
        for i in range(3):
            tmp[d].append(data)
```

Na nultom elementu odnosno prvoj tablici provjerava se nalazi li se u *data* riječ „Noćenja“ te ako se nalazi u *tmp* se tri puta dodaje varijablu *data* i zatim dva puta prazan string. U slučaju da se pronađe neka druga vrijednost provjeri se da li se u *tmp* listi nalazi *None*, te ako se nalazi na toj poziciji se dodaje novu listu sa pronađenom vrijednosti, inače se provjeri da li je vrijednost u *data* različita od „Dolasci“ te ako je dodaje se u listu:

```
# prva tablica
if d == 0:
    if data == "Noćenja":
        for i in range(3):
            tmp[d].append(data)
        for i in range(2):
            tmp[d].append("")
    else:
        if tmp[d] is None:
            tmp[d] = [data]
        else:
            if data != "Dolasci":
                tmp[d].append(data)
```

Na prvom elementu provjeri se da li se *data* nalazi u listi *date_2018* u kojoj je popis datuma za 2018. godinu u formatu mjesec godina, te da li je brojač *c2018* manji od 1 tj. da li je pronađen prvi datum u retku. Ako je uvjet zadovoljen, u *tmp* listu dodaje se dva puta *data* i povećava brojač za jedan. U slučaju da uvjet nije zadovoljen znači da je pronađena druga vrijednost koju se sprema na isti način kao i u prethodnom slučaju samo što ovdje postoji uvjet da u *data* nije spremljeno „Dolasci“ i „Noćenja“:

```
# druga tablica
if d == 1:
    if data in date_2018 and c2018 < 1:
        for i in range(2):
            tmp[d].append(data)
        c2018 += 1
    else:
        if tmp[d] is None:
            tmp[d] = [data]
        else:
            if data != "Dolasci" and data != "Noćenja":
                tmp[d].append(data)
```

Za drugi element liste provjerava se da li je u *data* spremljeno „Dolasci“, te ako je, najprije se u *tmp* dodaje prazan string i zatim tri puta vrijednost iz *data*. Ako je spremljeno „Noćenja“ u *tmp* se dodaje tri puta vrijednost iz *data* pa onda prazan string. Ako je *data* u popisu datuma za 2017. godinu, najprije se poveća brojač *c2017* za jedan i zatim provjeri ako je vrijednost brojača jedan te onda dodaje prazan string pa vrijednost iz *data* u *tmp* listu, inače se u *tmp* listu spremi podatak koji je pronađen:

```
# treća tablica
if d == 2:
    if data == "Dolasci":
        tmp[d].append("")
        for i in range(3):
            tmp[d].append(data)
    elif data == "Noćenja":
        for i in range(3):
            tmp[d].append(data)
        tmp[d].append("")

    elif data in date_2017:
        c2017 += 1
        if c2017 == 1:
            tmp[d].append("")
            tmp[d].append(data)
        else:
            tmp[d].append(data)
```

Ako je u *data* spremljeno „indeksi“, povećava se brojač *cIdx* za jedan. Ako je vrijednost brojača 2, u *tmp* se dodaje vrijednost iz *data* i tri puta prazan string, inače se spremi vrijednost iz *data*. Ako je u *data* spremljeno „Ukupno“, „Total“, „Domaći turisti“ ili „Strani turisti“, tada se u *tmp* dodaje *data* pa zatim prazan string:

```
elif data == "indeksi":
    cIdx += 1
    if cIdx == 2:
        tmp[d].append(data)
        for i in range(3):
            tmp[d].append("")
    else:
        tmp[d].append(data)


elif data == "Ukupno" or data == "Total" or
data == "Domaći turisti" or data == "Strani turisti":
    tmp[d].append(data)
    tmp[d].append("")
```

Kada je u *data* spremljena vrijednost „Camping sites4)“ u *tmp* se dodaje ta vrijednost. Ako je vrijednost brojača *cCs* jednaka 0, tada se dodaje dva puta prazan string u *tmp* pa poveća vrijednost brojača za jedan. Ovo je potrebno jer u retku poslije navedene vrijednosti nedostaju dvije ćelije i ovako ih se umjetno dodaje:

```
elif data == "Camping sites4)":
    tmp[d].append(data)
    if cCs == 0:
        for i in range(2):
            tmp[d].append("")
    cCs += 1
```

Ako je u *data* spremljena vrijednost „Domestic tourists“ ili „Foreign tourists“, u *tmp* se dodaje *data* i zatim prazan string. Kada niti jedan od prethodno navedenih uvjeta nije zadovoljen, provjerava se da li je u *tmp* spremljena vrijednost *None* te ako je na tu se poziciju sprema lista sa vrijednosti *data*, inače se provjeri da li vrijedi postavljeni uvjet i spremi vrijednost iz *data* u listu na poziciji *d*:

```
elif data == "Domestic tourists" or data == "Foreign tourists":
    tmp[d].append(data)
    tmp[d].append("")

else:
    if tmp[d] is None:
        tmp[d] = [data]
    else:
        if data != "Dolasci" and data != "Noćenja" and data not in date_2017
        and data != "indeksi":
            tmp[d].append(data)
```

Za treći element se provjerava da li je u *data* riječ „Individualno“ ili „Organizirano“, te ako je dodaje se šest puta u *tmp* listu. Ako je spremljena neka druga riječ postupak je isti kao i u prethodnim slučajevima uz neke izmijenjene uvjete u drugoj *else* grani:

```
# četvrta tablica
if d == 3:
    if data == "Individualno" or data == "Organizirano":
        for i in range(6):
            tmp[d].append(data)
    else:
        if tmp[d] is None:
            tmp[d] = [data]
        else:
            if data != "dolasci" and data != "noćenja":
                tmp[d].append(data)
```

Za četvrti element imamo tri uvjeta – provjeravamo nalazi li se *data* u listi *indices* koja sadrži string „Indeksi mjesec godina“ i ako da taj string se dodaje tri puta u *tmp*. Ako je u *data* spremljeno „ukupno“, onda se u *tmp* tri puta dodaje prazan string pa zatim vrijednost iz *data*. Ako je u *data* spremljeno „Domaći“, „Strani“ ili „Ukupno“ u *tmp* se dodaje string iz *data* spojen sa stringom iz *url_date* koji sadrži mjesec i godinu u kojoj je izdana publikacija. To je potrebno da bi se moglo razlikovati kojem mjesecu i godini pripadaju podaci navedenih stupaca. *Else* grana i u ovom slučaju služi za spremanje ostalih pronađenih vrijednosti:

```
# peta tablica
if d == 4:
    if data in indices:
        for i in range(3):
            tmp[d].append(data)
    if data == "ukupno":
        for i in range(3):
            tmp[d].append("")
        tmp[d].append(data)
    if data == "Domaći" or data == "Strani" or data == "Ukupno":
        tmp[d].append(data + url_date)
    else:
        if tmp[d] is None:
            tmp[d] = [data]
        else:
            if data not in indices:
                tmp[d].append(data)
```

Za zadnji element liste provjerava se da li je u *data* spremljeno „Dolasci“ ili „Noćenja“ te ako je u *tmp* se dodaje četiri puta. Kada *data* sadrži riječ „muškarci“ provjeri se da li je vrijednost brojača *cm* jednaka 0 i ako je u *tmp* se dodaje prazan string. Brojač se poveća za jedan i u *tmp* se dodaje vrijednost iz *data* dva puta. Kada je u *data* spremljen string „žene“, brojač *cw* se

poveća za jedan i u *tmp* se dva puta dodaje vrijednost, a kada brojač dođe do vrijednosti 2, u *tmp* se dodaje prazan string:

```
# šesta tablica
if d == 5:
    if data == "Dolasci"  or data == "Noćenja":
        for i in range(4):
            tmp[d].append(data)

    elif data == "muškarci":
        if cm == 0:
            tmp[d].append("")
        cm += 1
        for i in range(2):
            tmp[d].append(data)

    elif data == "žene":
        cw += 1
        for i in range(2):
            tmp[d].append(data)
        if cw == 2:
            tmp[d].append("")
```

Ako je u *data* spremljena vrijednost „domaći“ ili „strani“, vrijednost brojača *ct* poveća se za jedan. Kada je vrijednost brojača 1, u *tmp* se dodaje prazan string i zatim vrijednost iz *data* kojoj dodajemo vrijednost iz varijable *url_date* koja iz linka stranice uzima datum i godinu. Dodavanje datuma i godine je potrebno da bi se moglo odrediti kojem mjesecu i godini pripada navedena vrijednost. Ako je vrijednost brojača 8, tada se najprije dodaje vrijednost iz *data* spojena sa *url_date* te zatim prazan string. U svim ostalim slučajevima dodaje se pronađena vrijednost koja nije zadovoljila prethodne uvjete:

```
elif data == "domaći" or data == "strani":
    ct += 1
    if ct == 1:
        tmp[d].append("")
        tmp[d].append(data + url_date)
    elif ct == 8:
        tmp[d].append(data + url_date)
        tmp[d].append("")
    else:
        tmp[d].append(data + url_date)

else:
    if tmp[d] is None:
        tmp[d] = [data]
    else:
        tmp[d].append(data)
```

Vrijednosti iz varijable *tmp* spremaju se u varijablu *all_data*. Kako tablice imaju različiti broj stupaca potrebno je odrediti koji redovi pripadaju kojem stupcu. Nulti, prvi, drugi i peti element liste ima deset ćelija u retku tablice te se brojač *addTd* zaustavlja kada vrijednost bude 10 i ponovno počinje od vrijednosti 0. Kada je vrijednost brojača jednaka 0, u *tr_data* dodaje se lista sa vrijednosti *data* i poveća vrijednost brojača. U slučaju bilo koje druge vrijednosti u prethodnu listu dodaje se vrijednost iz *data* i provjerava da li je vrijednost brojača 10 te ako je postavlja se na 0 da bi se mogao spremiti sljedeći redak tablice. Treći i četvrti element *all_data* liste imaju različit broj redaka u tablici, treći ima 14, a četvrti 8 te je za njih napisan poseban uvjet koji podatke raspoređuje na isti način kao i u prethodnom slučaju:

```
for d in range(len(all_data)):
    for data in all_data[d]:
        if d in [0, 1, 2, 5]:
            if(addTd == 0):
                tr_data.append([data])
                addTd += 1
            else:
                tr_data[-1].append(data)
                addTd += 1

                if(addTd == 10):
                    addTd = 0
        if d == 3:
            if(addTd == 0):
                tr_data.append([data])
                addTd += 1
            else:
                tr_data[-1].append(data)
                addTd += 1

                if(addTd == 14):
                    addTd = 0
        if d == 4:
            if(addTd == 0):
                tr_data.append([data])
                addTd += 1
            else:
                tr_data[-1].append(data)
                addTd += 1

                if(addTd == 8):
                    addTd = 0
```

U funkciji *edit_tables()* najprije se odredi koja lista elemenata (redak) pripada kojoj tablici. To se postiže na način da se iterira lista *tr_data* i provjeri da li se u unutarnjoj listi nalazi string „Dolasci“, „Individualno“ ili „Domaći“ + *url_date* te ako se nalazi u *tmp* listu spremimo *tr_data[data]* listu kao element *tmp* liste, a u *else* grani se dodaju liste u prethodnu listu koja je zadovoljila uvjet iz *if* grane:

```
url_date = " " + url[-11:-4].replace("_", " ")
tmp = []
for data in range(len(tr_data)):
    if("Dolasci" in tr_data[data] or "Individualno" in tr_data[data]
    or "Domaći" + url_date in tr_data[data]):
        tmp.append([tr_data[data]])
    else:
        tmp[-1].append(tr_data[data])

tr_data = tmp
```

Zatim se zaglavlje tablice objedinjuje u jedan redak na način da za nulti, prvi, drugi i četvrti element liste *tr_data* i prvi element unutarnje liste koji je također lista spojimo svaki element sa onim na sljedećoj poziciji, a vrijednost elemenata liste sa sljedeće pozicije postavi se na prazan string. Za treći i peti element spoji se svaki element prve unutarnje liste sa elementima sljedeće dvije liste, a elementi tih dviju lista postave se na prazan string:

```
for data in range(len(tr_data)):
        for d in range(len(tr_data[data])):
            for k in range(len(tr_data[data][d])):
                if data in [0, 1, 2, 4] and d == 0:
                    tr_data[data][d][k] = tr_data[data][d][k] + " " + tr_data[data][d+1][k]
                    tr_data[data][d+1][k] = ""
                if data in [3, 5] and d == 0:
                    tr_data[data][d][k] = tr_data[data][d][k] + " " + tr_data[data][d+1][k]
                            + " " + tr_data[data][d+2][k]
                    tr_data[data][d+1][k] = ""
                    tr_data[data][d+2][k] = ""
```

Za spremanje podataka u datoteku poziva se funkcija *save_to_files()* u kojoj se prvo u varijablu *titles* spremaju naslovi tablica pomoću funkcije *get_title()*. Funkcija *get_title()* sprema pronađene naslove u *titles* listu te je modificirana tako da je dodan još jedan argument koji se prosljeđuje u listu – *url* varijabla u koju je spremljen link stranice. Naslov druge, četvrte i šeste tablice izluči se na način da se u *html_soup* varijabli pronađu sva pojavljivanja *bold tag-a* i spremi u varijablu *bolds*. Iterira se kroz listu *bolds* i za svaki element pronalazi *span tag*. Brojač *c* poveća se za jedan i pronađeni *span tag* spremi se u varijablu *span*. Ako je vrijednost brojača

13, 14 ili 15 pronađeni su traženi naslovi i sprema ih se u *spans* listu te se vrijednost brojača postavlja na 0:

```
# 2., 4., 6. naslov
bolds = html_soup.find_all('b')
for bold in bolds:
    if bold.find('span', {'lang':'EN-US'}):
        c += 1
        span = bold.find('span', {'lang':'EN-US'})
        if c in [13, 14, 15]:
            spans.append(span.text)
c = 0
```

Prvi, treći i peti naslov nalaze se unutar *spans tag-a* koji se izlučuje i sprema u *spans2* varijablu. Iterira se kroz sve pronađene *spans tag-ove* spremljene u *spans2* listi te se u *temp* listu spremaju naslovi koji su na 14., 25. i 52. poziciji u listi:

```
# 1., 3., 5. naslov
spans2 = html_soup.find_all('span', {'lang':'EN-GB'})
for i in range(len(spans2)):
    if i in [14, 25, 52]:
        temp.append(spans2[i].text)
spans2 = temp
```

Dvije liste koje sadrže naslove spajaju se u jednu listu pomoću *zip()* funkcije koja najprije uzima jedan element prve pa jedan element druge liste. Elementi listi *spans2* i *spans* spremaju se u listu *titles*. U listi *titles* četiri elementa odnosno naslova ne sadrži mjesec i godinu te im se dodaje dio string-a od četvrtog elementa koji ima mjesec i godinu u naslovu. Svim elementima zamijene ne nepotrebni znakovi i lista sada sadrži naslove uređene i spremne za korištenje:

```
'''
spajanje dvije liste u jednu
'''
for s1, s2 in zip(spans2, spans):
    titles.append(s1)
    titles.append(s2)
'''
4 naslova ne sadrže mjesec i godinu pa se to dodaje od stringa naslova
na 4. poziciji
'''
for title in range(len(titles)):
    if title != 4:
        titles[title] = titles[title] + titles[4][-17:]
    titles[title] = titles[title].replace(u'\xa0', u' ').replace("  ","")
    .replace("   "," ").replace("\r"," ").replace("\n"," ").lower()[2:].lstrip()
```

Sad kada su i naslovi spremljeni unutar *save_to_files()* funkcije u *titles* varijablu spremaju se podaci u .csv datoteke u mapu „dolasci turista“. Sa *for* petljom prolazi se kroz *tables* listu, otvori se datoteka u koju će se spremiti podaci te se sa još jednom *for* petljom prolazi kroz listu koja sadrži liste (retke pojedine tablice). Provjerava se da li svi elementi liste nisu prazni znakovi te

ako nisu u .csv datoteku sprema se lista kao jedan redak tablice. Na ovaj način se eliminiraju prazne liste koje su u publikaciji bile prazni redovi:

```
for data in range(len(tables)):
    with open("C:\\Users\\Sanja\\Documents\\FAKS\\diplomski rad\\data\\dolasci turista\\"
              + titles[data] + ".csv",'w', newline='', encoding='utf-8') as csv_file:
        for tr in range(len(tables[data])):
            if all('' == s for s in tables[data][tr]) is False:
                table_writer = csv.writer(csv_file, delimiter=',')
                table_writer.writerow(tables[data][tr])
```

Skripta *tourism_scraper_2017.py* pomoću koje se preuzimaju tablice Dolasci i noćenja turista, Republika Hrvatska, NKPJS 2012. – 2. razina, županije, gradovi, općine i Dolasci i noćenja turista prema dobnim skupinama sadrži određene preinake koda iz skripte *tourism_scraper.py*.

Tablice koje se žele preuzeti nalaze se na šestom i sedmom elementu liste *tables* u kojoj su spremljeni svi *table tag-ovi* pronađeni na stranici. Postupak izlučivanja podataka iz tablice je isti kao u prethodnoj skripti sa izmjenom da se traži podatak unutar *p tag-a* umjesto *span* i sprema se u *all_data* listu:

```
tables = html_soup.find_all('table', attrs={'class':'MsoNormalTable'})
tables = tables[6:8]
```

U ovom se slučaju peta tablica nalazi na nultoj poziciji, a šesta na prvom poziciji liste *all_data*. U ovoj funkciji izmijenjen je dio koda koji se odnosi na dodavanje naziva kako bi svi reci imali jednaki broj stupaca i to za šestu tablicu.

Na prvoj poziciji liste *all_data* provjerava se da li je u *data* spremljen string „DolasciArrivals“ ili „NoćenjaNights“. Ako je iterira se kroz listu *arrival* i ako se pronađe neki od elemenata liste u varijabli *data* onda ga se zamijeni za prazan string. Na taj način u *data* ostaje samo riječ „Dolasci“ ili „Noćenja“:

```
# šesta tablica
if d == 1:
    if data == "DolasciArrivals"  or data == "NoćenjaNights":
        arrival = ["Arrivals", "Nights"]
        for a in arrival:
            if a in data:
                data = data.replace(a, "")
        for i in range(4):
            tmp[d].append(data)
```

Ako je u *data* spremljen string „muškarciMen“ kao i u prethodnoj skripti provjerava se da li je vrijednost brojača *cm* jednaka 0 i ako je u *tmp* se dodaje prazan string i poveća vrijednost brojača za jedan. Zatim se u *tmp* dva puta dodaje vrijednost iz *data* ali u ovom slučaju se dio stringa koji sadrži riječ „Men“ zamijeni na prazan string. Sličan postupak je i kod pronalaženja stringa „ženeWomen“ gdje se dio stringa „Women“ zamijeni za prazan string:

```
elif data == "muškarciMen":
    if cm == 0:
        tmp[d].append("")
    cm += 1
    for i in range(2):
        tmp[d].append(data.replace("Men", ""))

elif data == "ženeWomen":
    cw += 1
    for i in range(2):
        tmp[d].append(data.replace("Women", ""))
    if cw == 2:
        tmp[d].append("")
```

Kada je u *data* spremljen string „domaći turistiDomestic tourists“ ili "strani turistiForeign tourists" brojač *ct* povećamo za jedan. Zatim se ukloni engleski naziv iz stringa na način da se iterira lista *tourist* u koju je spremljen taj naziv i sa funkcijom *replace()* zamijenimo taj dio stringa sa praznim stringom i na taj način ga obrišemo. Kada je vrijednost brojač *ct* jedanaka 1 ili 8, kao i u prethodnoj skripti na string u varijabli *data* dodaje se datum i godina iz varijable *url_date*. U svim ostalim slučajevima podatak iz *data* spremimo u *tmp* listu:

```
elif data == "domaći turistiDomestic tourists" or data == "strani turistiForeign tourists":
    ct += 1
    tourist = ["Domestic tourists", "Foreign tourists"]
    for t in tourist:
        if t in data:
            data = data.replace(t, "")
```

Obje tablice imaju isti broj stupaca kao u publikaciji za 2018. godinu te se oni određuju na isti način kao u prethodnoj skripti.

Funkcije *edit_tables()* identična je kao u prethodnoj skripti ali u ovoj kod određivanja koja lista elemenata pripada kojoj tablici ima jedan uvjet manje te se provjerava da li se u listi *tr_data[data]* nalazi string „Dolasci“ ili „Domaći“ + *url_date*.

Funkcija za spremanje datoteka, *save_to_files()* sada sprema samo dvije te je u *get_title()* funkciji potrebno izlučiti samo dva naziva datoteka.

Funkcija *get_title()* sadrži jedan parametar više i to je link stranice, a koristi se da bi se mogao provjeriti mjesec za koji se preuzima publikacija. U *date* varijablu izlučuje se datum i godina da bi se moglo dodati u naslov kojem nedostaje taj podatak. Datum i godina izlučuje se tako da se pronađu svi *h1 tag-ovi* i uzme prvi pronađeni te izluči samo string od sedme pozicije na dalje te izbace zadnja dva znaka:

```
h1 = html_soup.find_all('h1')[0].text[7:]
date = h1[:-2].lower()
```

Naslovi se nalazi u istim *tag-ovima* kao i u prethodnoj skripti te se sada samo izluče sa odgovarajuće pozicije iz liste. Peti naslov nalazi se na 60. poziciji u listi *spans2* i sprema se u listu *temp*. Šesti naslov se nalazi na 12. poziciji. Prvo se peti naslov iz *temp* liste dodaje u listu *titles* pa zatim šesti naslov:

```
# 5. naslov
spans2 = html_soup.find_all('span', {'lang':'EN-GB'})
for i in range(len(spans2)):
    if i == 60:
        temp.append(spans2[i].text)

# 6. naslov
bolds = html_soup.find_all('b')
for bold in bolds:
    if bold.find('span', {'lang':'EN-US'}):
        span = bold.find('span', {'lang':'EN-US'})
        spans.append(span.text)

titles = temp
titles.append(spans[12])
```

Šesti naslov ne sadrži mjesec i godinu za prva tri mjeseca publikacija te se to dodaje iz naslova publikacije. Ako je u varijabli *url_date* spremljena vrijednost manja od 4 (za prva tri mjeseca) tada se uzima naslov sa prve pozicije liste *titles* i očisti od nepotrebnih znakova te mu se dodaje datum i godina iz varijable *date*. Ako je vrijednost veća ili jednaka 4 tada se naslov samo očisti od nepotrebnih znakova. Naslov pete tablice samo se očisti od nepotrebnih znakova i spremi u listu *titles*:

```
if url_date < 4:
    title = titles[1].replace(u'\xa0', u' ').replace("  ","").replace("    "," ")
    .replace("\r"," ").replace("\n"," ").lower()[2:].lstrip()
    titles[1] = title + date
else:
    titles[1] = titles[1].replace(u'\xa0', u' ').replace("  ","").replace("    "," ")
    .replace("\r"," ").replace("\n"," ").lower()[2:].lstrip()

titles[0] = titles[0].replace(u'\xa0', u' ').replace("  ","").replace("    "," ")
.replace("\r"," ").replace("\n"," ").lower()[2:].lstrip()
```

Publikacija o kružnom putovanju stranih brodova sadrži tri tablice:

- Kružna putovanja stranih brodova,
- Kružna putovanja stranih brodova, dani boravka i putnici prema zastavi broda,
- Kružna putovanja stranih brodova, dani boravka i putnici po mjesecima.

Preuzet će se druga i treća tablica, te je za svaku napisana posebna skripta.

Skripta *vessels_scraper_1.py* sadrži kod za preuzimanje tablica Kružna putovanja stranih brodova, dani boravka i putnici prema zastavi broda. Tablica sadrži podatke za tekući mjesec i godinu te je potrebno preuzeti publikaciju za svaki mjesec kroz 2017. i 2018. godinu.

U funkciji *get_tables()* inicijalizirane su dvije liste koje sadrže podatke iz zaglavlja tablice. Lista *date* sadrži nazive mjeseca zapisane rimskim brojevima, a lista *date_to* mjesece zapisane kroz vremenski period, npr. I. – V.:

```
date = ["V.", "VI.", "VII.", "VIII.", "IX.", "X.", "XI.", "XII."]
date_to = ["I. - V.", "I. - VI.", "I. - VII.", "I. - VIII.", "I. - IX.",
           "I. - X.", "I. - XI.", "I. - XII."]
```

U listi *tables* uzima se treći element odnosno tablica i pronalazi podatak unutar ćelije kao i u prethodnim skriptama. Provjerava se da li je u *data* spremljeno „Putovanja“, „Dani boravka brodova“, „Dani boravka“ ili „Putnici na brodu“ te ako je sprema se dva puta u listu *all_data*. Ako se vrijednost spremljena u *data* nalazi u listi *date* i brojač *addV* ima vrijednost nula, u *all_data* sprema se prazan string pa vrijednost iz *data* i poveća vrijednost brojača za jedan. Ako je vrijednost iz *data* u listi *date_to*, brojač *addIV* poveća se za jedan te se provjerava da li je vrijednost brojača jednaka tri i ako je u listu *all_data* doda se vrijednost iz *data* i prazan string,

a ako nije dodaje se samo vrijednost iz *data*. Kada nije zadovoljen niti jedan od uvjeta vrijednost iz *data* sprema se u listu:

```
if(data == "Putovanja" or data == "Dani boravka brodova" or
   data == "Dani boravka" or data == "Putnici na brodu"):
    for i in range(2):
        all_data.append(data)
elif(data in date and addV == 0):
   all_data.append("")
   all_data.append(data)
   addV += 1
elif(data in date_to):
   addIV += 1
   if addIV == 3:
      all_data.append(data)
      all_data.append("")
   else:
      all_data.append(data)
else:
    all_data.append(data)
```

Da bi se odredili reci u tablici brojač *addTd* dolazi do vrijednosti osam te se tada ponovo postavlja na nula jer svaki redak tablice ima osam stupaca. Postupak određivanja redaka je isti kao u skripti *cross_border_scraper.py*.

U funkciji *edit_tables()* da bi se zaglavlje tablice objedinilo u jedan redak provjerava se da li je u listi *tr_data[data]* sadržan string „Putovanja“ te ako je svi elementi te liste se spajaju sa elementima naredne liste, a elementi naredne liste se postavljaju na prazan string da bi se uklonili suvišni podaci. Zatim se brišu liste koje sadrže prazne stringove jer predstavljaju prazne retke tablice, a postupak je isti kao i u skripti *cross_border_scraper.py*:

```
for data in range(len(tr_data)):
    if("Putovanja" in tr_data[data]):
        for d in range(len(tr_data[data])):
            tr_data[data][d] += " " + tr_data[data+1][d]
            tr_data[data+1][d] = ""
```

save_to_files() funkcija ista je kao u prethodnoj skripti *tourism_scraper_2017.py*, a ova i sve ostale datoteke spremaju se u mapu *kruzna putovanja*. Da bi se dobio naziv tablica i koristio kao naziv datoteka u funkciji *get_title()* pronalaze se naslovi unutar *span taga* kao i u prethodnoj skripti. Obzirom da se u nekim publikacijama nalaze na drugim mjestima pomoću *url_date* varijable u kojoj je spremljen mjesec za koji je publikacija izdana provjerava se da li je publikacija izdana za peti ili sedmi mjesec. Ako je u varijablu *title* spremamo element sa devete

pozicije liste *spans*, a ako nije, elemente sa devete i desete pozicije spajamo i spremamo u varijablu *title*:

```
if url_date == 1 or url_date == 3:
    title = spans[9].replace(u'\xa0',' ').replace("  ","").replace("    "," ")
    .replace("\r"," ").replace("\n"," ").lower()[2:].lstrip()
else:
    spans = spans[9:11]
    for span in spans:
        title += span.replace(u'\xa0',' ').replace("  ","").replace("    "," ")
        .replace("\r"," ").replace("\n"," ").lower()[2:].lstrip()
```

Skripta *vessels_scraper_2.py* sadrži kod za preuzimanje tablice Kružna putovanja stranih brodova, dani boravka i putnici po mjesecima i to za 2017./2018. godinu. Preuzima se zadnja publikacija izdana za 2018. godinu jer se u njoj nalaze podaci za sve mjesece u godini, a link na publikaciju spremljen je u varijabli *url*.

U *get_tables()* funkciji uzima se četvrti element liste *tables* u kojoj su spremljeni svi *table tag-ovi* pronađeni na stranici publikacije. Kada se pronađe i očisti podatak u ćeliji, provjerava se da li je u *data* spremljeno „Putovanja“, „Dani boravka brodova“ ili „Putnici na brodu“. Ukoliko je u listu *all_data* ta vrijednost se sprema tri puta. Ako je u *data* spremljen string „2017.“ i vrijednost brojača *c2017* jednaka nula, u *all_data* sprema se prazan string i zatim vrijednost iz *data* te se poveća vrijednost brojača za jedan. U slučaju da vrijednost spremljena u *data* ne zadovoljava niti jedan od uvjeta tada se ona sprema u listu *all_data*:

```
if(data == "Putovanja" or data == "Dani boravka brodova"
   or data == "Putnici na brodu"):
    for i in range(3):
        all_data.append(data)

elif (data == "2017." and c2017 == 0):
    all_data.append("")
    all_data.append(data)
    c2017 += 1
else:
    all_data.append(data)
```

Da bi se odredili reci u tablici brojač *addTd* dolazi do vrijednosti deset te se tada ponovo postavlja na nula jer svaki redak tablice ima deset stupaca. Postupak određivanja redaka je isti kao u prethodnoj skripti.

Funkcija *edit_tables()* ista je kao u prethodnoj skripti. Jedina izmjena nalazi se u funkciji *get_title()* u kojoj se uzima jedanaesti element liste *spans* i čisti od nepotrebnih znakova te se primijeni funkcija *save_to_files()*.

Zaključno sa ovom skriptom preuzete su sve tablice iz publikacija DZS-a i mogu se izlučiti tablice iz PDF formata koje sadrže podatke o turizmu Plitvičkih jezera.

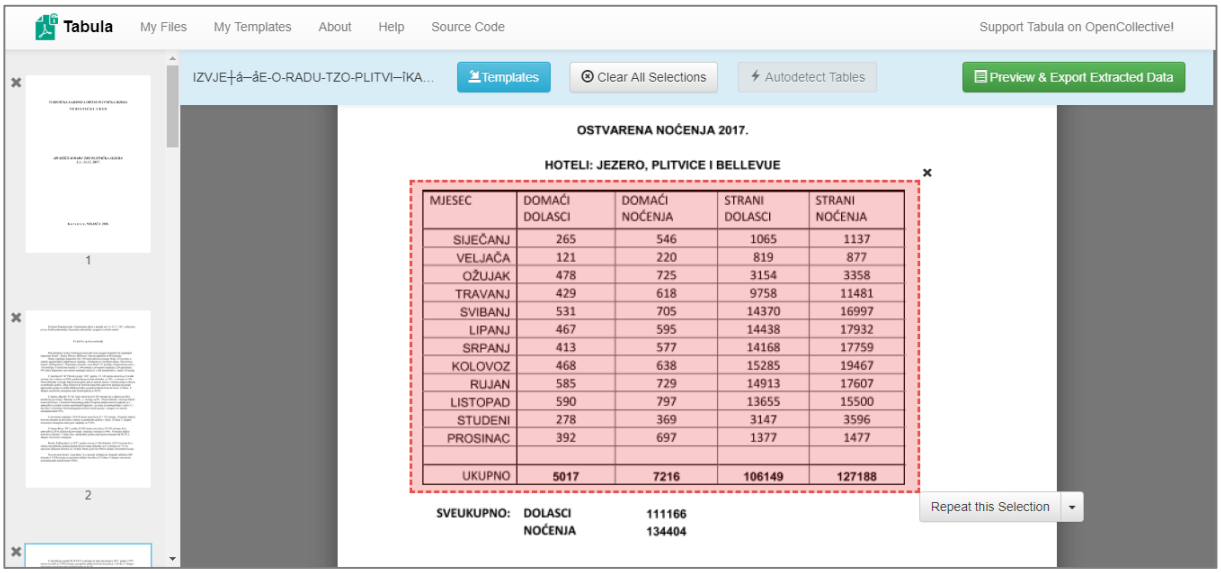
1.2 Ekstrakcija podataka o turizmu Plitvičkih jezera

Kako bi se analizirao broj dolazaka i noćenja turista, iz kojih zemalja turisti dolaze te gdje odsjedaju u posjetu Plitvičkim jezerima preuzeta su Izvješća o radu TZO Plitvička jezera za 2017. i 2018. godinu te jedno izvješće koje sadrži podatke za usporedbu te dvije godine kroz prvih devet mjeseci.

Sva izvješća su spremljena u PDF formatu, a tablice koje će se koristiti za analizu su sljedeće:

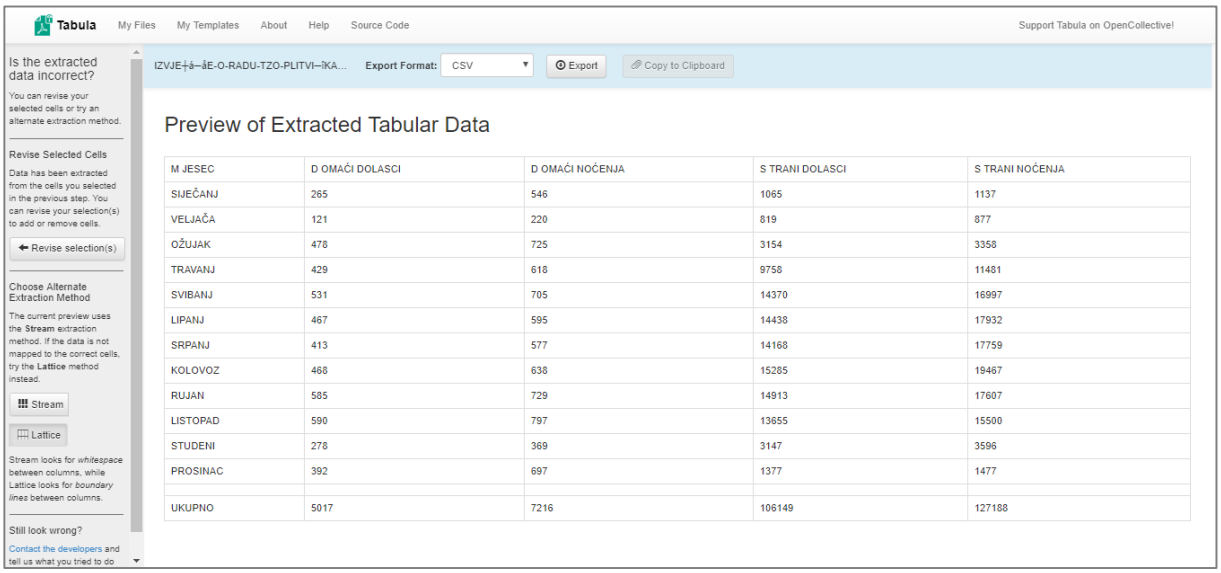
- Ostvarena noćenja po vrstama smještajnih kapaciteta,
- Dolasci i noćenja – turistički promet po smještajnim kapacitetima i državama,
- Dolasci i noćenja posjetitelja po državama,
- Ostvarena noćenja po naseljima/mjesecima,
- Raspoloživi kapaciteti privatnog smještaja po naseljima.

Za izlučivanje tablica iz PDF datoteke koristi se alat Tabula te je u nastavku primjer izlučivanja jedne od tablica. Za korištenje ovog alata potrebno ga je preuzeti na računalo. Za Windows operacijski sustav dostupan je u .zip datoteci. Raspakira se u željeni direktorij i pokrene .exe datoteka te se sam program otvori u web pregledniku. Klikom na *import* gumb otvori se prozor te se odabere datoteka iz koje se želi izlučiti tablica. U ovom slučaju izlučuje se iz datoteke *IZVJEŠĆE-O-RADU-TZO-PLITVIČKA-JEZERA-01.01.-31.12.2017.pdf*. Kada se datoteka učita sa kursorom se označi tablica koju se želi izlučiti i oko nje se pojavi crveni okvir sa isprekidanim rubom kao na Slici 1:



Slika 1 Označavanje željene tablice

Klikom na *Preview & Export Extracted Data* otvara se novi prozor u kojem je vidljiva izlučena tablica (Slika 2). U padajućoj listi *Export Format* odabire se format u kojem će se spremiti izlučena tablica (.csv) i klikom na gumb *Export* datoteka se preuzima na računalo.



Slika 2 Pretpregled izlučene tablice

Sve tablice iz datoteka mape *raw_data* izlučuju se na isti način i spremaju se u mapu *plitvicka jezera*. Pojedine tablice nedostaju te su podaci sintetizirani.

1.3 Čišćenje podataka i spajanje tablica pomoću Python skripte

Svaka publikacija sadržava više tablica i tako za svaki mjesec. Radi jednostavnosti, zbog daljnje izrade grafikona, tablice istih naziva spojene su u jednu tako da se svi podaci kroz dvije godine nalaze u istoj .csv datoteci.

Pojedini podaci su nedostajali te su oni ručno unijeti kako bi sve tablice prije spajanja imale isti broj redaka.

Skripta *concat_files.py* sadrži kôd pomoću kojeg se učitavaju .csv datoteke, spajaju u jednu, brišu nepotrebni stupci tablica te u konačnici spremaju u novu .csv datoteku. U *path* varijabli navede se putanja do mape u kojoj se nalaze datoteke koje se žele učitati. Unutar *glob()* funkcije sa *join()* spoji se putanja do mape sa svim datotekama koje imaju ekstenziju .csv te se sve pronađene putanje spreme u varijablu *filenames*. *file_name* varijabla sadrži naziv datoteka koje će se pretraživati i učitati:

```
'''spremanje imena datoteka'''
path =r'../data/dolasci turista'
filenames = glob.glob(os.path.join(path, "*.csv"))
file_name = "dolasci i noćenja  turista, republika hrvatska, nkpjs 2012. - 2. razina, županije, gradovi,  općine"
```

Pomoću *for* petlje iterira se kroz elemente liste *filenames* i provjeri da li je u listi pronađenih putanja datoteka ona datoteka sa nazivom spremljenim u varijablu *file_name*. Ako je sa *read_csv()* funkcijom učita se sadržaj datoteke, bez zaglavlja, te spremi kao *data frame* u listu *dfs*:

```
'''
učitavanje svih datoteka koje odgovaraju nazivu
spremljenom u file_name
'''
dfs = []

for filename in filenames:
    if file_name in filename:
        dfs.append(pd.read_csv(filename, header=None))
```

U *dfs* listi su sada elementi tablice iz datoteka spremljene kao *data frame-ovi*. Da bi se spojile u jedan *data frame* koristi se *concat()* funkcija kojoj se proslijedi lista *dfs*, sa *axis=1* se naznači da se spaja po stupcima te sa *ignore_index=True* resetira se indeks *data frame-a*. Kako se prva dva stupca svake tablice ponavljaju, potrebno je ukloniti viškove i ostaviti samo prva dva stupca prve tablice. To se napravi na način da se izrade dvije liste *l1* i *l2* koje se popune brojevima sa *for* petljom kojoj odredimo početni i zadnji indeks stupca kojeg se želi obrisati te korak kako bi se izbrisao svaki osmi stupac. Te dvije liste spajaju se u jednu listu – *l*. Sa funkcijom *drop()*

brišu se nepotrebni stupci na način da se kao indeks stupca proslijedi izrađena lista *l*, sa *axis=1* odredi da se brišu stupci te sa *inplace=True* brišu se stupci unutar postojećeg *data frame-a*:

```
'''spajanje svih df u jedan i brisanje dupliranih stupaca'''
all_files = pd.concat(dfs, axis=1, ignore_index=True)
l1 = []
l2 = []

for i in range(8,185,8):
    l1.append(i)

for i in range(9,186,8):
    l2.append(i)

l = l1+l2

all_files.drop(l, axis=1, inplace=True)
```

Kada je nova velika tablica uređena u *filename_save* varijablu sprema se naziv nove datoteke i u *save_file* određuje putanja gdje će se datoteka spremiti. Za putanju se iskoristi postojeća i nadoda naziv nove datoteke. Sa *to_csv()* funkcijom *all_files data frame* sprema se u datoteku *save_file*, ne zapisuju se indeksi retka niti zaglavlje te se naznačuje da se sprema sa *encodingom utf-8*:

