# Prikaz podataka i tehnike vizualizacije

**Smajić, Alma**

Sveučilište Jurja Dobrile u Puli

Fakultet informatike

**ALMA SMAJIĆ**

# PRIKAZ PODATAKA I TEHNIKE VIZUALIZACIJE

Završni rad

Pula, 13. rujna 2021.

Sveučilište Jurja Dobrile u Puli

Fakultet informatike

**ALMA SMAJIĆ**

# PRIKAZ PODATAKA I TEHNIKE VIZUALIZACIJE

Završni rad

JMBAG: 0303082451, redovna studentica

STUDIJSKI SMJER: informatika

KOLEGIJ: statistika

MENTOR: doc.dr.sc. Siniša Miličić

Pula, 13. rujna 2021.

**IZJAVA O KORIŠTENJU AUTORSKOG DJELA**

Ja, Alma Smajić dajem odobrenje Sveučilištu Jurja Dobrile u Puli, kao nositelju prava iskorištavanja, da moj Završni rad pod nazivom Prikaz podataka i tehnike vizualizacije

koristi na način da gore navedeno autorsko djelo, kao cjeloviti tekst trajno objavi u javnoj internetskoj bazi Sveučilišne knjižnice Sveučilišta Jurja Dobrile u Puli te kopira u javnu internetsku bazu završnih radova Nacionalne i sveučilišne knjižnice (stavljanje na raspolaganje javnosti), sve u skladu s Zakonom o autorskom pravu i drugim srodnim pravima i dobrom akademskom praksom, a radi promicanja otvorenoga, slobodnoga pristupa znanstvenim informacijama.

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

U Puli, 13. rujna 2021.

Potpis

Alma Smajić

**IZJAVA O KORIŠTENJU AUTORSKOG DJELA**

Ja, Alma Smajić dajem odobrenje Sveučilištu Jurja Dobrile u Puli, kao nositelju prava iskorištavanja, da moj Završni rad pod nazivom Prikaz podataka i tehnike vizualizacije

koristi na način da gore navedeno autorsko djelo, kao cjeloviti tekst trajno objavi u javnoj internetskoj bazi Sveučilišne knjižnice Sveučilišta Jurja Dobrile u Puli te kopira u javnu internetsku bazu završnih radova Nacionalne i sveučilišne knjižnice (stavljanje na raspolaganje javnosti), sve u skladu s Zakonom o autorskom pravu i drugim srodnim pravima i dobrom akademskom praksom, a radi promicanja otvorenoga, slobodnoga pristupa znanstvenim informacijama.

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

U Puli, 13. rujna 2021.

Potpis

Alma Smajić

## Sadržaj

# UVOD

Vizualizacija podataka postala je svojevrstan jezik koji spaja kategoričke/numeričke varijable sa slikom. Korisna je kada postoji potreba za povećanjem ljudskih sposobnosti umjesto njihove zamjene računalnim metodama. Prava bi vizualizacija trebala:

- uštedjeti na vremenu,
- imati jasnu svrhu,
- uključivati relevantan kontekst, te
- prikazivati podatke na prikladan način.

Prikladan način podrazumijeva profesionalnost pojedinca; kako u sferi znanosti - programiranje koda koji će dati očekivani grafički ishod tako i u sferi umjetnosti - vizualno usklađivanje boja i njihov utjecaj na ciljanu publiku.

Da bi oboje bilo osigurano vrlo je važan pristup neovisno o kojim i kakvim podacima taj pojedinac raspolaže. Obrada podataka bi trebala biti stručna i sveobuhvatna kako bi se izbjegle potencijalne anomalije i nedostaci u budućoj vizualizaciji. Ako su ti postulati zadovoljeni prelazi se na ostale, direktno vezane uz istu. Kronološkim redom to su:

- izbor ispravne vizualizacijske metode
- determiniranje stupnja točnosti dobivenog grafa/dijagrama
- raspored elemenata i izbor boja za pripadajuće.

Uz sva spomenuta teorijska objašnjenja prikazan je i autorski kod na deset primjera s popratnim vizualnim prikazom. Korišteni programski jezik je Python u Jupyter Notebook okruženju sa pripadajućim bibliotekama *Matplotlib, Seaborn, Plotly* i pomoćnom *Numpy*.

MATEMATIČKO-STATISTIČKA TEORIJA

Polazišna točka u teoriji je uzorak definiran kao „*podskup osnovnog statističkog skupa izabran tako da se reprezentativnom metodom s pomoću njega mogu procijeniti svojstva svih elemenata osnovnog skupa*[1]“. Uzorak se formulom prikazuje kao:

$$\bar{x} = (x_1, \cdots x_n)$$

Za višedimenzionalne slučajeve poput matplotliba koji je 2D, formula je:

$$x = \big((x_1, y_1), (x_2, y_2), \cdots (x_n, y_n)\big)$$

Uzorak sam po sebi može biti kvantitativan (numerički)– prikazuje brojčanu vrijednost koju kategorija podataka postiže ili kvalitativan (kategorijski) – predstavlja pripadnost određenoj skupini. Uz to, svaki uzorak ima svoj mod koji predstavlja najčešću vrijednost. Ovisno o distribucijama koje opisuju koliko često se pojavljuju vrijednosti podataka, razlikuju se:

- unimodalne – prisutnost samo jednog moda, tj. jedne najviše vrijednosti
- bimodalne – prisutnost dvaju modova
- mutlimodalne – prisutnost više modova

Nastavno na mod, kao deskriptivne mjere sredine povezuju se još i prosjek (aritmetička sredina) te medijan. Prosjek predstavlja sumu svih produkata u uzorku podijeljenu na ukupan broj podataka uzorka:

$$\bar{x} = \frac{\sum x_i \cdot \sum n_i}{n},\ n\ = \sum n_i$$

Drugi „prijatelj“ moda je prethodno spomenuti medijan – položajna središnja vrijednost uzorka. Da bi shvatili princip određivanja pozicije medijana potrebno je poznavati funkciju distribucije – opisuje vjerojatnosna svojstva za oba tipa slučajnih varijabli (diskretna i kontinuirana):

$$Q(p) = inf\{x: F(x) \geq p\}, p \in (0,1)$$

[1]Izvor: https://www.enciklopedija.hr/Natuknica.aspx?ID=63577 , pristup: 11.9.2021.