```
'''spremanje u csv datoteku'''
filename_save = 'dolasci i noćenja turista, republika hrvatska, nkpjs 2012. -
2. razina, županije, gradovi, općine 2017. i 2018. godine.csv'
save_file = path + '/' + filename_save
all_files.to_csv(save_file, index=False, header=False, encoding='utf-8')
```

Kako se učitavaju različite datoteke potrebno je promijeniti naziv datoteke koja se učitava, putanju do te datoteke i odrediti novi naziv .csv datoteke u koju će se spremiti spojene tablice.

Kada su sve tablice spojene prelazi se na analizu i izradu grafikona, koji će biti prikazani u poglavlju 2.

2 Vizualizacija podataka u Tableau

U ovom poglavlju su obuhvaćene najvažnije vizualizacije pripremljenih i prikupljenih podataka o hrvatskom turizmu izrađene u alatu Tableau. Najprije su prikazani podaci o graničnom prometu gdje se usporede ulasci i izlasci domaćih i stranih turista iz države. Posebno se analizira svaka vrsta graničnog prometa, koji granični prijelazi su bili najprometniji te broj putnika po vrstama vozila. Analiziraju se zračne luke po broju putnika, koliki je broj putnika i dana provedenih na kružnom putovanju prema zastavi broda, broj putovanja, dana boravaka i broj putnika na brodu po mjesecima.

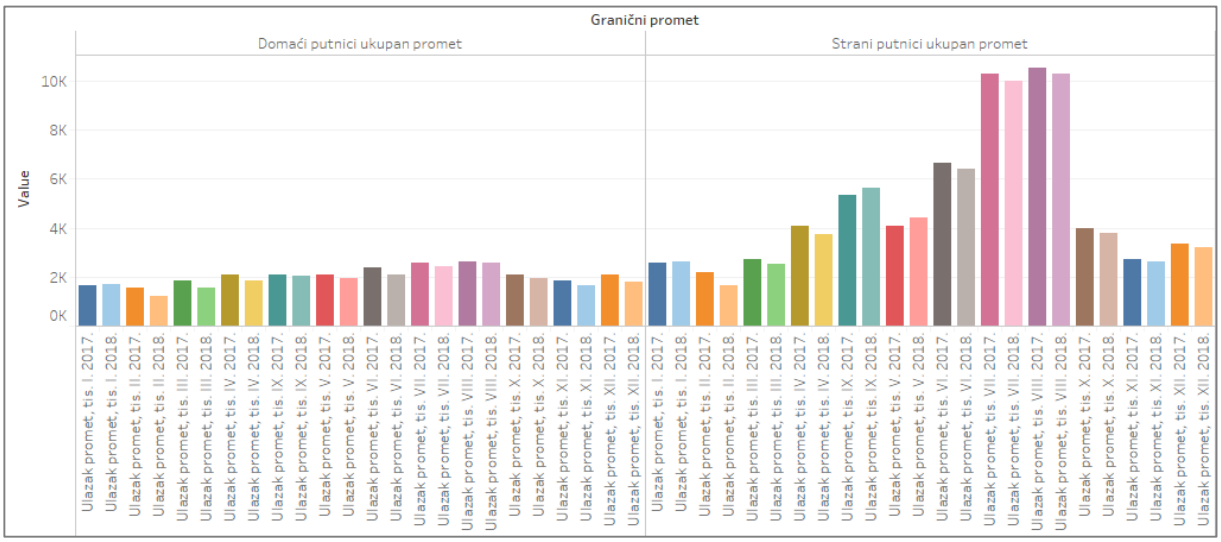
Prikazuje se broj dolazaka i noćenja domaćih i stranih turista, iz koje zemlje je došlo i noćilo najviše turista, u kojoj vrsti smještaja turisti vole odsjesti, način na koji su došli u posjet, u kojem dijelu Hrvatske dolazi najviše turista te kakav je omjer prema dobnim skupinama.

Posebna analiza provodi se za turizam Plitvičkih jezera gdje se analizira broj dolazaka i noćenja domaćih i stranih turista prema vrstama smještajnih kapaciteta, prema zemlji prebivališta, broj posjetitelja nacionalnog parka Plitvička jezera, kakav je omjer broja kreveta i noćenja po naseljima i gdje se nalaze naselja sa najvećim brojem dolazaka i noćenja prikazano na karti i u konačnici analizira se broj raspoloživog kapaciteta privatnog smještaja po naseljima.

2.1 Granični promet i putovanja turista

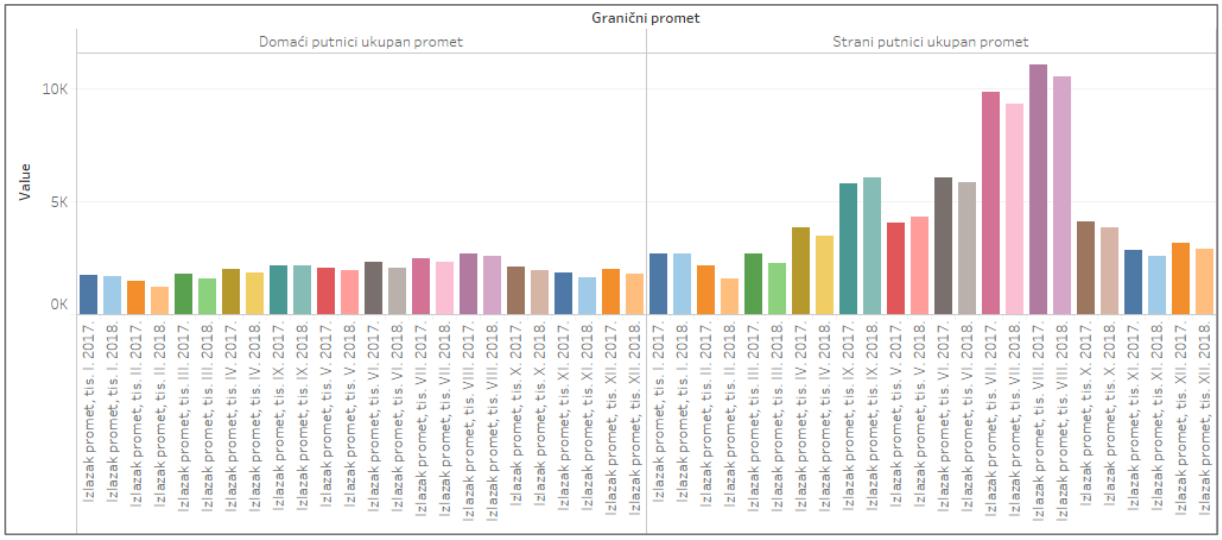
Često se kao pokazatelj dolazaka većeg broja turista uzima broj vozila na graničnim prijelazima. Iz tog razloga analizirat će se granični promet iz više aspekata kako bi se utvrdilo u kojim mjesecima je povećan promet vozila.

U analizi graničnog prometa prema vrsti putnika primjećuje se povećan broj ulazaka stranih putnika u ljetnim mjesecima, točnije u srpnju i kolovozu te i u rujnu. Srpanj i kolovoz smatraju se „srcem sezone“ te je očekivan povećan broj dolazaka stranih turista. U rujnu još dolaze turisti koji žele izbjeći ljetne gužve i uživati u mirnom odmoru. Tamnijom bojom prikazani su podaci iz 2017., a svjetlijom podaci iz 2018. godine. Iz grafikona na Slici 3 je vidljivo da je prisutan pad broja ulazaka u 2018. g. ali nije toliko značajan.



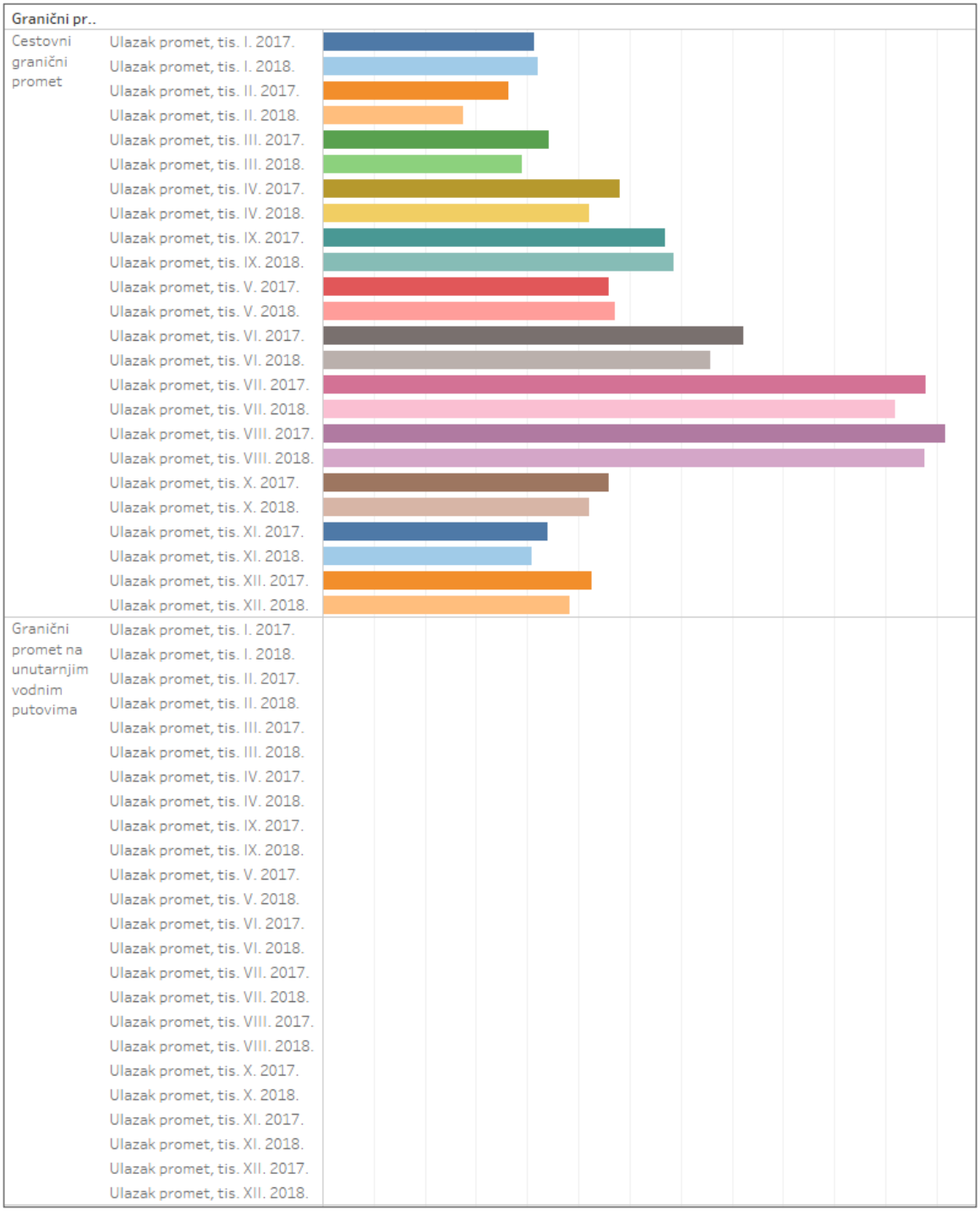
Slika 3 Granični promet po vrstama putnika - ulazak

U istim mjesecima pojačan je broj izlazaka stranih putnika iz zemlje te je daleko veći od izlazaka domaćih putnika kao što se vidi na Slici 4. U tim mjesecima treba računati na velike gužve i kolone na cestama što se obično i dešava svaki vikend u ta tri mjeseca.

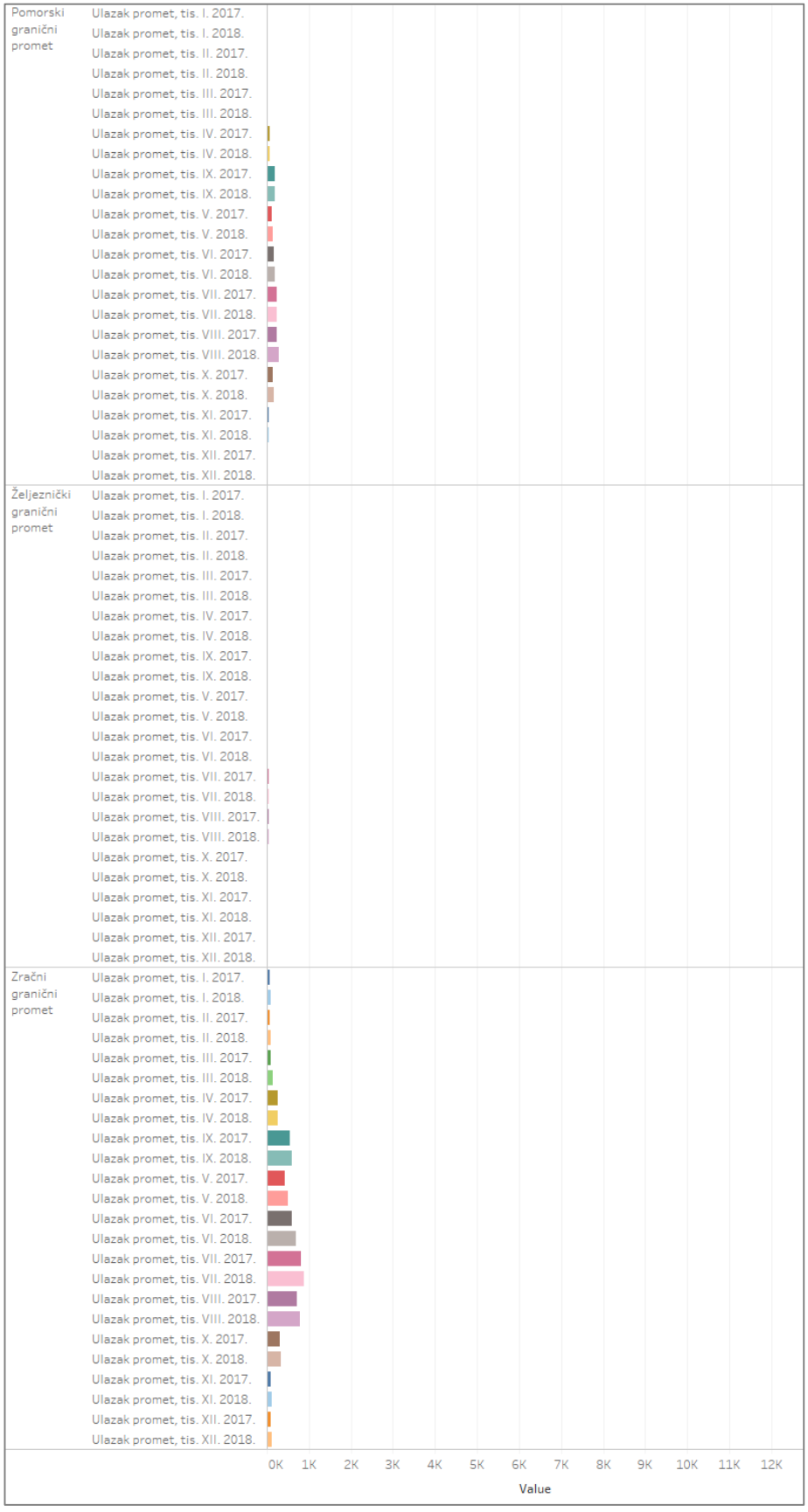


Slika 4 Granični promet po vrstama putnika - izlazak

Od svih vrsta prijevoza najviše se koristi cestovni i zračni. Cestovni promet najgušći je u ljetnim mjesecima – lipnju, srpnju i kolovozu te je 2018. g. u opadanju. U rujnu je također povećan cestovni promet te je 2018. u porastu. Razlog opadanja cestovnog prometa u ljetnim mjesecima je povećanje zračnog prometa koji se poslije cestovnog najviše koristi kao način putovanja (Slika 5 i Slika 6).

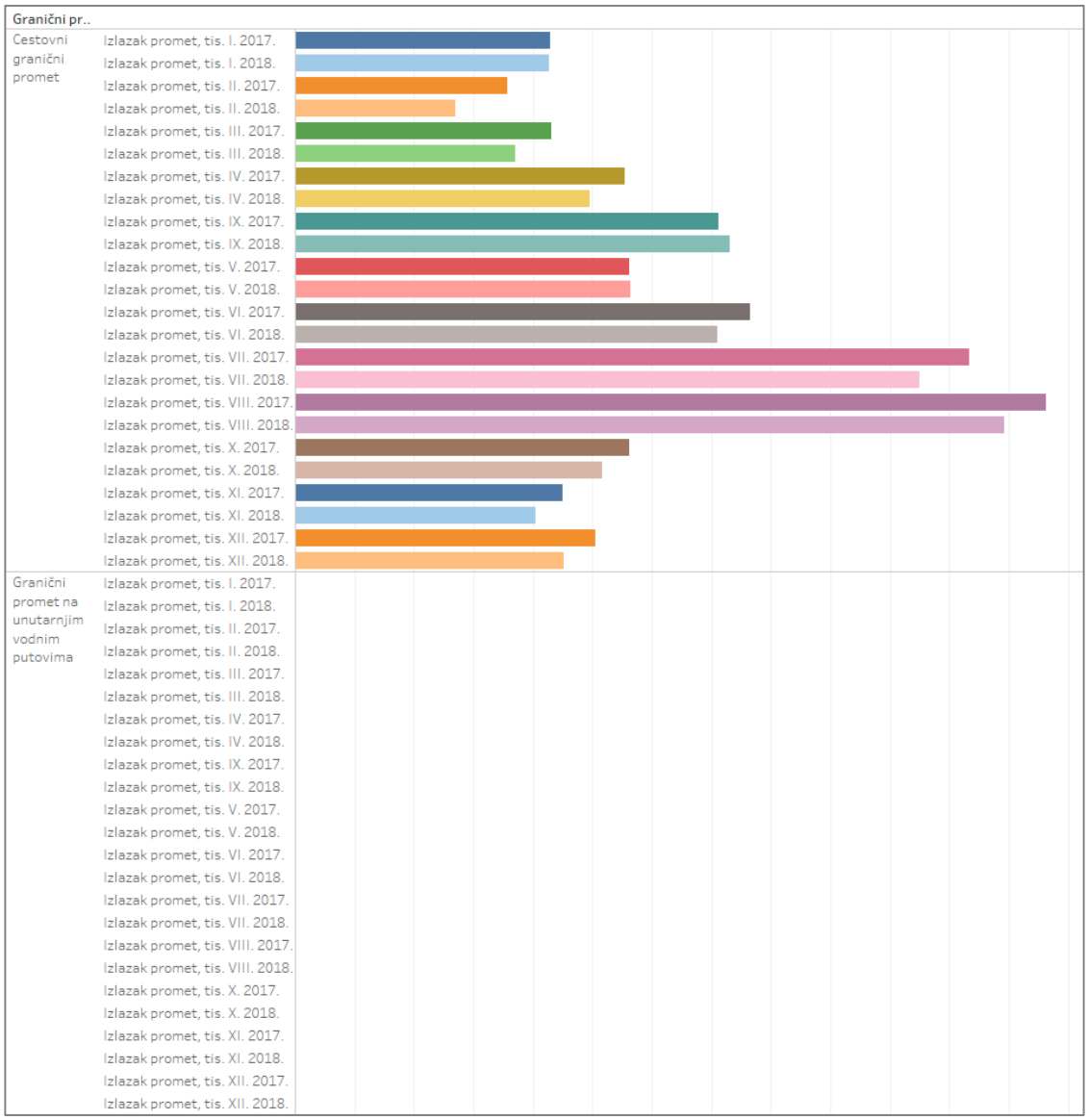


Slika 5 Količina graničnog prometa prema vrstama graničnog prometa - ulazak (1)

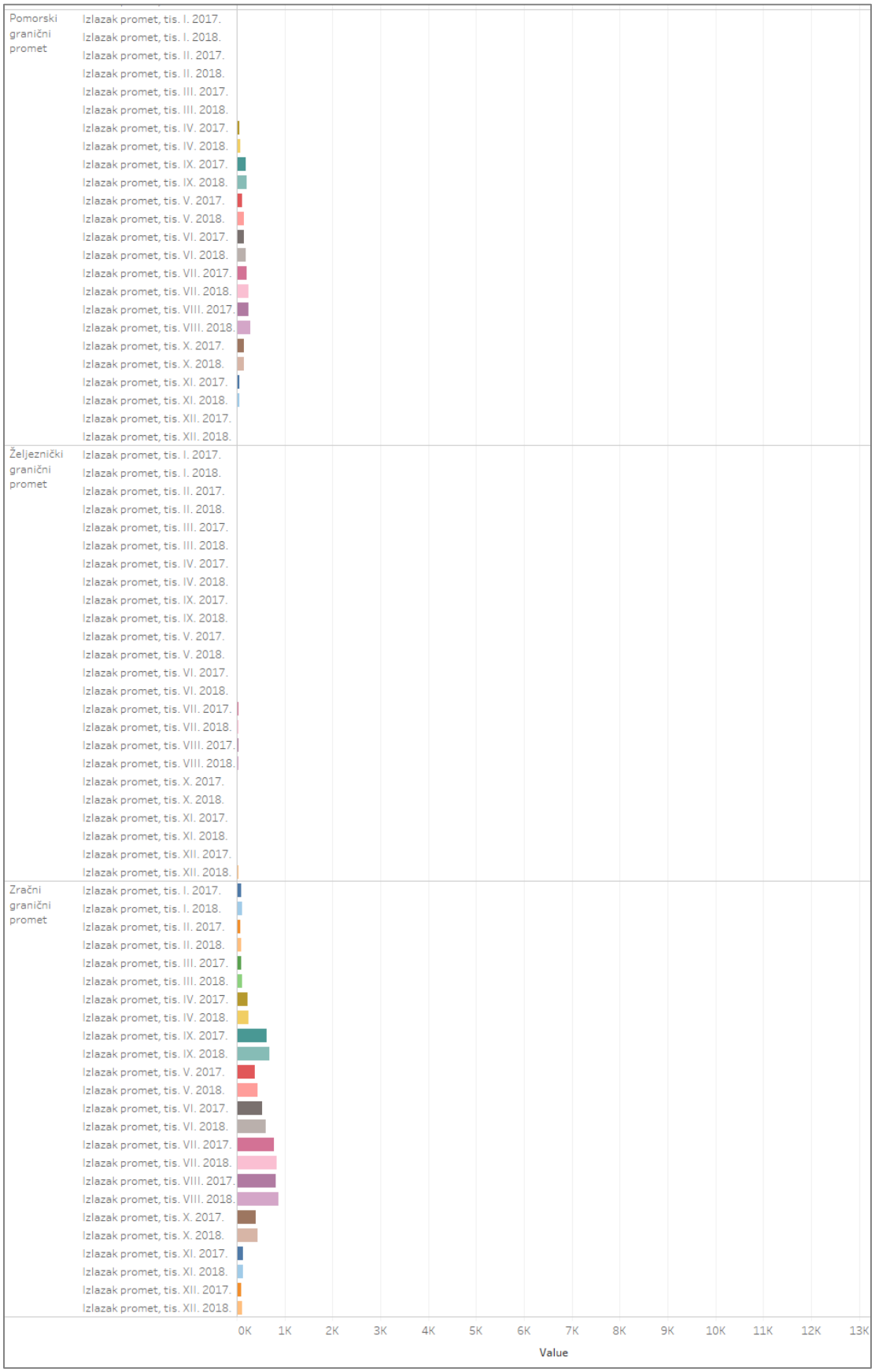


Slika 6 Količina graničnog prometa prema vrstama graničnog prometa - ulazak (2)

Cestovni promet je povećan i u prosincu te travnju zbog zimskih odnosno proljetnih praznika. To je vidljivo i na Slikama 7 i 8 koje prikazuju izlazak putnika iz zemlje. Kada se usporede grafikoni za ulaz i izlaz iz zemlje vidljivo je da je veći broj izlaza nego ulaza, naročito u ljetnim mjesecima.

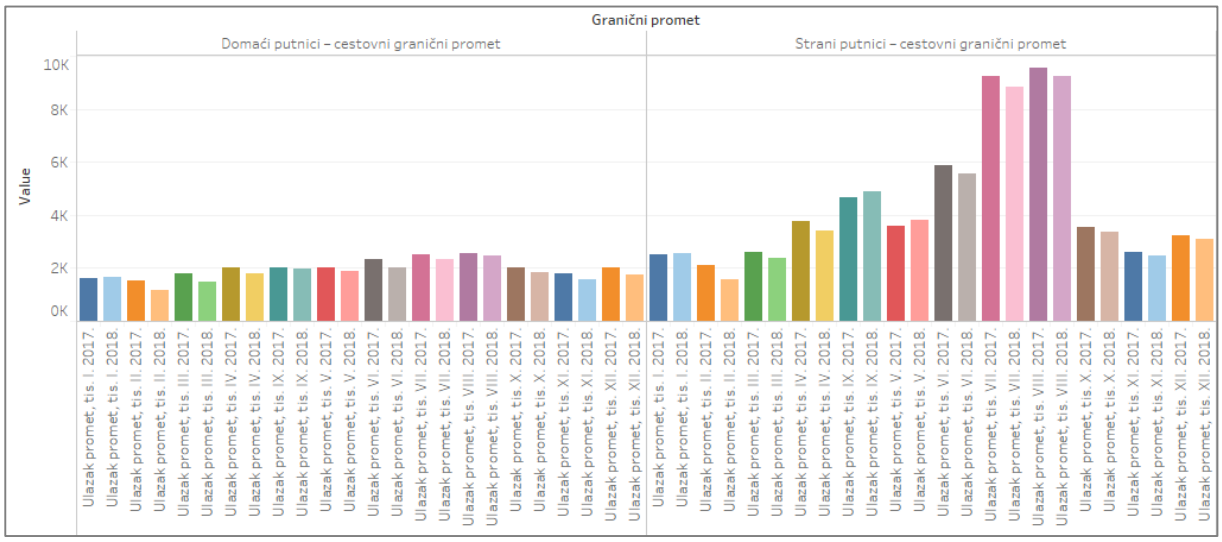


Slika 7 Količina graničnog prometa prema vrstama graničnog prometa - izlazak (1)



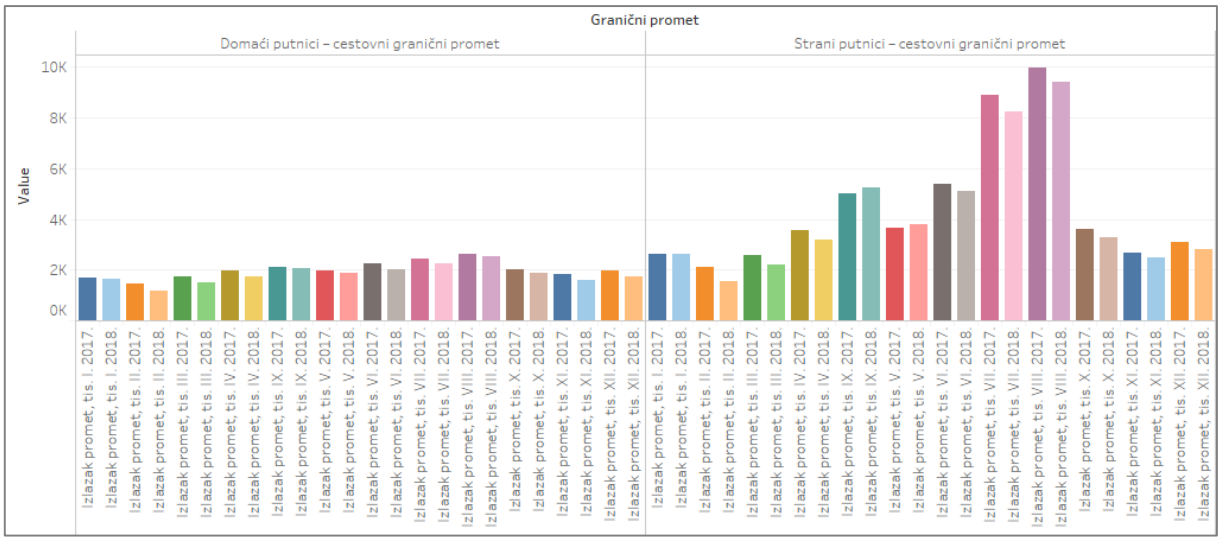
Slika 8 Količina graničnog prometa prema vrstama graničnog prometa - izlazak (2)

Analizom cestovnog prometa na Slici 9 vidljivo je da veliki broj stranih putnika ulazi u zemlju u ljetnim mjesecima, te nešto manje u prosincu. Broj domaćih putnika koji ulaze u zemlju je jednoličan te u oba slučaja opada cestovni promet izuzev prometa stranih putnika u svibnju i rujnu 2018. kada se promet povećao u odnosu na prethodnu godinu.



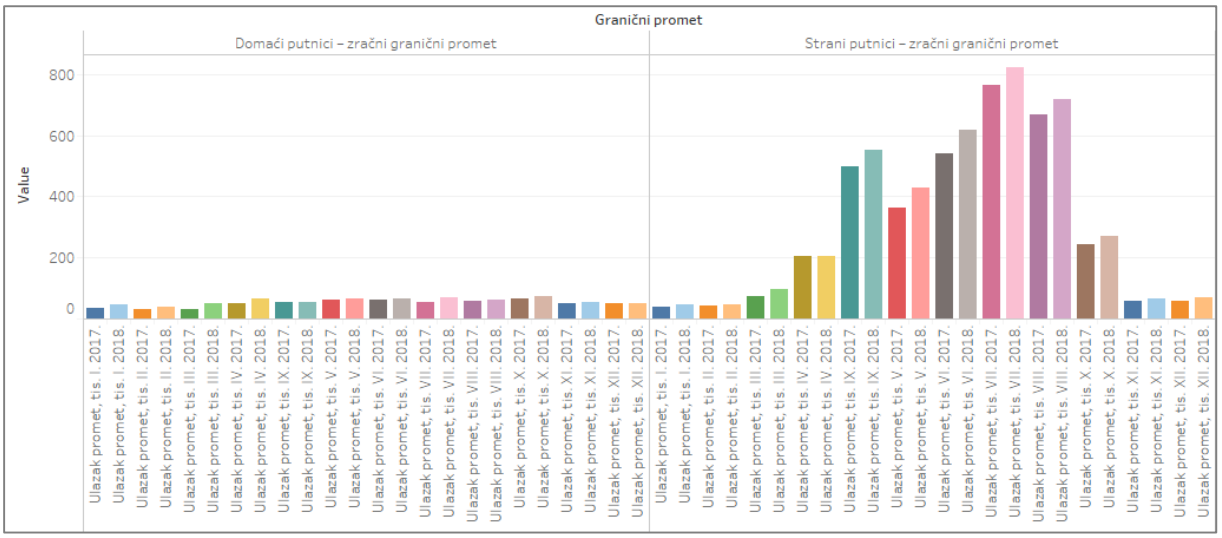
Slika 9 Cestovni granični promet po vrstama putnika - ulazak

Domaći putnici više izlaze iz zemlje u ljetnim mjesecima te je povećan broj u travnju i prosincu zbog proljetnih i zimskih praznika. Izlazak stranih putnika povećan je u ljetnim mjesecima kad je povećan i ulazak. Grafikon je prikazan na Slici 10.



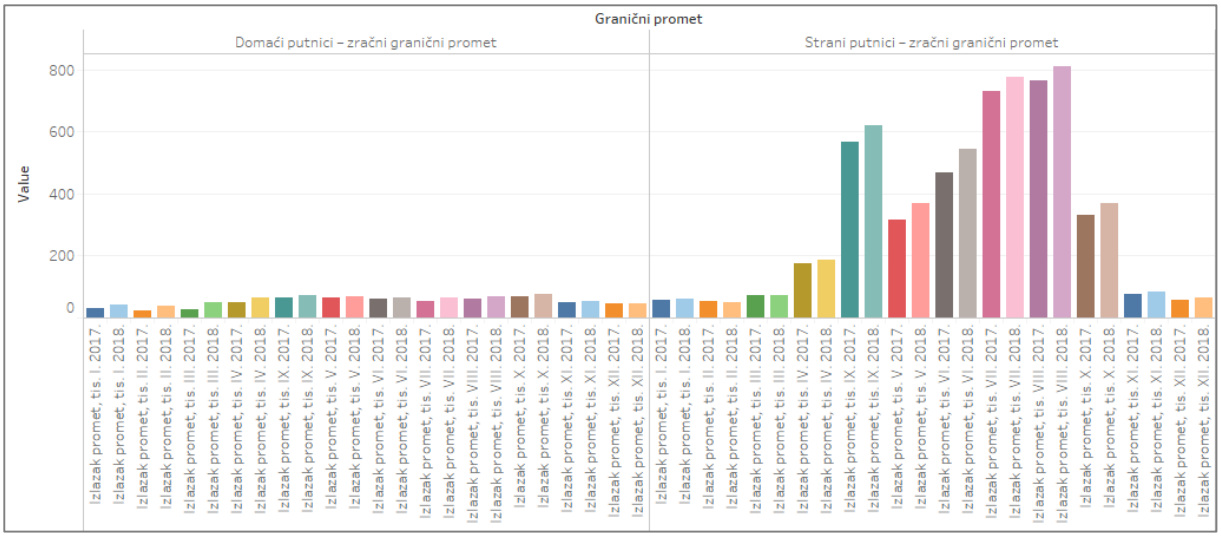
Slika 10 Cestovni granični promet po vrstama putnika - izlazak

Zračni promet koriste strani putnici koji zrakoplovom najviše putuju od svibnja do rujna, a najveći broj letova zabilježen je u srpnju te promet raste što je i vidljivo na Slici 11.



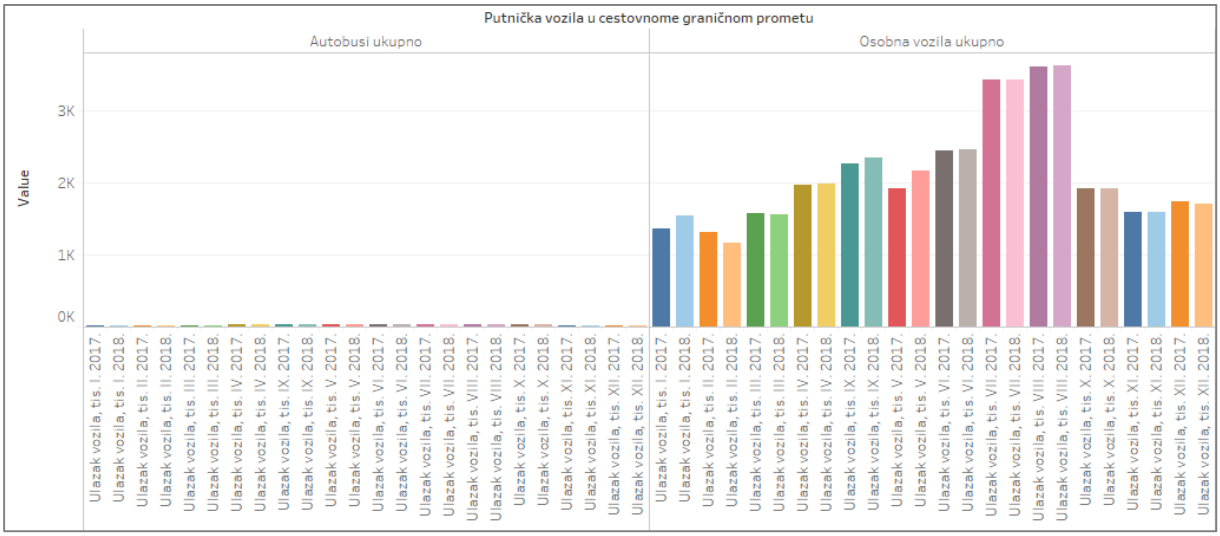
Slika 11 Zračni granični promet po vrstama putnika - ulazak

Isti trend nastavlja se i kod izlaska iz zemlje te i u tom slučaju strani putnici masovno koriste zrakoplove kao prijevozno sredstvo dok ga domaći putnici koriste u jako malom broju, ali je vidljiv mali porast u zračnom prometu (Slika 12).



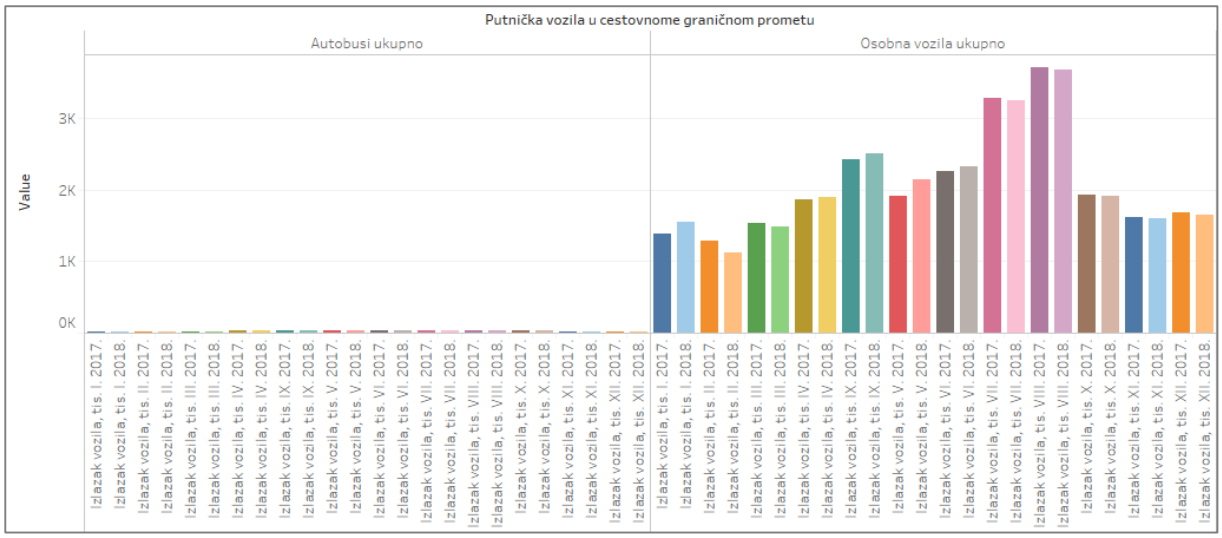
Slika 12 Zračni granični promet po vrstama putnika - izlazak

Putnici u cestovnom prometu kao prijevozno sredstvo najviše koriste osobna vozila, a autobus se koristi jako malo u odnosu na osobna vozila. Najveći broj osobnih vozila na cestama je u srpnju i kolovozu te je taj broj lagano u opadanju. U siječnju raste broj osobnih vozila na cestama zbog skijaške sezone kada je svu opremu praktičnije ponijeti u osobnom vozilu (Slika 13).



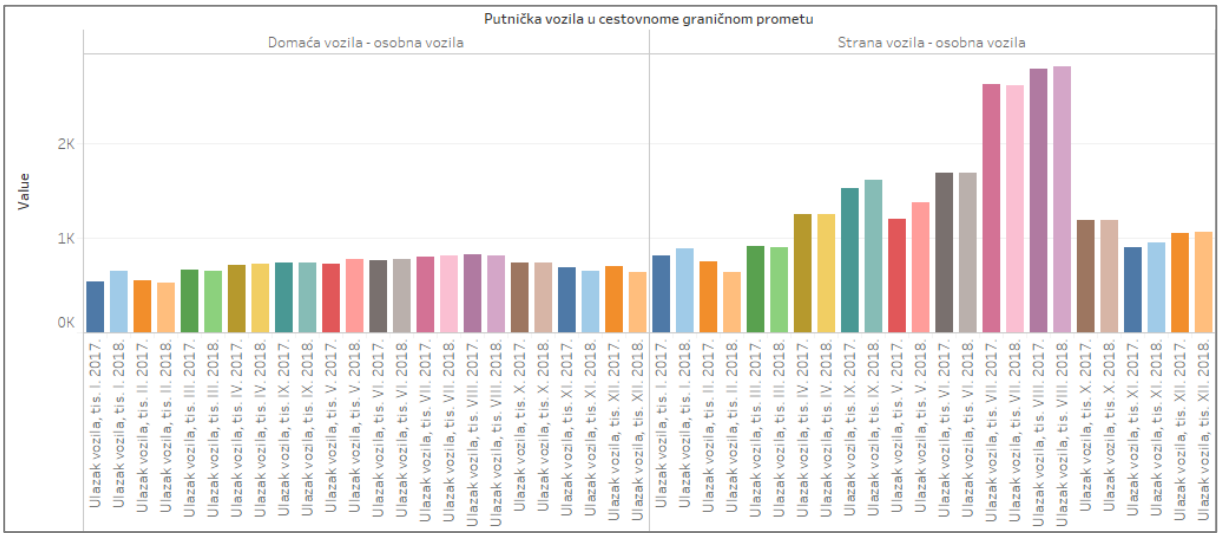
Slika 13 Vrste putničkih vozila u cestovnome graničnom prometu - ulazak

Identičan grafikon prikazuje putnička vozila u cestovnome graničnom prometu na izlazu iz zemlje na Slici 14. Kada se usporede brojke veći je broj ulazaka nego broj izlazaka.



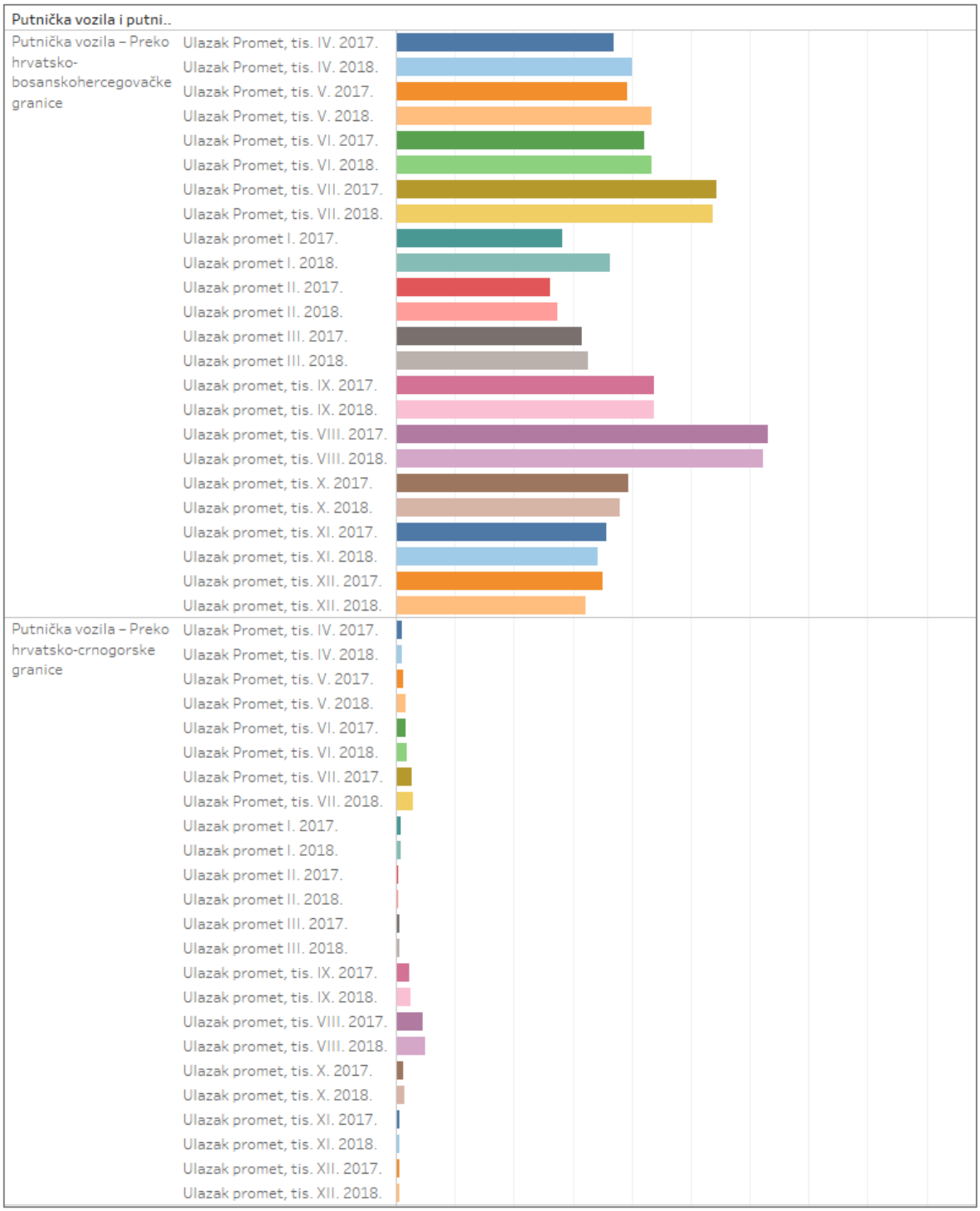
Slika 14 Vrste putničkih vozila u cestovnome graničnom prometu - izlazak

Udio stranih vozila u osobnim vozilima u ljetnim mjesecima je duplo veći na ulazu u državu. Srpanj i kolovoz su udarni mjeseci kada se broj stranih osobnih vozila udvostručuje u odnosu na ostale mjesece. Broj vozila se u dvije godine nije drastično promijenio, u pojedinim mjesecima broj je neznatno porastao ili se smanjio. Na Slici 15 nalazi se grafikon koji prikazuje ulazak osobnih vozila prema vrsti putnika, a grafikon koji prikazuje izlazak je identičan te stoga nije naveden.

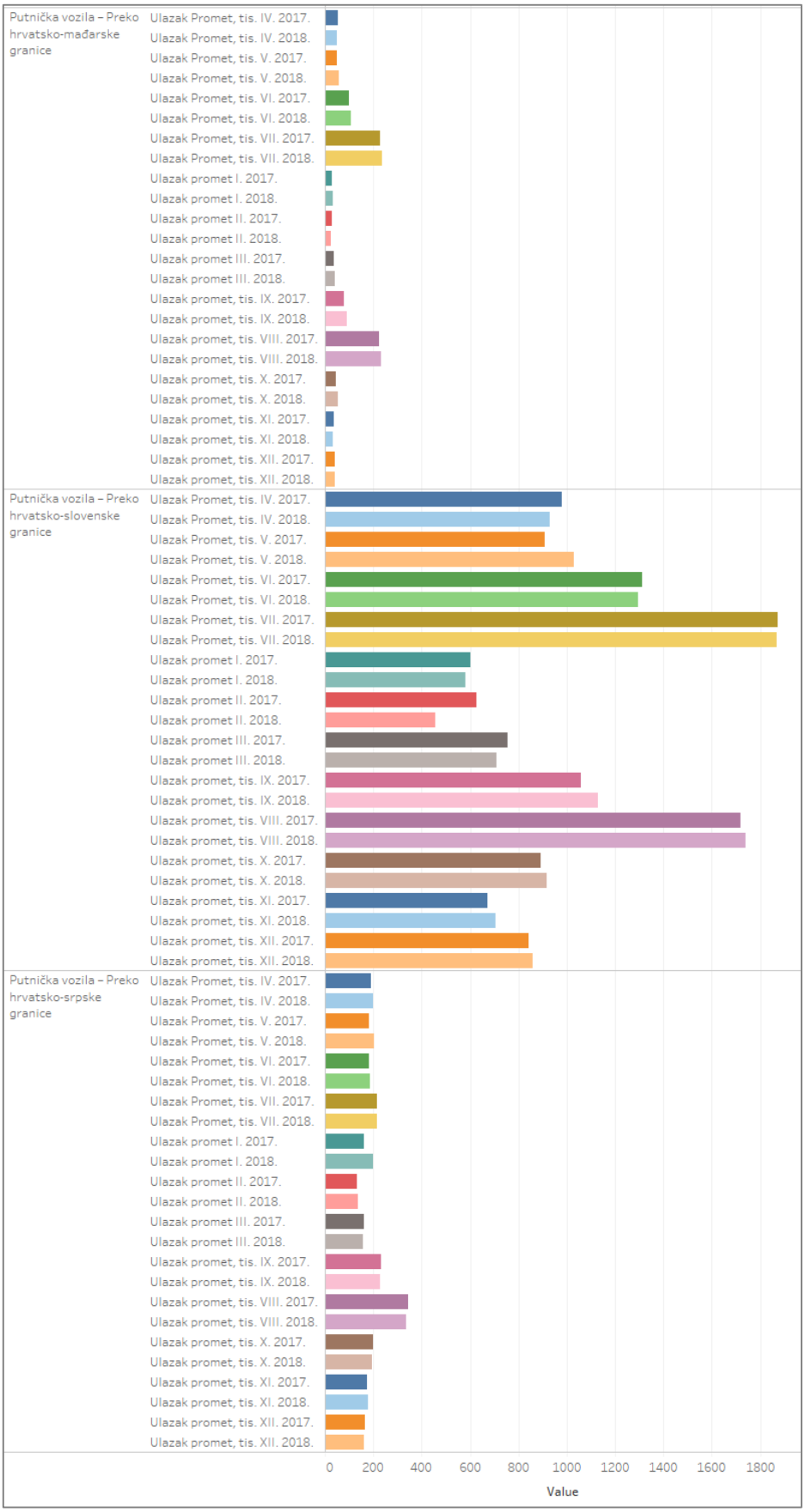


Slika 15 Udio domaćih i stranih vozila u osobnim vozilima cestovnog prometa - ulazak

Analizom graničnih prijelaza otkriveno je da je najveći broj putničkih cestovnih vozila ušlo preko hrvatsko-bosanskohercegovačke granice te preko hrvatsko-slovenske granice. Veći broj vozila ulazi preko hrvatsko-slovenske granice što je posljedica toga da turisti iz Slovenije, Austrije i Njemačke moraju proći kroz Sloveniju da bi došli u posjet. Najgušći promet je standardno u srpnju i kolovozu, a generalno gledajući na obje granice promet je intenzivan u proljeće i ljeto, te nešto manje u jesen, dok zimi nema gužvi (Slika 16 i Slika 17). Identičan promet je i na izlazu iz države te taj grafikon nije naveden.

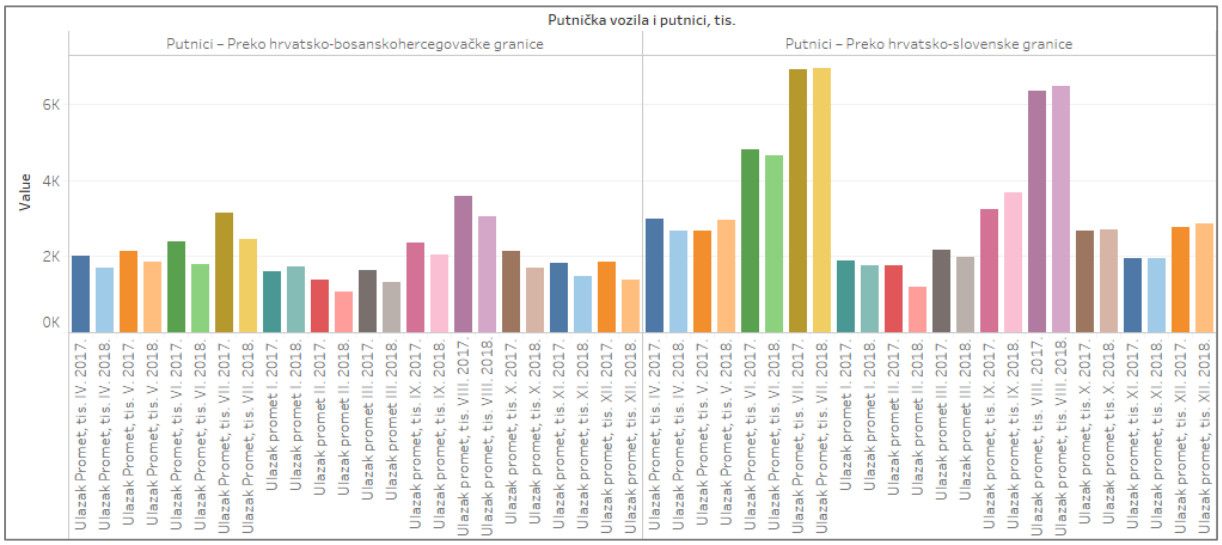


Slika 16 Granični promet putničkih cestovnih vozila po smjerovima kretanja – ulazak(1)



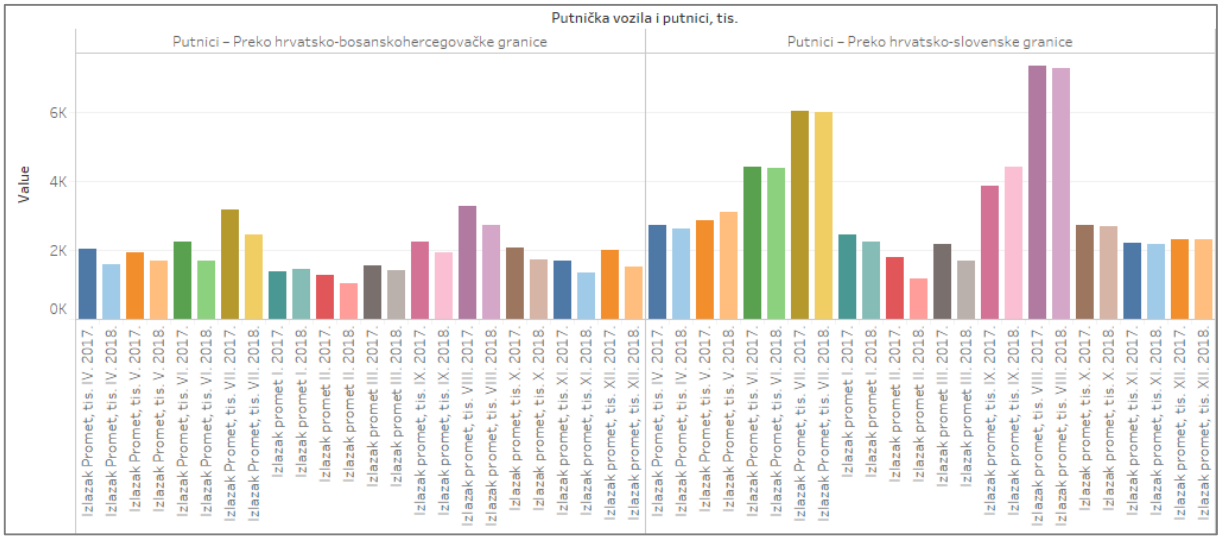
Slika 17 Granični promet putničkih cestovnih vozila po smjerovima kretanja – ulazak(2)

Putnici koji ulaze na navedenim graničnim prijelazima su brojniji na hrvatsko-slovenskoj granici gdje u srpnju i kolovozu uđe čak 6,9 mil. putnika. Najveći broj putnika ulazi u srpnju te je u tom mjesecu ušlo duplo više putnika na hrvatsko-slovenskoj granici nego na hrvatsko-bosanskohercegovačkoj (Slika 18).



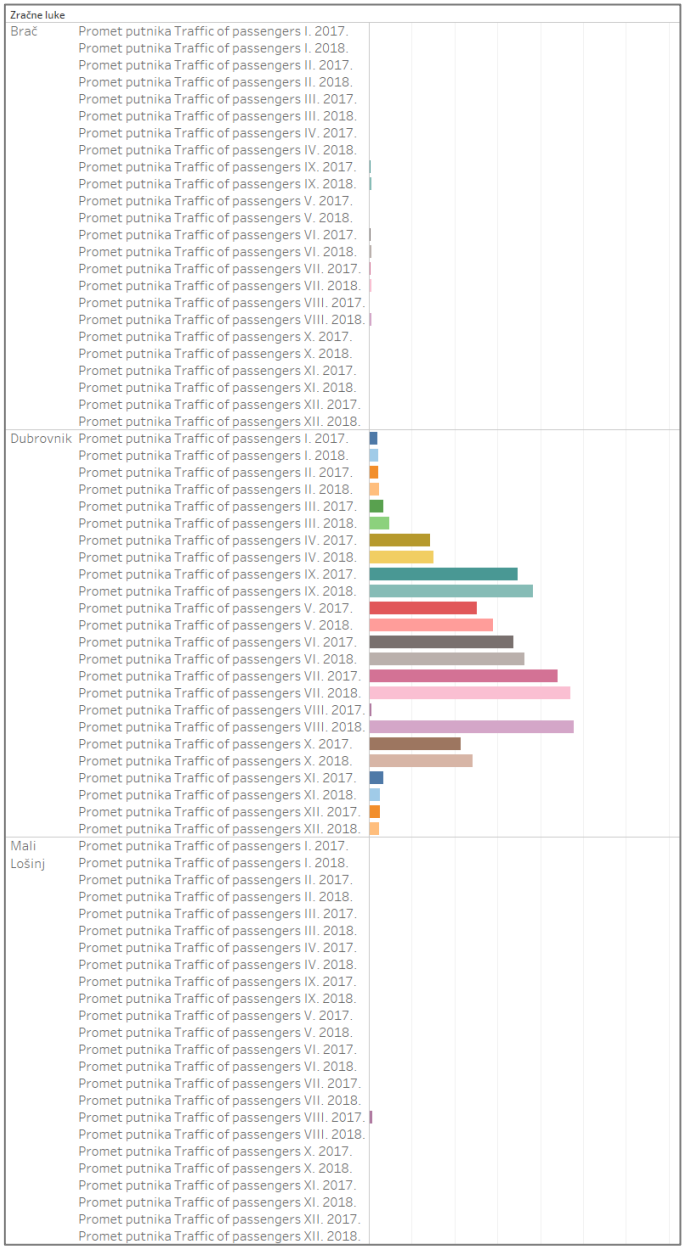
Slika 18 Granični promet putnika po smjerovima kretanja - ulazak

Kod izlaska putnika na gore navedenim graničnim prijelazima najveći broj putnika izašao je na hrvatsko-slovenskoj granici u kolovozu 2017. g., a na bosansko-hercegovačkoj granici nastavlja se trend opadanja broja putnika, kako na ulasku u Hrvatsku tako i na izlasku. U srpnju i kolovozu taj pad je izraženiji nego u ostalim mjesecima, dok se na hrvatsko-slovenskoj granici ne vidi značajan pad putnika (Slika 19).

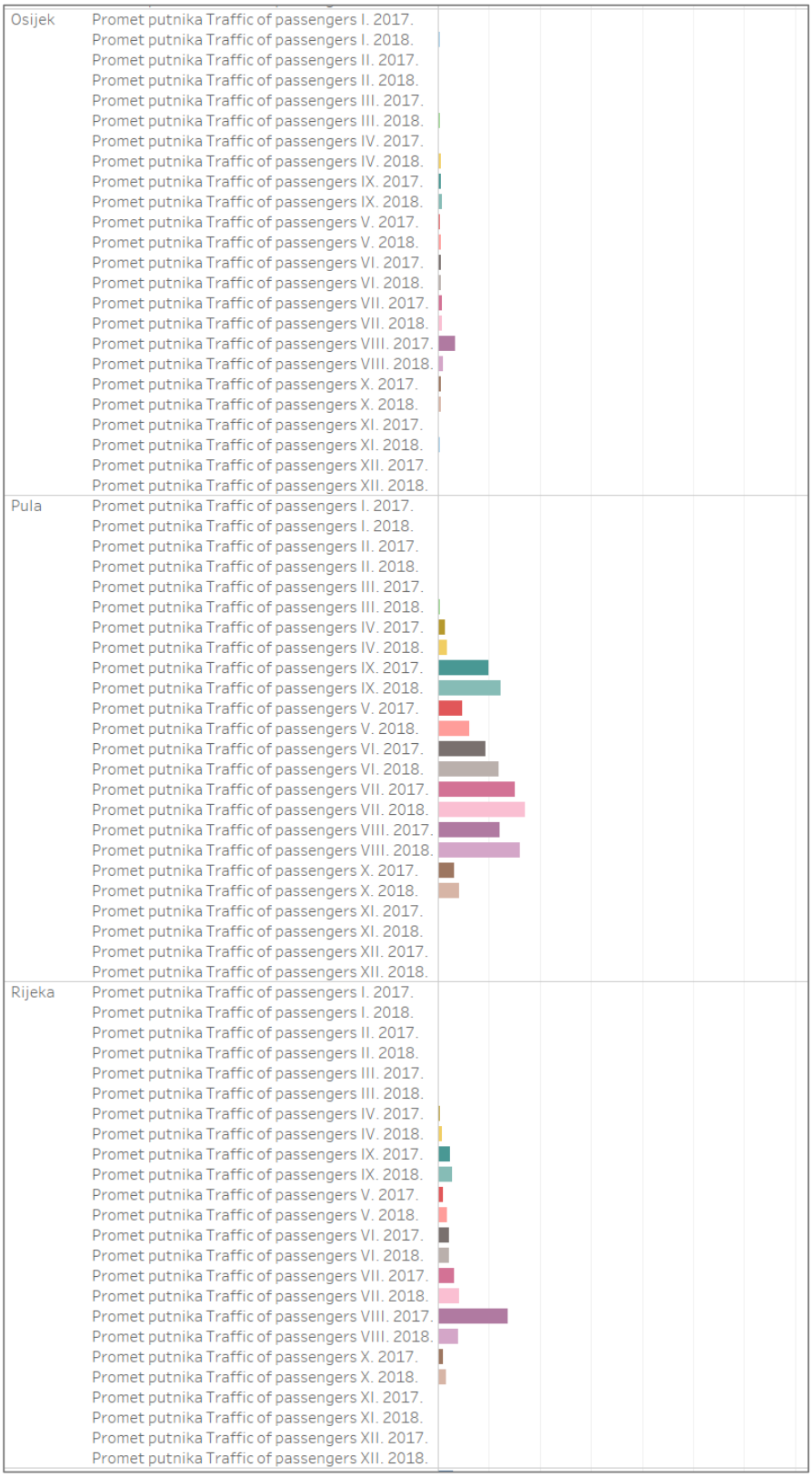


Slika 19 Granični promet putnika po smjerovima kretanja - izlazak

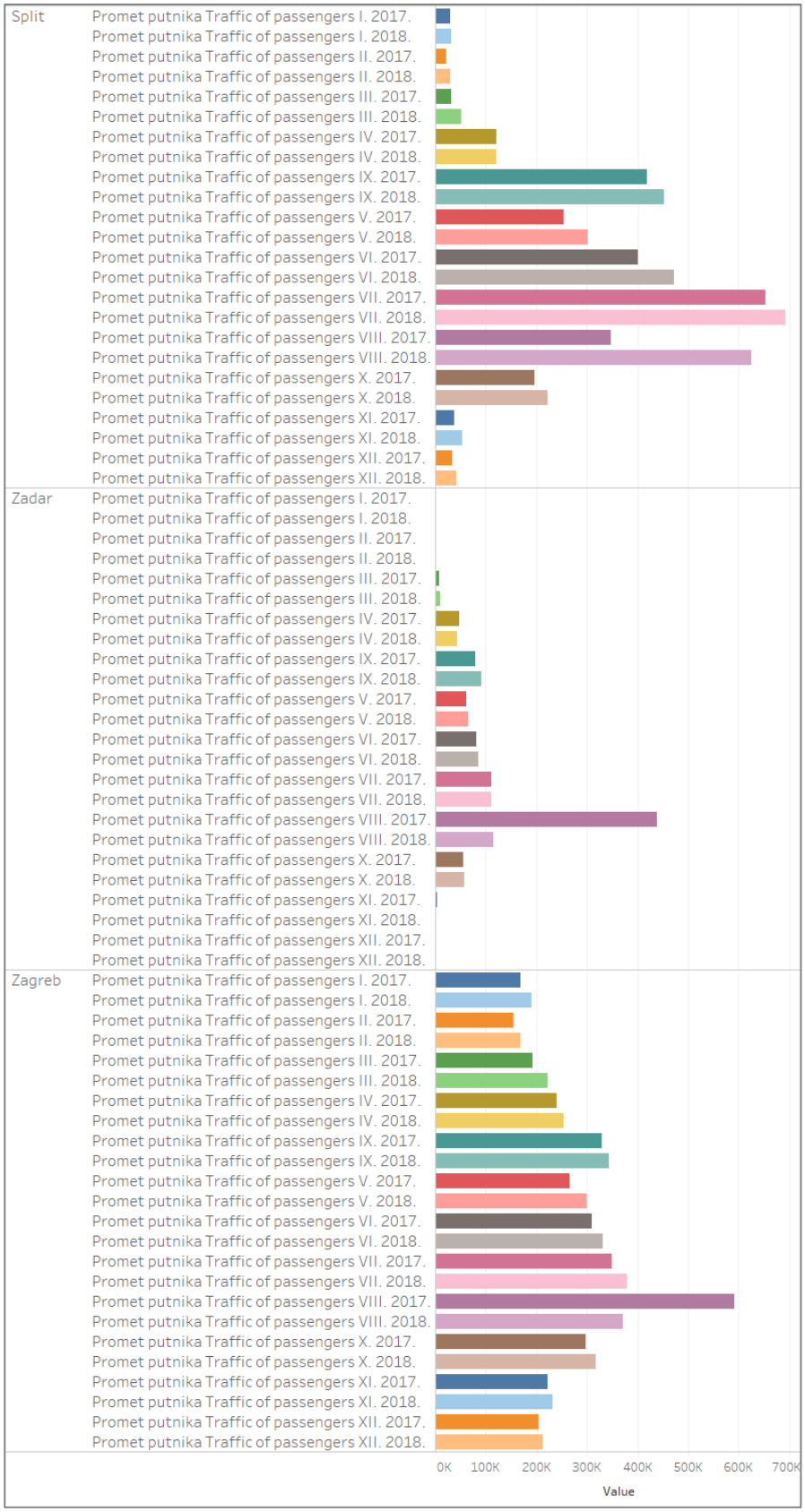
Nadalje, analizira se promet u zračnim lukama. Analizom devet zračnih luka otkriveno je da najveći promet ima Zračna luka Zagreb te zatim Zračne luke Split i Dubrovnik. Zračna luka Zagreb imala je najveći promet u kolovozu 2017. g. kada je zabilježeno 590 tisuća putnika. U kolovozu 2018. g. broj putnika se drastično smanjio dok u ostalim mjesecima raste. Najveći broj putnika u sve tri zračne luke ostvari se u ljetnim mjesecima, a najmanji u zimskim mjesecima. U Zračnoj luci Split broj putnika u srpnju i kolovozu penje se do 650 tisuća te je u ta dva mjeseca promet intenzivan u odnosu na ostale mjesece. Zračna luka Dubrovnik je u kolovozu 2017. g. ostvarila jako mali broj putnika te je pretpostavka da nisu zabilježeni svi podaci. Ostale zračne luke imaju jako mali broj putnika jedino je Zračna luka Zadar u srpnju 2017. g. ostvarila veliki broj putnika i to 438 tisuća (Slika 20 do Slika 22).



Slika 20 Promet u zračnim lukama (1)

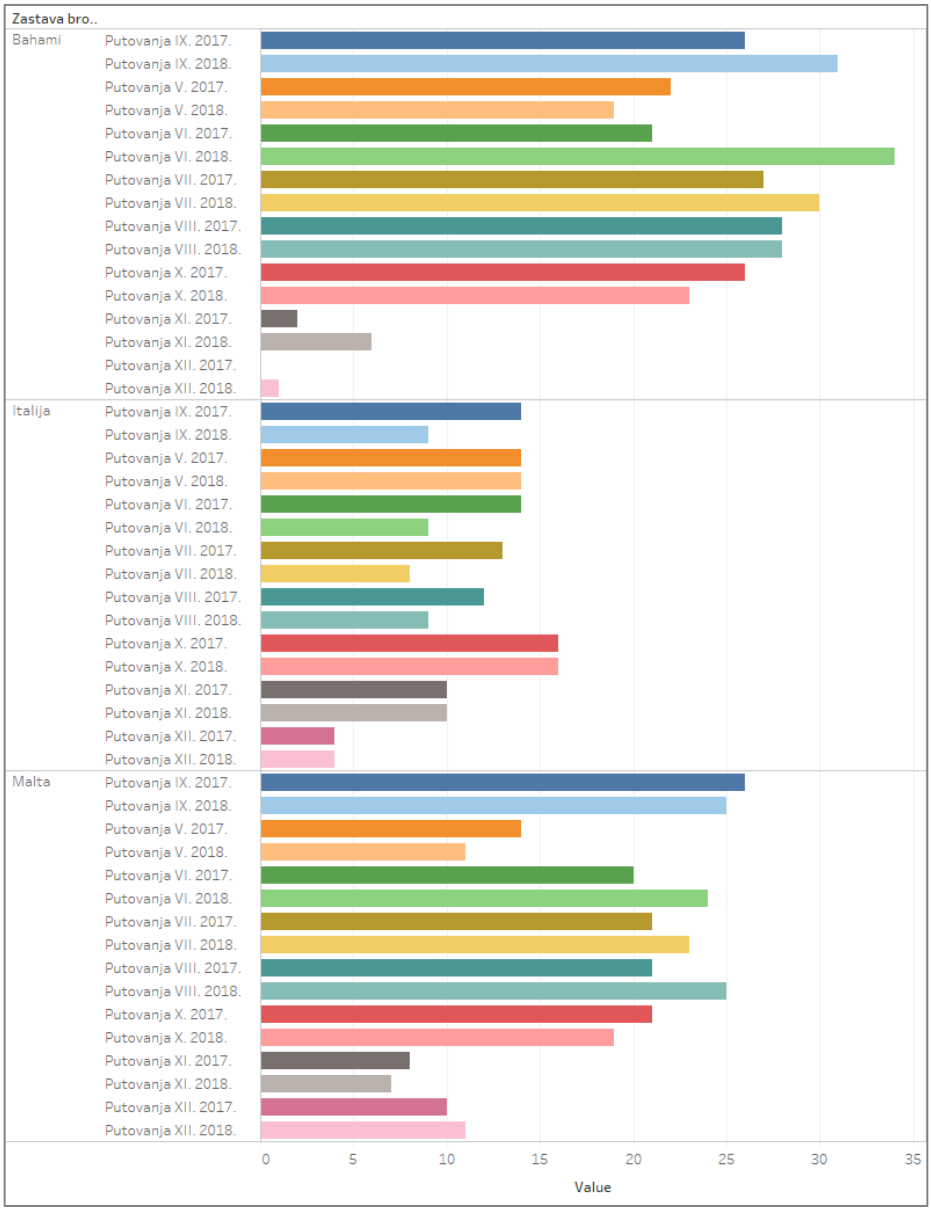


Slika 21 Promet u zračnim lukama (2)



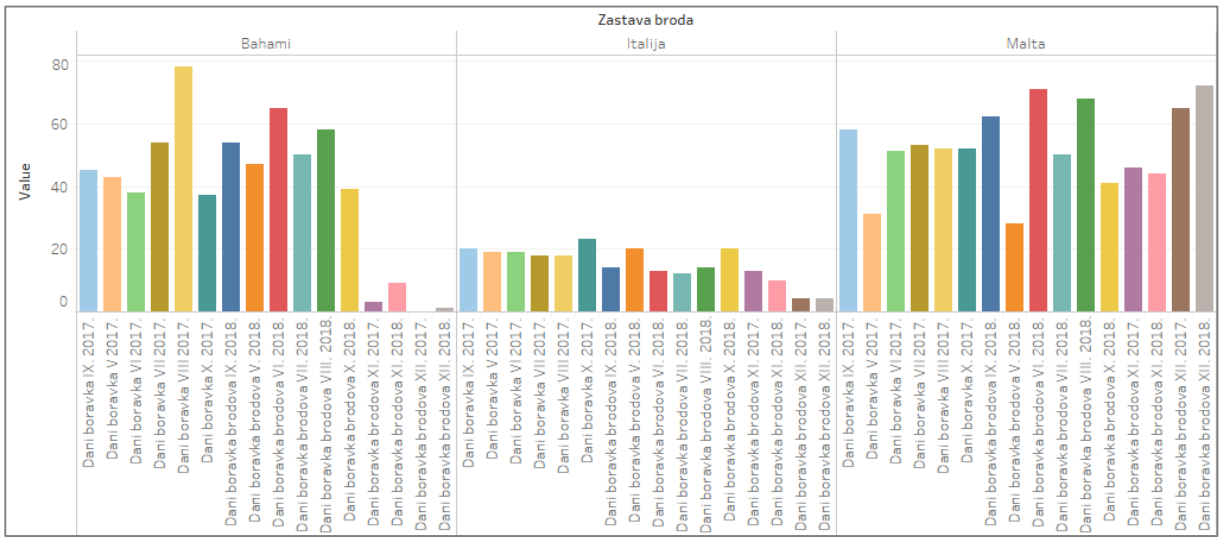
Slika 22 Promet u zračnim lukama (3)

Danas su sve popularnija kružna putovanja brodovima. Brod za kružna putovanja jest putnički brod čija namjena nije prijevoz osoba od jedne luke do druge luke odredišta, nego je namijenjen za uživanje u putovanju i sadržajima na brodu (luksuznom ugođaju, avanturama, obrazovanju, kulturi, zabavi, sportskim aktivnostima, wellnessu ili rekreaciji). Opremljen je za višednevni boravak putnika i mora pružati usluge prehrane, pića i napitaka te usluge smještaja. Brod za kružna putovanja pristaje na svojem putu u lukama koje su zanimljive turističke destinacije [1]. Najveći broj kružnih putovanja stranih brodova prema zastavi broda pripada brodovima sa Bahama, Malte i Italije. Brodovi sa Bahama dolaze u sve većem broju te ih je najviše u lipnju, srpnju i rujnu dok je broj brodova u svibnju i listopadu u opadanju. Brodovi sa Malte najviše dolaze u ljetnim mjesecima i rujnu, dok je broj talijanskih brodova najveći u listopadu (Slika 23).



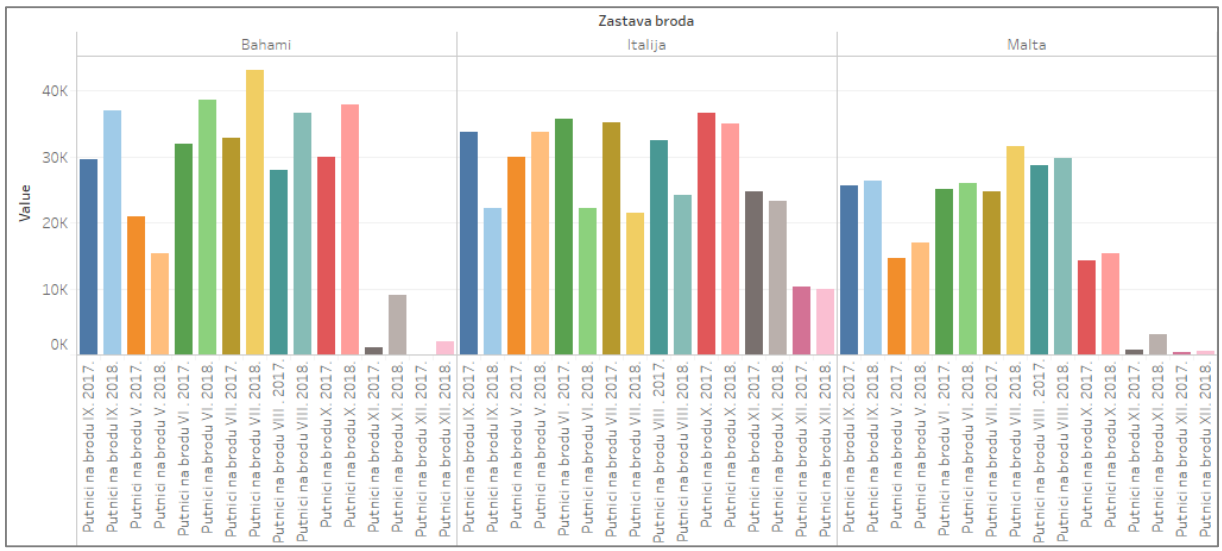
Slika 23 Broj kružnih putovanja stranih brodova prema zastavi broda

Putnici sa Malte proveli su najviše dana na kružnom putovanju te je broj dana sve veći. Sličan trend vlada i na putovanjima sa Bahama ali oni provedu jako malo dana u Hrvatskoj u studenom i prosincu. Putnici sa talijanskih brodova provedu upola manje dana u Hrvatskoj te je broj dana provedenih na našem teritoriju sve manji (Slika 24).



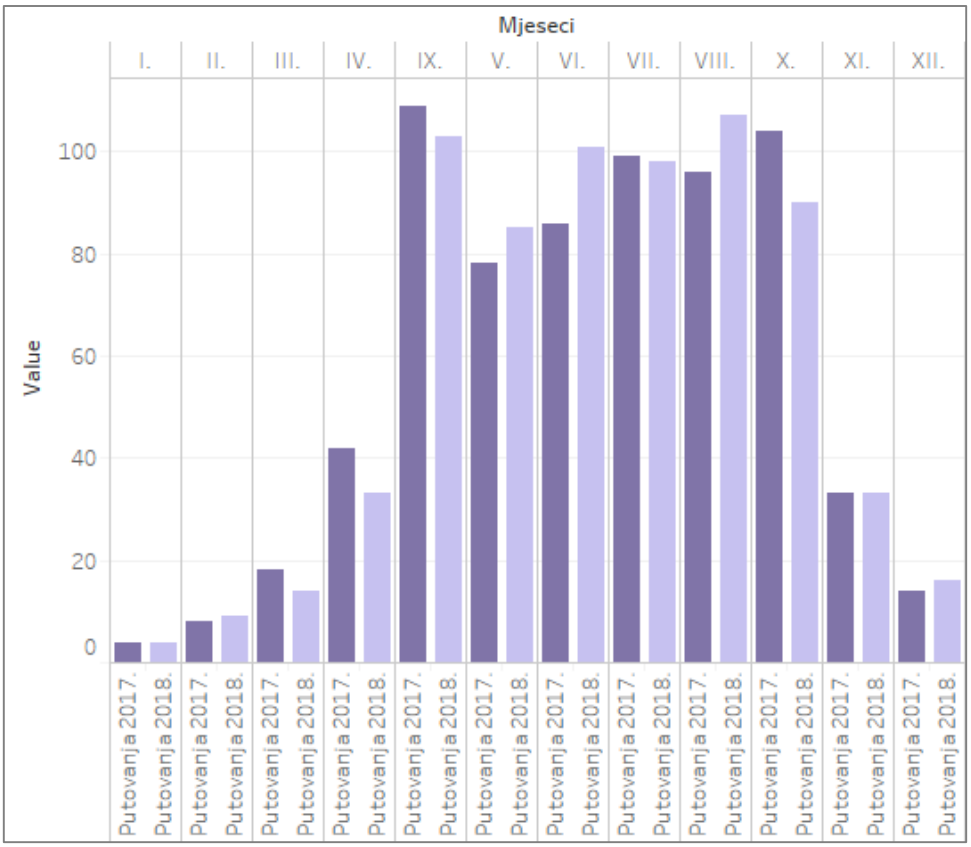
Slika 25 Dani boravka brodova prema zastavi broda

Najveći broj putnika dolazi na brodovima sa Bahama gdje je oko 38 000 putnika u ljetnim mjesecima i u jesen te ih dolazi sve veći broj. Zanimljivo je to da se broj talijanskih putnika drastično smanjio u 2018. godini naročito u lipnju, srpnju i rujnu kada je inače najveći broj pristiglih brodova (Slika 25).



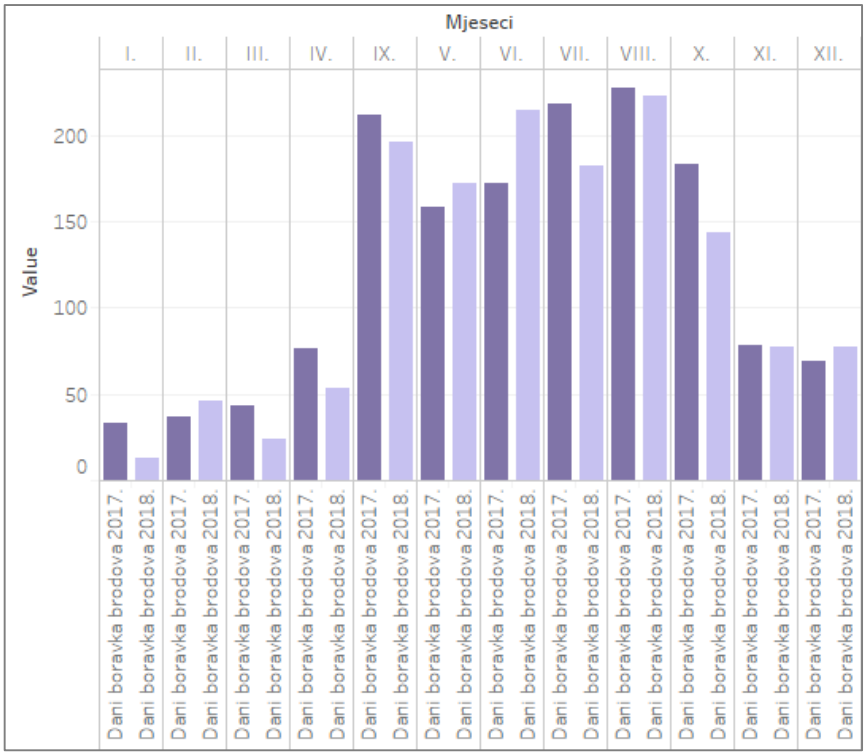
Slika 24 Broj putnika na brodu prema zastavi broda

Kada se analizira broj kružnih putovanja kroz cijelu godinu vidljivo je da su ljeto i jesen dva godišnja doba u kojima dolazi najveći broj brodova. Također je 2018. godina imala veći broj putovanja, jedino je u rujnu i listopadu sezona bila lošija nego 2017. godine što se može pripisati vremenskim uvjetima (Slika 26).



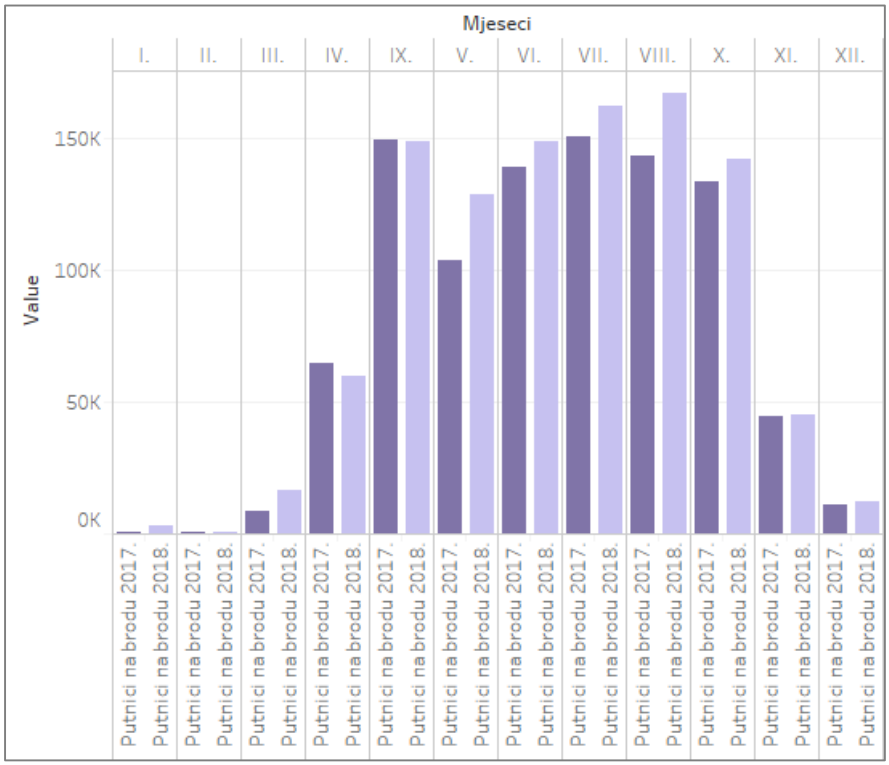
Slika 26 Broj kružnih putovanja po mjesecima

Iako je bila bolja po broju putovanja, 2018. godina lošija je po broju dana boravaka brodova. U srpnju i kolovozu kada se ostvari najveći broj dana boravaka sezona je bila lošija u odnosu na 2017. godinu, ali je zato broj dana boravaka u lipnju bio skoro jednak kao broj dana boravaka u srpnju 2017. te se može pretpostaviti da je određeni broj brodova došao ranije nego obično pa je taj dio putnika pretočen u mjesec lipanj (Slika 27).



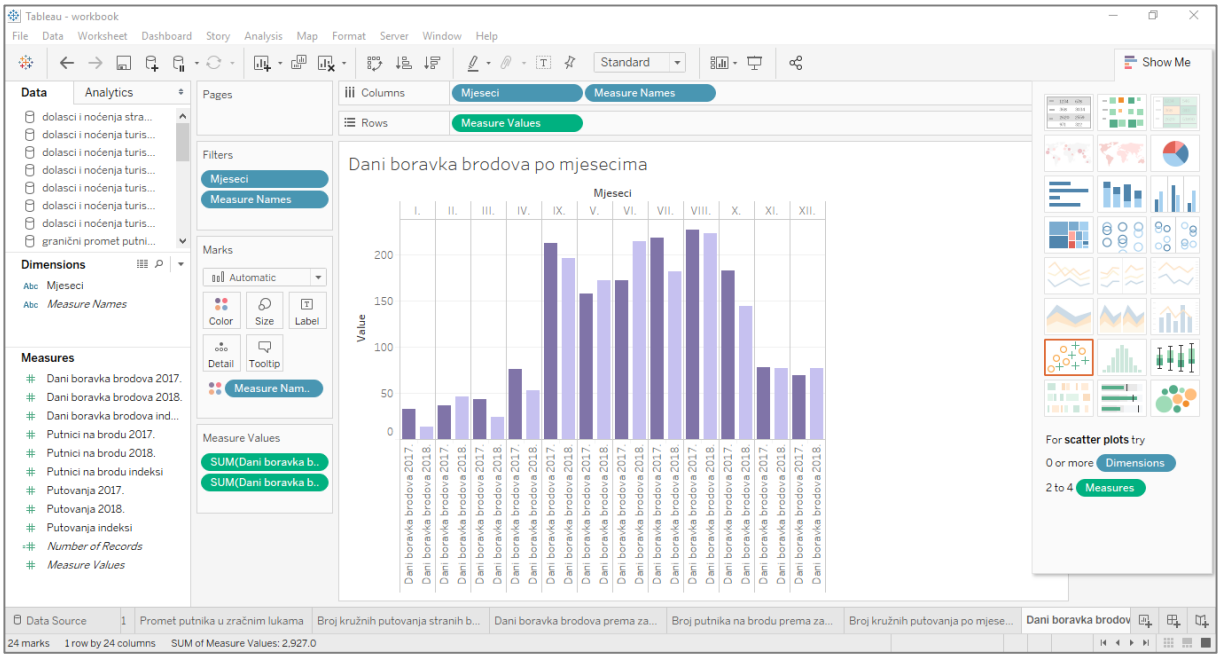
Slika 27 Dani boravka brodova po mjesecima

Iako je broj dana provedenih u Hrvatskoj kraći dolazi sve veći broj turista te je zanimljivo primijetiti da se povećava broj putnika koji dolaze zimi. I dalje glavnina putnika dolazi ljeti te ih je sve više ali povećava se i broj putnika u post sezoni te je u ožujku 2018. došlo duplo više putnika nego prethodne godine (Slika 28).



Slika 28 Broj putnika na brodu po mjesecima

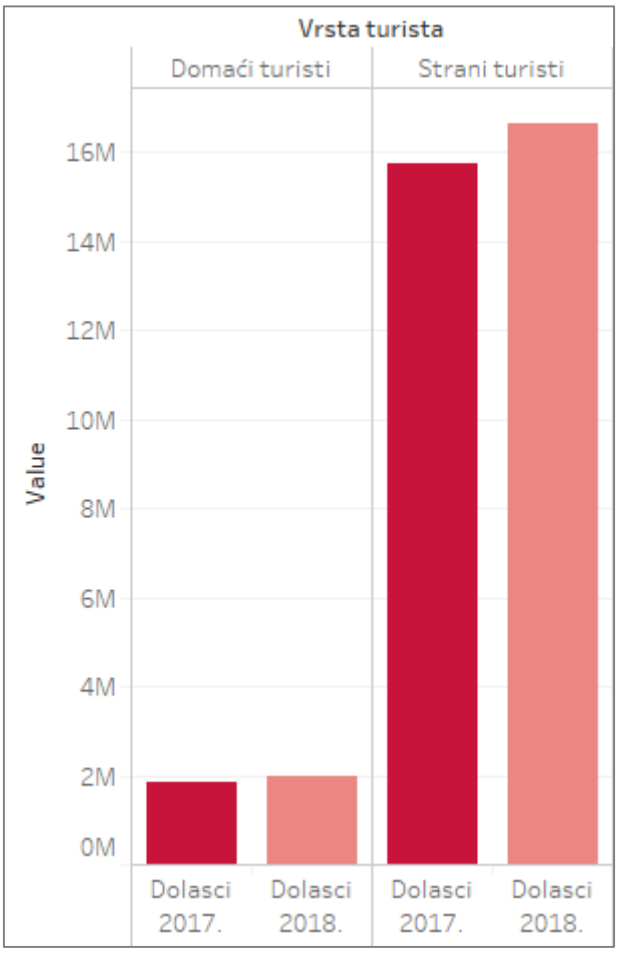
Na Slici 29 prikazan je primjer *dashboard-a* izrađenih grafikona u kojem će se nastaviti izrađivati i ostali grafikoni. Svaki grafikon izrađen je na posebnom *worksheet-u* koji na lijevoj strani ima karticu *Data* u kojoj se nalazi popis svih tablica pomoću kojih su izrađeni grafikoni. Kartica *Dimensions* sadrži dimenzije tablice, a kartica *Measures* sve mjere. Željene dimenzije smještene su u karticu *Columns*, a mjere u karticu *Rows*. Zatim se iz kartice *Show me* odabere željena vrsta grafikona, u ovom slučaju to je *side-by-side-bars*.



Slika 29 Primjer dashboarda i izrađenog grafikona

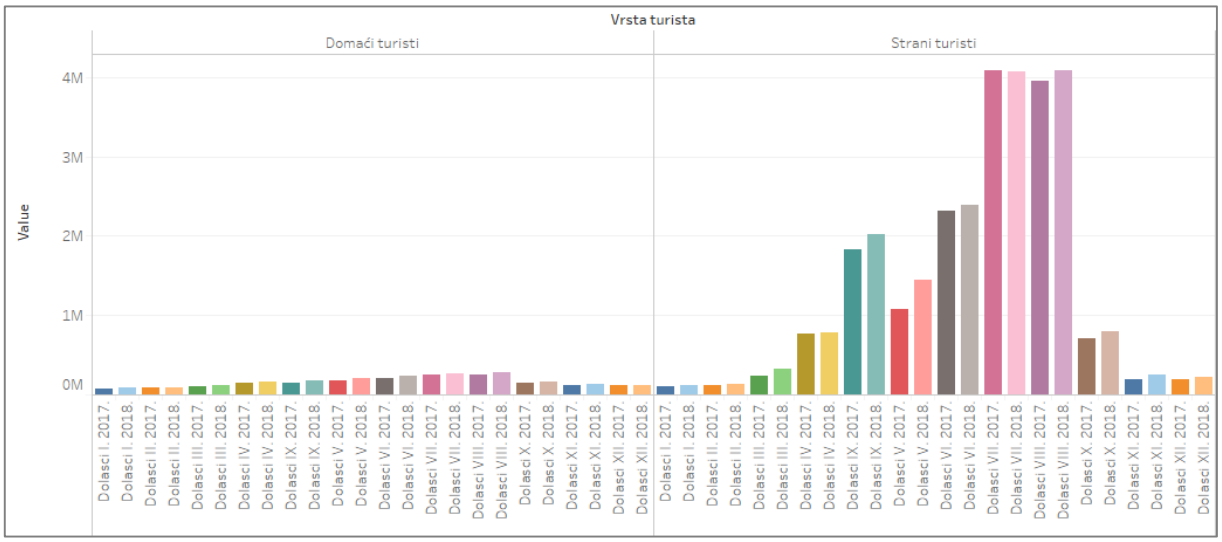
2.2 Dolasci i noćenja turista

Daljnja analiza obuhvaća dolaske i noćenja turista u komercijalnim smještajnim objektima. Broj dolazaka stranih turista veći je od broja dolazaka domaćih turista za čak 8 puta te je 2018. godine ostvareno 16,6 milijuna dolazaka, za 900 000 više nego u 2017. godini. Ostvareni broj dolazaka domaćih turista popeo se do 2 milijuna te je za 200 000 veći nego u 2017. godini što je vidljivo na Slici 30.



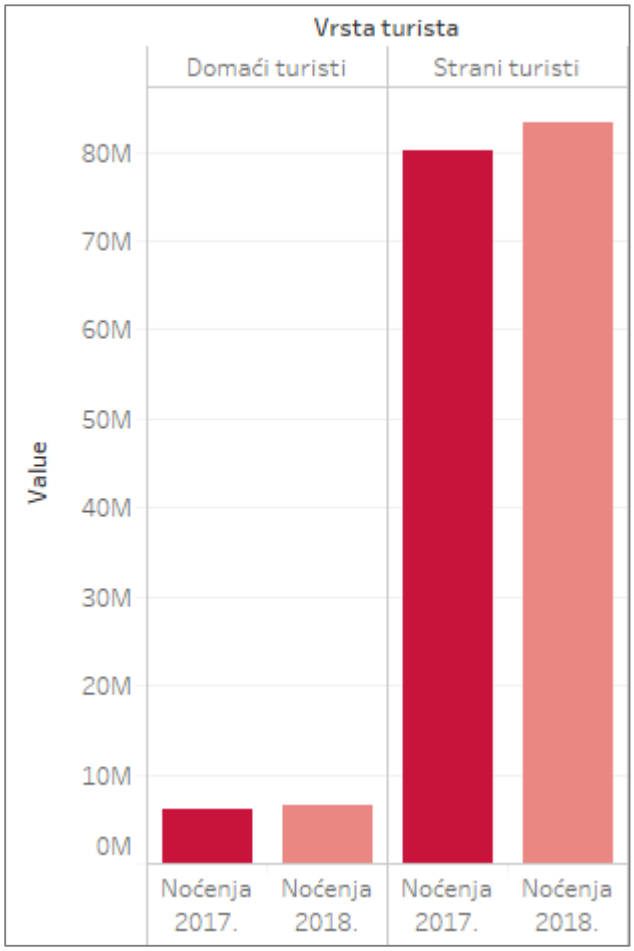
Slika 30 Dolasci turista 2017. i 2018. godine

Navedene brojke analiziraju se i po mjesecima gdje je vidljivo da i strani i domaći turisti najviše dolaze u ljetnim mjesecima. Među domaćim i stranim turistima vlada trend porasta broja dolazaka te je broj stranih turista u kolovozu 2018. veći nego u istom mjesecu prethodne godine i čak je prestigao broj dolazaka iz srpnja 2017. što znači da sve više turista dolazi u tom mjesecu (Slika 31).



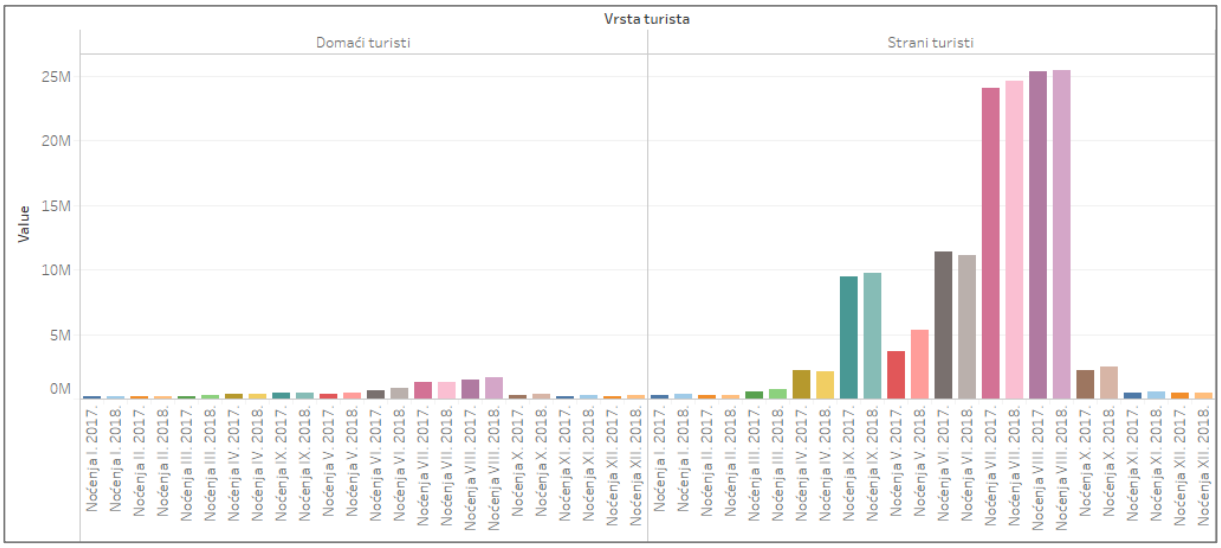
Slika 31 Dolasci turista po mjesecima za 2017. i 2018. godinu

U 16 milijuna dolazaka stranih turista ostvareno je 83 milijuna noćenja, oko 3 milijuna više nego 2017. Domaći turisti ostvarili su 6 milijuna noćenja, daleko manje nego strani turisti. Vidljivo je da strani turisti ostaju puno dulje u smještajima nego domaći turisti koji u prosjeku borave 3 dana, dok strani turisti borave 5 dana (Slika 32).



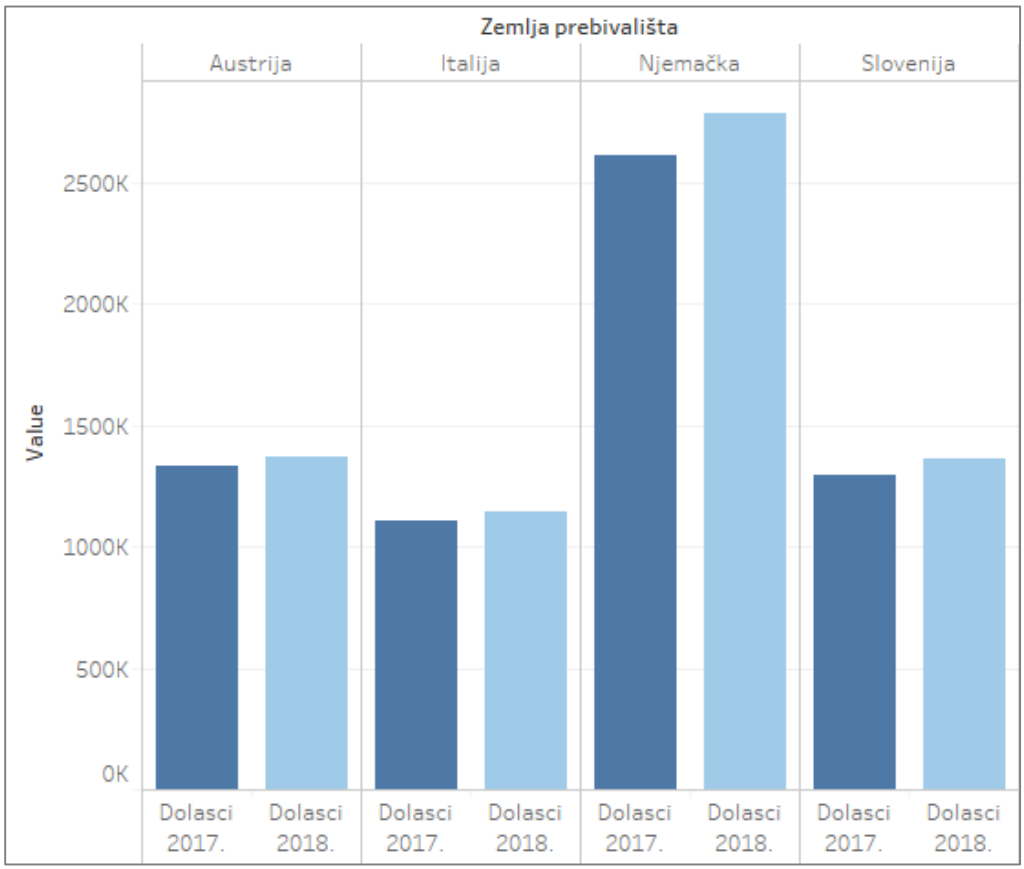
Slika 32 Noćenja turista 2017. i 2018. godine

Ako se broj noćenja turista prikaže po mjesecima vidljivo je da je najveći broj u srpnju i kolovozu te da taj broj raste i kod domaćih i kod stranih turista. Lagani pad u broju noćenja stranih turista vidljiv je u lipnju 2018. ali se to nadoknadilo kroz srpanj i kolovoz. Navedeno je prikazano na Slici 33.



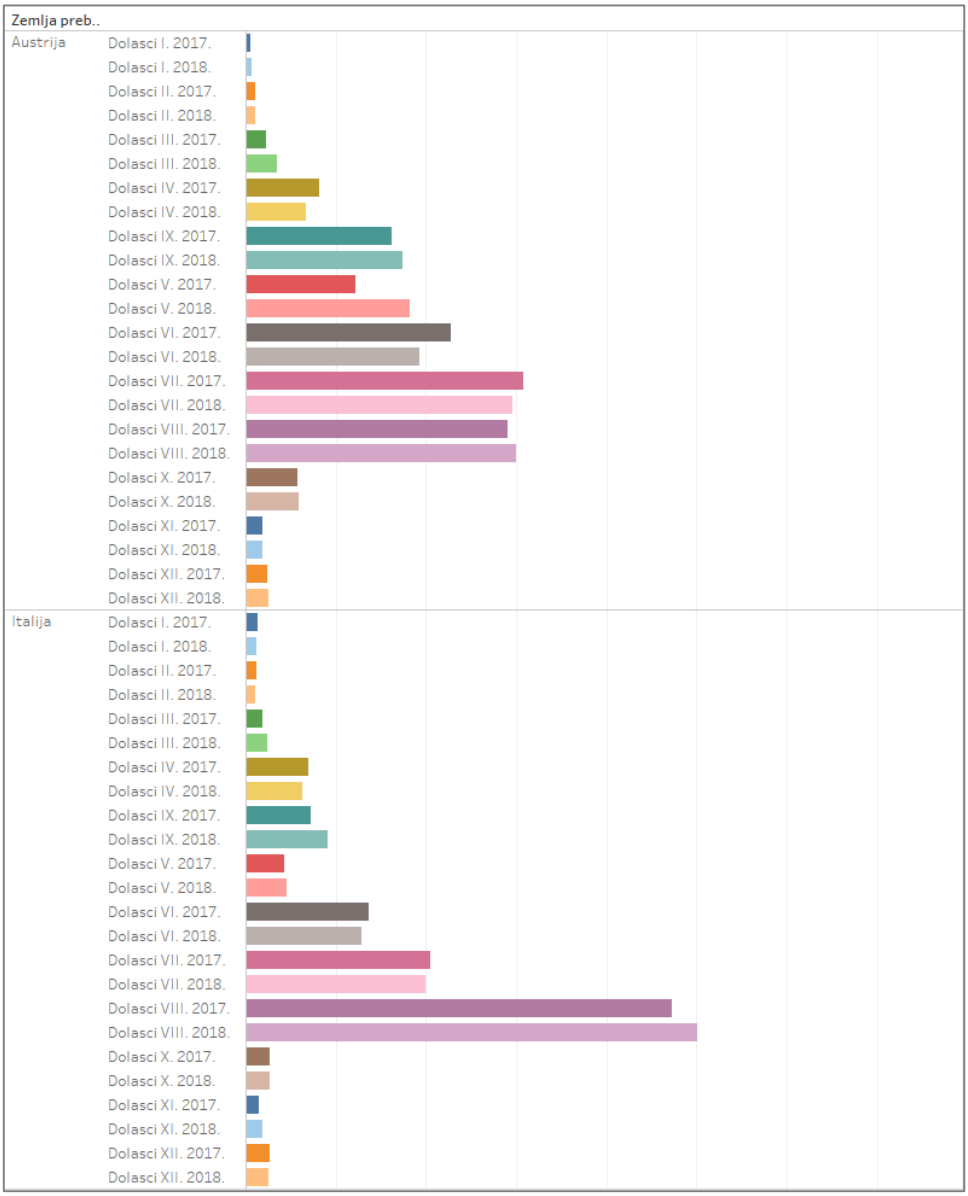
Slika 33 Noćenja turista po mjesecima 2017. i 2018. godine

Među stranim turistima najveći broj dolazaka imaju Nijemci kojih na godišnjoj razini dođe čak 2,6 milijuna. 2017. došlo ih je 2,6 milijuna, a 2018. skoro 2,8 milijuna. Sljedeći narod najbrojniji po dolascima su Austrijanci i Slovenci kojih dođe oko 1,3 milijuna i zatim Talijani kojih je 1,1 milijun (Slika 34).

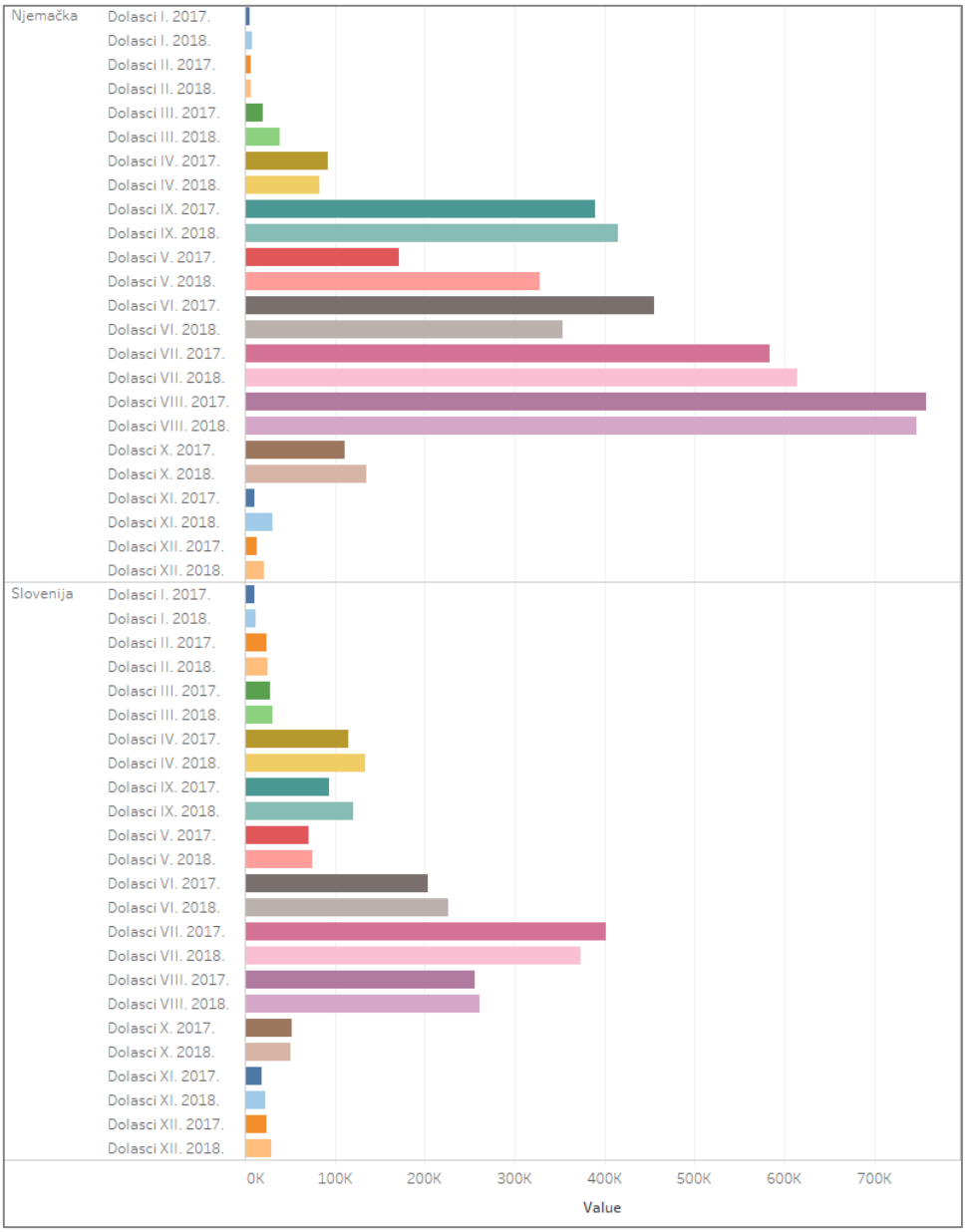


Slika 34 Dolasci stranih turista prema zemlji prebivališta 2017. i 2018.

Analizom dolazaka po mjesecima vidljivo je da polovica dolazaka njemačkih turista dođe u srpnju i kolovozu. Broj turista koji su dolazili u lipnju se smanjio ali je zato porastao broj turista u svibnju. Nijemci većinom preferiraju ljeto i ranu jesen te ih za vrijeme ostalih godišnjih doba dolaze znatno manje. Sličan trend vlada i kod Austrijanaca, dok Slovenci najviše dolaze u srpnju, a Talijani u kolovozu što je i očekivano jer u tom mjesecu imaju kolektivni godišnji odmor te se broj talijanskih turista udvostruči u odnosu na srpanj (Slika 35 i Slika 36).

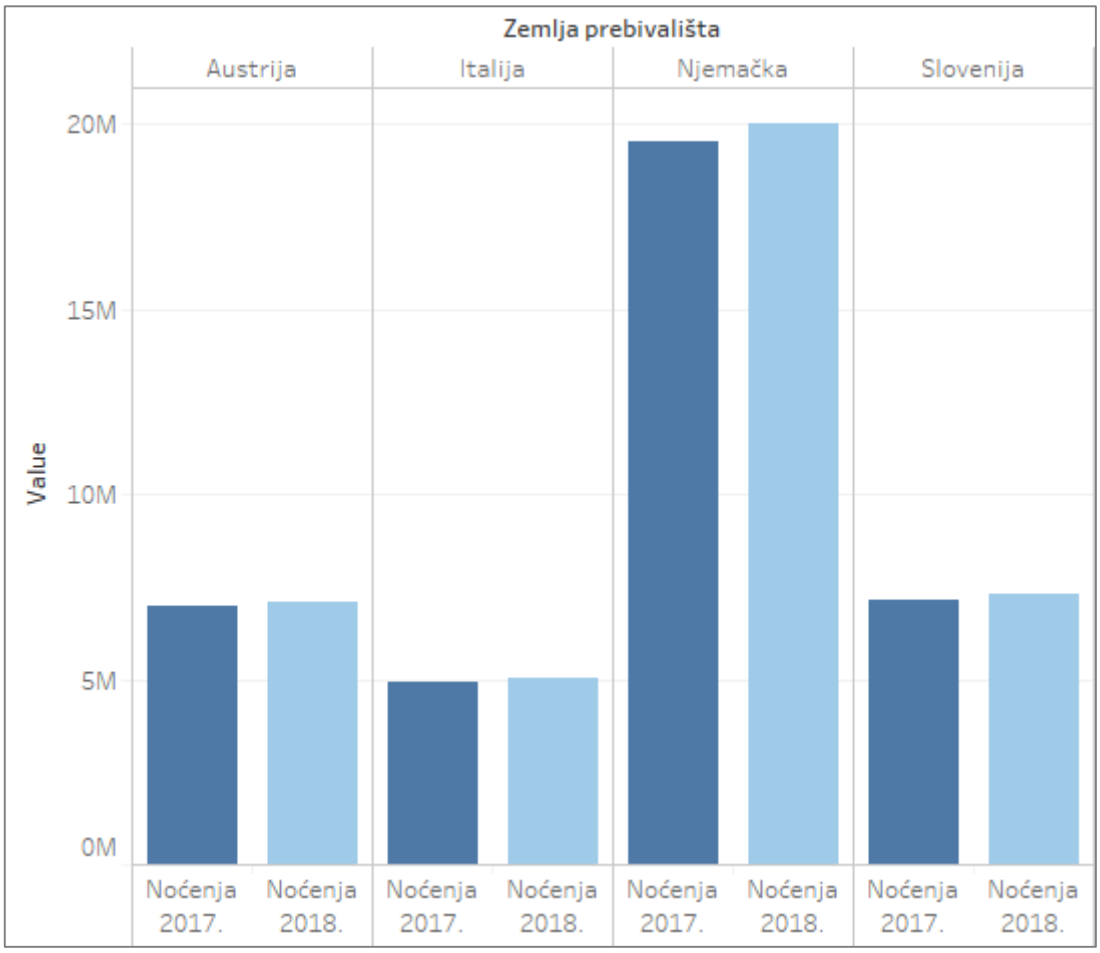


Slika 35 Dolasci stranih turista prema zemlji prebivališta po mjesecima (1)



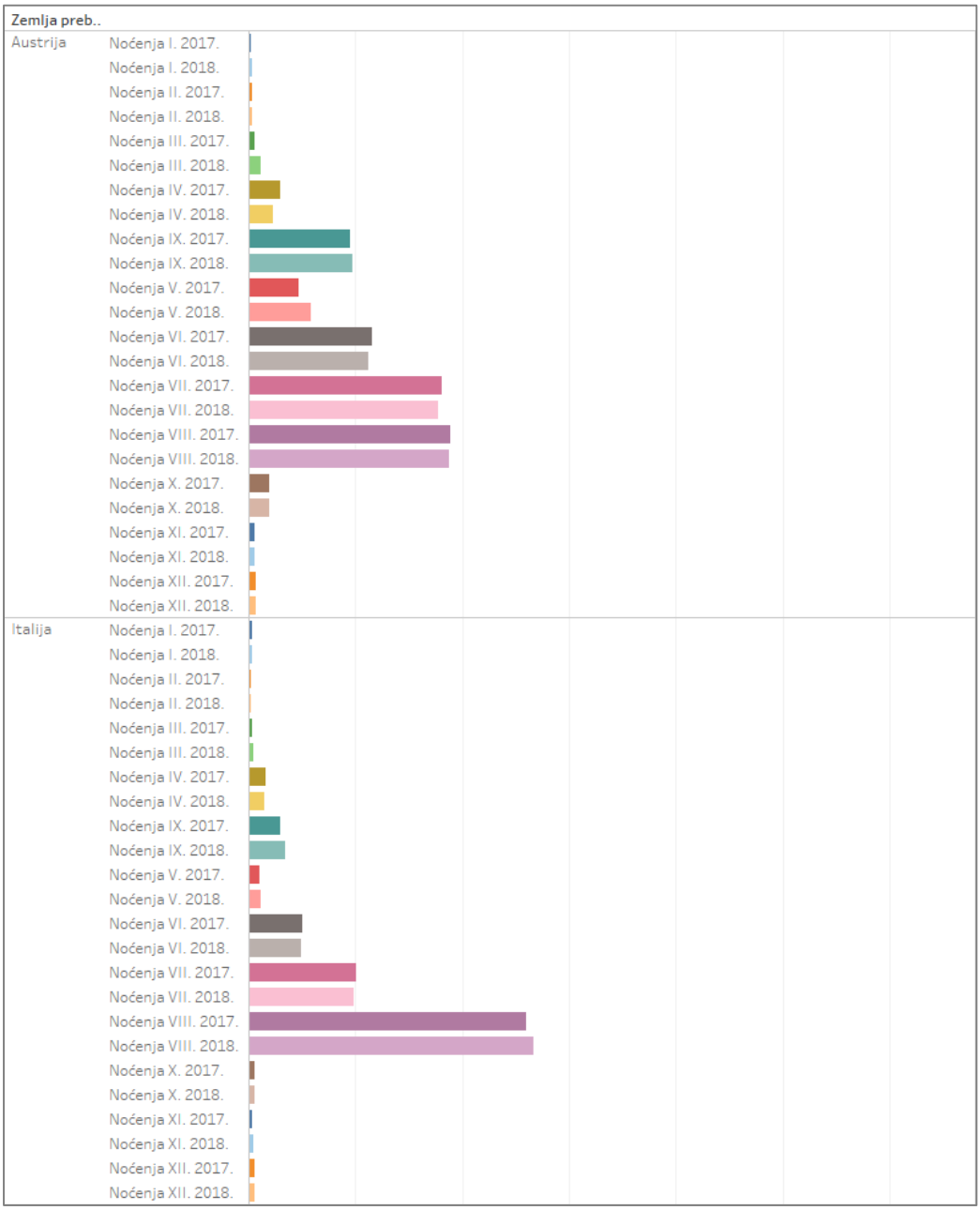
Slika 36 Dolasci stranih turista prema zemlji prebivališta po mjesecima (2)

U 2,6 milijuna dolazaka njemačkih turista ostvareno je gotovo 20 milijuna noćenja što bi značilo da u prosjeku provedu 7 dana u Hrvatskoj. Slovenci iako imaju manji broj dolazaka u ovom slučaju imaju više noćenja od Austrijanaca, a obje nacije su ostvarile oko 7 milijuna noćenja svake godine. Talijani ostvare oko 5 milijuna noćenja te sve nacije imaju porast broja noćenja (Slika 37).

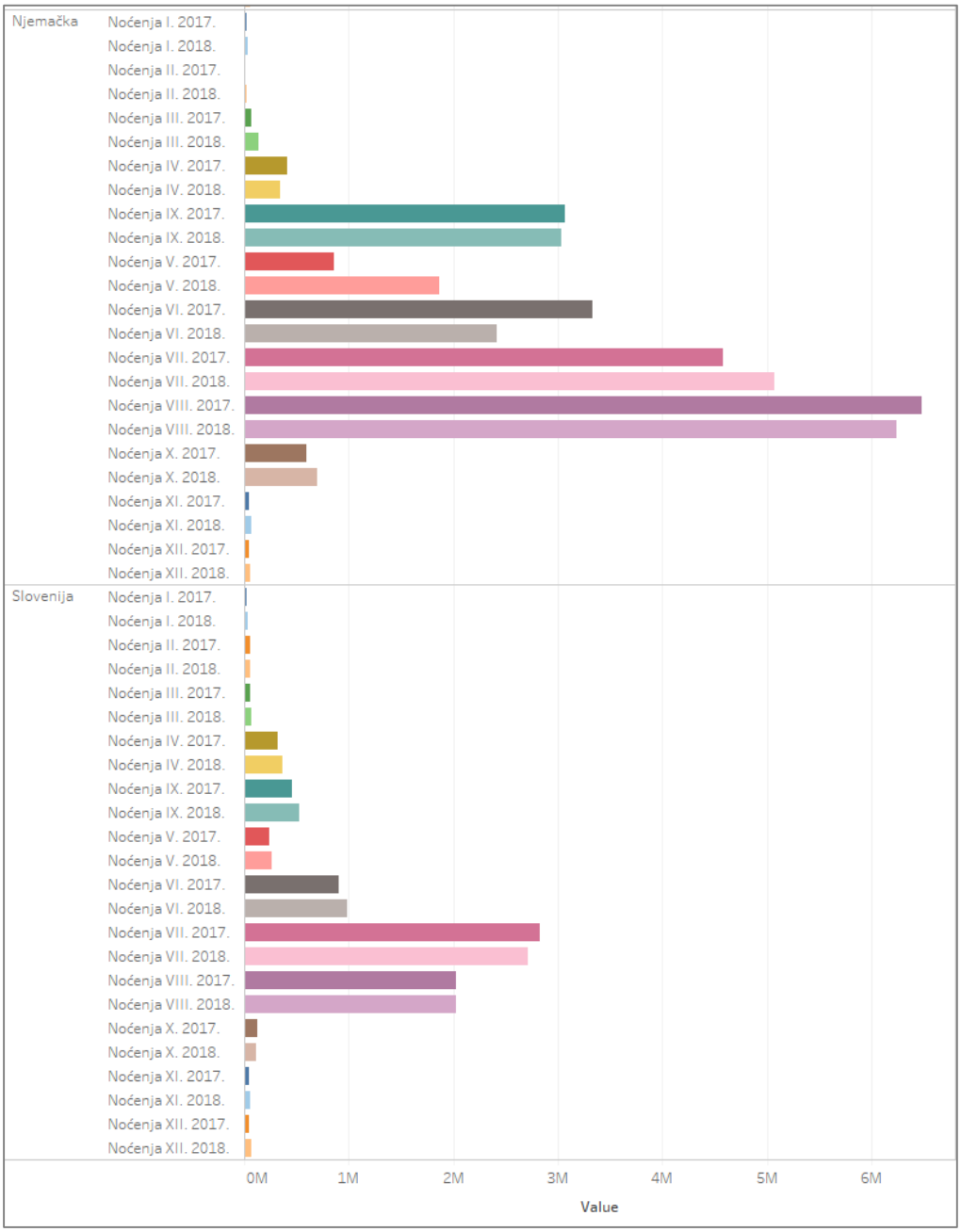


Slika 37 Noćenja stranih turista prema zemlji prebivališta 2017. i 2018. godine

Njemački turisti glavninu noćenja ostvare ljeti u srpnju i kolovozu, a broj noćenja u srpnju se povećao, dok se broj noćenja u kolovozu shodno tome smanjio. Veliki broj noćenja ostvari se i u post sezoni u runju, a broj turista u lipnju (predsezoni) opada. Austrijanci ostvare podjednak broj noćenja u srpnju i kolovozu, te u lipnju i runju kada ljetna sezona nije u punom jeku. Talijani najveći broj noćenja ostvare u kolovozu, a Slovenci u srpnju (Slika 38 i Slika 39).

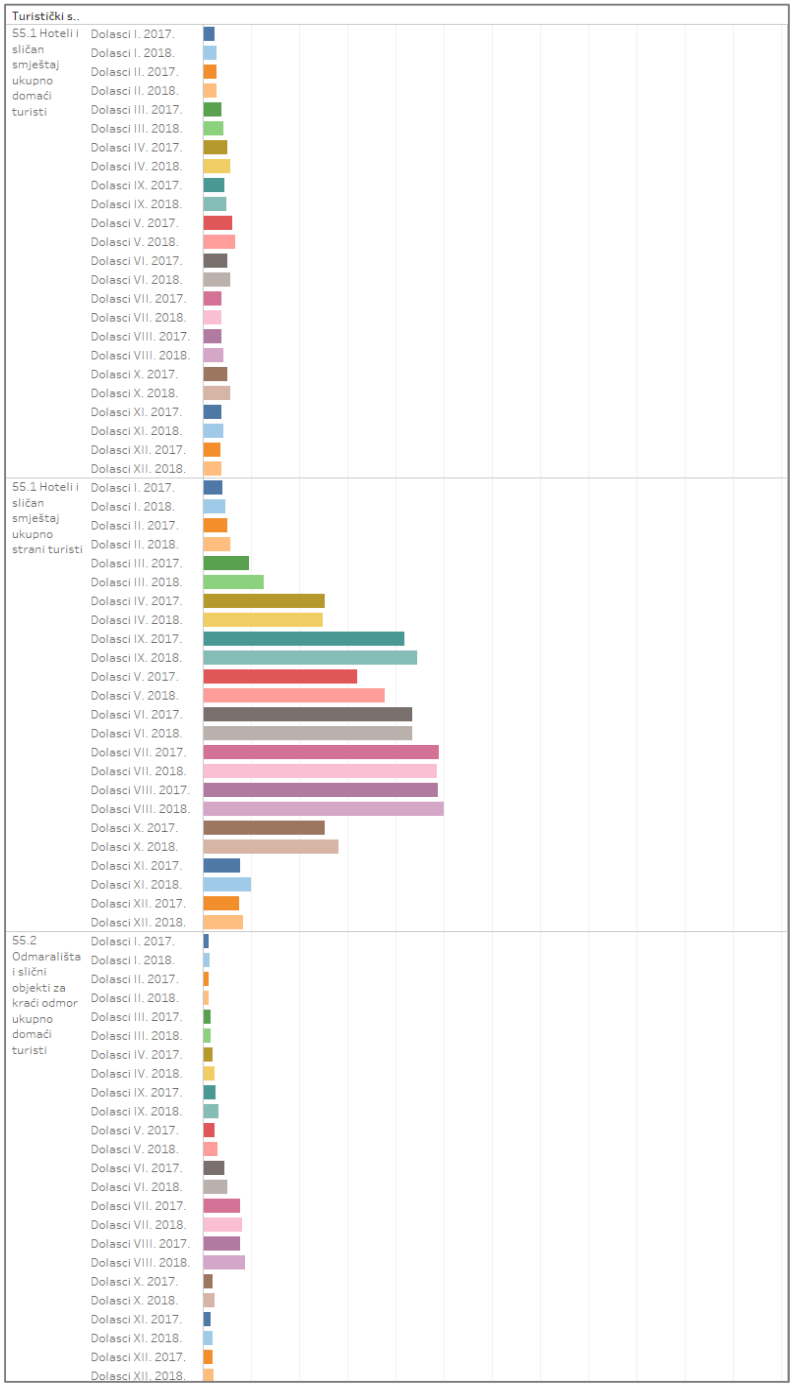


Slika 38 Noćenja stranih turista prema zemlji prebivališta po mjesecima (1)

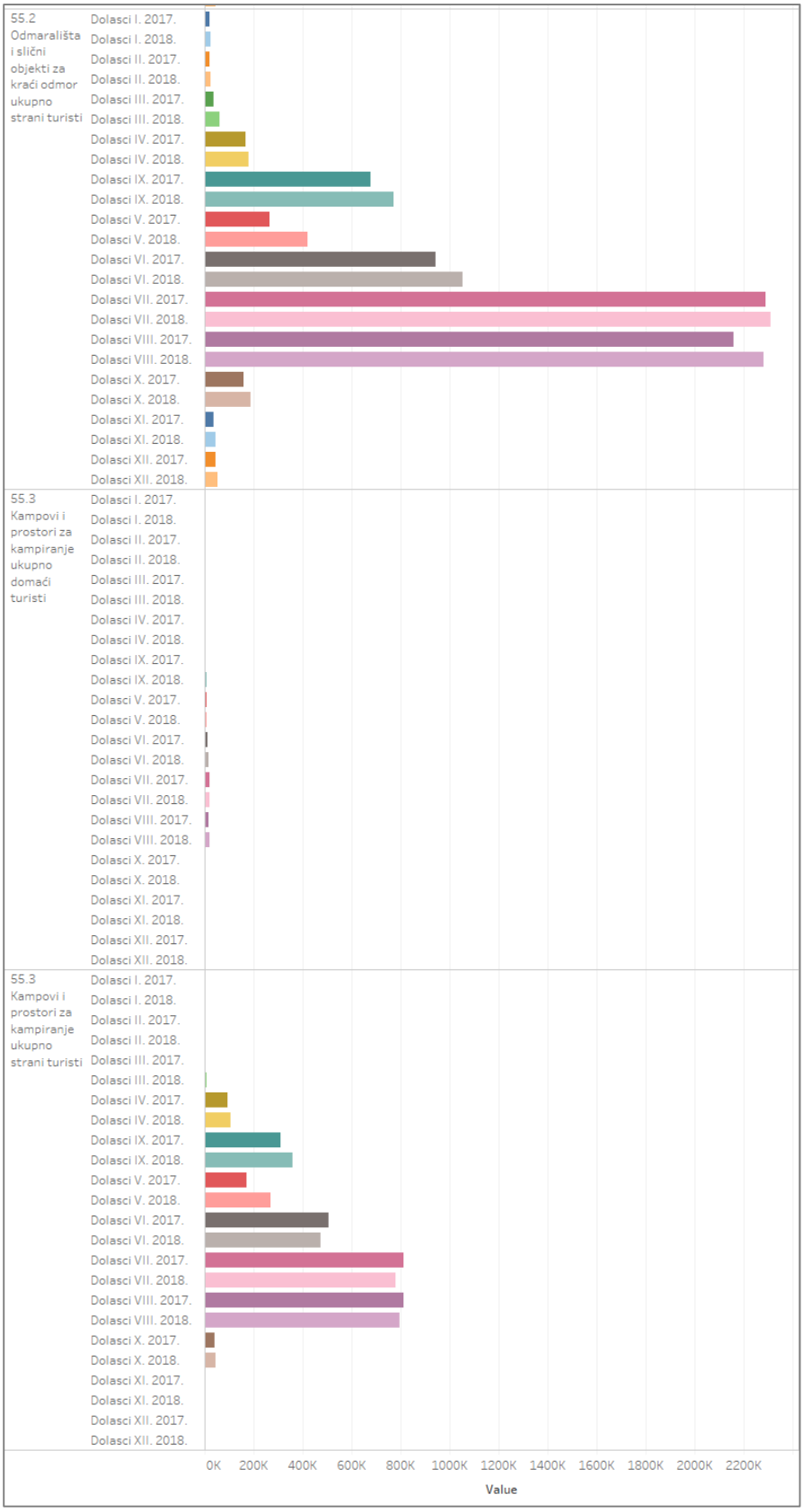


Slika 39 Noćenja stranih turista prema zemlji prebivališta po mjesecima (2)

Strani turisti najviše dolaze u odmarališta i slične objekte za kraći odmor te u hotele, a popularni su i kampovi naročito u ljetnim mjesecima. Najveći broj ih dolazi u srpnju i kolovozu u odmarališta, a broj dolazaka u hotele i kampove u tim mjesecima je duplo manji nego u odmarištima. Domaći turisti najviše dolaze u hotele i odmarališta. U srpnju i kolovozu veći broj ih dolazi u odmarališta dok je u travnju, svibnju i listopadu veći broj turista u hotelima. Domaćim turistima kampovi nisu zanimljivi te radije biraju hotele ili odmarališta (Slika 40 i Slika 41) .

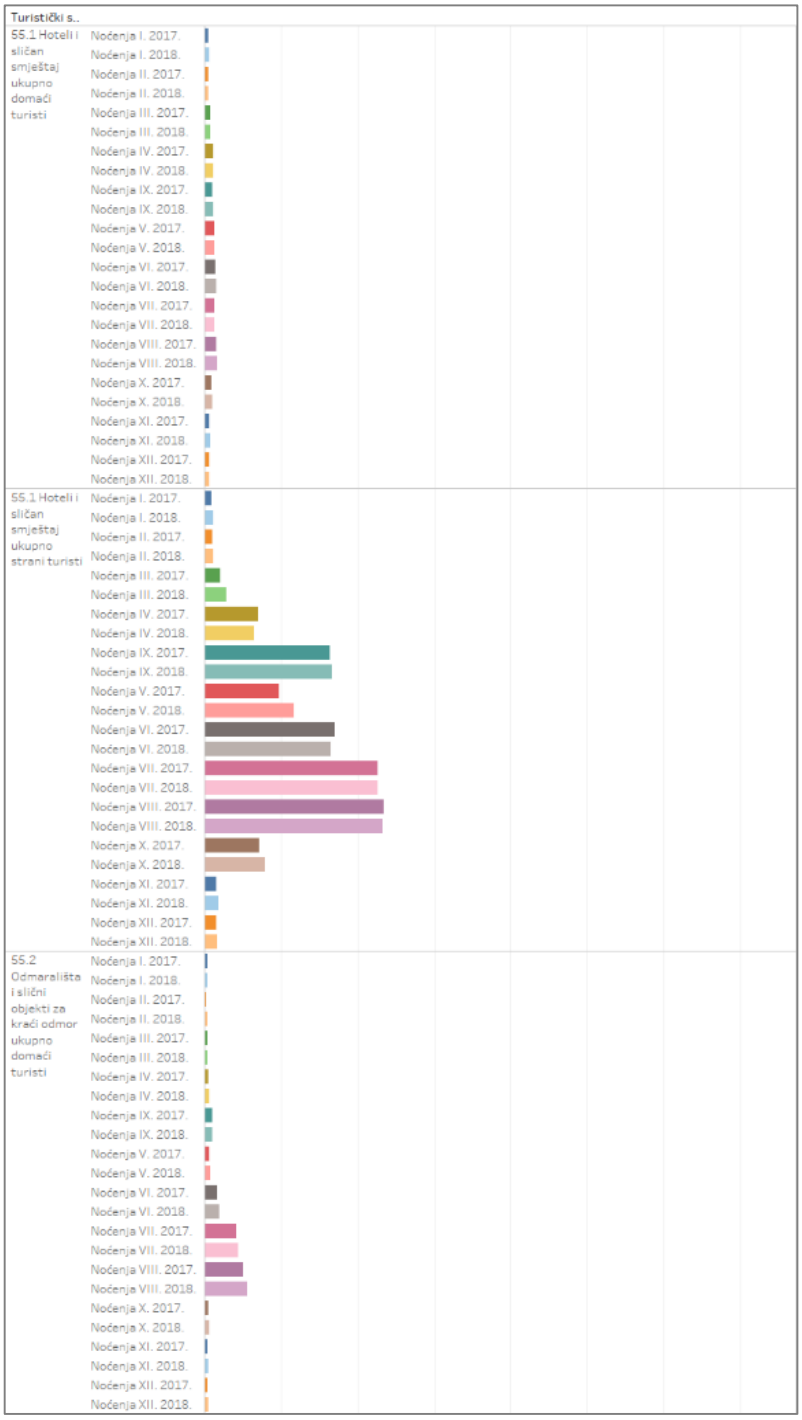


Slika 40 Broj dolazaka u turističke smještajne objekte (1)

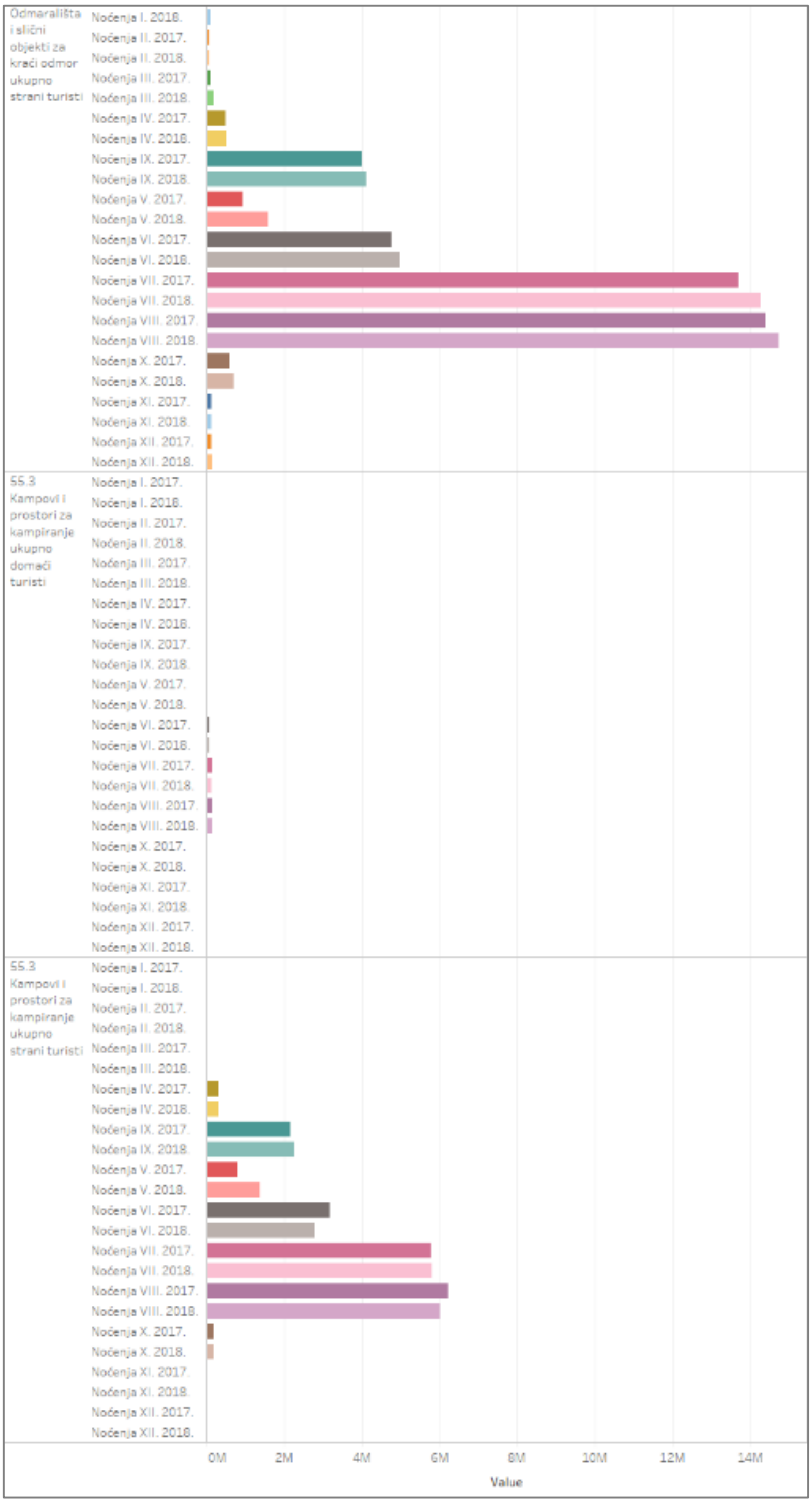


Slika 41 Broj dolazaka u turističke smještajne objekte (2)

Strani turisti u odmarališima u srpnju i kolovozu ostvare oko 14 milijuna noćenja mjesečno što bi značilo da tamo provedu u prosjeku 7 dana. U usporedbi sa njima, domaći turisti ostvare 800 000 do milijun noćenja, odnosno 9 noćenja u prosjeku. U hotelima se ostvari manji broj noćenja – oko 4,5 milijuna ostvare strani turisti, a domaći od 250 do 300 tisuća. U prosjeku strani turisti provedu 4 dana, a domaći 1. Kako su kampovi zanimljivi samo stranim turistima oni tu ljeti ostvare oko 6 milijuna noćenja po mjesecu, odnosno 7,5 dana u prosjeku za svakog turista (Slika 42 i Slika 43).

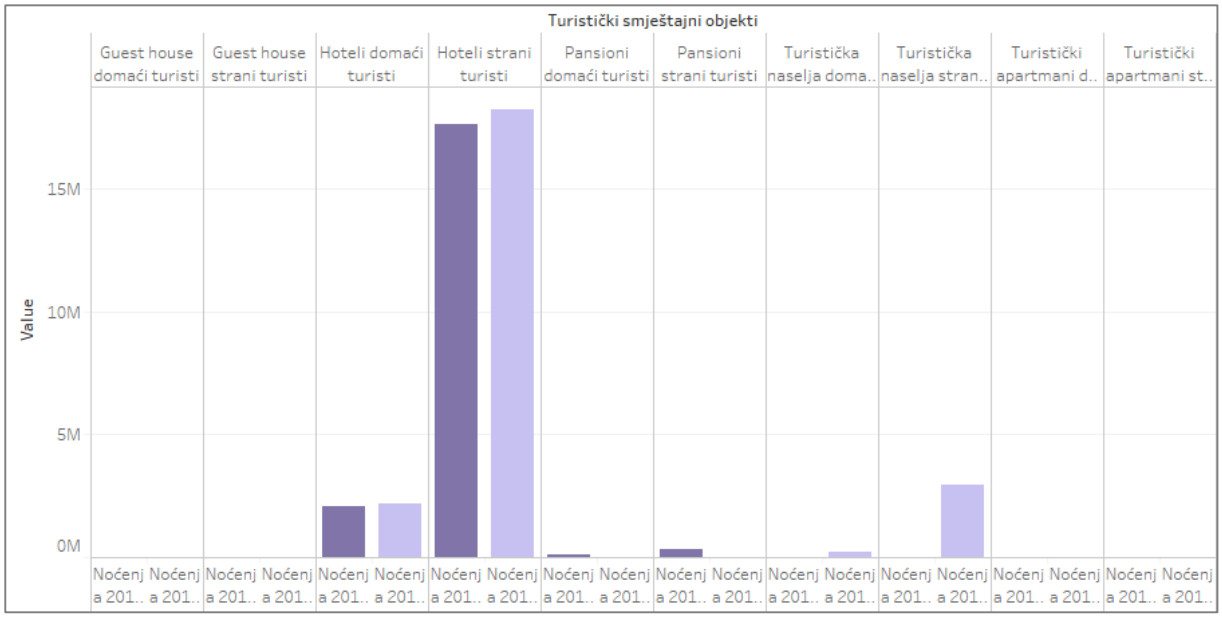


Slika 42 Broj noćenja po turističkim smještajnim objektima (1)



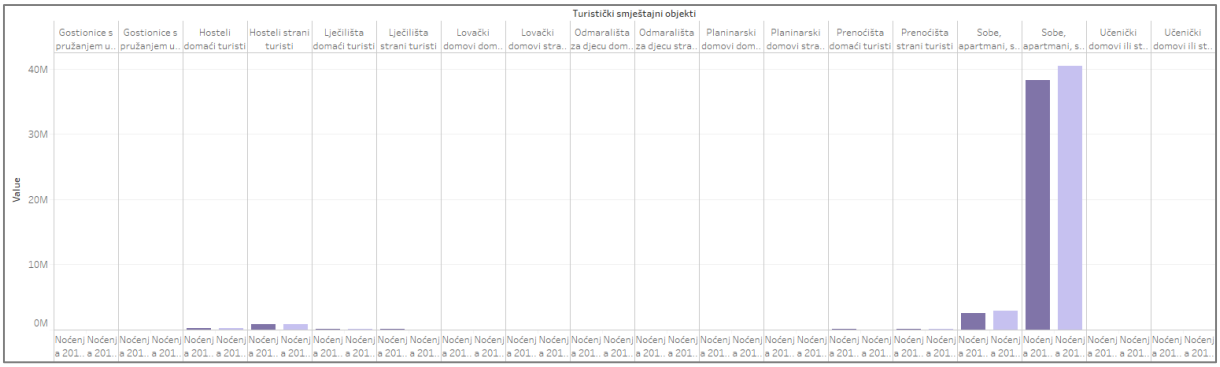
Slika 43 Broj noćenja po turističkim smještajnim objektima (2)

Kad se analizira vrsta hotelskog smještaja vidljivo je da i domaći i strani turisti najviše odsjedaju u hotelima te da je broj noćenja u blagom porastu. Stranim turistima postala su zanimljiva turistička naselja, a pansioni više ne interesiraju niti domaće niti strane turiste (Slika 44).



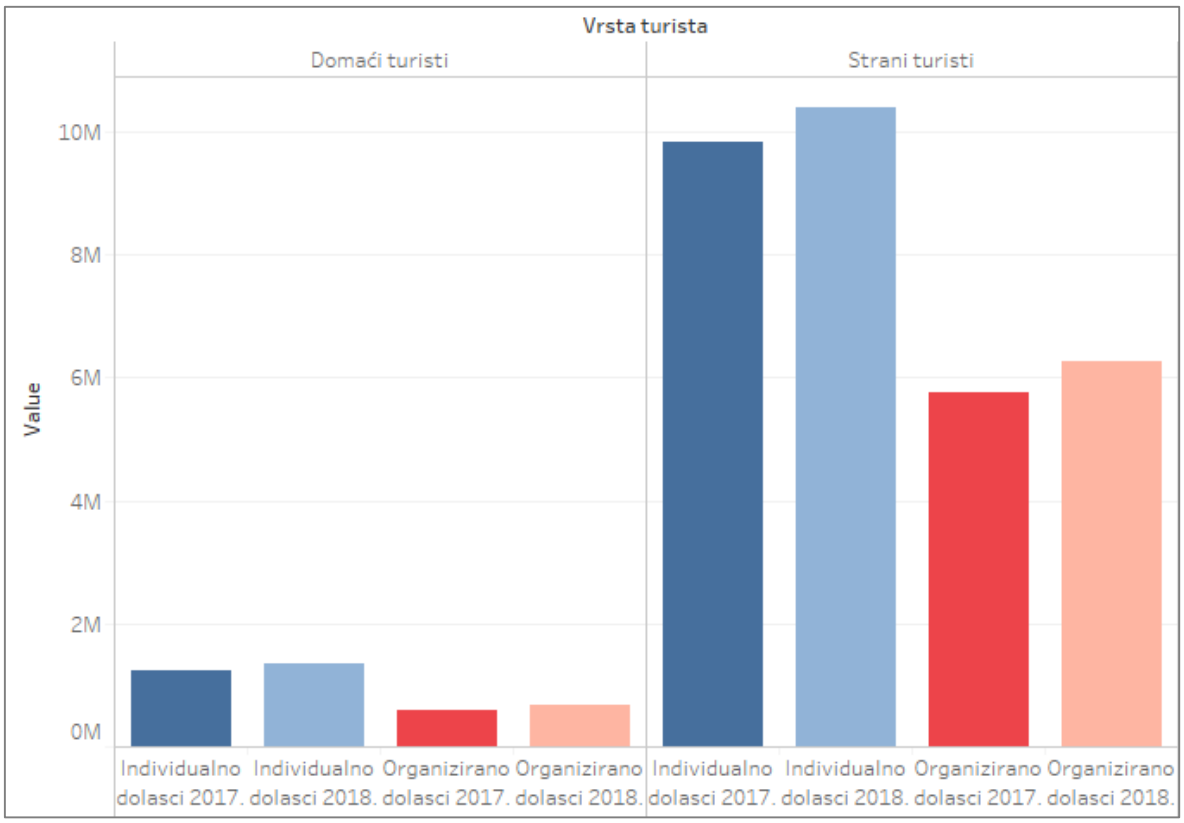
Slika 44 Noćenja domaćih i stranih turista u hotelima i sličnom smještaju 2017. i 2018. godine

Među odmarališima i sličnim objektima za odmor najpopularniji tip smještaja su sobe, apartmani, studio-apartmani i kuće. Mali broj domaćih i stranih turista odsjeda u hostelima, a ostali tip smještaja poput prenoćišta i lječilišta im nije toliko zanimljiv (Slika 45).



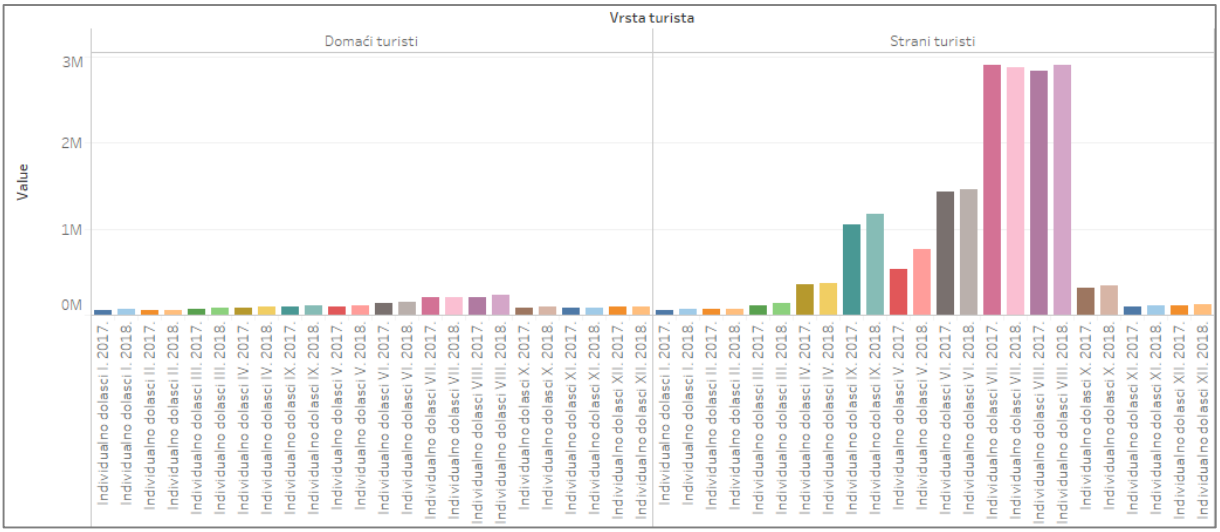
Slika 45 Noćenja domaćih i stranih turista u odmarališima i sličnim objektima za odmor 2017. i 2018. godine

Analizom načina na koji turisti dolaze vidimo da strani turisti većinom dolaze individualno, ali ih veliki broj dolazi i organizirano. Domaći turisti dolaze individualno, a organizirano ih dolazi upola manje (Slika 46).



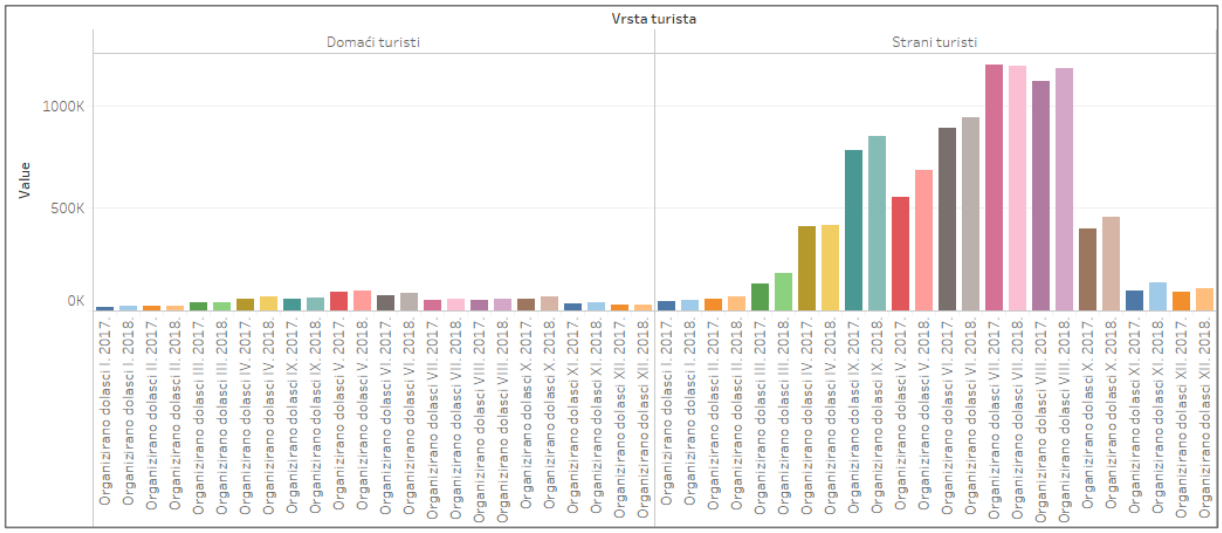
Slika 46 Individualni i organizirani način dolazaka turista prema vrstama turista

Strani turisti individualno najviše dolaze u srpnju i kolovozu te u predsezoni i post sezoni odnosno lipnju i rujnu. Domaći turisti dolaze u srcu sezone tj., srpnju i kolovozu (Slika 47).



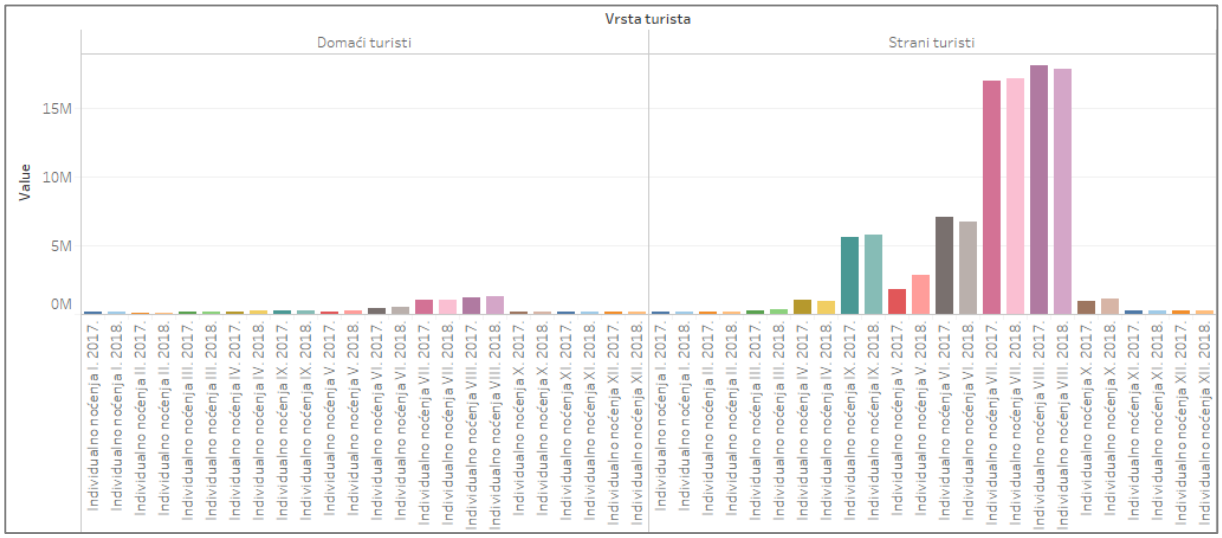
Slika 47 Individualni način dolazaka turista prema vrstama turista

Strani turisti intenzivno dolaze organizirano tokom cijele godine, jedino je taj broj smanjen zimi te se u proljeće opet polako povećava. Najveći broj organiziranih dolazaka domaćih turista je u svibnju, a broj dolazaka u lipnju se približava tom broju. Od ostalih mjeseca mogu se istaknuti travanj, rujan i listopad, te je pretpostavka da u tim mjesecima ljudi organizirano idu u obilaske određenih znamenitosti (Slika 48).



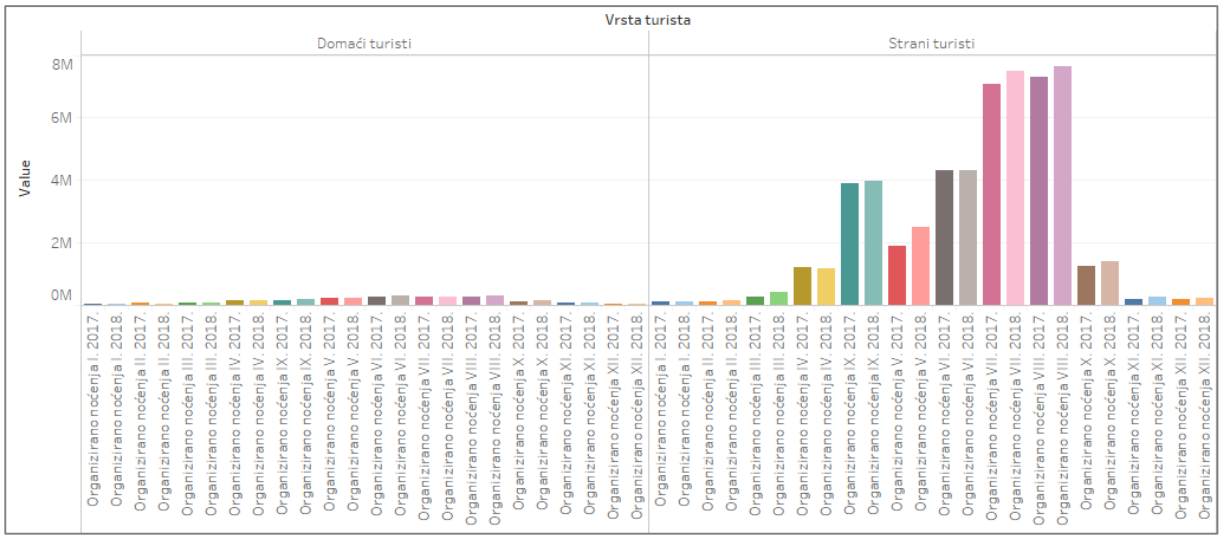
Slika 48 Organizirani način dolazaka turista prema vrstama turista

Domaći turisti koji su individualno došli jedino u ljetnim mjesecima noće te ostvare u prosjeku 5 noćenja. Strani turisti koji su došli u najfrekventnijim mjesecima ostvare oko 6 noćenja. Kako je stranih turista došlo više logično je da su ostvarili puno više noćenja (Slika 49).



Slika 49 Broj noćenja turista koji su došli individualno

Domaći turisti koji su došli organizirano ostvaruju vrlo mali broj noćenja. Strani turisti ostvaruju daleko veći broj noćenja, najviše ljeti kada u prosjeku ostvare 7 noćenja, a u lipnju i rujnu se broj noćenja smanji te ostvare u prosjeku 6 noćenja (Slika 50).



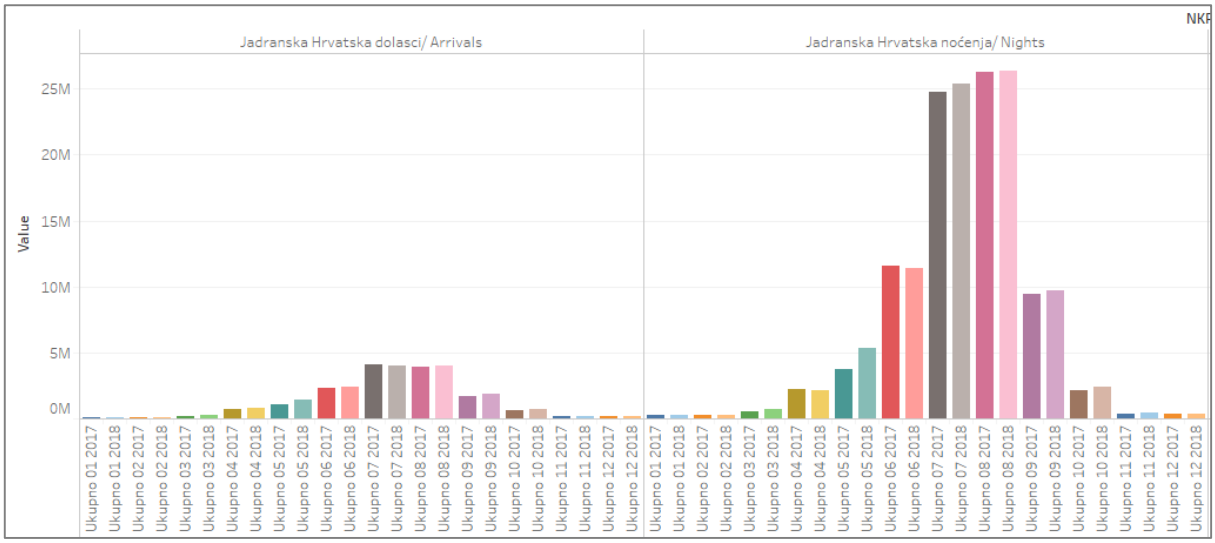
Slika 50 Broj noćenja turista koji su došli organizirano

Zanimljivo je istražiti u koji dio Hrvatske je došao najveći dio turista te se analiziraju dolasci i noćenja turista dvije razine prostornih jedinica za statistiku. Zbog statističkih potreba Hrvatska je podijeljena na tri razine prostornih jedinica za statistiku prema Nacionalnoj klasifikaciji prostornih jedinica za statistiku 2012.

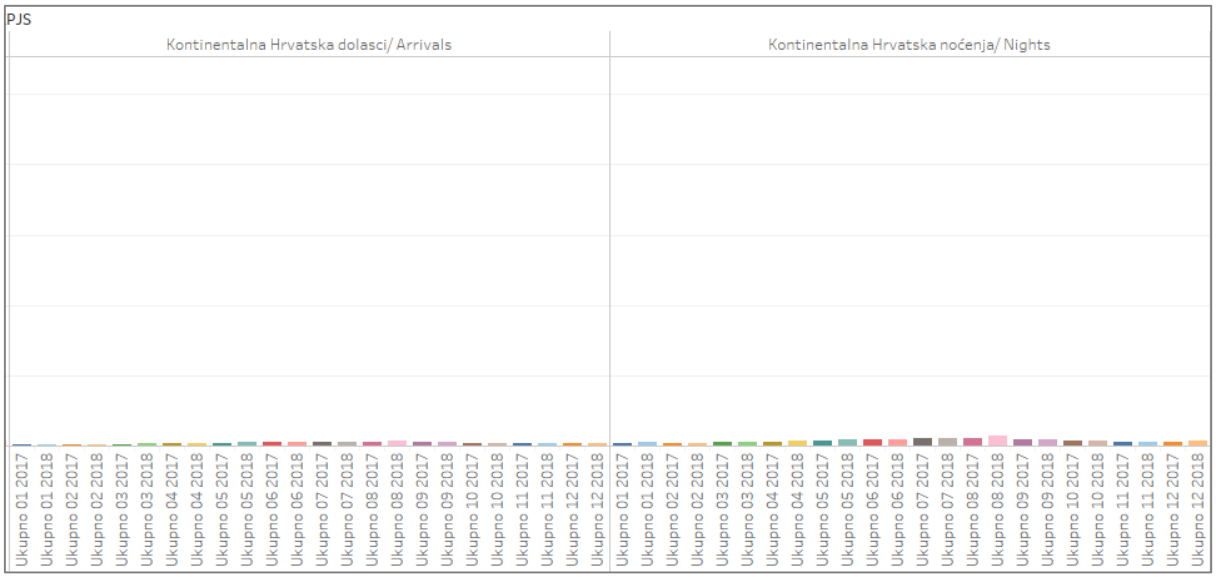
Nacionalna klasifikacija prostornih jedinica za statistiku 2012. (NKPJS 2012.) statistički je standard koji se koristi za prikupljanje, upisivanje, obradu, analizu i diseminaciju podataka regionalne statistike prema razinama prostorne podjele Republike Hrvatske. NKPJS 2012. je hijerarhijska klasifikacija kojom se uspostavljaju prostorne jedinice za statistiku 1., 2. i 3. razine prema kojima se dijeli prostor Republike Hrvatske za svrhe regionalne statistike.

- Prostorna jedinica za statistiku 1. razine je Republika Hrvatska kao administrativna jedinica.
- Prostorne jedinice za statistiku 2. razine sastoje se od 2 neadministrativne jedinice nastale grupiranjem županija kao administrativnih jedinica niže razine.
- Prostorne jedinice za statistiku 3. razine sastoje se od 21 administrativne jedinice (20 županija i Grad Zagreb) [7].

Prostorne jedinice za statistiku 2. razine su Kontinentalna i Jadranska Hrvatska. Na grafikonu se može vidjeti kako gotovo svi turisti dolaze u Jadransku Hrvatsku i tu ostvaruju oko 26 milijuna noćenja za vrijeme sezone. Može se vidjeti kako broj dolazaka raste od travnja i u srpnju i kolovozu dosegne svoj maksimum, te se u rujnu ponovno smanjuje. Isto vrijedi i za broj noćenja kojih je u Jadranskoj Hrvatskoj ljeti oko 26 milijuna, a u Kontinentalnoj oko 500 tisuća (Slika 51 i Slika 52).

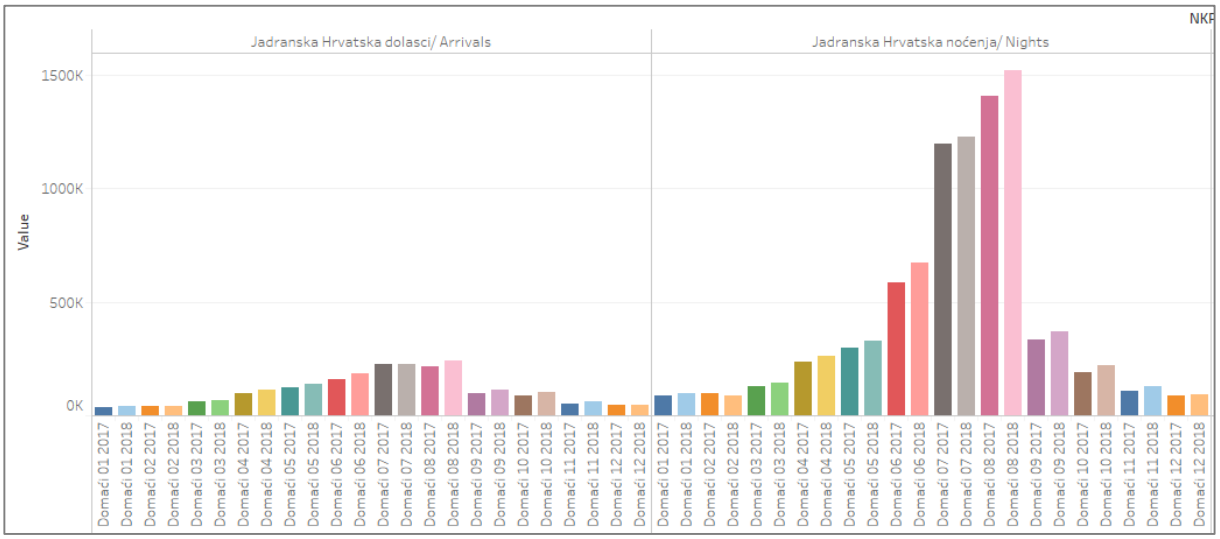


Slika 51 Broj dolazaka i noćenja turista po mjesecima prema NKPJS 2. razine (1)

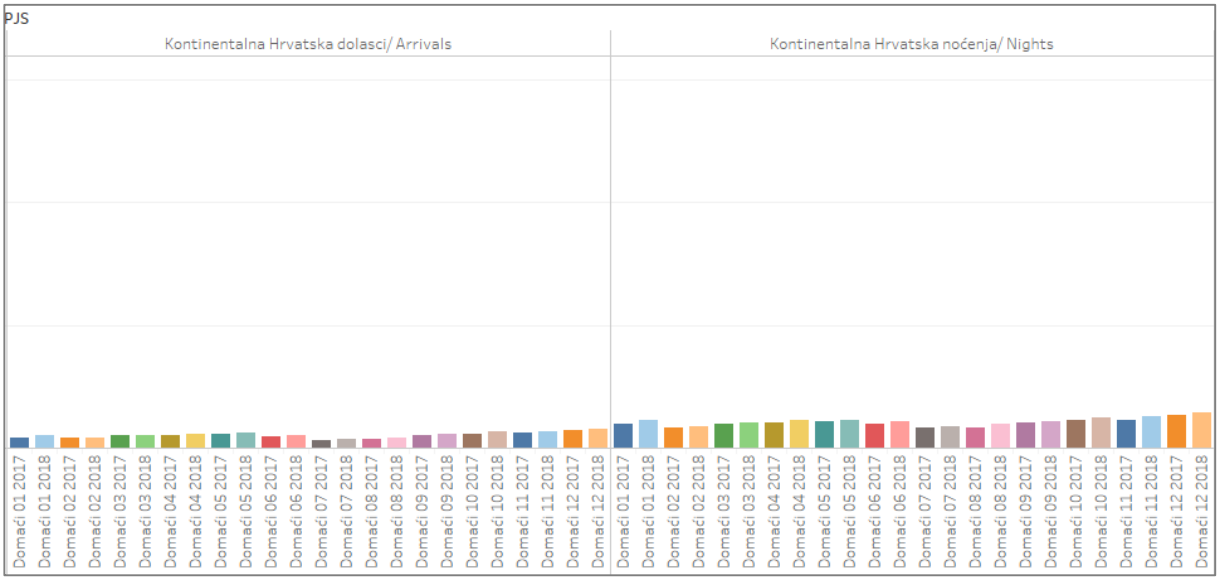


Slika 52 Broj dolazaka i noćenja turista po mjesecima prema NKPJS 2. razine (2)

Najveći broj domaćih turista u Jadranskoj Hrvatskoj ostvari se u kolovozu kada 250 000 turista ostvari oko 1,5 milijuna noćenja što je u prosjeku 6 noćenja po osobi. Veći broj dolazaka i noćenja ostvare zimi i u proljeće kada idu na skijanje ili uživanje u prirodi. Najveći broj dolazaka i noćenja u zimskom periodu postigne se u prosincu i siječnju kada je sezona skijanja, a u proljeće u travnju i svibnju (Slika 53 i Slika 54).

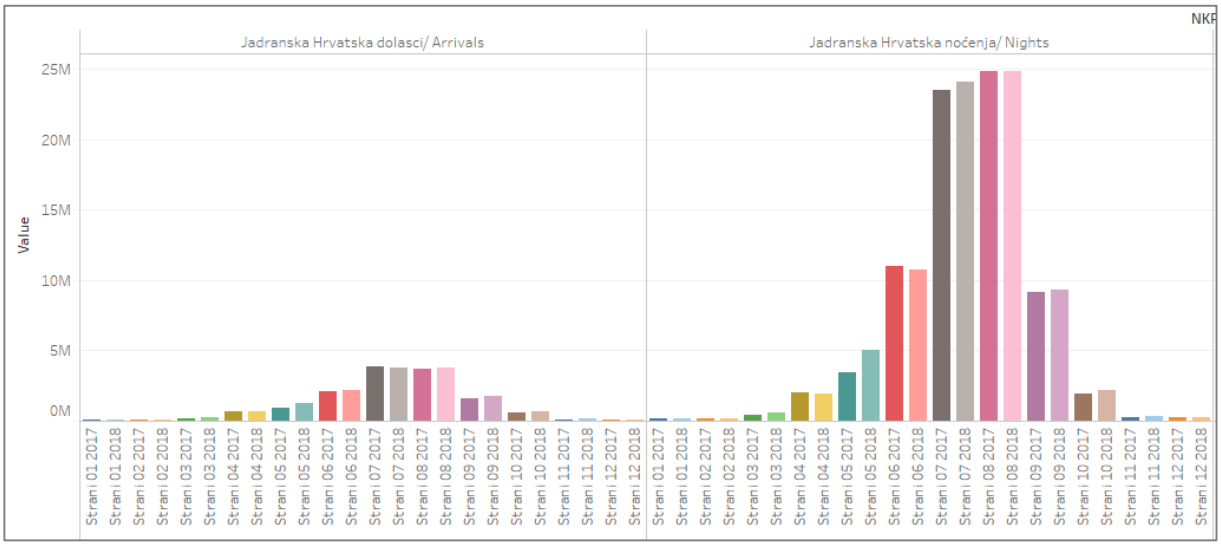


Slika 53 Broj dolazaka i noćenja domaćih turista po mjesecima prema NKPJS 2. razine (1)

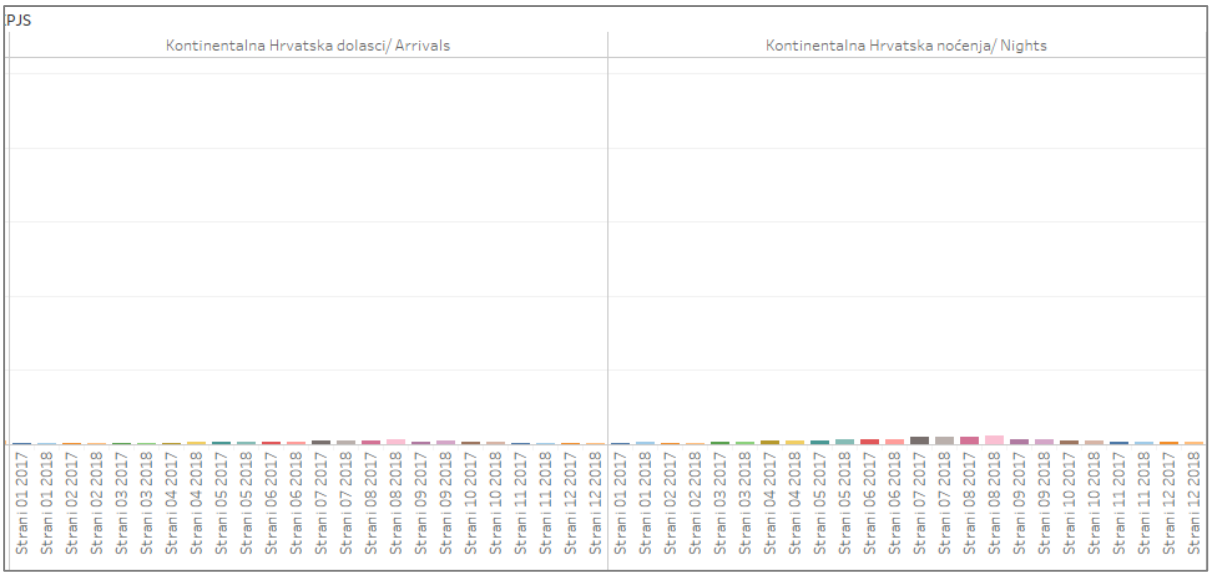


Slika 54 Broj dolazaka i noćenja domaćih turista po mjesecima prema NKPJS 2. razine (2)

Strani turisti koji posjete Jadransku Hrvatsku kao i domaći turisti ostvare najveći broj dolazaka i noćenja u kolovozu i srpnju, te se broj turista počne naglo povećavati u travnju, a počne opadati u rujnu kada se broj dolazaka i noćenja prepolovi. Kontinentalna Hrvatska im također nije toliko zanimljiva za posjetiti ali ju za razliku od domaćih turista najviše posjećuju u srpnju i kolovozu, dok je broj posjeta zimi vrlo malen (Slika 55 i Slika 56).

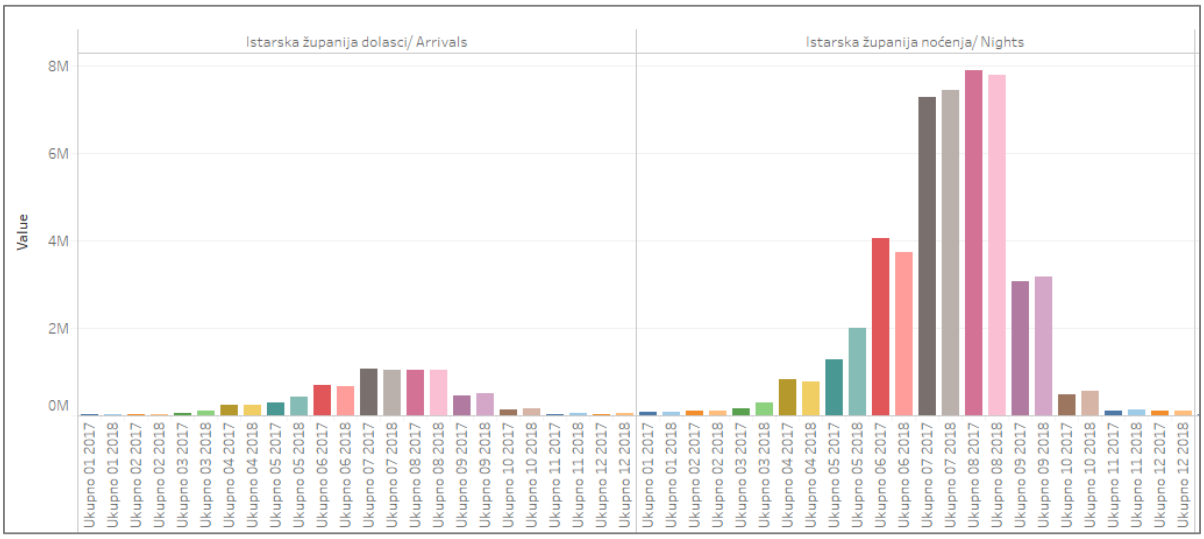


Slika 55 Broj dolazaka i noćenja stranih turista po mjesecima prema NKPJS 2. razine (1)

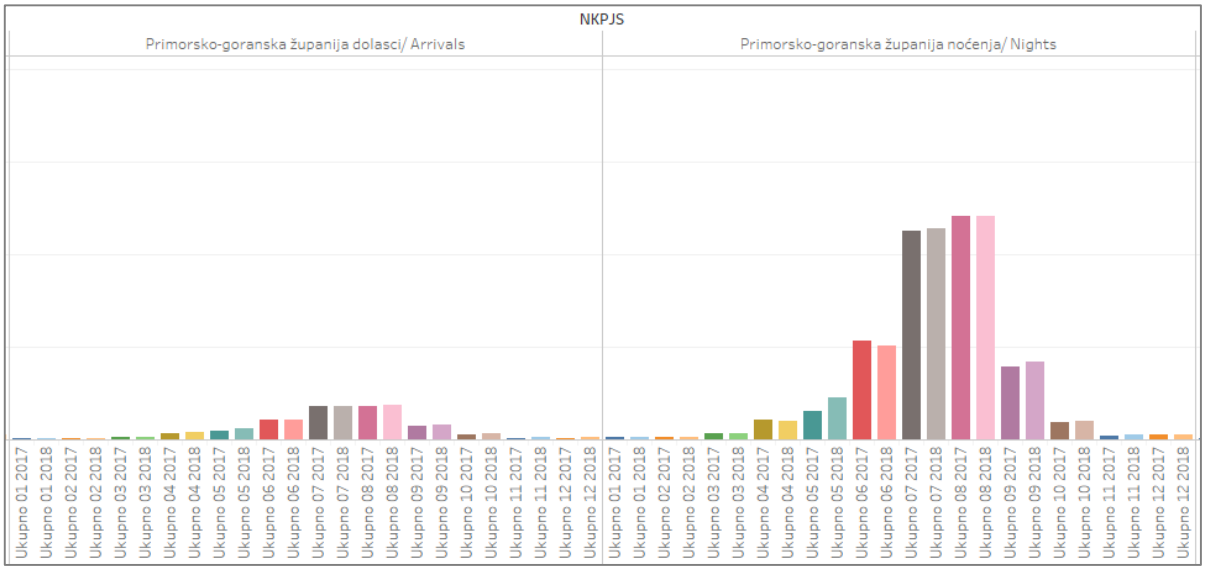


Slika 56 Broj dolazaka i noćenja stranih turista po mjesecima prema NKPJS 2. razine (2)

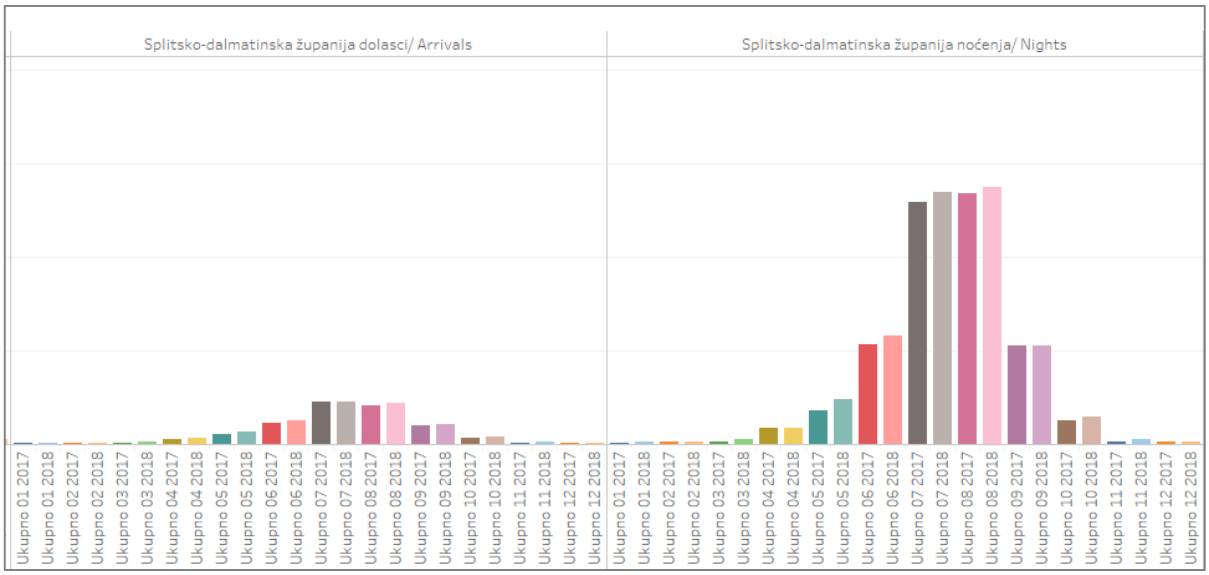
U Jadranskoj Hrvatskoj županija sa najvećim brojem dolazaka i noćenja je Istarska županija koja u srpnju i kolovozu ostvari oko milijun dolazaka i blizu 8 milijuna noćenja. Druge dvije županije koje ostvare veliki broj dolazaka i noćenja su Primorsko-goranska županija te Splitsko-dalmatinska koje su najposjećenije u ljetnim mjesecima (Slika 57 do Slika 59).



Slika 57 Broj dolazaka i noćenja turista Jadranske Hrvatske na razini županija (1)

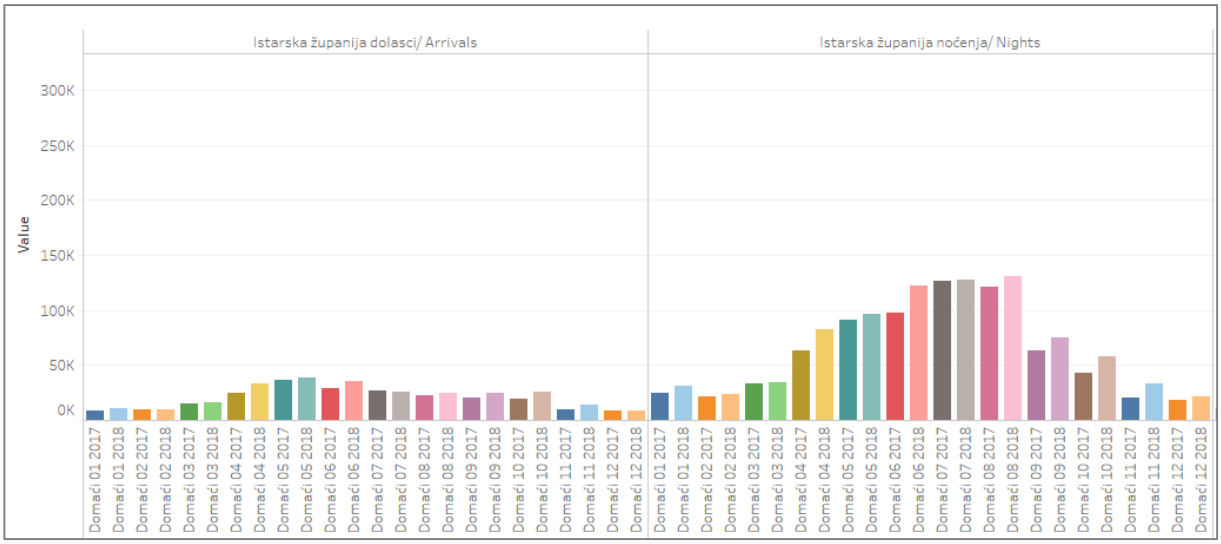


Slika 58 Broj dolazaka i noćenja turista Jadranske Hrvatske na razini županija (2)

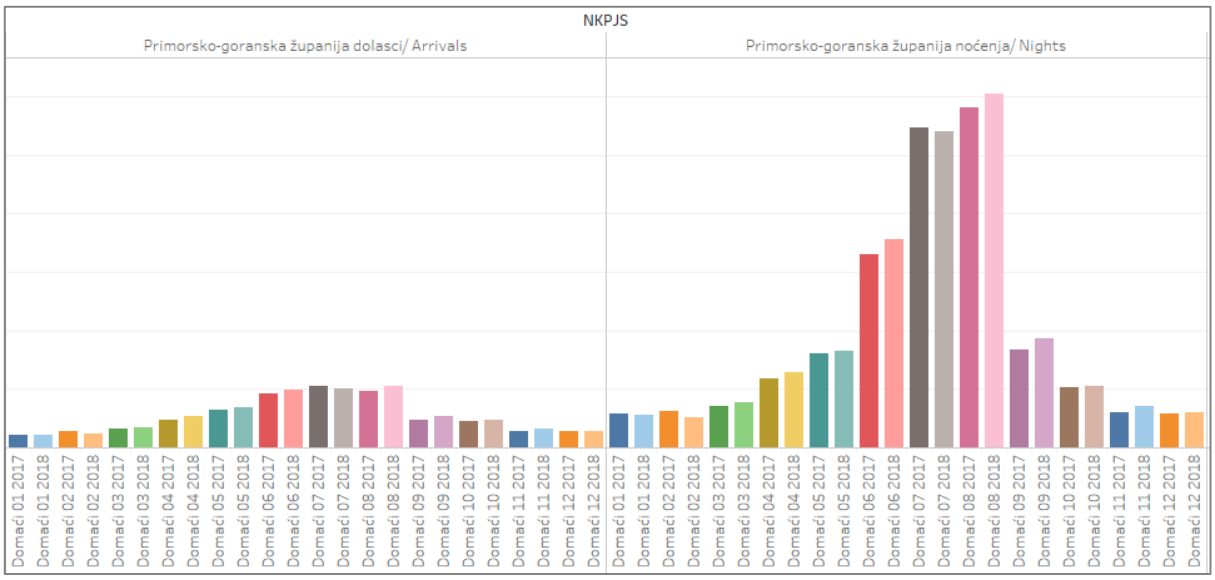


Slika 59 Broj dolazaka i noćenja turista Jadranske Hrvatske na razini županija (3)

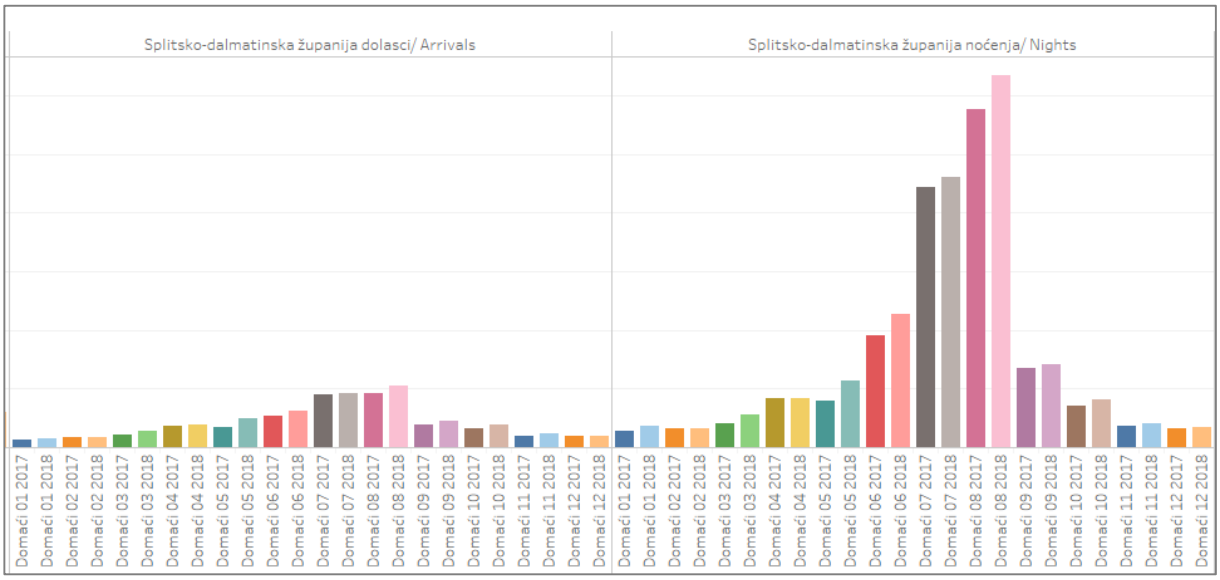
Domaći turisti u Istarsku županiju najviše dolaze od travnja do svibnja ali najveći broj noćenja ostvaruju od svibnja do kolovoza kada se najduže zadržavaju u toj županiji. U Primorsko-goranskoj županiji najviše dolazaka i noćenja ostvari se od lipnja do kolovoza, a sličan trend je i u Splitsko-Dalmatinskoj županiji u kojoj je ujedno ostvaren najveći broj noćenja u ovom djelu Hrvatske i to u kolovozu (Slika 60 do Slika 62).



Slika 60 Broj dolazaka i noćenja domaćih turista Jadranske Hrvatske na razini županija (1)

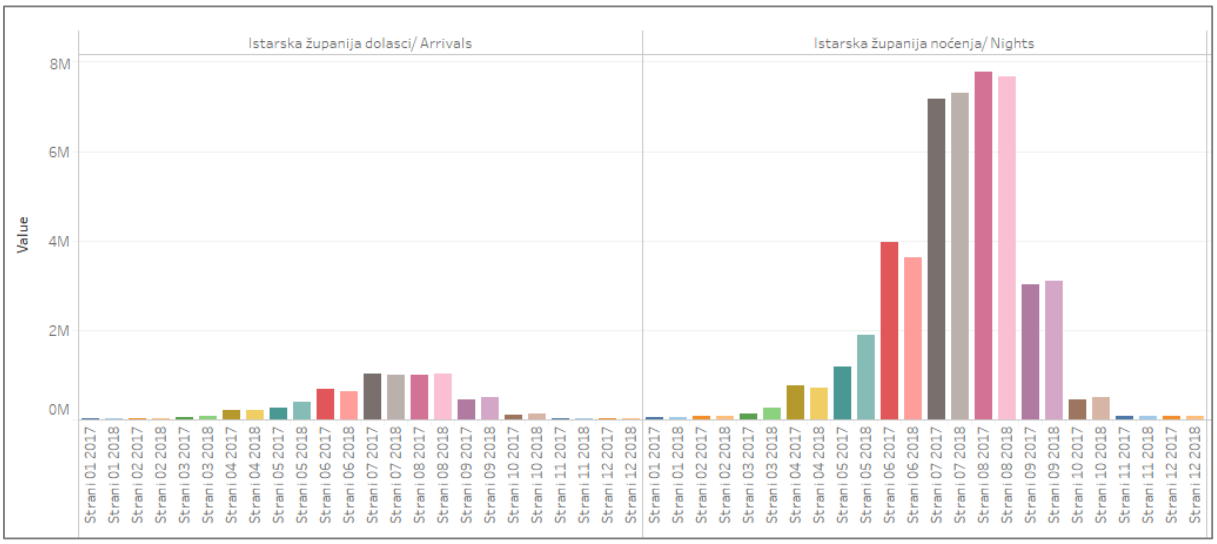


Slika 61 Broj dolazaka i noćenja domaćih turista Jadranske Hrvatske na razini županija (2)

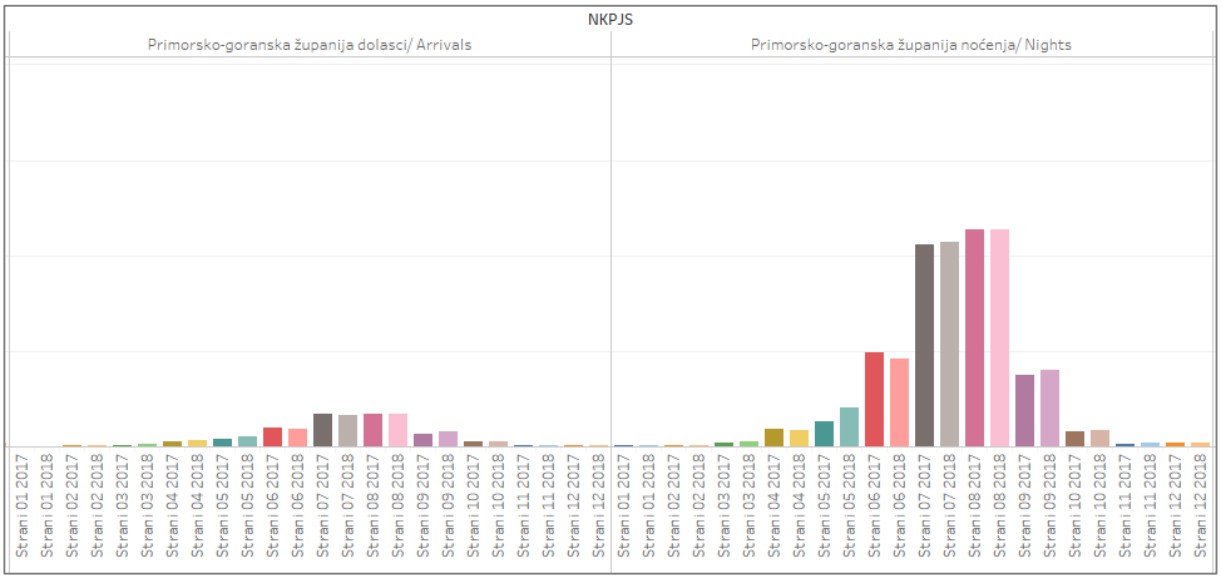


Slika 62 Broj dolazaka i noćenja domaćih turista Jadranske Hrvatske na razini županija (3)

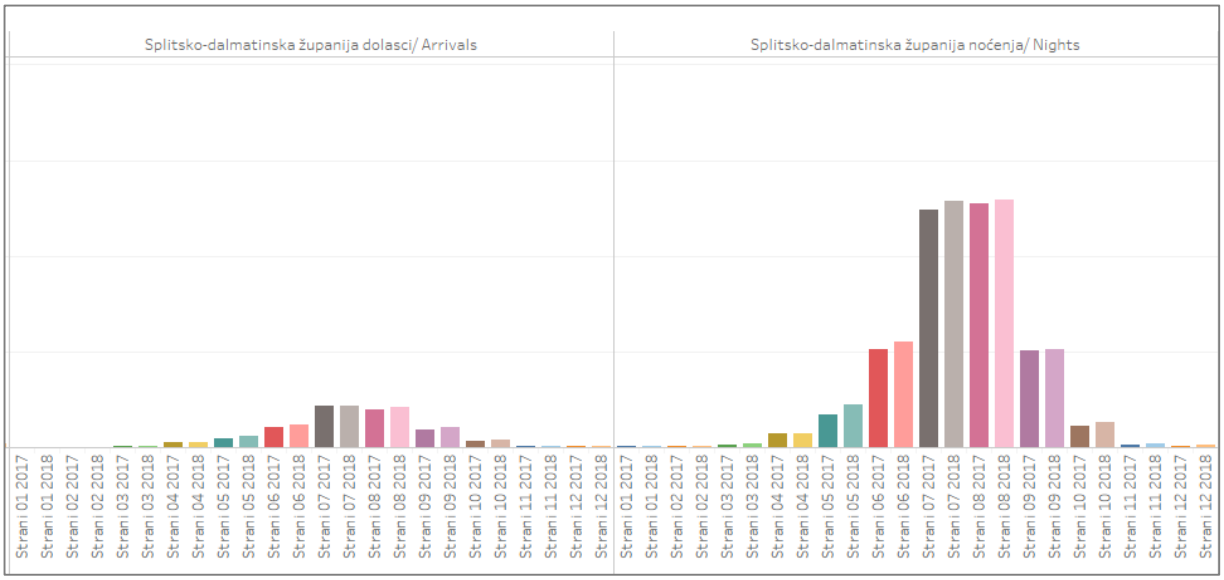
Strani turisti u sve tri županije najviše dolaze u ljetnim mjesecima, pogotovo u srpnju i kolovozu. Najveći broj noćenja ostvari se u Istarskoj županiji, dok je u Primorsko-goranskoj i Splitsko-Dalmatinskoj taj broj približno jednak (Slika 63 do Slika 65).



Slika 63 Broj dolazaka i noćenja stranih turista Jadranske Hrvatske na razini županija (1)



Slika 64 Broj dolazaka i noćenja stranih turista Jadranske Hrvatske na razini županija (2)

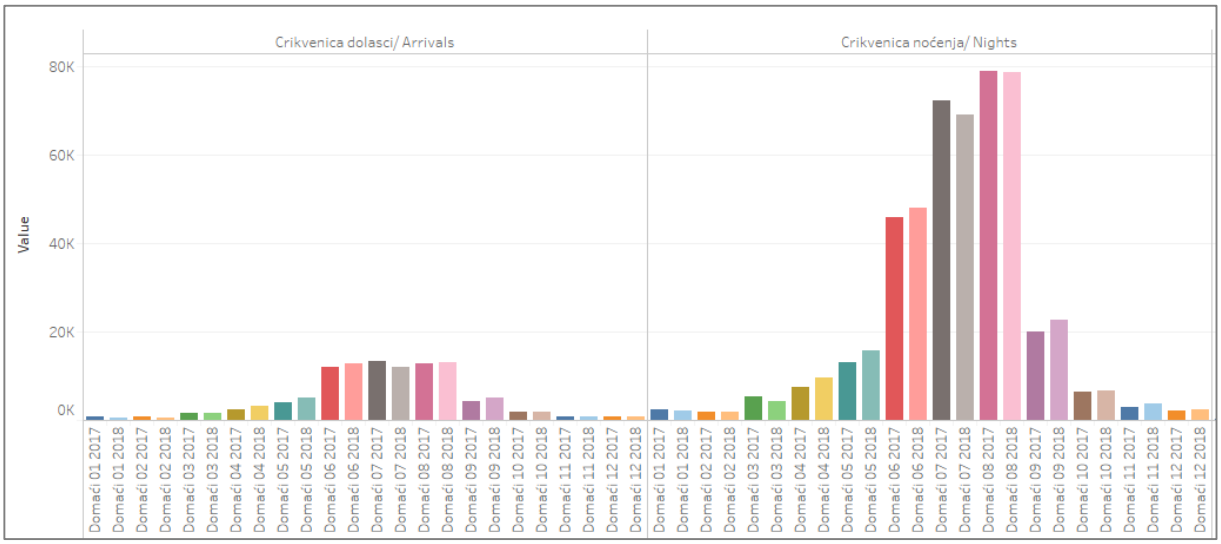


Slika 65 Broj dolazaka i noćenja stranih turista Jadranske Hrvatske na razini županija (3)

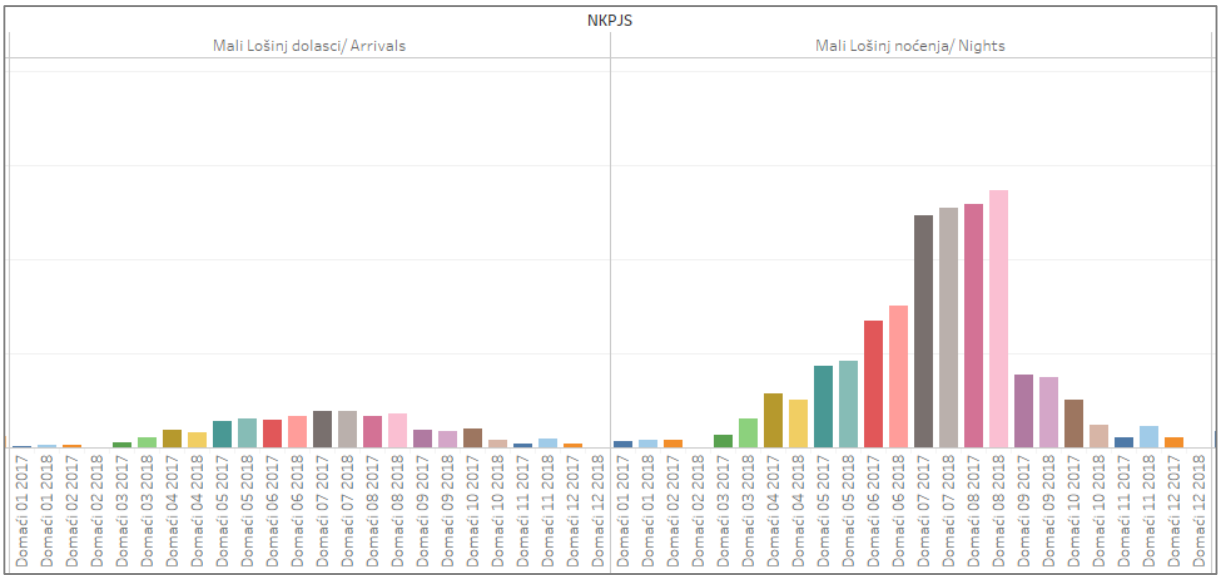
Kako je domaćih turista najviše u Primorsko-goranskoj županiji provedena je analiza u kojim gradovima i općinama je najveći broj dolazaka i noćenja.

Opatija ima najveći broj dolazaka te su dolasci intenzivniji u svibnju i listopadu, a ljeti ih ima nešto manje. Shodno tome i broj noćenja je veći u proljeće i jesen nego ljeti, a najveći broj noćenja ostvari se u svibnju i lipnju ali je broj noćenja u lipnju počeo opadati (Slika 68).

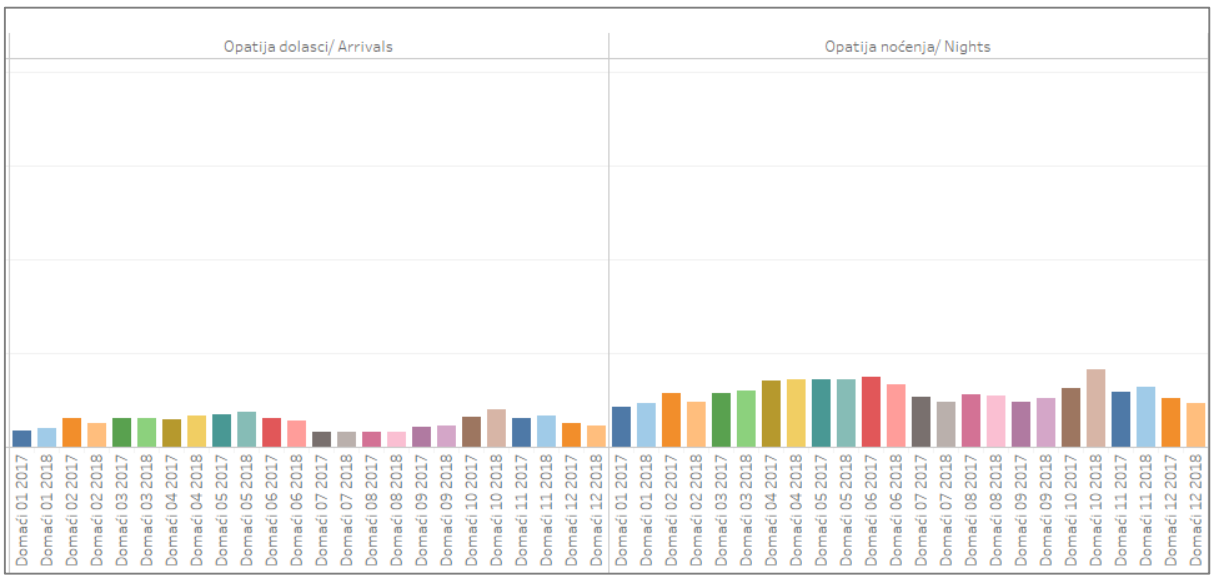
U Crikvenici i Malom Lošinju broj dolazaka i noćenja raste od travnja te je u srpnju i kolovozu najveći broj, a zatim u rujnu počinje opadati. Crikvenica je grad koji je ostvario najveći broj noćenja u Primorsko-goranskoj županiji (Slika 66 i Slika 67).



Slika 66 Broj dolazaka i noćenja domaćih turista u gradovima PGŽ (1)

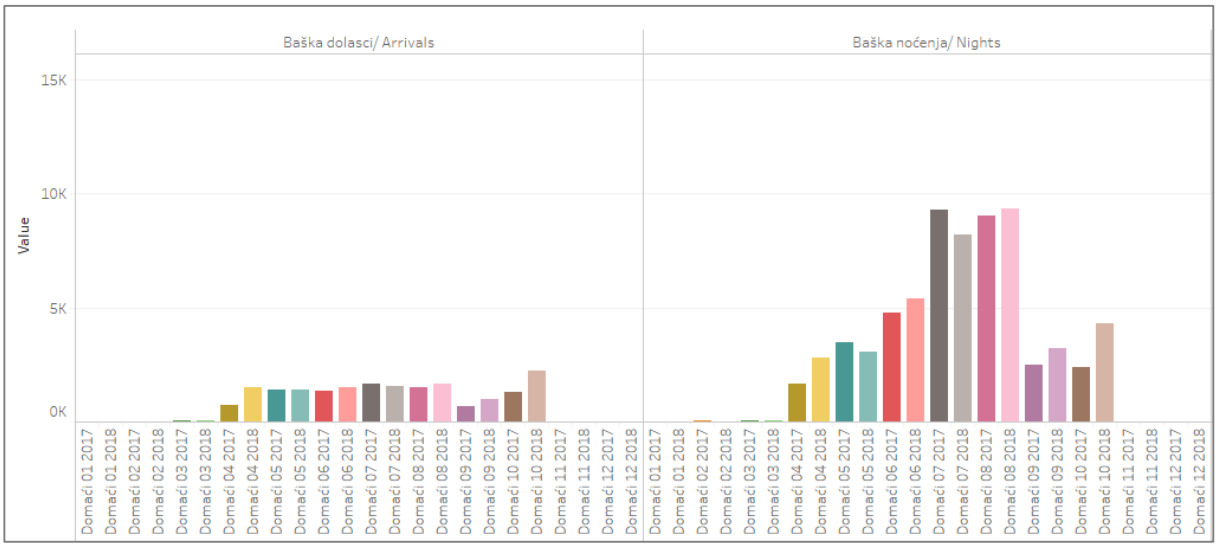


Slika 67 Broj dolazaka i noćenja domaćih turista u gradovima PGŽ (2)

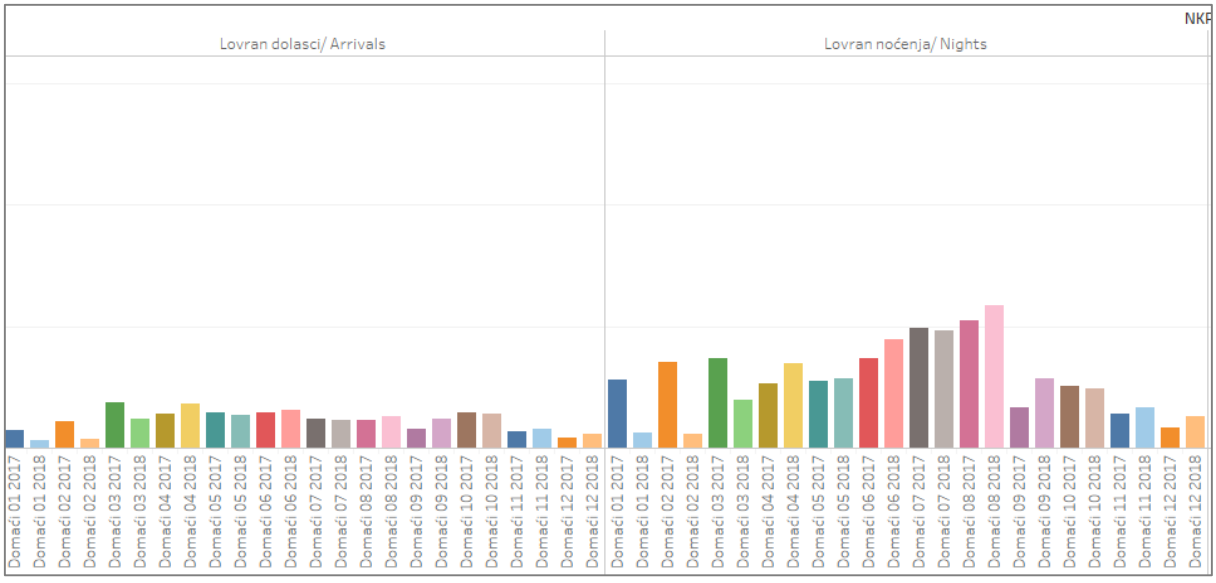


Slika 68 Broj dolazaka i noćenja domaćih turista u gradovima PGŽ (3)

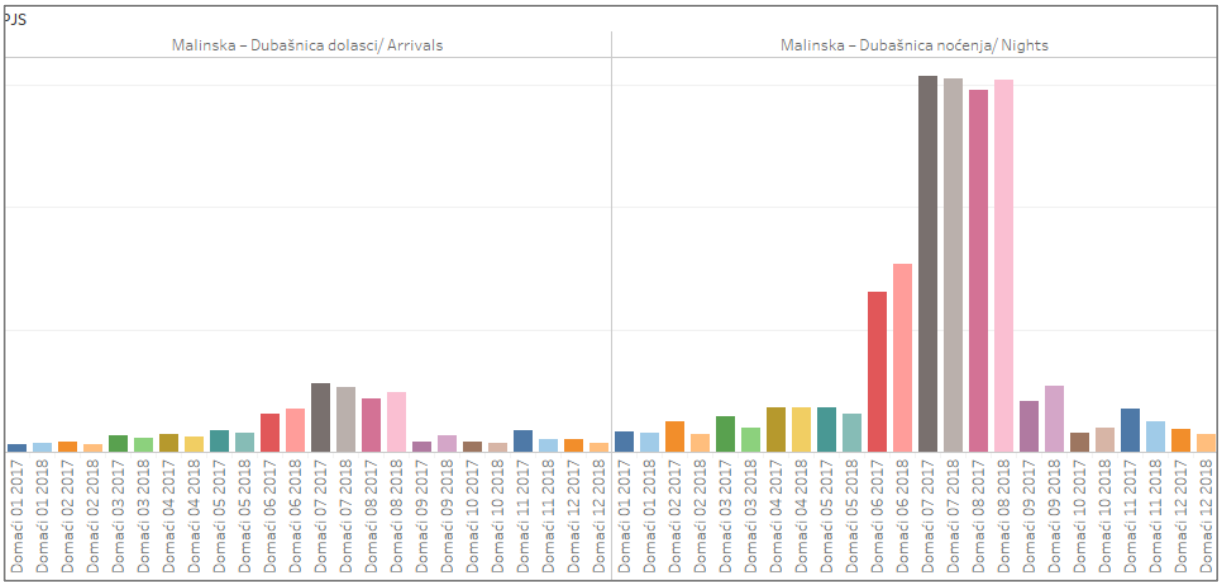
Općina Lovran ostvarila je najveći broj dolazaka domaćih turista te se 2017. godine ističe broj dolazaka u ožujku i travnju. Iste godine u siječnju, veljači i ožujku ostvaren je veliki broj noćenja, ali se najveći broj noćenja ipak postigne u ljetnim mjesecima. Malinska i Omišalj imaju identičnu frekvenciju dolazaka i noćenja, ali je Malinska ipak ostvarila najveći broj noćenja od svih općina. Baška ima zanimljivu frekvenciju dolazaka i noćenja – dolasci i noćenja ostvaruju se od travnja do listopada, a u ostalim mjesecima broj dolazaka i noćenja je vrlo malen (Slika 69 do Slika 72).



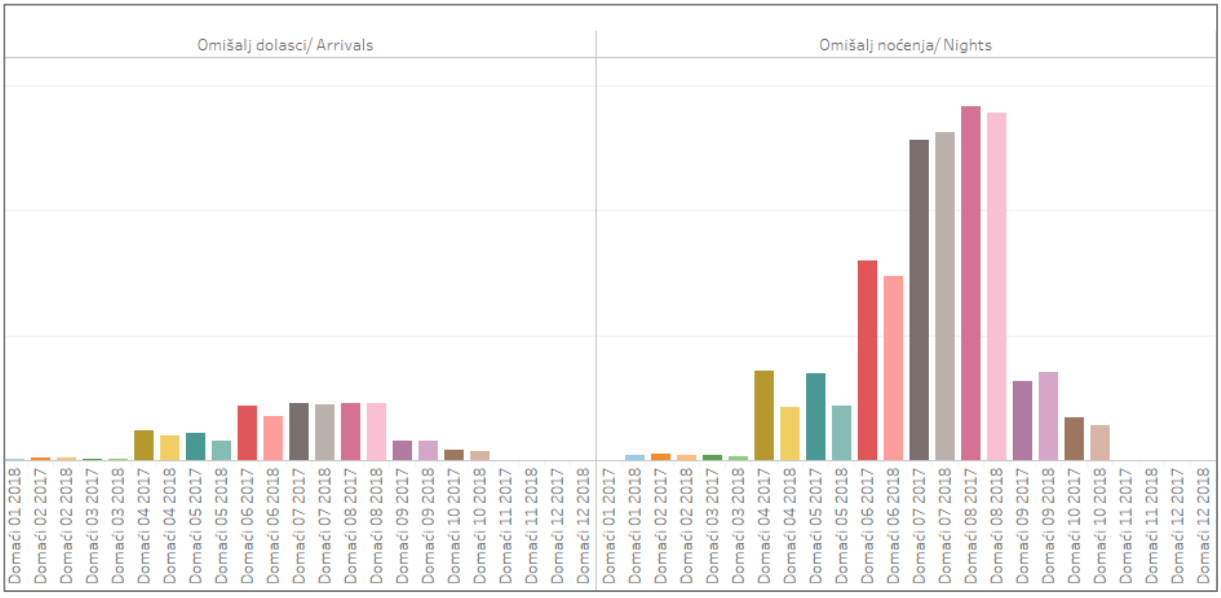
Slika 69 Broj dolazaka i noćenja domaćih turista u općinama PGŽ (1)



Slika 70 Broj dolazaka i noćenja domaćih turista u općinama PGŽ (2)

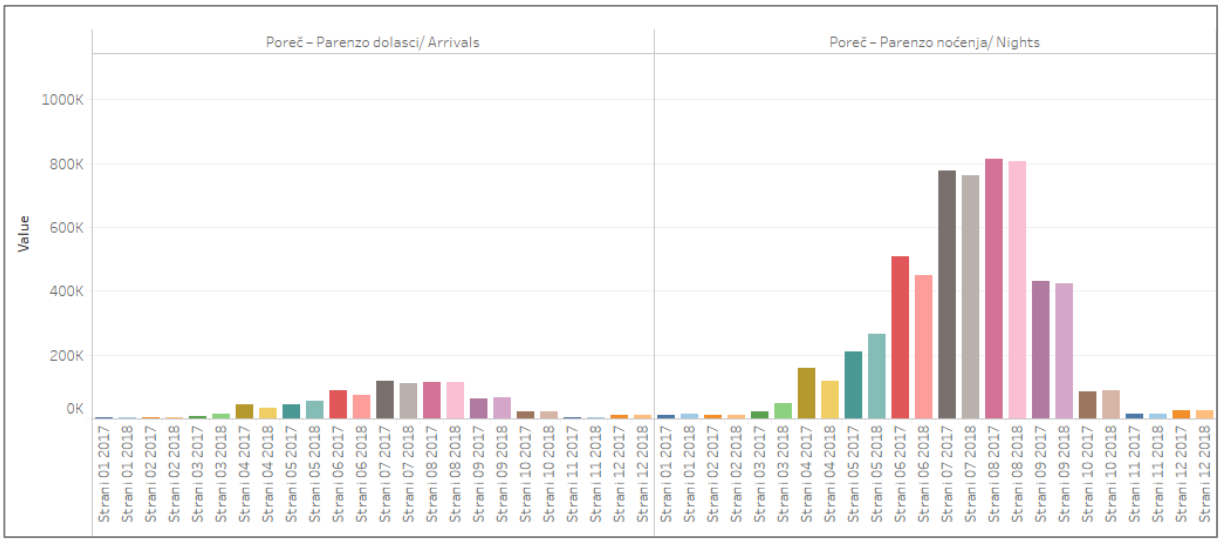


Slika 71 Broj dolazaka i noćenja domaćih turista u općinama PGŽ (3)

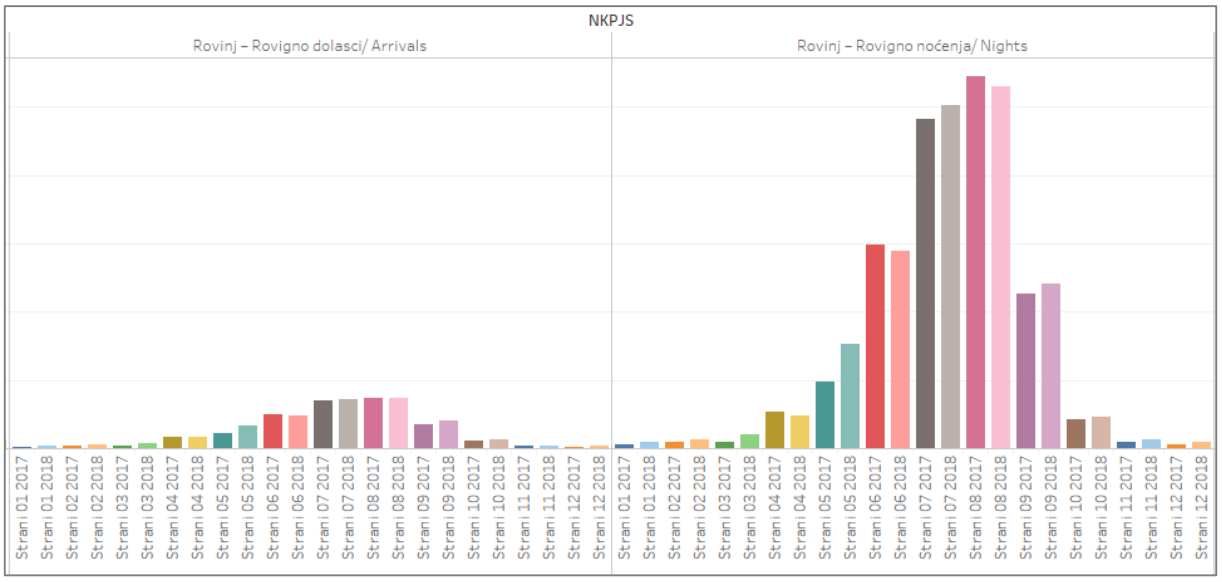


Slika 72 Broj dolazaka i noćenja domaćih turista u općinama PGŽ (4)

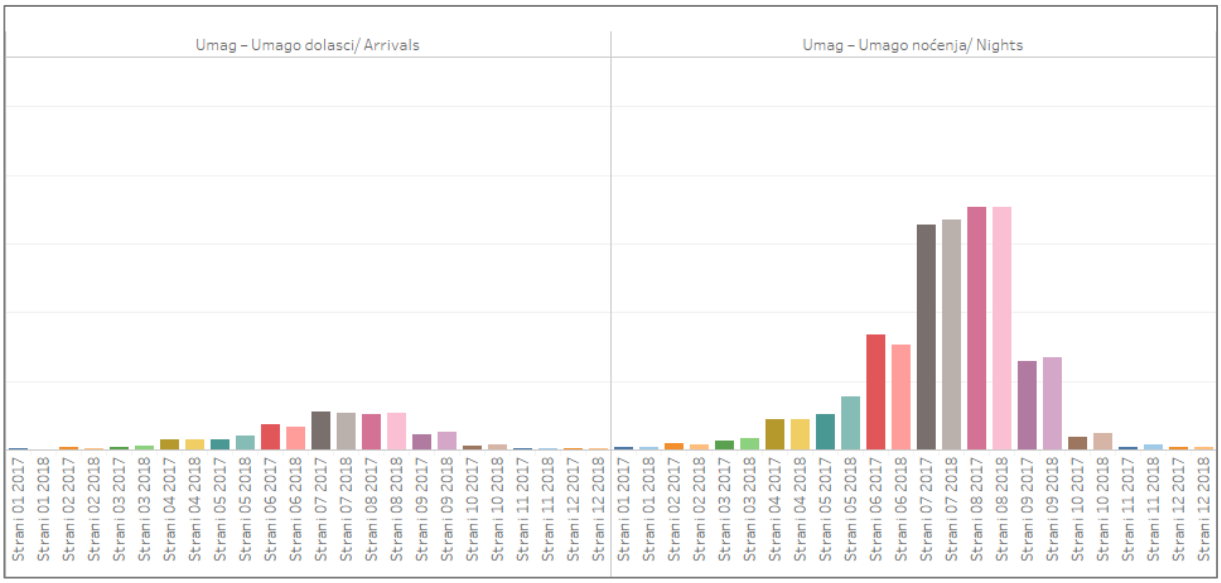
Strani turisti najviše posjećuju Istarsku županiju te imaju najveći broj dolazaka i noćenja u gradu Rovinju gdje ljeti dolaze u velikom broju. Ostali gradovi su također najviše popularni ljeti ali imaju nešto manji broj dolazaka i noćenja, naročito Umag. Zanimljivo je da Poreč ima jako mali broj dolazaka u odnosu na ostale gradove ali zato ima veliki broj noćenja što u prosjeku znači da turisti borave 7 do 8 dana. Zimi istarski gradovi nisu zanimljivi za posjete (Slika 73 do Slika 75).



Slika 73 Broj dolazaka i noćenja stranih turista u gradovima Istarske županije (1)

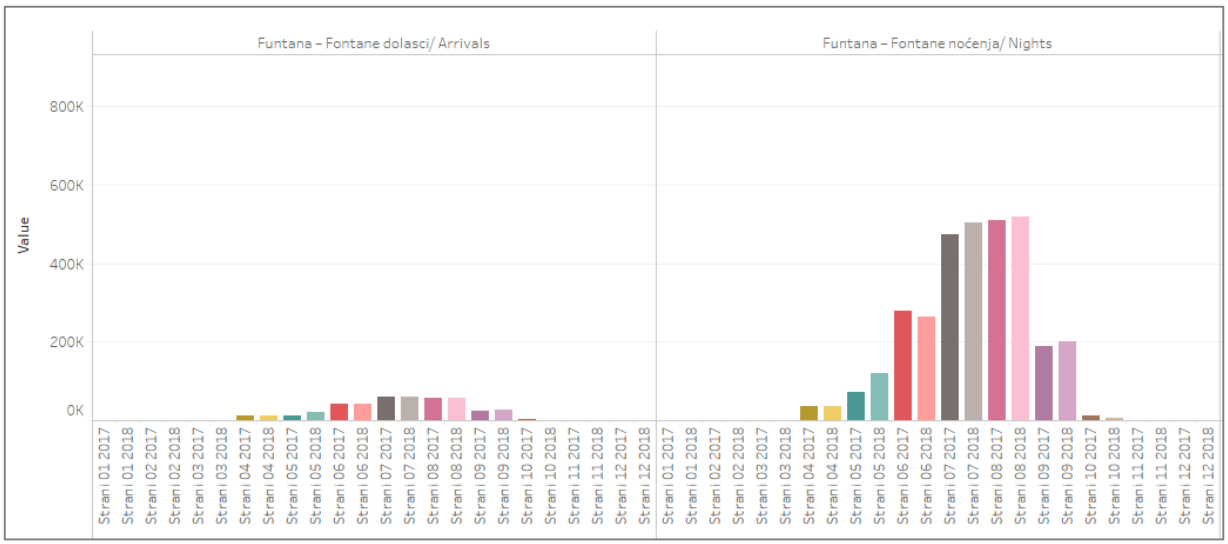


Slika 74 Broj dolazaka i noćenja stranih turista u gradovima Istarske županije (2)

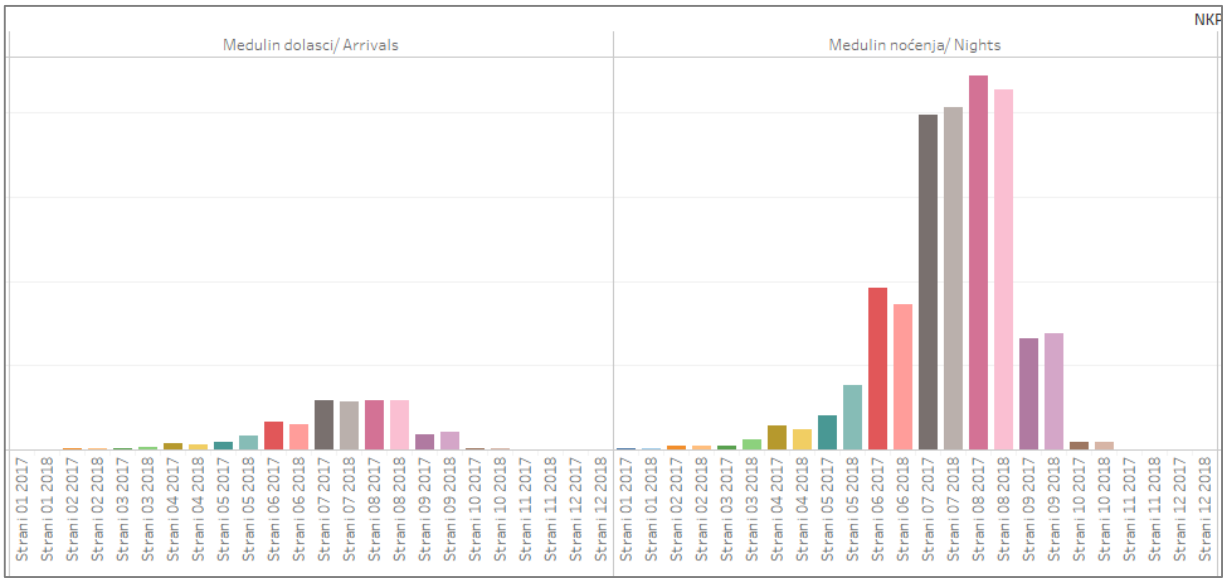


Slika 75 Broj dolazaka i noćenja stranih turista u gradovima Istarske županije (3)

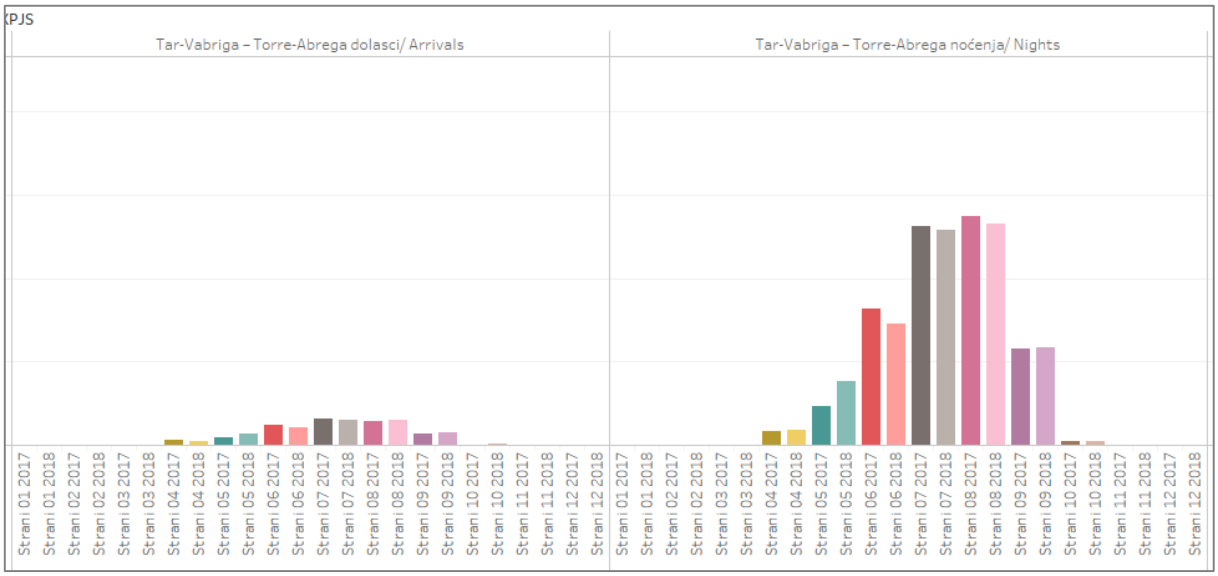
Od istarskih općina najveći broj dolazaka i noćenja turista ima Medulin koji u srcu sezone mjesečno ostvari oko 110 000 dolazaka i čak 850 000 noćenja što je u prosjeku 8 noćenja. Karakteristično za sve općine je da ih se posjećuje od travnja do listopada što se već moglo vidjeti za Bašku, a najveći broj dolazaka i noćenja ostvari se u srpnju i kolovozu. Kada se usporedi sezona 2017. i 2018. vidljivo je da broj dolazaka i noćenja lagano opada za većinu općina, jedino u Funtani raste broj noćenja (Slika 76 do Slika 79).



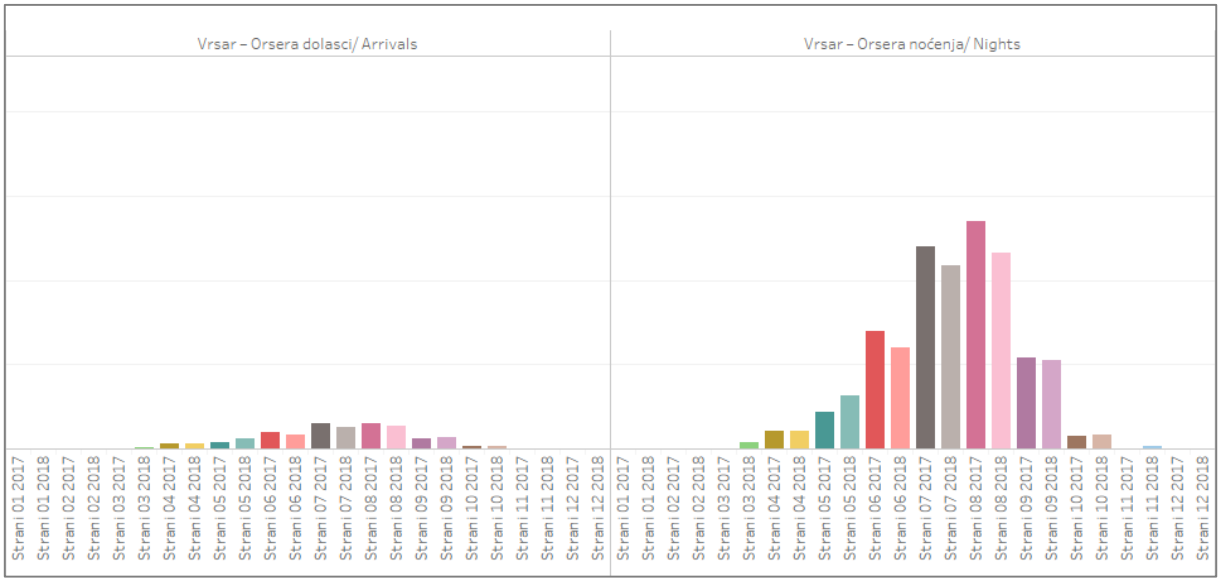
Slika 76 Broj dolazaka i noćenja stranih turista u općinama Istarske županije (1)



Slika 77 Broj dolazaka i noćenja stranih turista u općinama Istarske županije (2)

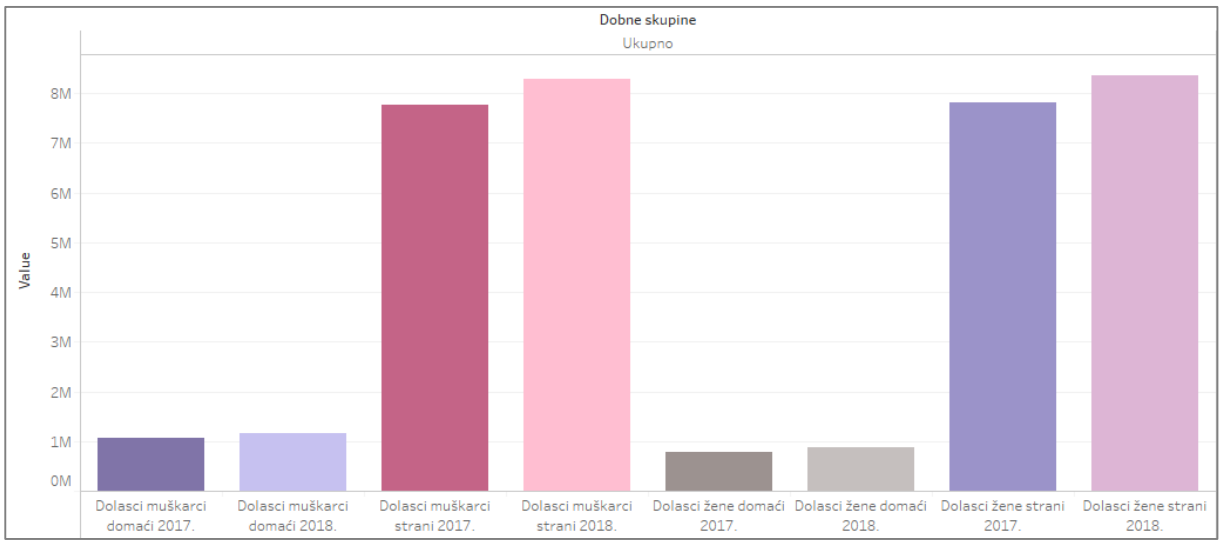


Slika 78 Broj dolazaka i noćenja stranih turista u općinama Istarske županije (3)



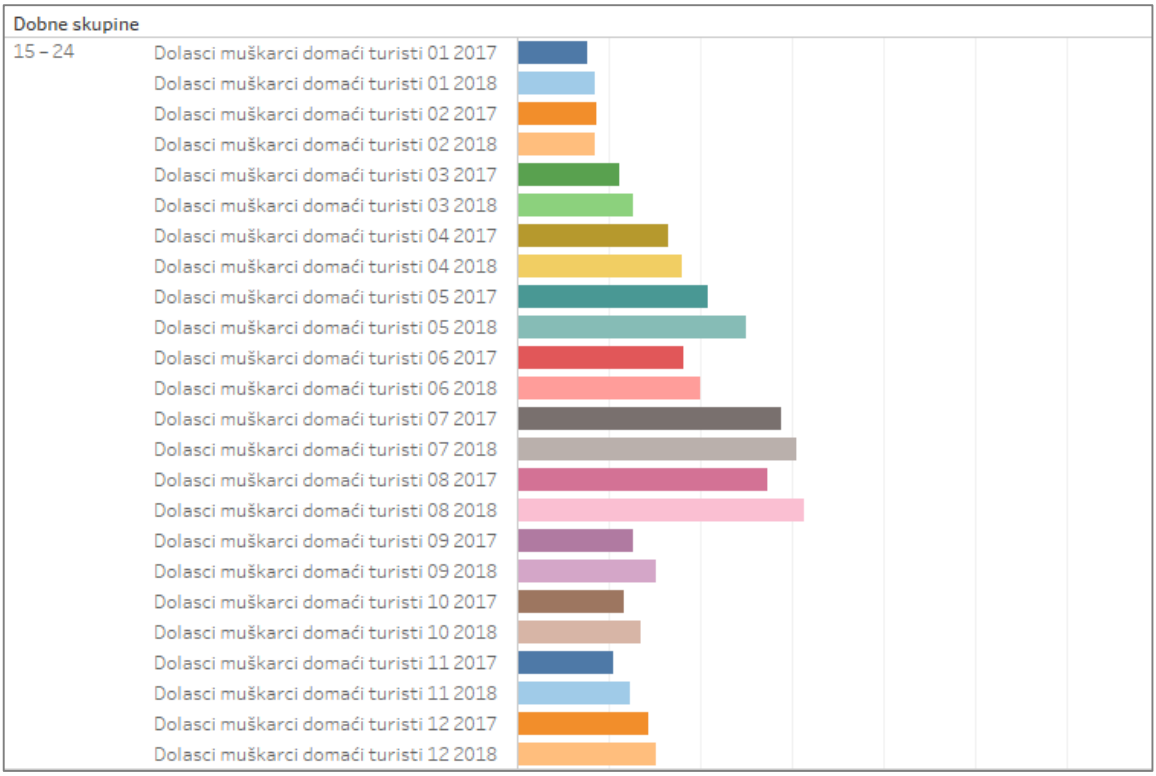
Slika 79 Broj dolazaka i noćenja stranih turista u općinama Istarske županije (4)

Analizirajući spol turista na Slici 80 vidimo da je došlo više muškaraca, te da je među domaćim turistima obje godine došlo više muškaraca, a među stranim više žena.

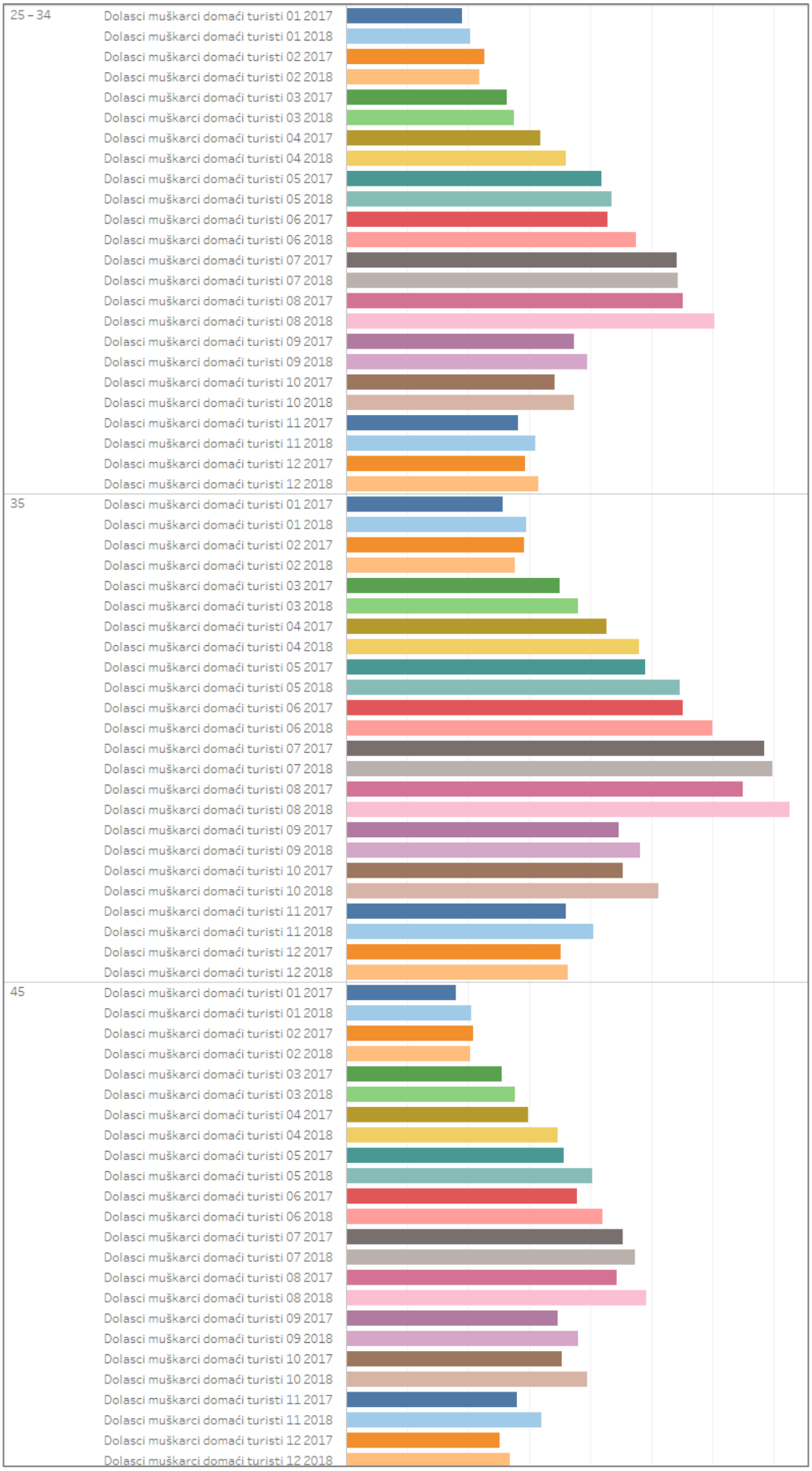


Slika 80 Omjer muškaraca i žena među turistima

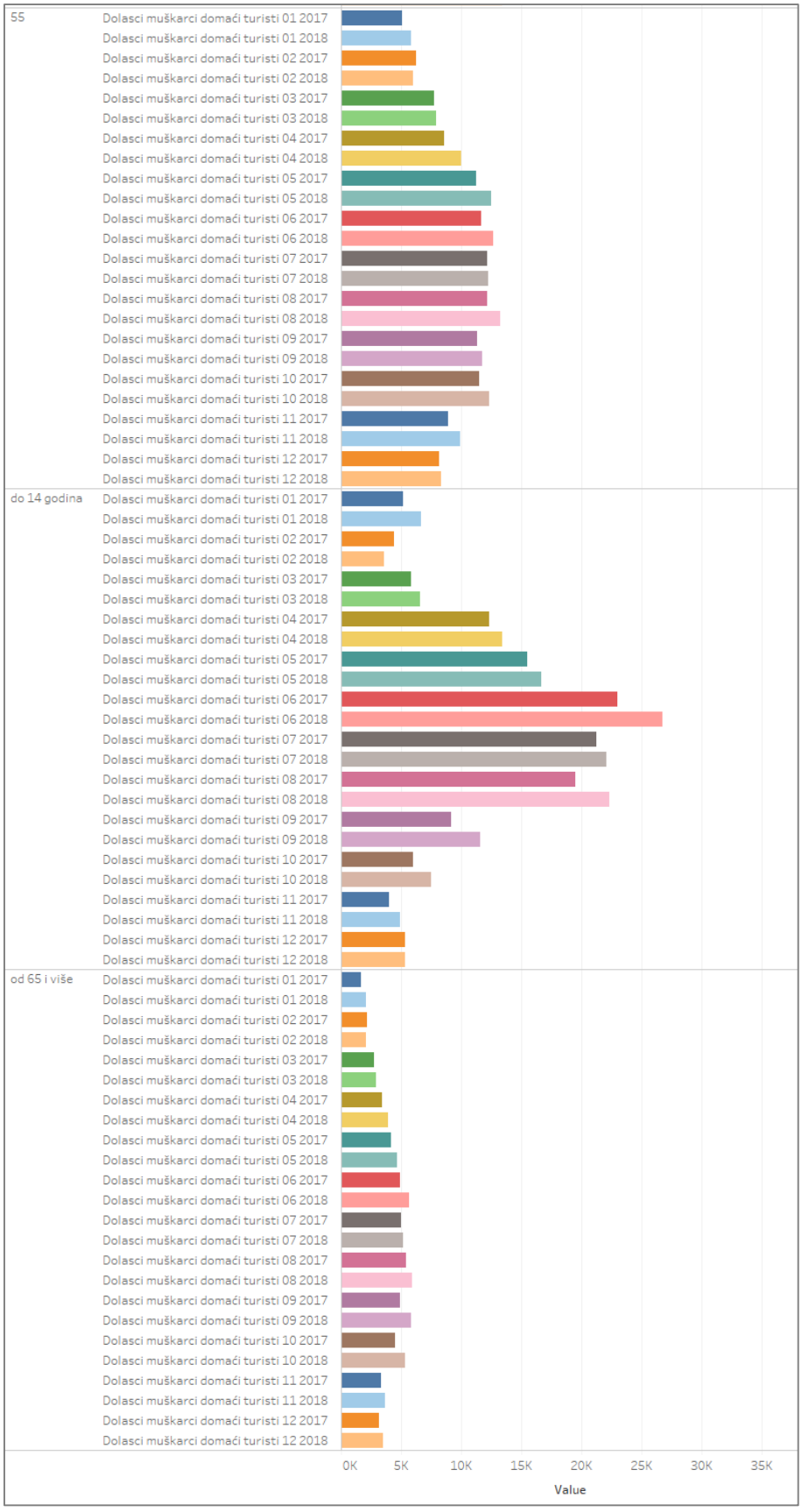
Domaći turisti najviše dolaze ljeti te ih je došlo najviše u dobi od 35 godina, a najmanje u dobi od 65 i više. Približno je došlo i muškaraca u dobi od 25 do 34 godine te u dobi od 45 (Slika 81 do Slika 83).



Slika 81 Dolasci domaćih turista muškaraca prema dobnim skupinama (1)

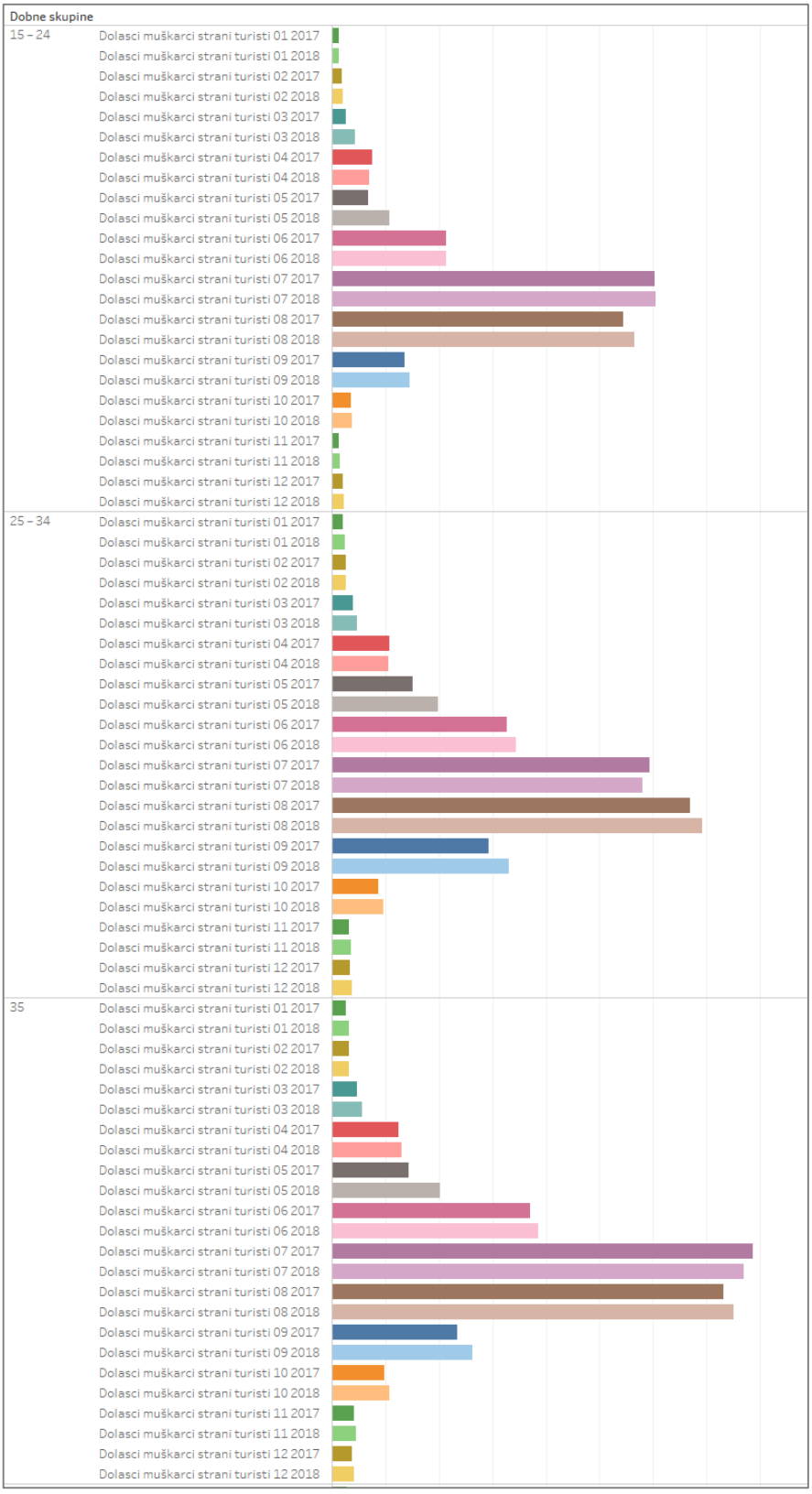


Slika 82 Dolasci domaćih turista muškaraca prema dobnim skupinama (2)

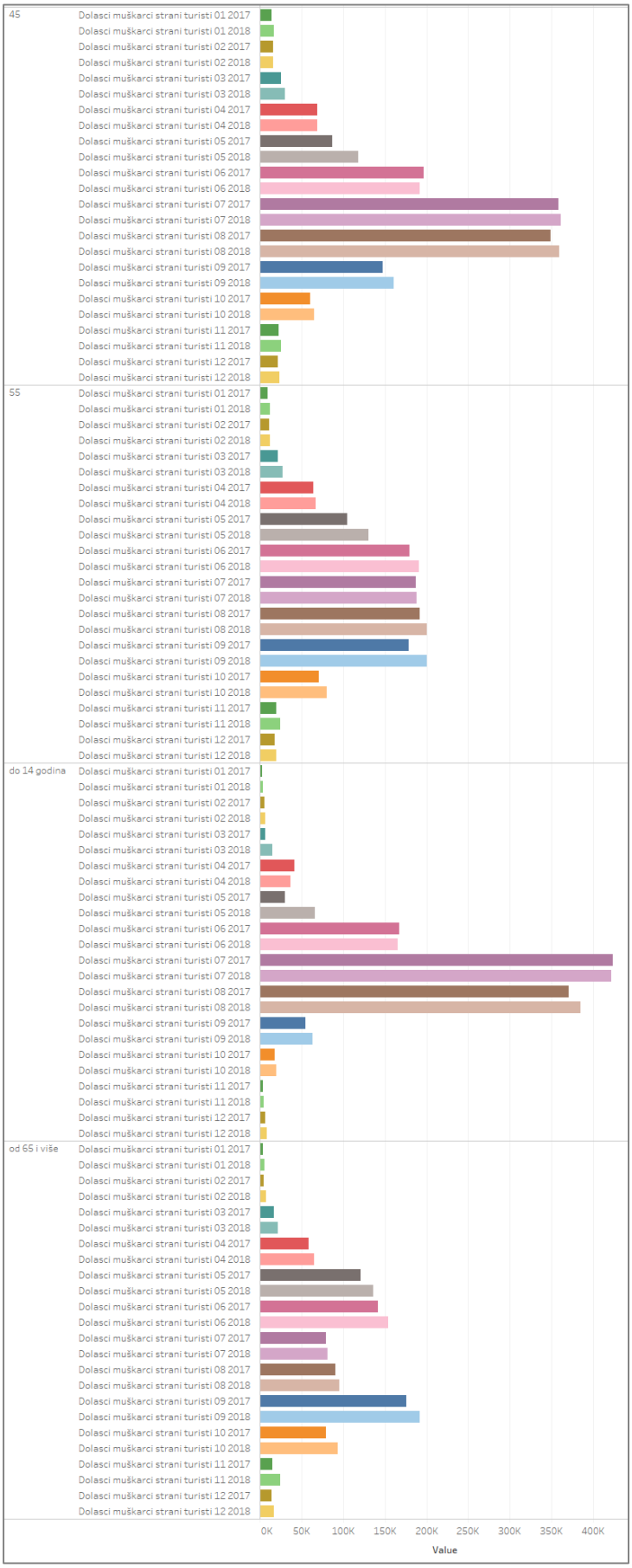


Slika 83 Dolasci domaćih turista muškaraca prema dobnim skupinama (3)

Stranih turista najviše dolazi u dobi od 45 godina te ih je kao i među domaćima najmanje u dobi od 65 godina i više. Za razliku od domaćih turista, stranih je jako malo u zimskim mjesecima (Slika 84 i Slika 85).



Slika 84 Dolasci stranih turista muškaraca prema dobnim skupinama (1)



Slika 85 Dolasci stranih turista muškaraca prema dobnim skupinama (2)

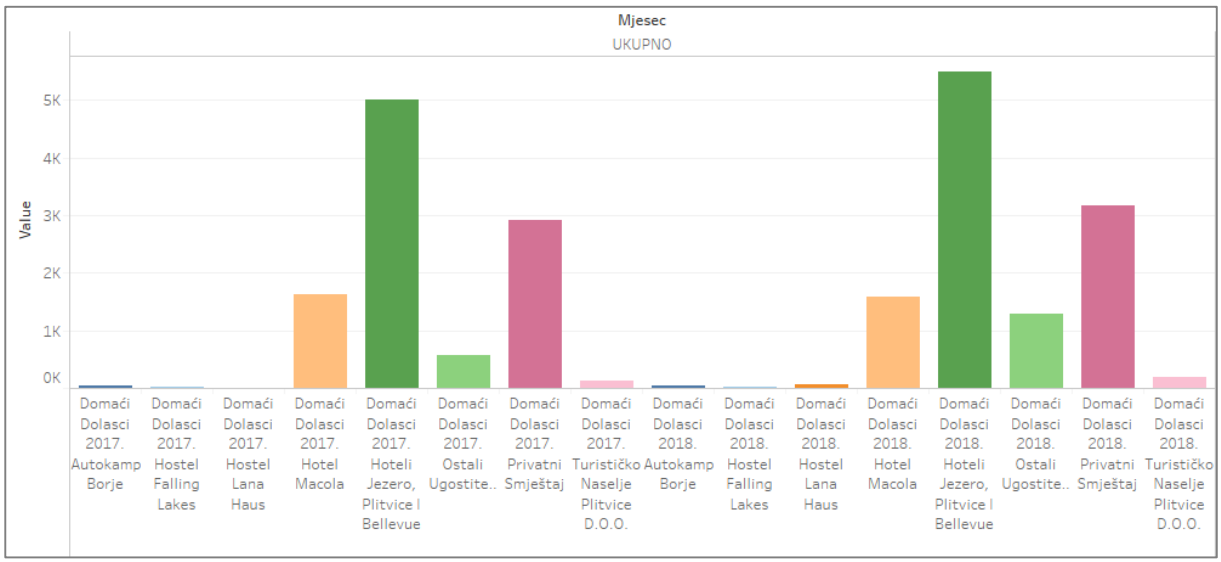
Kao i kod domaćih turista muškaraca i turistkinja ima najviše ljeti te ima najmanje turistkinja u dobi od 65 godina. Grafikon za svaku dobnu skupinu je identičan te nije naveden grafikon o dolascima domaćih turista žena prema dobnim skupinama. Grafikoni i za strane turistkinje su isti kao i za strane turiste te niti ti grafikoni nisu navedeni da bi se izbjeglo ponavljanje.

Grafikoni koji prikazuju noćenja također su identični gore navedenim te ni oni nisu navedeni.

2.3 Turizam Plitvičkih jezera

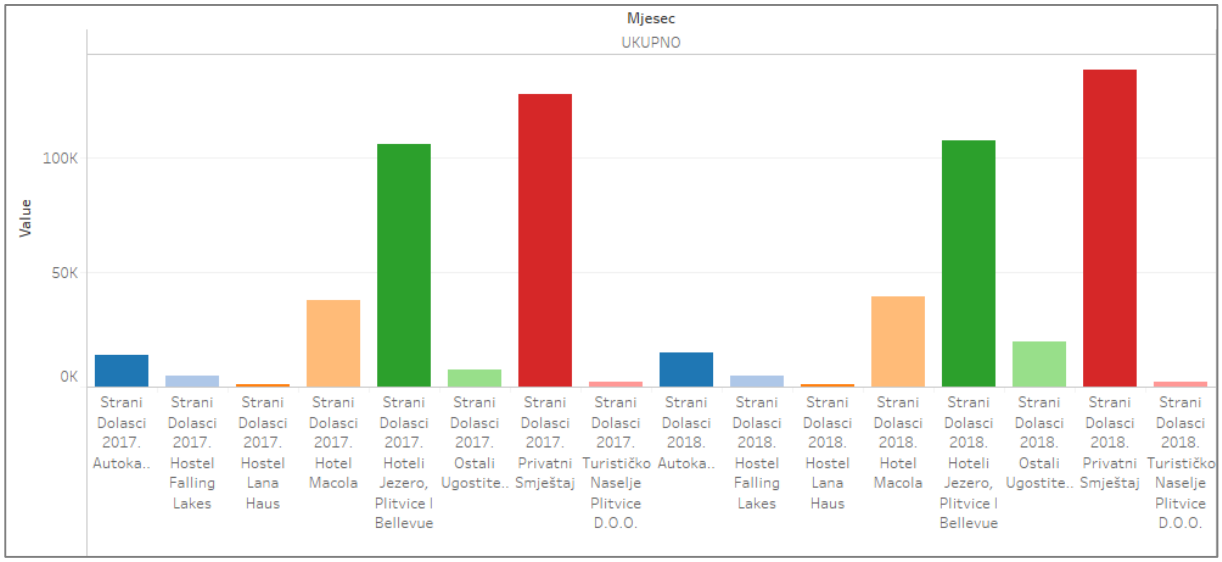
U nastavku se vrši analiza posjećenosti Plitvičkih jezera, najvećeg i najstarijeg nacionalnog parka. Istražit će se u koji smještaj dolazi najviše domaćih i stranih turista te koji je ostvario najveći broj noćenja, u kojem mjesecu ima naviše posjetitelja, koje nacije preferiraju koji smještaj, koliko dugo odsjedaju i drugi faktori. U konačnici će se izlučiti 4 najposjećenija smještaja za koje će se analizirati mišljenja posjetitelja prikupljena sa stranice TripAdvisor.

Domaći turisti najviše dolaze u hotele Jezero, Plitvice i Bellevue te privatni smještaj i hotel Macola. U ostalim smještajima broj dolazaka je vrlo mali, a u navedenim jedino za hotel Macola opada broj dolazaka (Slika 86).



Slika 86 Dolasci domaćih turista u ugostiteljske objekte

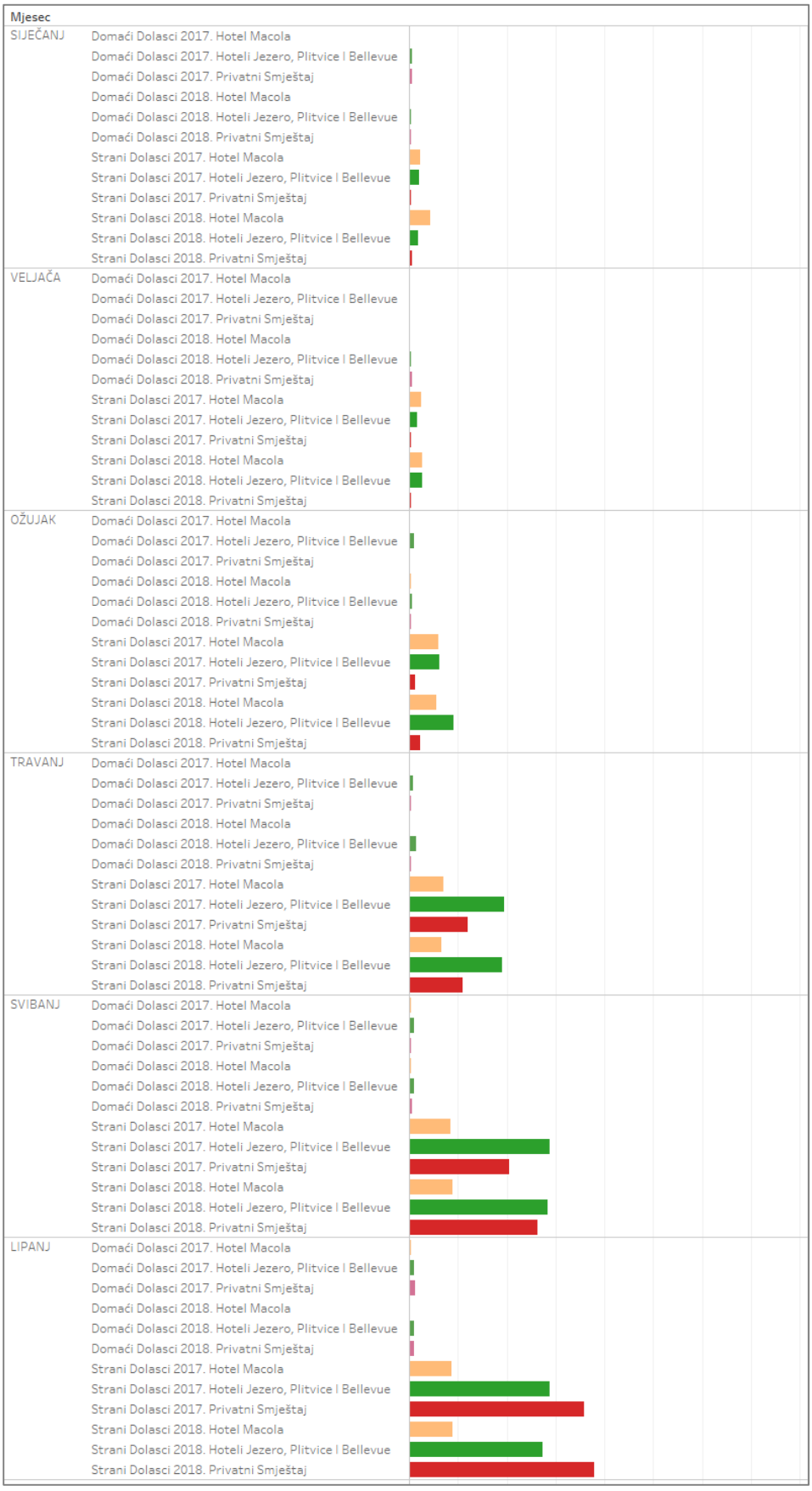
Strani turisti najviše dolaze u privatne smještaje te zatim u hotele Jezero, Plitvice i Bellevue i hotel Macola. Za razliku od domaćih turista dolaze u jako velikom broju, primjerice u privatni smještaj 2017. došlo je 127 499 stranih turista, a 2018. 138 000, a domaćih turista dolazi oko 3 000. Strani turisti sve više počinju dolaziti u autokamp Borje te je broj dolazaka porastao za 2 000 (Slika 87).



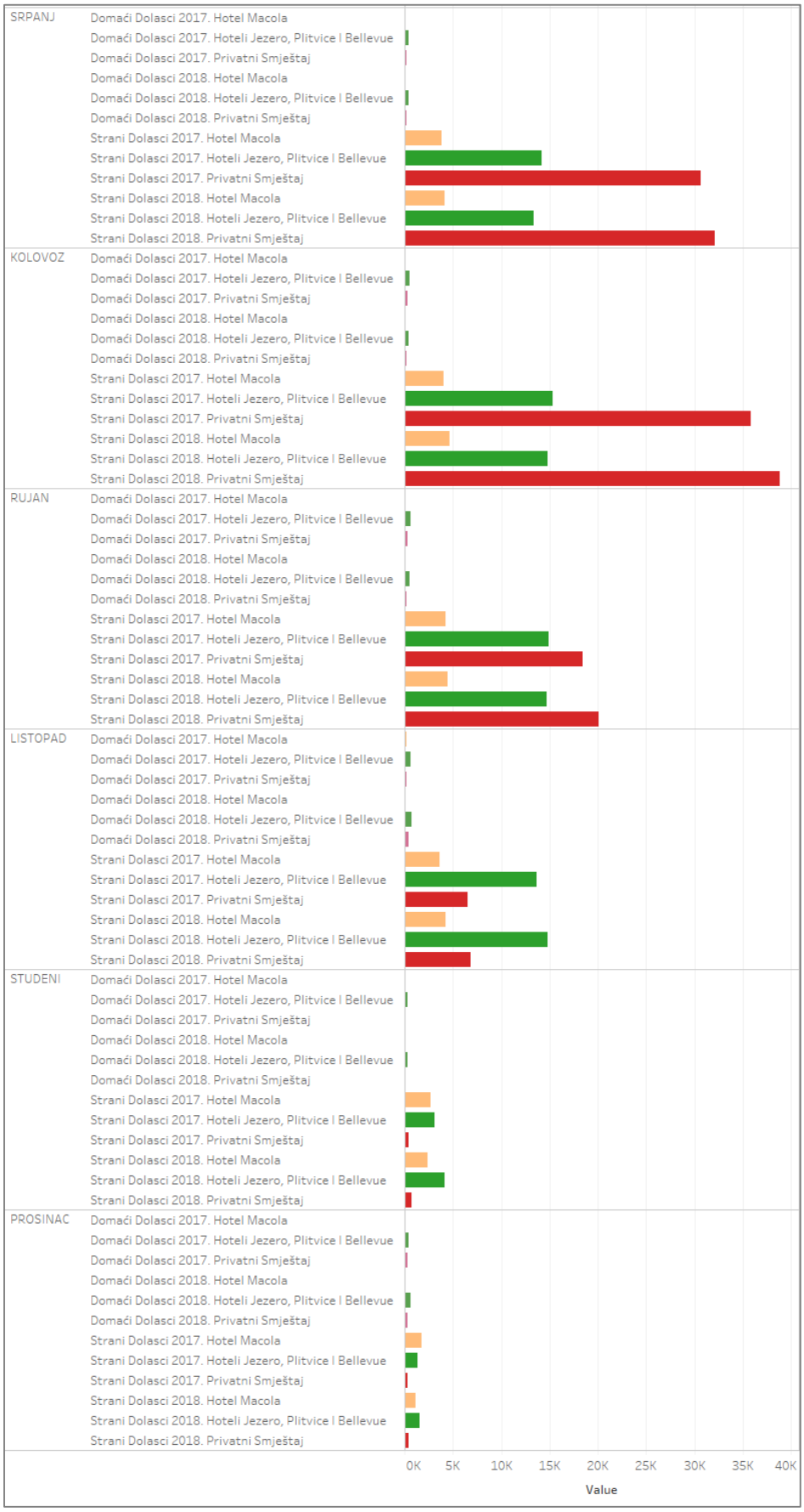
Slika 87 Dolasci stranih turista u ugostiteljske objekte

Kada se analiziraju dolasci po mjesecima vidimo da strani turisti od siječnja do ožujka najviše dolaze u hotel Macola te hotele Jezero, Plitvice i Bellevue. Od travnja naglo raste broj dolazaka u hotele Jezero, Plitvice i Bellevue i privatni smještaj te u lipnju broj dolazaka u privatni smještaj premaši broj dolazaka u hotele. U rujnu broj dolazaka u privatne smještaje i hotele opada te do kraja godine ponovno imamo najviše dolazaka u hotel Macola te hotele Jezero, Plitvice i Bellevue.

Domaći turisti najviše dolaze u hotele Jezero, Plitvice i Bellevue od travnja do listopada te je u ostalim mjesecima taj broj daleko manji. Broj dolazaka u ostale smještaje je tijekom godine ujednačen (Slika 88 i Slika 89).

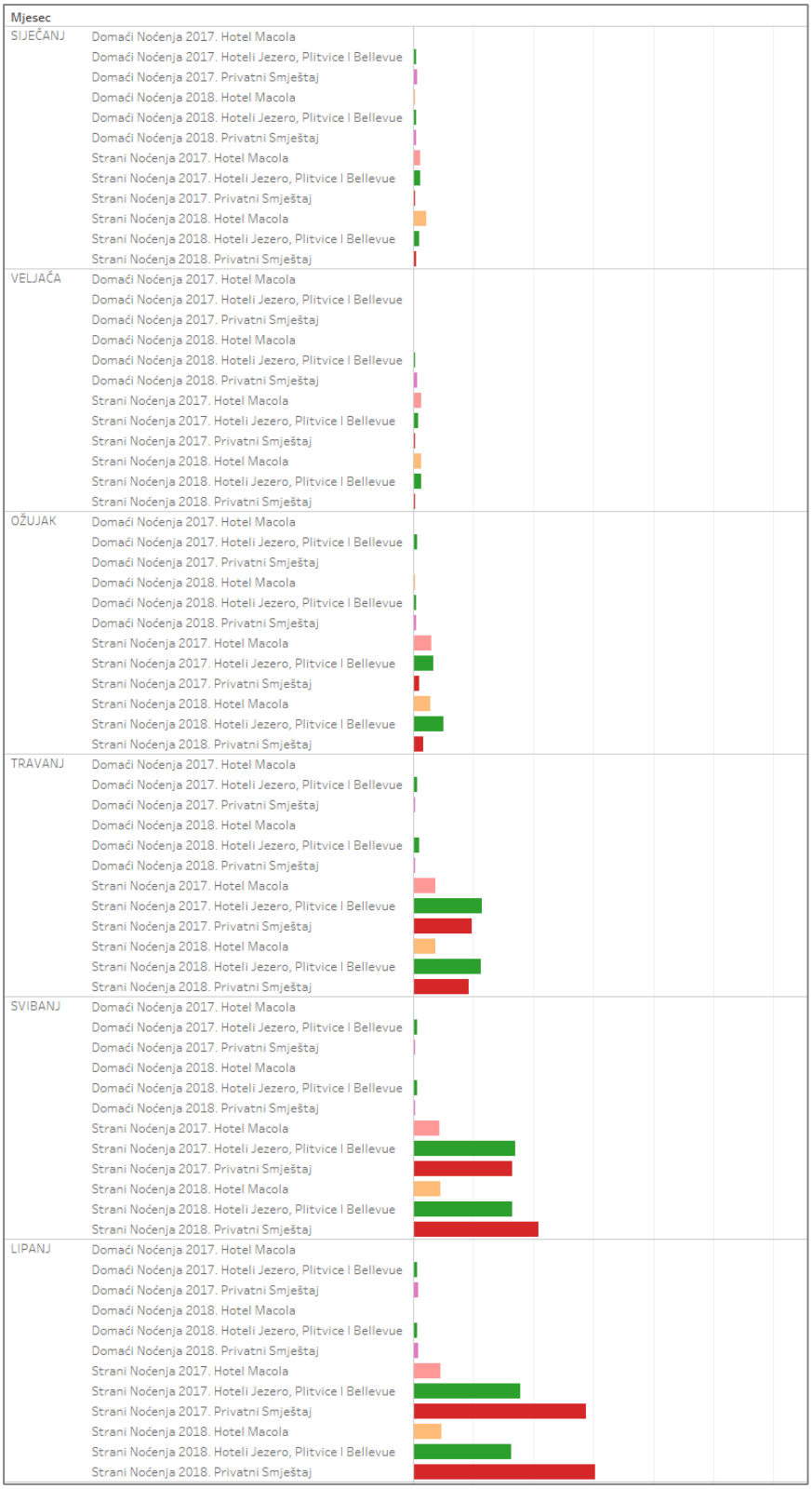


Slika 88 Najposjećeniji smještajni objekti po mjesecima i vrsti turista (1)

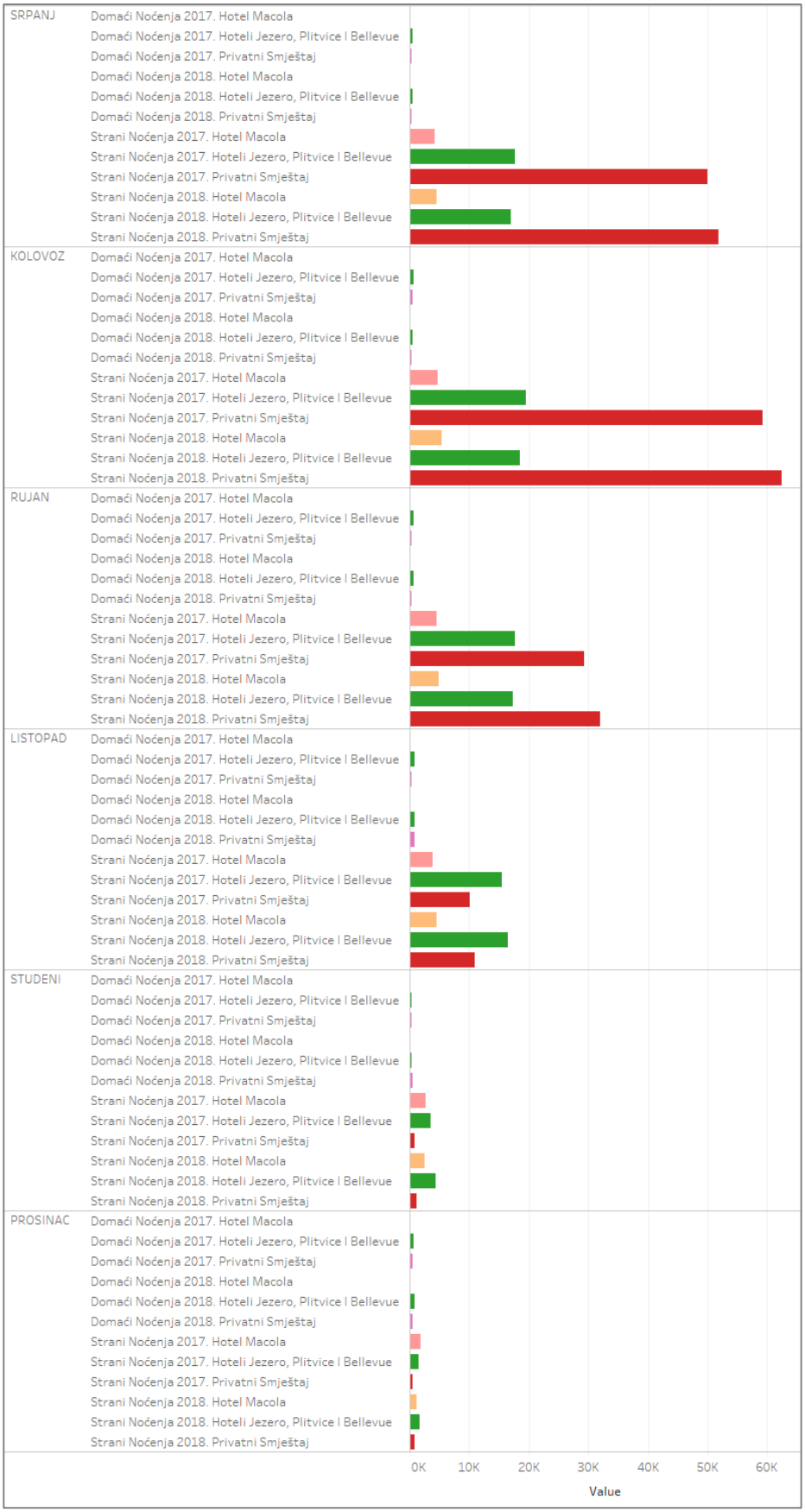


Slika 89 Najposjećeniji smještajni objekti po mjesecima i vrsti turista (2)

Noćenja domaćih i stranih turista u gore navedenim smještajnim objektima prati broj dolazaka, te je najveći broj noćenja u kolovozu, a u rujnu se broj noćenja stranih turista u privatnom smještaju prepolovi (Slika 90 i Slika 91).



Slika 90 Broj noćenja u najposjećenijim smještajnim objektima po mjesecima i vrsti turista (1)



Slika 91 Broj noćenja u najposjećenijim smještajnim objektima po mjesecima i vrsti turista (2)

Za hotele Jezero, Plitvice i Bellevue i hotel Macola do rujna tekuće godine spremljen je broj dolazaka i noćenja prema zemlji prebivališta. U hotele Jezero, Plitvice i Bellevue najviše dolaze posjetitelji iz Japana, Republike Koreje i Tajvana. Turisti iz Australije, Belgije i SAD-a dolaze samo u ove hotele. U hotel Macola najviše dolaze posjetitelji iz Republike Koreje, a samo u ovaj smještaj dolaze posjetitelji iz Hrvatske, Mađarske, Njemačke i Španjolske (Slika 92).

Slika 92 Dolasci posjetitelja najposjećenijih smještajnih objekata prema zemlji prebivališta

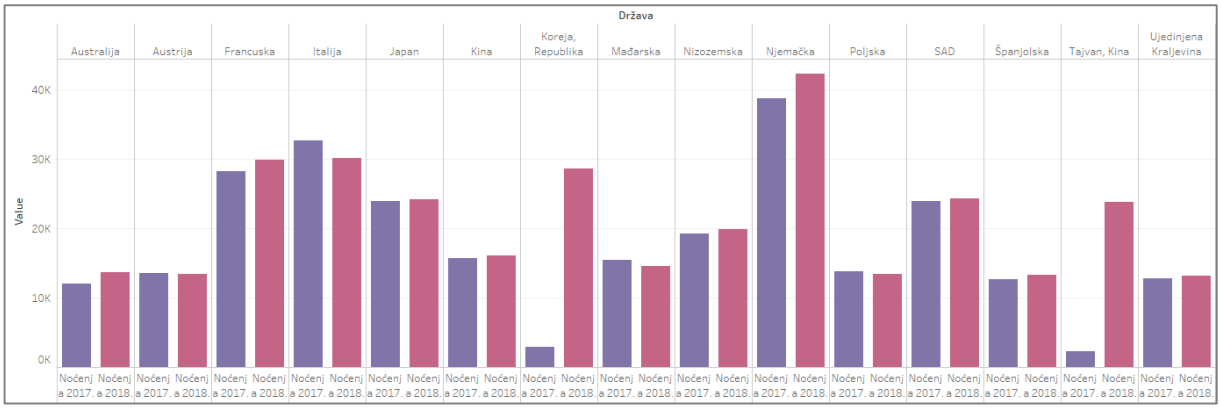
U hotelima Jezero, Plitvice i Bellevue i hotelu Macola turisti iz Republike Koreje, Japana i Tajvana ostvarili su jednak broj dolazaka i noćenja. Turisti iz Australije i SAD-a koji su došli samo u hotele Jezero, Plitvice i Bellevue 2018. godine ostvarili su veliki broj noćenja obzirom da ih je došlo u manjem broju nego 2017. 2018. turisti iz Japana su u gore navedena tri hotela ostvarili daleko veći broj noćenja nego godinu ranije kada su rijetko ostvarili pokoje noćenje. Turisti koji su posjetili samo hotel Macola uglavnom imaju isti broj dolazaka i noćenja (Slika 93).

Slika 93 Noćenja posjetitelja najposjećenijih smještajnih objekata prema zemlji prebivališta

Ako se analiziraju dolasci turista prema zemlji prebivališta na godišnjoj razini vidimo da je najviše turista iz Republike Koreje te općenito azijskih zemalja. Veliki broj ih dolazi iz SAD-a, Francuske, Njemačke i susjedne Italije te broj turista iz SAD-a i Njemačke raste, a Francuske i Italije opada. Može se uočiti da je došlo do naglog porasta broja turista iz Španjolske i da ih veliki broj dolazi iz daleke Australije, čak i više nego iz Austrije (Slika 94).

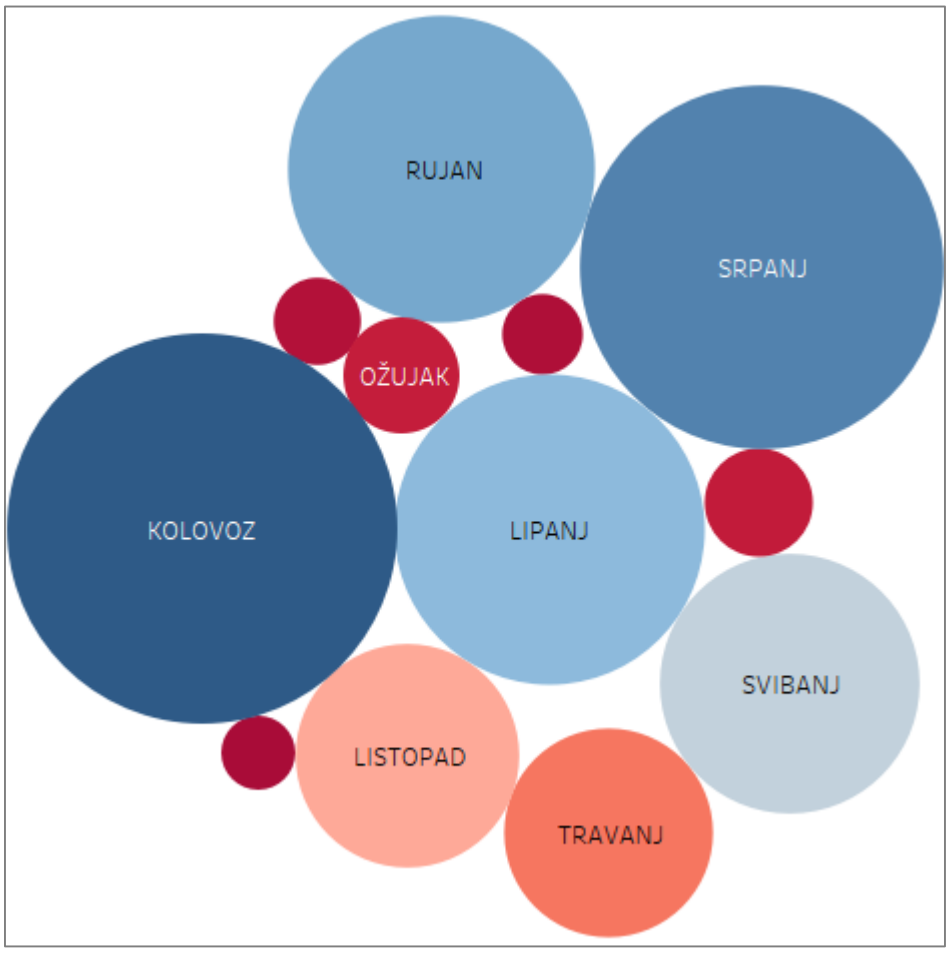
Slika 94 Dolasci turista prema zemlji prebivališta

Kada se analiziraju noćenja zanimljivo je uočiti kako se 2018. naglo povećao broj noćenja turista iz Republike Koreje i Tajvana. Njemački turisti ostvaruju najveći broj noćenja iako turisti Republike Koreje imaju najveći broj dolazaka što znači da se oni duže zadržavaju (Slika 95).



Slika 95 Noćenja turista prema zemlji prebivališta

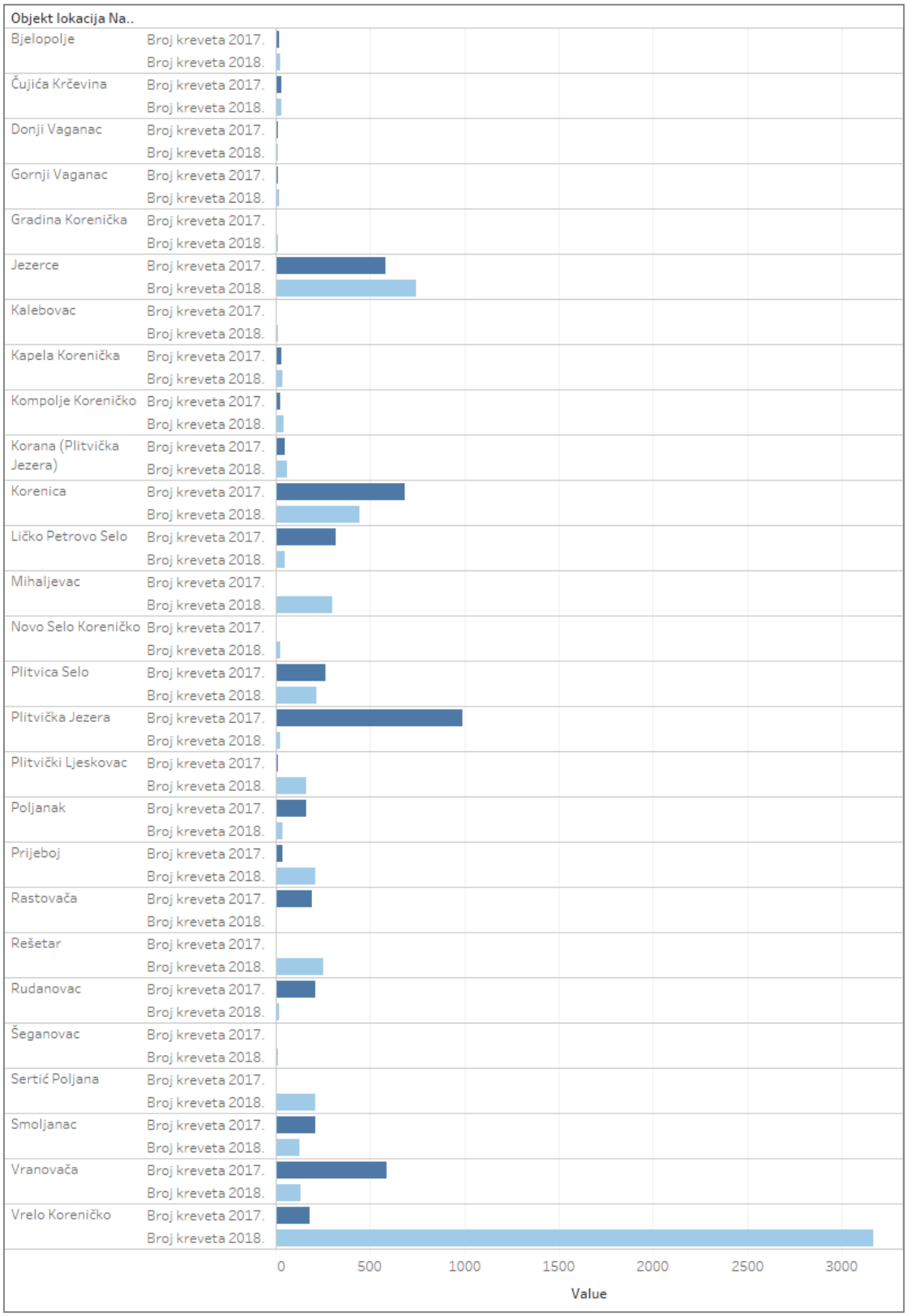
Turisti koji dolaze samo u posjet Nacionalnom parku Plitvička jezera dolaze od lipnja do rujna, a najveći broj ih dođe u kolovozu kao što je prikazano na Slici 96 u mjehurićastom dijagramu. Od studenog do veljače broj posjeta je vrlo mali u odnosu na ostale mjesece. Na grafikonu su prikazani podaci sakupljeni kroz dvije godine.



Slika 96 Posjetitelji NP Plitvička jezera po mjesecima

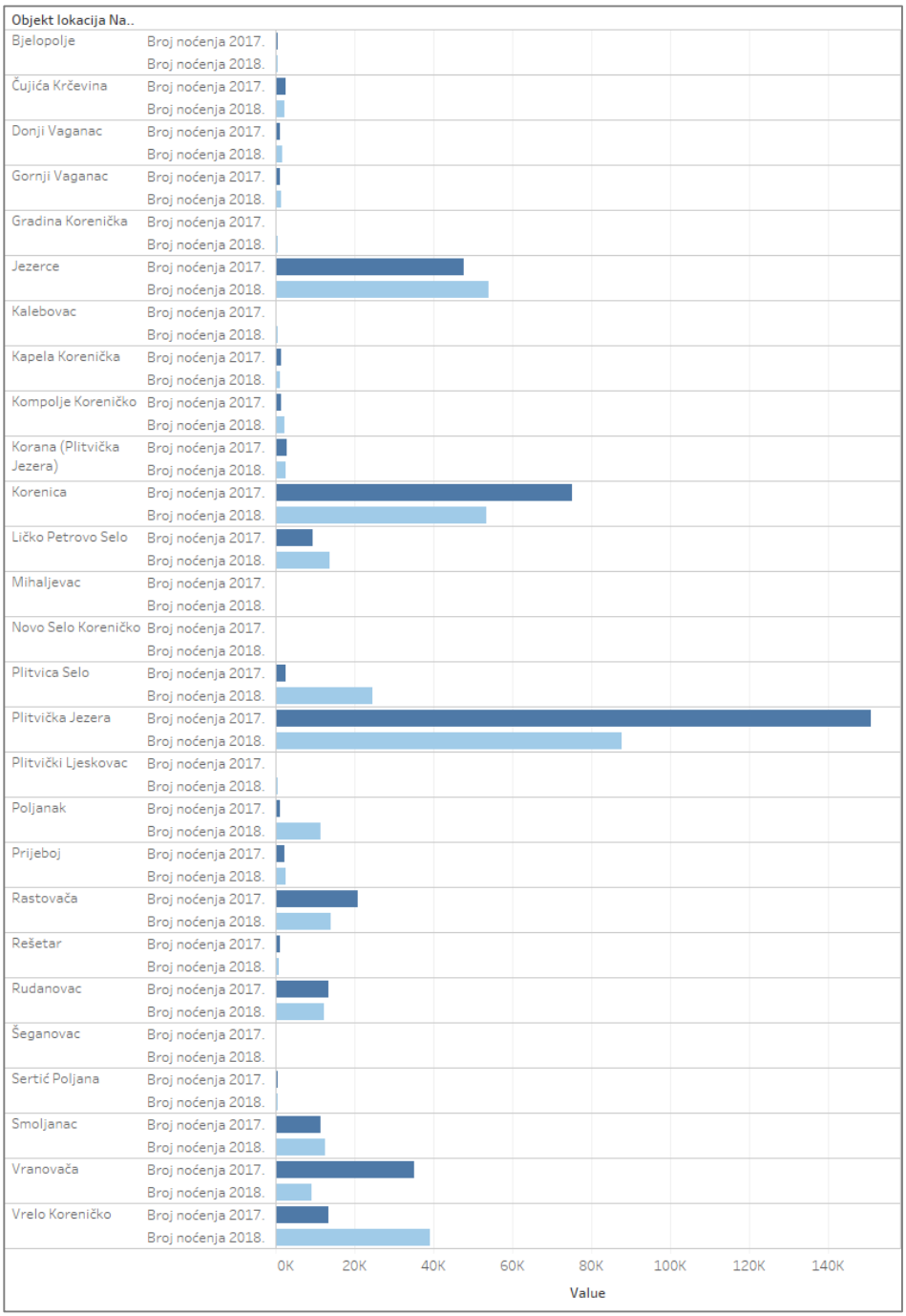
Kada dolaze u Plitvička jezera turisti biraju smještaj po nekom kriteriju. Pojedina naselja nude veći broj kreveta, a neka manji.

U Vrelu Koreničkom 2018. se nudio najveći broj kreveta i naglo je porastao sa 133 na 3166 kreveta. U Jezercu i Korenici se također nudi veliki broj kreveta, a u Plitvičkim jezerima broj kreveta se naglo smanjio. Postoje mjesta u kojima je broj ponuđenih kreveta vrlo mali, do 60 kreveta u naselju, te ona sa 200 kreveta i više (Slika 97).



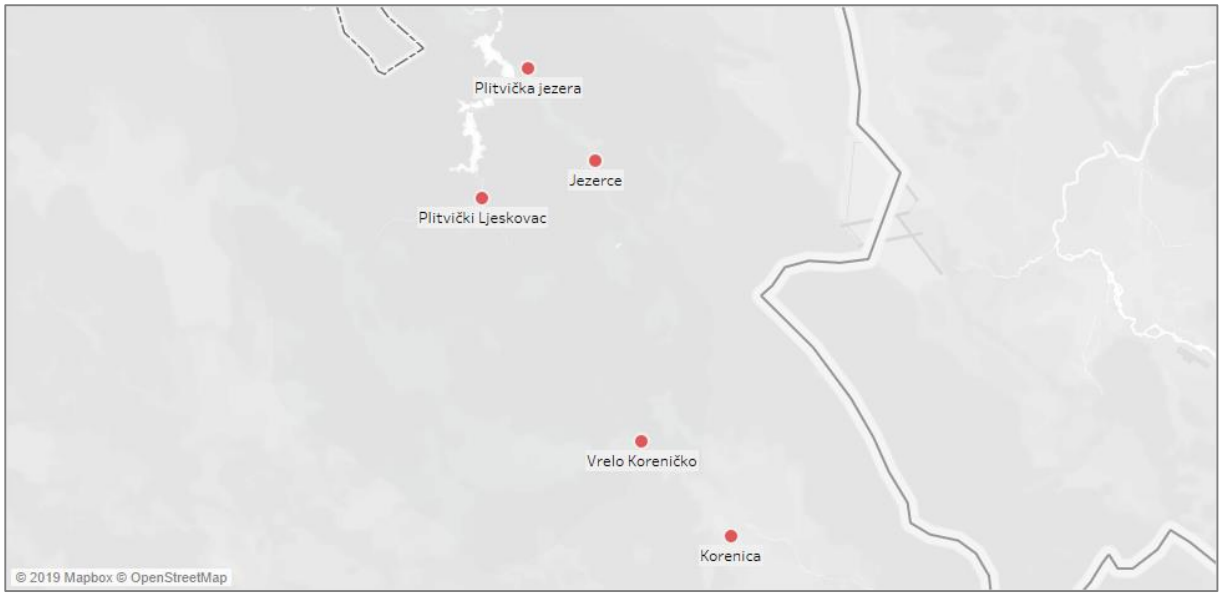
Slika 97 Broj kreveta po naseljima

Iako Vrelo Koreničko nudi najveći broj kreveta tamo se ne ostvaruje najveći broj noćenja nego u Plitvičkim jezerima što znači da turisti žele biti što bliže nacionalnom parku. 2018. u Plitvičkom Ljeskovcu ostvario se veliki broj noćenja iako ne nudi puno kreveta što znači da je zanimljiv turistima zbog svoje blizine nacionalnom parku. U naseljima koja imaju najveći broj kreveta ujedno se ostvari najveći broj noćenja, a u onima sa manje kreveta manje noćenja što znači da turisti biraju smještaj prema recenzijama ili popularnosti ili pak žele bolje opremljenu sobu koju si mogu priuštit sa svojim budžetom (Slika 98).



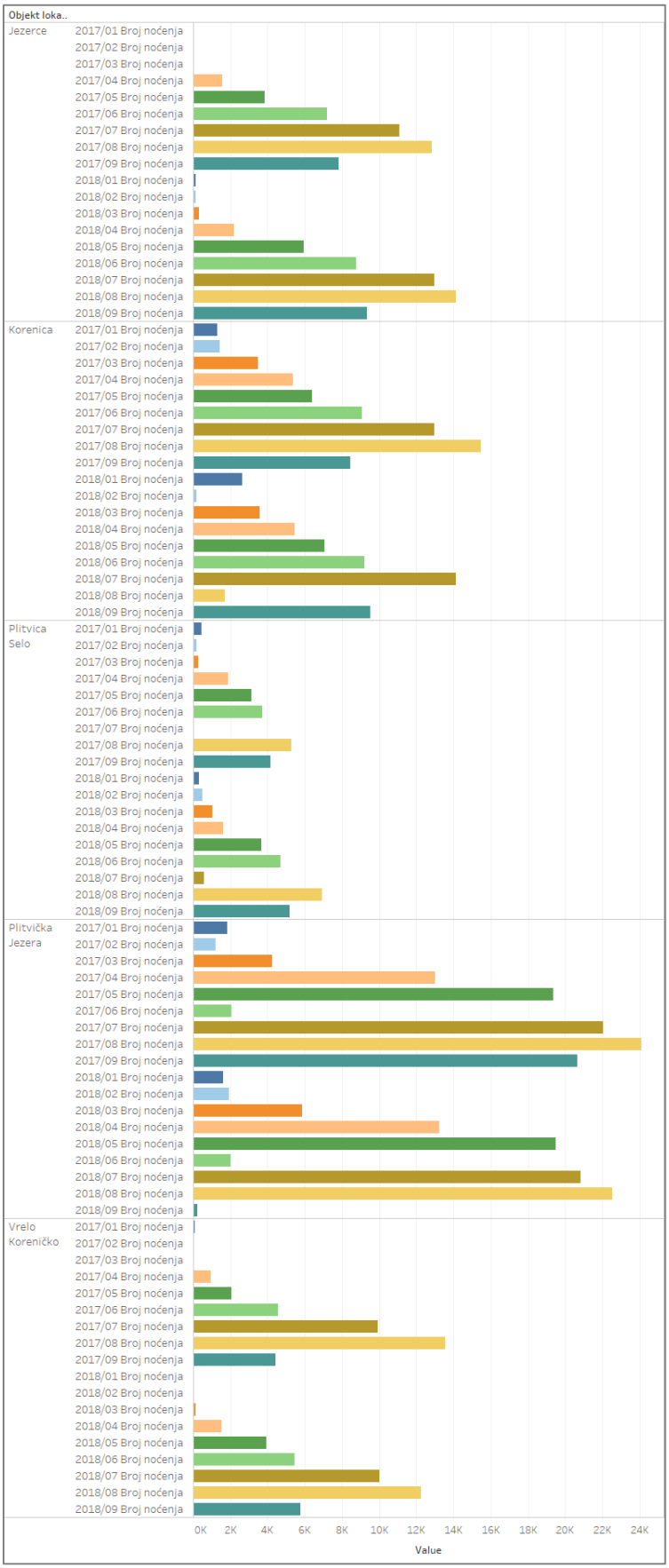
Slika 98 Broj noćenja po naseljima

Na Slici 99 prikazano je 5 naselja sa najvećim brojem kreveta i noćenja na geografskoj karti. Tri naselja nalaze se u području nacionalnog parka – Plitvička jezera koje je kraj glavnog sadržaja i u kojem su tri hotela, Jezero, Plitvice i Bellevue, te dva u rubnom dijelu nacionalnog parka – Jezerce i Plitvički Ljeskovac. Druga dva naselja, Vrelo Koreničko i Korenica, nešto su udaljenija od nacionalnog parka ali su istaknuta jer se u Vrelu Koreničkom nalazi Kamp Borje, a u Korenici hotel Macola koji je u prva tri smještaja sa brojem dolazaka i noćenja turista (Slika 99).



Slika 99 Karta naselja sa najvećim brojem kreveta i noćenja

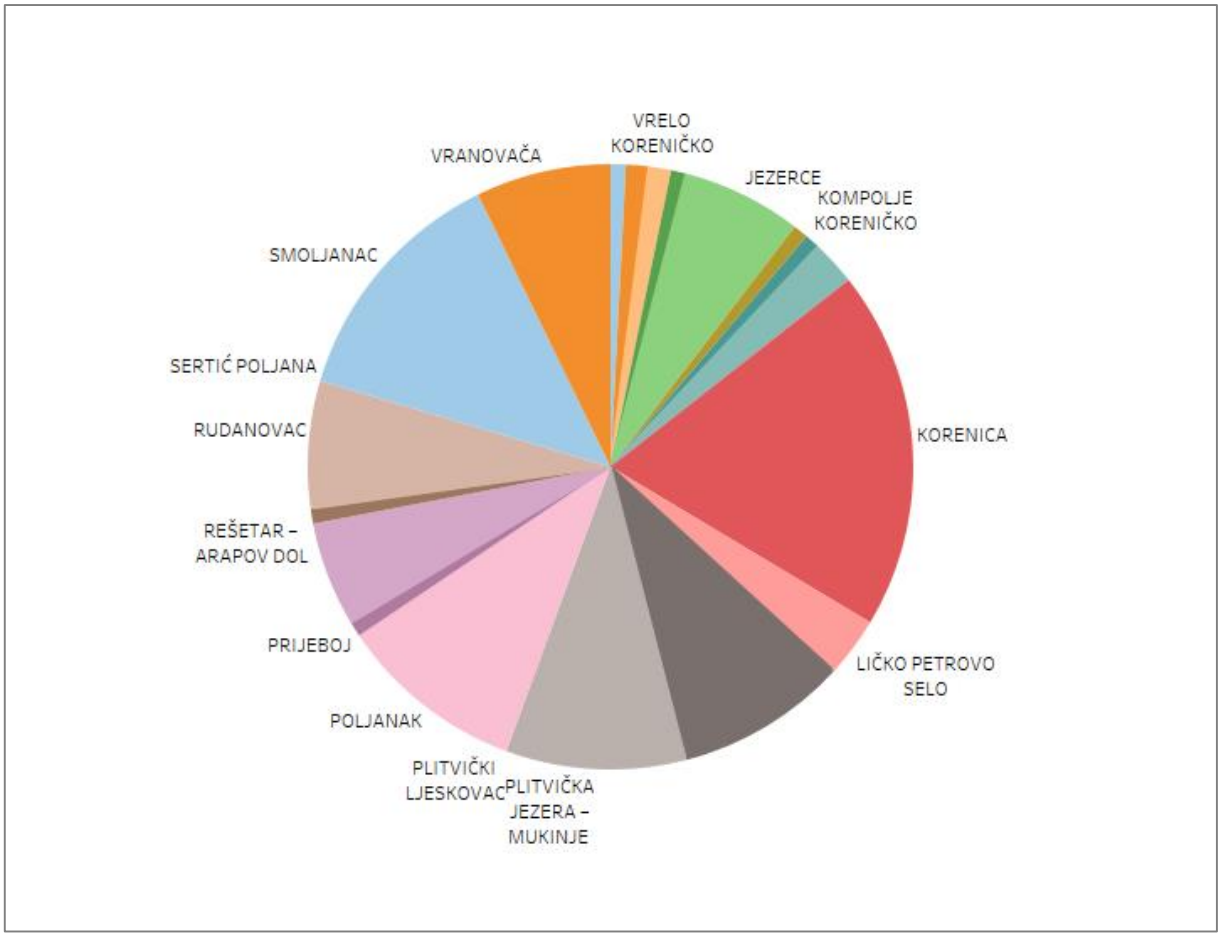
Naselje Plitvička jezera ostvaruje najveći broj noćenja do rujna, a najveći broj noćenja se ostvari u kolovozu, ne samo u ovom naselju već i u ostalima. Valja istaknuti travanj, svibanj, srpanj i rujan kao mjesece u kojima je povećan broj noćenja, a u ostali je taj broj vrlo mali za ovo naselje. U Jezercu, Korenici i Vrelu Koreničkom broj noćenja je u svim mjesecima vrlo sličan, jedino Jezerce ima manji broj dolazaka u siječnju, veljači i rujnu. Zbog Plitvičkih jezera, Plitvica Selo ima jako mali broj noćenja u srpnju, te se općenito turisti više opredjeljuju za Plitvička jezera koje je susjedno naselje ali ima više kreveta.



Slika 100 Top 5 naselja po broju ostvarenih noćenja do rujna tekuće godine

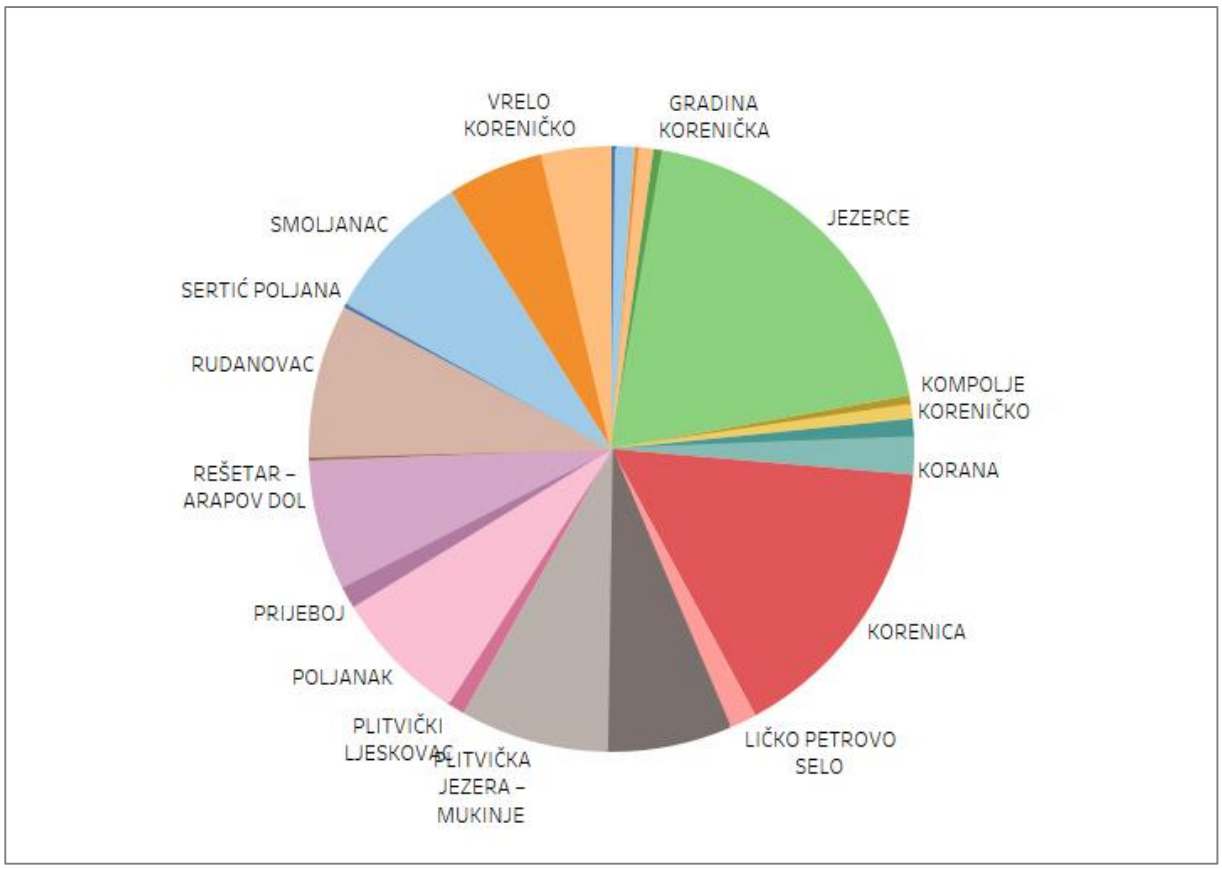
Kako veliki broj turista odsjeda u privatnom smještaju u nastavku se analizira u kojim naseljima se nalazi najviše apartmana i domaćinstava koji nude smještaj.

Najveći broj apartmana nudi se u Korenici te u Smoljancu. U Korenici je porastao broj apartmana te se može očekivati daljnji porast jer ima gotovo isti broj apartmana kao i naselja u neposrednoj blizini nacionalnog parka – Plitvica Selo i Plitvička jezera – Mukinje (siva nijansa) imaju zajedno isti broj apartmana kao i Korenica koja je udaljena 20 km. Značajan dio apartmana nalazi se i u Poljanku, susjednom naselju Plitvica Sela (Slika 101).



Slika 101 Udio apartmana u naseljima

U Jezercu se nalazi veliki broj domaćinstava koji nudi smještaj, a slijedi ga Korenica koja opet ima isti broj smještaja kao Plitvica Selo i Plitvička jezera – Mukinje. Smoljanac, Rudanovac, Rastovača i Poljanjak imaju podjednak broj domaćinstava koja nude smještaj, te je Rudanovac najudaljeniji od Plitvičkih jezera, a ostala tri naselja su susjedna (Slika 102).



Slika 102 Udio domaćinstava u naseljima

3 Analiza recenzija najposjećenijih smještaja u Plitvičkim jezerima

Prethodnom analizom hoteli Jezero, Plitvice, Bellevue i Macola izdvojeni su kao četiri smještaja sa najvećim brojem dolazaka i noćenja u Plitvičkim jezerima. Da bi se otkrilo zašto ti hoteli imaju veliki broj noćenja analizirat će se mišljenja gostiju preuzeta sa stranice TripAdvisor. Za preuzimanje recenzija gostiju izrađena je Python skripta *tripAdvisor_scraper.py*, a za njihovu analizu skripta *sentiment_analyzer.py*.

3.1 Ekstrakcija recenzija sa TripAdvisor-a

Skripta *tripAdvisor_scraper.py* izrađena je tako da se nakon pokretanja od korisnika traži unos linka stranice sa koje će se preuzeti recenzije. Nakon unosa linka preuzima se komentar po komentar sa svake stranice i sprema u *review.csv* datoteku.

TripAdvisor stranica sadrži *anti-scraping* softver te je potrebno lažirati ip adrese i *user-agent-a*. Da bi se to izvelo u glavnoj metodi najprije se dohvaćaju slobodni *proxyji* sa stranice Free Proxy List [8]. Za dohvaćanje *proxija* napisana je funkcija *get_proxies()* koja *scrape-a* stranicu Free Proxy List na način da pronađe tablicu i uzme 30 redaka tablice. Ako stupac Https, koji je sedmi element liste td, sadrži riječ *Yes* (znači da ta ip adresa podržava HTTPS), spoji ip adresu i port sa znakom „:“ i spremi u varijablu *proxy* koju se zatim dodaje u listu *proxies*. Na ovaj način se izluče svi *proxiji* i u glavnoj metodi spreme u varijablu *proxies*_listed putem funkcije *get_proxies()* koja vraća preuzete *proxije*:

```
def get_proxies():
    url = 'https://free-proxy-list.net/'
    response = get(url)
    parser = fromstring(response.text)
    proxies = []
    for i in parser.xpath('//tbody/tr')[:30]:
        if i.xpath('.//td[7][contains(text(),"yes")]'):
            proxy = ":".join([i.xpath('.//td[1]/text()')[0], i.xpath('.//td[2]/text()')[0]])
            proxies.append(proxy)
    return proxies

#MAIN
if __name__=='__main__':

    proxies_listed = get_proxies()
```

Kako postoji mogućnost da dohvaćeni *proxiji* neće raditi u glavnoj metodi najprije se provjeri da li oni rade na način da se pristupi stranici example.com. U *try* bloku sa *get()* funkcijom dohvaća se stranica *example.com*. Kao parametar *get()* funkcije proslijedi se odabrani *proxi* iz liste preuzetih *proxija* i postavi vrijeme isteka dohvaćanja stranice na 5 sekundi. Ako se stranica u 5 sekundi ne uspije dohvatiti ulazi se u prvi *except* blok i ispisuje „*Connection timed out*." U slučaju pojave neke druge iznimke ispisuje se neka od poruka iz naredna dva *except* bloka. Ako *get()* metoda vrati *status code* 200 znači da se stranica dohvatila i *proxi* koji se u tom *get* zahtjevu koristio sprema se u listu *working_proxies*.

Za preuzimanje recenzija napisana je funkcija *get_review_container()* kojoj se prosljeđuju dva parametra – *url* stranice sa koje se dohvaćaju recenzije i lista *proxija* koji rade:

```
working_proxies = []
print(len(proxies_listed))
for proxy in proxies_listed:
    proxies = {"http": proxy, "https": proxy}

    print("Trying GET example.com using proxy {}.".format(proxy))
    try:
        response = get('https://example.com/', proxies=proxies, timeout=5)
        print(response)
    except exceptions.ConnectTimeout:
        print("Connection timed out.")
        continue
    except exceptions.ProxyError:
        print("Proxy error.")
        continue
    except:
        print(sys.exc_info())
        continue
    if response.status_code == 200:
        print("Working.")
        working_proxies.append(proxy)

print(working_proxies)

url = input("Unesite link: ")

get_review_container(url, working_proxies)
```

get_review_container() funkcija sadrži listu *user-agenta* (*user_agent_list*) koji se pomoću funkcije *random()* uzimaju nasumično i prosljeđuju u *header* parametar funkcije *get()*. Prije pristupanja stranici sa recenzijama provjeri se da li je lista proslijeđenih *proxija* prazna i ako je ispiše se poruka i izlazi iz funkcije *get_review_container()*.

Ako imamo listu *proxyja*, uzima se prvi iz liste i stvara nova sesija. Funkcijom *Retry()* pokušat će se stranici pristupiti 3 puta i između svakog pokušaja postojat će vrijeme odgode koje se dobije sa *backoff_factor*:

```
for proxy in proxies_avail:
    proxies = {"http": proxy, "https": proxy}

    session = Session()
    retry = Retry(connect=3, backoff_factor=0.5)
    adapter = HTTPAdapter(max_retries=retry)
    session.mount('http://', adapter)
    session.mount('https://', adapter)
```

U try bloku pokuša se pristupiti stranici sa recenzijama koristeći *get()* metodu koja u parametrima ima proslijeđen odabran *proxy* i *user-agent*. Ako se stranicu uspije dohvatiti izlazi se iz *try* bloka, a ukoliko se ne dohvati stranica ulazi se u *except* blok i ispisuje poruka o neuspjelom dohvaćanju te da će se pokušati sa sljedećim *poxyjem*, a za sljedeći zahtjev čeka se određeni vremenski period:

```
print("Trying GET {} using proxy {}.".format(url, proxy))
try:
    response = session.get(url, headers = headers, proxies = proxies)
    print(response.status_code)
    print(len(response.text))
    break
except:
    print("Skipping proxy {}. Connnection error. Will try next proxy.".format(proxy))
    time.sleep(random.uniform(10.2,20.05))
```

Po izlasku iz *try* bloka sa *BeautifulSoup()* *scrapea* se stranica sa recenzijama. U *reviews_container* spremaju se svi pronađeni *div tag*-ovi koji sadrže recenziju. Za svaki *tag* pronalazi se ocjena recenzije i sprema u varijablu *rating*. Zatim se sam tekst recenzije pronalazi unutar *tag-a q* i sprema u varijablu *review*.

```
html_soup = BeautifulSoup(response.text, 'html.parser')

reviews_container = html_soup.findAll('div', class_ = 'hotels-review-list-
                                        parts-SingleReview__mainCol--2XgHm')
print(len(reviews_container))
for rc in range(len(reviews_container)):
    rating = reviews_container[rc].find('span', class_ = 'ui_bubble_rating')
    rating = rating['class'][1][-2:]

    review = reviews_container[rc].find('q', class_ = re.compile("hotels-review-
                                list-parts-ExpandableReview__reviewText")).text
```

Ako recenzija sadrži ocjenu manju od 30, u reviews.csv datoteku sprema se recenzija i zatim ocjena, pa dva prazna *stringa*. Spremljena recenzija je prema ocjeni negativna te se zato sa ocjenom sprema u prva dva stupca datoteke.

Ako je ocjena recenzije veća ili jednaka 30, najprije se u datoteku spremaju dva prazna *stringa* i zatim recenzija i ocjena te će oni označavati pozitivnu recenziju:

```
if int(rating) < 30:
    with open('../data/recenzije/reviews.csv','a', newline='', encoding='utf-8') as csv_file:
        writer = csv.writer(csv_file, delimiter=',')
        writer.writerow([review, rating, '', ''])

else:
    with open('../data/recenzije/reviews.csv','a', newline='', encoding='utf-8') as csv_file:
        writer = csv.writer(csv_file, delimiter=',')
        writer.writerow(['', '', review, rating])
```

Kako je na stranici prikazano po 5 recenzija, da bi se dohvatilo ostale potrebno je iz gumba *Next* izvući link na sljedeću stranicu i dohvatiti ju. To se radi na način da se novi *url* proslijedi funkciji *get_review_container()* sa određenim vremenom čekanja ukoliko se na stranici pronađe gumb *Next*. Na ovaj način funkcija za preuzimanje recenzija postaje rekurzivna:

```
if html_soup.find('a', class_ = 'ui_button nav next primary'):
    nextBtn = html_soup.find('a', class_ = 'ui_button nav next primary').get('href')
    print(nextBtn)
    nextBtn = "https://www.tripadvisor.com" + nextBtn
    time.sleep(random.uniform(15.2,25.05))
    get_review_container(nextBtn, proxies_avail)
```

3.2 Analiza sentimenta recenzija

Na prethodno prikupljenim recenzijama provodi se analiza sentimenta. Istražit će se koje su se riječi najčešće pojavljivale u recenzijama te koje su se pojavile u pozitivnim i negativnim recenzijama. Zatim se izrađuje model i ocjenjuje točnost klasifikacije riječi. Za dio analize korišten je kôd preuzet sa stranice Kaggle [9], a cjelokupni kôd nalazi se u skripti *sentiment_analyzer.py*.

U glavnoj metodi učitava se prethodno izrađena datoteka *reviews.csv* i sprema u *data frame reviews_df*. Stupcima *data frame-a* daju se redom imena: *negative_review, negative_rating, positive_review, positive_rating* kako bi se imenovao svaki učitani stupac iz datoteke.

Ako su u stupcima koji sadrže pozitivne i negativne recenzije nepostojeći podaci (*nan*) umjesto *nan* vrijednosti dodaje se prazan *string*. Postupak za pozitivne i negativne ocjene recenzija je identičan ali se umjesto praznog *stringa* dodaje 0. Zatim se u stupac *review* spajaju stupci koji sadrže pozitivne i negativne recenzije, a u stupac *rating* pozitivne i negativne ocjene recenzija. Na osnovu ocjene dodaje se stupac *is_bad_review* koji sadrži 0 ako recenzija nije negativna odnosno 1 ako je, a to se određuje na temelju ocjene recenzije iz stupca *rating*. U *data frame reviews_df* spremaju se samo stupci *review, is_bad_review* i *rating*:

```
#MAIN
if __name__=='__main__':
    reviews_df = pd.read_csv("../data/recenzije/reviews.csv")
    reviews_df.columns = ['negative_review', 'negative_rating', 'positive_review', 'positive_rating']

    reviews_df[['negative_review', 'positive_review']] = reviews_df[['negative_review', 'positive_review']].fillna('')
    reviews_df[['negative_rating', 'positive_rating']] = reviews_df[['negative_rating', 'positive_rating']].fillna(0)
    reviews_df['review'] = reviews_df['negative_review'] + reviews_df['positive_review']
    reviews_df['rating'] = reviews_df['negative_rating'] + reviews_df['positive_rating']
    # create the label matching bad or good review
    reviews_df['is_bad_review'] = reviews_df["rating"].apply(lambda x: 1 if x < 30 else 0)
    reviews_df = reviews_df[["review", "is_bad_review", "rating"]]
    print(reviews_df.head())
```

Na Slici 103 prikazano je prvih 5 redaka *reviews_df data frame-a*:

```
                                              review  is_bad_review  rating
0  Hotel is perfect location for the National Par...              0    30.0
1  Stayed for 2 nights with family. Clean, basic ...              0    40.0
2  Stayed here as part of a bus tour. Room is saf...              0    50.0
3  Stayed for one night, hotel is like stuck in c...              0    30.0
4  This place will make your lake trip easier. Bo...              0    30.0
```

Slika 103 Sadržaj review_df data frame-a

U stupcu koji sadrži recenzije potrebno je „očistiti“ tekst recenzija od riječi i znakova koji nisu relevantni za analizu. Za to je napisana funkcija *clean_text()* kojoj se proslijedi tekst recenzije, a u njoj se sva slova riječi pretvore u mala slova, uklone se interpunkcijski znakovi, uklone se riječi koje sadrže broju, izbace se sve zaustavne riječi (*the, a, this*, itd.), uklone se prazni tokeni te se za svaku riječ označi kojoj vrsti riječi pripada (POS – engl. Part-of-Speech taging). U POS označavanju koristi se funkcija *get_wordnet_pos()* kojoj se proslijedi riječ i na osnovu slova na koje počinje odredi se kojoj vrsti riječi pripada. U konačnici se riječi lematiziraju – nađe se osnovni oblik riječi, te se izbace sve riječi koje imaju samo jedno slovo, a ostala spoje i tako očišćena recenzija sprema se u novi stupac – *review_clean*:

```
def get_wordnet_pos(pos_tag):
    if pos_tag.startswith('J'):
        return wordnet.ADJ
    elif pos_tag.startswith('V'):
        return wordnet.VERB
    elif pos_tag.startswith('N'):
        return wordnet.NOUN
    elif pos_tag.startswith('R'):
        return wordnet.ADV
    else:
        return wordnet.NOUN

def clean_text(text):
    # lower text
    text = text.lower()
    # tokenize text and remove puncutation
    text = [word.strip(string.punctuation) for word in text.split(" ")]
    # remove words that contain numbers
    text = [word for word in text if not any(c.isdigit() for c in word)]
    # remove stop words
    stop = stopwords.words('english')
    text = [x for x in text if x not in stop]
    # remove empty tokens
    text = [t for t in text if len(t) > 0]
    # pos tag text
    pos_tags = pos_tag(text)
    # lemmatize text
    text = [WordNetLemmatizer().lemmatize(t[0], get_wordnet_pos(t[1])) for t in pos_tags]
    # remove words with only one letter
    text = [t for t in text if len(t) > 1]
    # join all
    text = " ".join(text)
    return(text)
```

Kada su recenzije uređene određuje se njihov sentiment odnosno sentiment riječi sadržanih u recenziji. Sentiment će se koristiti kao jedan od značajki za izradu modela i daljnju analizu. Za određivanje sentimenta koristi se NLTK model Vader koji koristi leksikon riječi da bi pronašao pozitivne i negativne riječi, te uzima u obzir kontekst rečenica kako bi odredio ocjenu polariteta. Za svaki tekst vraća 4 vrijednosti:

- ocjenu neutralnosti,

- ocjenu pozitivnosti,
- ocjenu negativnosti,
- ocjenu koja sažima prethodne vrijednosti.

Funkciji *polarity_scores()* proslijedi se recenzija te ona vrati gore navedene 4 ocjene. Obzirom da se sve spremi u jedan stupac *data frame-a* potrebno je svaku ocjenu spremiti u poseban stupac i to se postiže tako da se podatke iz stupca *sentiments* spremi u formatu *Pandas Series*:

```
# add sentiment anaylsis columns
sid = SentimentIntensityAnalyzer()
reviews_df["sentiments"] = reviews_df["review"].apply(lambda x: sid.polarity_scores(x))
reviews_df = pd.concat([reviews_df.drop(['sentiments'], axis=1), reviews_df['sentiments'].apply(pd.Series)], axis=1)
```

Zatim se za još značajki dodaje broj riječi i slova u svakoj recenziji:

```
# add number of characters column
reviews_df["nb_chars"] = reviews_df["review"].apply(lambda x: len(x))
# add number of words column
reviews_df["nb_words"] = reviews_df["review"].apply(lambda x: len(x.split(" ")))
```

Sljedeći se korak sastoji u izvlačenju vektorskih prikaza za svaku recenziju. Svaki se tekst može pretvoriti u numeričke vektore pomoću vektora riječi (*Doc2Vec*). Isti tekstovi će imati slične vektore i zato se ti vektori mogu koristiti kao značajke. *TaggedDocument()* funkcija označava „očišćene“ recenzije tako da svaku recenziju spremi kao listu riječi i kao ključ dodijeli broj retka u kojem se nalazi. Tako označene recenzije proslijede se funkciji *Doc2Vec()* koja će vratiti 5-dimenzionalni vektor, najveća udaljenost između trenutne riječi i predviđene je 2, ignorirat će se riječi koje su se jednom pojavile u tekstu i za izradu modela koristit će se 4 niti. Naučenim modelom određuje se 5-dimenzionalni vektor značajki za svaku „očišćenu“ recenziju i sprema se u *data frame doc2vec_df* koji se zatim dodaje u *reviews_df data frame*:

```
# create doc2vec vector columns
documents = [TaggedDocument(doc, [i]) for i, doc in enumerate(reviews_df["review_clean"].apply(lambda x: x.split(" ")))]

# train a Doc2Vec model with our text data
model = Doc2Vec(documents, vector_size=5, window=2, min_count=1, workers=4)

# transform each document into a vector data
doc2vec_df = reviews_df["review_clean"].apply(lambda x: model.infer_vector(x.split(" "))).apply(pd.Series)
doc2vec_df.columns = ["doc2vec_vector_" + str(x) for x in doc2vec_df.columns]
reviews_df = pd.concat([reviews_df, doc2vec_df], axis=1)
```

Na kraju se u značajke dodaje i TF-IDF (engl. Term Frequency - Inverse Document Frequency) vrijednost za svaku riječ i svaku recenziju. Kako bi se odredila važnost pojedine riječi u recenziji sa TF se računa koliko puta se ta riječ pojavila u recenziji, a IDF računa relativnu važnost riječi koja ovisi o tome koliko se recenzija riječ može pronaći. Svakoj riječi koja se pojavljuje najmanje 10 puta dodaje se TF-IDF stupac u *tfidf_df data frame-u* i kao naziv stupca uzme se riječ „*word_*“ i ta riječ. *tfidf_df data frame* spaja se sa *reviews_df data frame-om* te je konačan sadržaj *reviews_df data frame-a* prikazan na Slici 104 i Slici 105:

```
# add tf-idfs columns
tfidf = TfidfVectorizer(min_df = 10)
tfidf_result = tfidf.fit_transform(reviews_df["review_clean"]).toarray()
tfidf_df = pd.DataFrame(tfidf_result, columns = tfidf.get_feature_names())
tfidf_df.columns = ["word_" + str(x) for x in tfidf_df.columns]
tfidf_df.index = reviews_df.index
reviews_df = pd.concat([reviews_df, tfidf_df], axis=1)
```

Index	review	is_bad_review	rating	review_clean	neg	neu	pos	compound	nb_chars	nb_words	doc2vec_vector_0	doc2vec_vector_1
0	Hotel is perfect loca…	0	30	hotel perfect location nat…	0	0.916	0.084	0.7845	508	91	-0.15739709	0.016016183
1	Stayed for 2 nights with …	0	40	stay night family clean…	0	0.722	0.278	0.9371	277	48	-0.09827989	0.026358014
2	Stayed here as part of a…	0	50	stayed part bus tour roo…	0	0.695	0.305	0.9545	277	53	0.009196473	0.028574985
3	Stayed for one night, h…	0	30	stayed one night hotel …	0.133	0.642	0.225	0.6802	210	39	-0.008208353	-0.012658497
4	This place will make yo…	0	30	place make lake trip ea…	0.021	0.826	0.153	0.7984	277	57	0.02763487	0.085026614
5	The hotel is very clean a…	0	40	hotel clean everything n…	0	0.779	0.221	0.8882	277	49	-0.09857721	0.0433276
6	We loved the staff they g…	0	40	loved staff greet warm w…	0	0.716	0.284	0.9506	277	57	0.018604567	0.10567309
7	It's mostly to spend the…	0	30	it's mostly spend night …	0.03	0.857	0.113	0.5301	273	51	-0.13423395	-0.04047439
8	The hotel was very near to…	0	30	hotel near plitvice lak…	0.048	0.686	0.265	0.918	277	49	-0.08197125	0.05978507
9	Fantastic location if …	0	30	fantastic location vis…	0	0.783	0.217	0.8858	278	49	-0.08015213	0.11679043
10	If you want to experienc…	0	30	want experience p…	0	0.837	0.163	0.8519	280	61	-0.008803945	-0.05642343
11	Good little hotel locate…	0	30	good little hotel locate…	0.05	0.82	0.13	0.5423	221	40	-0.0442401	0.078260876
12	Stayed here on a tour an…	0	40	stay tour perfect loca…	0.051	0.675	0.274	0.882	209	43	0.010137867	-0.03238505
13	We stayed 3 nights at th…	0	40	stayed night hotel give d…	0	0.701	0.299	0.9627	279	51	0.005518805	-0.0750494
14	The hotel is just next to…	0	40	hotel next entrance pli…	0	0.788	0.212	0.8828	255	53	0.009242439	0.027673582
15	I was staying at Hotel Bel…	0	50	stay hotel bellevue two…	0	0.791	0.209	0.8908	278	51	0.011784122	0.0996003
16	The hotel was easy to find…	0	40	hotel easy find little …	0	0.708	0.292	0.9476	241	47	-0.080179974	0.11223832
17	A perfect gateway to v…	0	30	perfect gateway visi…	0	0.894	0.106	0.6229	242	51	-0.11029051	0.019078756
18	Stayed here for the nigh…	0	30	stay night location gre…	0	0.801	0.199	0.8735	275	50	-0.02454526	0.051987223
19	The main reason for s…	0	40	main reason stay hotel w…	0	0.779	0.221	0.9169	277	55	0.041753225	-0.0510713

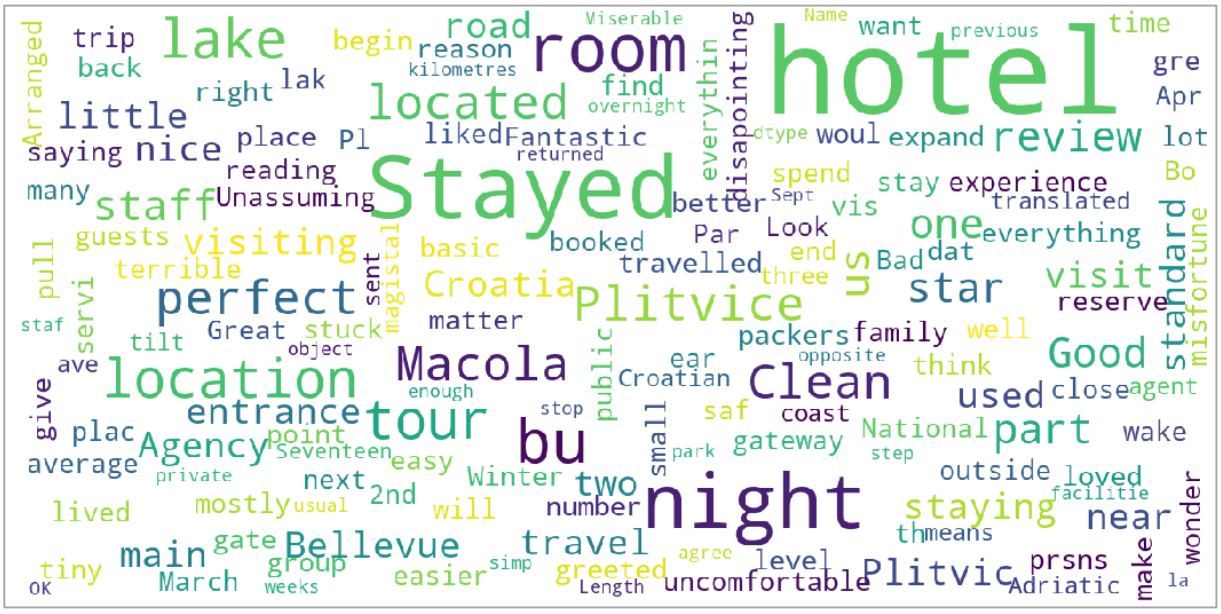
Slika 104 Sadržaj review_df data frame-a (1)

Index	doc2vec_vector_2	doc2vec_vector_3	doc2vec_vector_4	word_access	word_adequate	word_air	word_also	word_although	word_and	word_area	word_around	word_arrive
0	-0.3653045	-0.18202558	0.1360333	0	0	0	0	0	0	0	0.23145	0
1	-0.060085405	-0.05202642	0.18451804	0	0	0	0	0	0	0	0	0
2	-0.16683681	-0.047253676	0.15674362	0	0	0.29229	0	0	0	0	0	0
3	-0.07068773	-0.14596291	-0.0068158233	0	0	0	0	0	0	0	0	0
4	-0.26920968	0.002477484	0.20424525	0	0	0	0	0	0	0	0	0.21421
5	-0.05998343	-0.027848555	0.052730992	0	0	0	0	0	0	0	0	0
6	-0.19659188	0.022390114	0.044495173	0	0	0	0	0.290494	0	0	0	0
7	-0.18657391	-0.15221436	0.08719134	0	0	0	0	0	0	0	0	0
8	-0.13969064	-0.005972357	0.06777007	0	0	0	0	0	0	0	0	0
9	-0.25411415	-0.0038879316	0.20949891	0	0	0	0	0.264755	0	0	0	0
10	-0.14147082	-0.05764222	0.09103877	0	0	0	0	0	0	0.261237	0	0
11	-0.007084392	-0.0018156354	0.09840805	0	0	0	0	0	0	0	0	0
12	-0.07793624	-0.062678315	-0.04823441	0	0	0	0	0	0	0	0	0
13	-0.196124	0.010578447	0.14794068	0	0	0	0	0	0	0	0	0
14	-0.09257377	0.023417939	0.08352445	0	0	0	0	0	0	0	0	0
15	-0.062042393	0.023157356	0.07232322	0	0	0	0	0	0	0	0	0
16	-0.14681643	-0.018074807	0.19353344	0.310396	0	0	0	0	0	0	0	0
17	0.025941156	-0.033066794	-0.04588185	0	0	0	0	0	0	0	0	0
18	-0.17017677	-0.018125772	0.035866063	0	0	0	0	0.26751	0	0	0	0
19	-0.15909028	-0.0008017973	-0.0043346738	0	0	0	0	0	0	0	0	0

Slika 105 Sadržaj review_df data frame-a (2)

Kada se analizira broj pozitivnih i negativnih recenzija vidljivo je da skup podataka nije uravnotežen jer sadrži 88% pozitivnih i samo 12% negativnih komentara, a to će biti vrlo važno kod izrade klasifikatora.

U izrađenom oblaku riječi najčešće pojavljivane riječi u recenzijama su *hotel, stayed* i *night*. Pojavljuju se i imena hotela Macola i Bellevue, te komentira opći dojam smještaja: *room, clean, uncomfortable, nice, miserable, fantastic, good*. Kako se u hotele dolazi najviše zbog posjeta Nacionalnom parku Plitvička jezera u recenzijama se osvrće i na dojam koji je nacionalni park ostavio na posjetitelje, a to se vidi iz riječi: *lake, Plitvice, location, located, near, close, National*. Spominju se riječi *Winter, March* i *April* iz čega se može zaključiti da je većina ljudi koji su ostavili svoju recenziju na *TripAdvisor*-u došla zimi ili u proljeće (Slika 106).



Slika 106 Oblak riječi koje se pojavljuju u recenzijama

U nastavku je izdvojeno 10 najpozitivnijih, najneutralnijih i najnegativnijih recenzija odnosno samo one koje sadrže više od šest riječi. Na Slici 107 među pozitivnim recenzijama najviše se komentira izgled sobe i blizina Nacionalnom parku Plitvička jezera.

	review	pos
43	Basic but comfortable and clean rooms !.. Nice...	0.545
241	Pleasantly surprised as told it was a hangover...	0.487
223	Certainly a throwback to the regions communist...	0.454
23	Great location-close to the 2nd entrance to Pl...	0.453
60	Great place to stay, close to the number 2 ent...	0.431
140	Hotel Bellevue is right by the entrance of the...	0.430
67	Perfectly acceptable. Better than reviews. Ba...	0.425
215	Great. Nice hotel with good restourant. Diner ...	0.425
33	The rooms and restaurants are very basic, but ...	0.404
166	Hotel is clean and fairly large. Rooms are tid...	0.387

Slika 107 10 najpozitivnijih recenzija

Negativne recenzije odnose se na neudobnost ili dotrajalost smještaja, a jedan među njima istaknuo je nezadovoljstvo velikom udaljenosti od nacionalnog parka (Slika 108).

	review	neg
296	Clean but v uncomfortable back packers hotel. ...	0.314
288	We had to go here for food as part of a trip t...	0.249
148	Bellevue Hotel is a relic of the '80s. Our ro...	0.242
310	Seventeen kilometres from park. Miserable staf...	0.222
95	Nice room, although the balcony door did not l...	0.219
240	We were warned that this is a basic hotel, but...	0.182
116	Never had high hopes for this hotel based on t...	0.180
210	On the plus side - smallish room but it had c...	0.175
230	Although I was skeptical due to no a/c we had ...	0.169
222	Hotel is within the Plitvice National Park and...	0.161

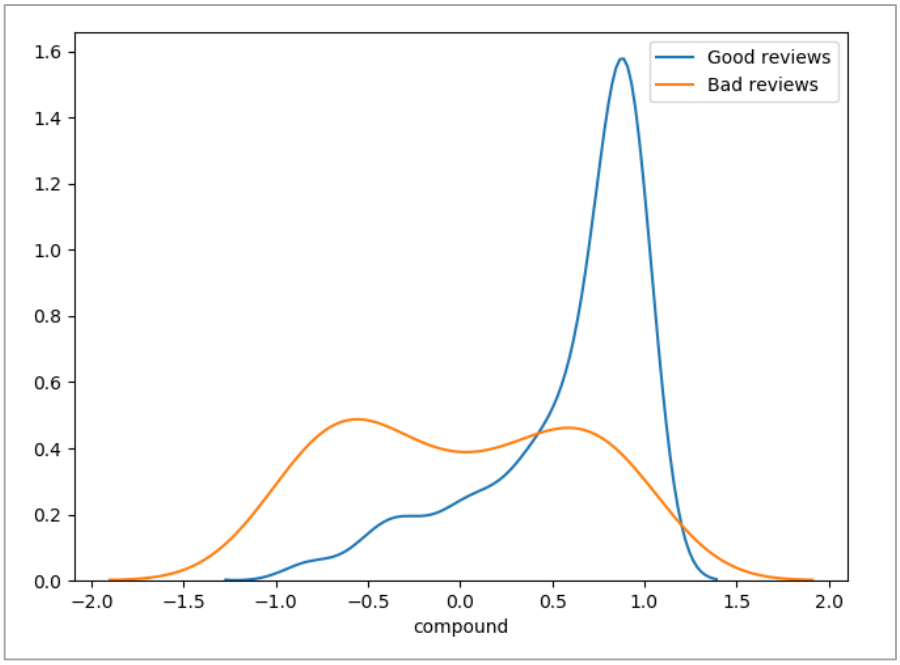
Slika 108 10 najnegativnijih recenzija

U neutralnim recenzijama prevladavaju turisti koji su odsjeli u nekom od hotela jer su bili u prolazu ili su ga odabrali jer je bio najisplativiji (Slika 109).

	review	neu
194	You need only walk down a small path to the tr...	1.000
65	We stayed here for one night after visiting th...	1.000
99	I first stayed at the Bellevue Hotel 26 years ...	1.000
122	We chose this hotel because it was cheapest on...	1.000
257	This is one of three hotels nearest the Plitic...	1.000
318	It is located just opposite the main road and ...	1.000
118	We stayed one night at the Bellevue to give us...	0.978
295	Bad,disapointing...of the all staff which we m...	0.975
168	We were traveling by car, but didn't want to h...	0.974
182	If you want to see the non-splendor of communi...	0.971

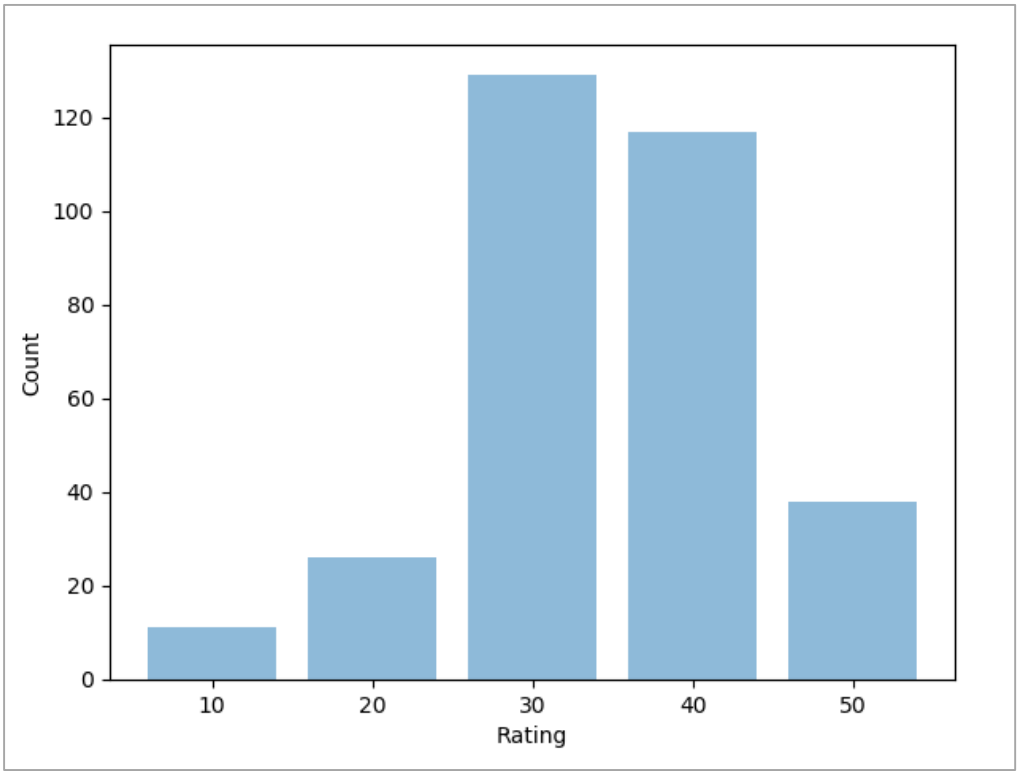
Slika 109 10 najneutralnijih recenzija

Grafikon na Slici 110 prikazuje distribuciju sentimenta recenzija prema pozitivnim i negativnim recenzijama. Pozitivne recenzije većinom se smatraju pozitivnim mišljenjem dok u negativnim recenzijama prevladava loše mišljenje o hotelima.



Slika 110 Distribucija sentimenta recenzija

Na Slici 111 prikazana je frekvencija ocjena pojedine recenzije. Osobe koje su napisale recenzije hotelima su najviše davale ocjene 30 i 40 što je dosta pozitivnih ocjena. Negativnih ocjena ima vrlo malo te kada bi se zbrojio broj ocjena 10 i 20 bio bi podjednak kao i broj ocjene 50 što je najviša moguća ocjena.



Slika 111 Frekvencije ocjena recenzija

Nakon analize slijedi izrada modela od izlučenih značajki. Za značajke koje će se koristiti u izradi modela uzima se sve osim oznake da li je recenzija pozitivna ili negativna te sama recenzija i „očišćena“ recenzija. Skup značajki podijeli se na skup za treniranje i testiranje u omjeru 80:20, te se za klasifikaciju koristi metoda slučajnih šuma (engl. *Random Forest Classifier*). Funkcija *RandomForestClassifier()* određuje da će se izraditi 100 stabla odluka, a *fit()* gradi navedena stabla odluke od skupa za treniranje:

```
''' making classifier'''
# feature selection
label = "is_bad_review"
ignore_cols = [label, "review", "review_clean"]
features = [c for c in reviews_df.columns if c not in ignore_cols]

# split the data into train and test
X_train, X_test, y_train, y_test = train_test_split(reviews_df[features], reviews_df[label],
                                                    test_size = 0.20, random_state = 42)

# train a random forest classifier - 100 desicion trees
rf = RandomForestClassifier(n_estimators = 100, random_state = 42)
rf.fit(X_train, y_train)
```

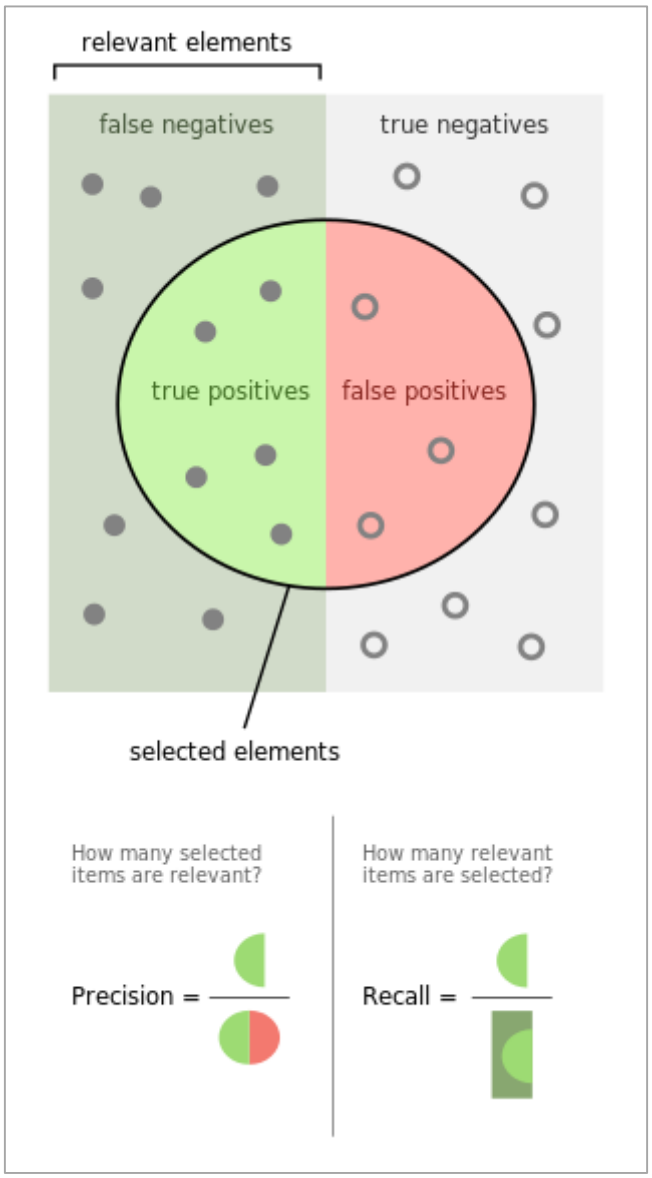
Najvažnije značajke koje su se koristile za izradu modela su ocjena recenzije te prethodno izlučene značajke kao što su sentiment te pojedine riječi koje imaju veliki značaj za izradu modela (npr. *great, reception, night, room, hotel*) (Slika 112).

	feature	importance
0	rating	0.286122
4	compound	0.056980
3	pos	0.031593
59	word_drive	0.028170
1	neg	0.026744
7	doc2vec_vector_0	0.018550
81	word_great	0.018159
158	word_sure	0.015318
135	word_really	0.015255
136	word_reception	0.015155
143	word_say	0.014789
115	word_night	0.013686
6	nb_words	0.013092
11	doc2vec_vector_4	0.012393
142	word_room	0.012172
160	word_the	0.011742
5	nb_chars	0.011383
87	word_hotel	0.011157
8	doc2vec_vector_1	0.011075
26	word_bad	0.011067

Slika 112 Najvažnije značajke korištene za izradu modela

Izrađeni model koristi se za klasifikaciju testnih podataka. Da bi se provjerila točnost klasifikacije koristi se funkcija *classification_report()* koja za informaciju vraća preciznost, odziv, *F1-score* i *support*. Testni skup sadrži 65 recenzija od kojih 51 pripada klasi 0 (pozitivna recenzija), a 14 kasi 1 (negativna recenzija). Iz preciznosti (prvi stupac) vidljivo je da su sve negativne recenzije točno klasificirane, a za pozitivne je preciznost 91%. Odziv označava da su

sve pozitivne recenzije uistinu označene kao pozitivne tj. kad god se pojavila pozitivna recenzija onda je i prepoznata, dok je postotak negativnih koje su i označene negativnim 64%. Iz toga se može zaključiti da je klasifikator naklonjeniji klasi 0, a razlog tome je nebalansirani skup podataka. Preciznost se može gledati kao mjera korektnosti (koji postotak primjera koji su označeni kao pozitivni to stvarno i jesu), dok je odziv mjera potpunosti (koji postotak pozitivnih primjera je označen takvim). Vizualni prikaz razlike između preciznosti i odziva nalazi se na Slici 113.



Slika 113 Preciznost i odziv (Izvor: Wikipedia)

Mjera *F1-score* je harmonijska sredina preciznosti i odziva, a računa se na sljedeći način:

$$F1 = 2 * \frac{precision * recall}{precision - recall}$$

Micro avg računa navedene tri metrike tako da zbroji broj točno pozitivnih, lažno pozitivnih i lažno negativnih. *Macro avg* izračuna metriku za svaku klasu, zbroji njihove izračune i podijeli sa brojem klasa te ne pazi na neuravnotežen broj instanca klase. *Weighted avg* računa metriku za svaku klasu i nađe njen prosjek tako da izračun podijeli sa brojem točnih instanci. Potpuno izvješće o točnosti klasifikatora prikazano je na Slici 114.

	precision	recall	f1-score	support
0	0.91	1.00	0.95	51
1	1.00	0.64	0.78	14
micro avg	0.92	0.92	0.92	65
macro avg	0.96	0.82	0.87	65
weighted avg	0.93	0.92	0.92	65

Slika 114 Izvješće o točnosti klasifikatora

Zaključak

U provedenoj analizi otkriveno je da je u graničnom prometu više stranih turista te da najviše ulaze i izlaze iz države u ljetnim mjesecima. Od svih vrsta graničnog prometa najviše se koristi cestovni i zračni. Zračni promet sve više se koristi ljeti i više ga koriste strani turisti te se od zračnih luka po prometu izdvajaju Zračna luka Zagreb, Split i Dubrovnik. U cestovnom prometu najviše se koriste osobni automobili osobito kod domaćih turista čiji je izlazak povećan u prosincu zbog sezone skijanja te travnju za vrijeme proljetnih praznika. Strani turisti dolaze i autobusima dok domaći tu vrstu prijevoza koriste vrlo malo.

U sve popularnijim kružnim putovanjima najveći broj brodova dolazi sa Bahama i Malte ljeti te iz Italije u listopadu. Putnici sa Malte provedu najviše dana u Hrvatskoj, a brod sa najviše putnika dolazi sa Bahama. Najviše kružnih putovanja ima ljeti i u jesen te se postupno povećava i u zimskim mjesecima.

Kako je stranih turista više u graničnom prometu shodno tome ostvare veći broj dolazaka i noćenja od domaćih turista. Jedni i drugi najviše dolaze u srpnju i kolovozu te se njihov broj i broj noćenja povećava. Među stranim turistima najviše je Nijemaca, Austrijanaca, Slovenaca i Talijana. Svi dolaze ljeti, Nijemci i Austrijanci u srpnju i kolovozu, Slovenci u srpnju te Talijani u kolovozu kada im je *ferragosto*.

Strani turisti najviše odsjedaju u odmarališima, hotelima i kampovima dok domaći odsjedaju u hotelima ili odmarališima. I domaći i strani turisti većinom dolaze individualno, dok organizirano više dolaze strani. Kada dođu individualno ostvare više noćenja, a domaći kada dođu organizirano ostvare mali broj noćenja što bi značilo da uglavnom idu na jednodnevne izlete.

Gledajući statistička područja Hrvatske većina turista dolazi u Jadransku Hrvatsku i to najviše ljeti. Domaći turisti najviše dolaze u Primorsko-goransku županiju u kojoj najveći broj dolazaka broji grad Opatija i to u proljeće i jesen, a Crikvenica ostvari najveći broj noćenja. Od općina Lovran se ističe po broju dolazaka, a Malinska po broju noćenja. Strani turisti najviše borave u Istarskoj županiji. Najveći broj dolazaka i noćenja broji Rovinj, a posebno je zanimljiv Poreč koji ostvaruje puno noćenja i tamo se turisti najduže zadržavaju.

Po spolu među domaćim turistima je više muškaraca, a među stranima žena. Najviše muškaraca dolazi u dobi od 35 do 45 godina, a najmanje od 65 i više te isto vrijedi i za žene.

U Plitvičkim jezerima domaći turisti većinom dolaze u hotele, a strani u privatni smještaj. Od hotela su se istaknuli Hotel Jezero, Plitvice, Bellevue i Macola. Broj noćenja u navedenim smještajima od travnja raste i pada u rujnu, a najviše ih se zabilježi u kolovozu. U hotele Jezero, Plitvice i Bellevue najviše dolaze Azijati, a u Hotel Macola državljani Republike Koreje, Hrvatske, Mađarske, Njemačke i Španjolske.

Nacionalni park Plitvička jezera posjećuje se od travnja do rujna, a najviše posjeta ostvari se u kolovozu. Kod biranja smještaja turisti žele biti što bliže nacionalnom parku te ih najviše dolazi u Plitvička jezera, Jezerce, Plitvički Ljeskovac, Vrelo Koreničko i Korenicu. Veći broj kreveta u pojedinim mjestima ne privlači ih koliko blizina nacionalnog parka te se to vidi iz ostvarenog broja noćenja.

Za gore navedene hotele analizirane su recenzije kojih je 88% bilo pozitivno. Turisti su uglavnom komentirali opći dojam smještaja te se osvrtali na dojam nacionalnog parka. Pozitivni komentari sadržavali su pohvale na smještaj i zadovoljstvo nacionalnim parkom, dok su oni negativni većinom kritizirali dotrajalost i neudobnost smještaja te preveliku udaljenost od nacionalnog parka.

Literatura

[1] »Državni zavod za statistiku,« [Mrežno]. Available: https://www.dzs.hr/. [Pokušaj pristupa 16. Lipanj 2019.].

[2] »Turistička zajednica općine Plitvička jezera,« [Mrežno]. Available: https://www.discoverplitvice.com/hr/. [Pokušaj pristupa 20. Lipanj 2019.].

[3] »WebHarvy,« SysNucleus, [Mrežno]. Available: https://www.webharvy.com/articles/what-is-web-scraping.html. [Pokušaj pristupa 15. Srpanj 2019.].

[4] »Tabula,« 4. lipanj 2018.. [Mrežno]. Available: https://tabula.technology/. [Pokušaj pristupa 5. rujan 2019.].

[5] A. Ronquillo, »Real Python,« 23. Siječanj 2019.. [Mrežno]. Available: https://realpython.com/python-requests/#getting-started-with-requests. [Pokušaj pristupa 15. Srpanj 2019.].

[6] »Computational Journalism,« 2016.. [Mrežno]. Available: http://www.compjour.org/warmups/govt-text-releases/intro-to-bs4-lxml-parsing-wh-press-briefings/. [Pokušaj pristupa 17. Srpanj 2019.].

[7] »Propisi.hr,« [Mrežno]. Available: http://propisi.hr/print.php?id=11863. [Pokušaj pristupa 30. Kolovoz 2019.].

[8] »Free Proxy List,« Disoft Ltd., [Mrežno]. Available: https://free-proxy-list.net/. [Pokušaj pristupa 10 Rujan 2019.].

[9] J. Oheix, »Kaggle,« 18. Prosinac 2018.. [Mrežno]. Available: https://www.kaggle.com/jonathanoheix/sentiment-analysis-with-hotel-reviews. [Pokušaj pristupa 9. Rujan 2019.].