Tek onda se njegova vrijednost definira:

- funkcijom kvantila podataka $F(x)$ – inverz funkcije distribucije prikazan kao:

$$F(x) = p(X \leq x) = p, \quad Q(p) = x$$

- $F\left(\frac{1}{2}\right)$ – vrijednost funkcije na polovini uzorka pri čemu treba imati na umu da pandas-ova funkcija `median()` ignorira sve vrijednosti koje nedostaju

Bliska asocijacija kod medijana, poznatog kao drugi kvartil, su preostali kvartili – kvantili koji dijele niz na 4 jednaka dijela. Q1 predstavlja donji ili prvi kvartil, Q3 je gornji ili treći kvartil, a njihova razlika – IQR – naziva se interkvartil:

$$IQR = Q_3 - Q_1$$

Kvantili kao vrijednosti statističkog obilježja koja statistički niz dijele na q jednakih dijelova, uz kvartile, razlikuju i decile – dijele niz na 10 jednakih dijelova i percentile – dijele niz na 100 jednakih dijelova.

Posljednje što se najviše ističe na svakom grafu su `outlieri`, tj. stršeće vrijednosti. Ovdje upadaju sve vrijednosti koje odstupaju od uzorka, a računaju se na dva načina:

- preko interkvartila - za koji vrijedi da je:
  - gornja granica: $Q_3 + 1.5(IQR)$
  - donja granica: $Q_1 - 1.5(IQR)$
- preko standardne devijacije:
  - formula za izračun standardne devijacije:

    $\sigma(x) = \sqrt{Var(x)} = \sqrt{\left(\sum_{i=0}^{k} pi\xi_i^2\right) - (\bar{x})^2}$, a sve vrijednosti koje su triput veće od nje smatraju se stršećima

Spomenuti pojam standardna devijacija označava pozitivnu vrijednost drugog korijena varijance uzorka, pri čemu je varijanca mjera disperzije koja prikazuje prosječnu sumu kvadrata odstupanja vrijednosti dobivenog uzorka od aritmetičke sredine.

Posljednja funkcija matematičkog poglavlja vezana je uz KDE graf i funkciju gustoće vjerojatnosti koja predstavlja relativnu vjerojatnost kontinuirane varijable da će poprimiti određenu vrijednost unutar intervala. Formulom prikazano kao:

$$P(a \leq X \leq b) = \int_a^b f(x)\, dx$$

Pri čemu su:

- $[a, b]$ – interval u kojem x leži
- $p(a \leq X \leq b)$ – vjerojatnost da je vrijednost x unutar tog intervala
- $dx$ – razlika b-a

## GRAFIČKI I VIZUALNI ASPEKTI

Korišteni podaci većinom su preuzeti sa Kaggle-a – zajednica podatkovnih znanstvenika i praktikanata strojnog učenja sa ulogom traženja i objavljivanja skupova podataka te izgradnje modela u okvirima podatkovne znanosti. Za ostatak primjera korišteno je vlastito očitavanje podataka koji su dostupni online ili u namirnicama koje nas okružuju.

Pojedini primjeri poput histograma i kutijastog dijagrama koriste iste skupove podataka i kako bi se pokazale prednosti svakog od grafova. Grafički aspekt koji ide histogramu u prilog je korisnost kod prikaza malih/velikih razlika među promatranim frekvencijama gdje bi kutijasti dijagram određene podatke prikazao na istoj plohi i time „normalizirao“ distribuciju. Ipak, kutijasti dijagram ima svoju prednost kod prikaza podataka gdje promatrane frekvencije imaju oscilacije pri čemu dijagram ima i dalje postojan izgled. Histogram bi u tom slučaju izgledao nesimetrično što bi moglo rezultirati pretpostavkom da su podaci iskrivljeni.

Neovisno o vrsti grafa/dijagrama i količini podataka sve vizualizacije bi trebale težiti istim vizualnim aspektima spomenutim u uvodu:

- usklađenost boja – prilagođenost svim korisnicima, posebice s naglaskom na osobe sa bilo kakvim oblikom vidnih poteškoća ili nemogućnosti raspoznavanja boja
- psihologija boja – reakcije i posljedična ponašanja osobe koja interpretira vizualizaciju na prisutnost određene boje, odnosno utjecaj na ciljanu publiku – je li vizualizacija postigla očekivano i je li prijenos informacija bio uspješan na razini postavljenih očekivanja

PRILAGOĐENOST BOJA

U vizualizaciji prije svega treba biti sveobuhvatan; od samog postavljanja grafa i njegovih boja do krajnjeg korištenja svih ispravnih podataka. Za početak, o bojama. Ako sami niste ili ne poznajete nekoga tko je daltonist vrlo teško da će Vam pasti na pamet prilagoditi svoje vizualizacije.

Svatko od nas vidi boju na svoj, drugačiji način, a „magija“ koja se krije iza svega su fotoreceptorne stanice. Upravo one su zaslužne za to kako će oči reagirati na pojedine svjetlosne valove. Govoreći o osobama koje slabije percipiraju pojedine boje, njihove fotoreceptorne stanice neće reagirati na jedan od oblika svjetlosnih nijansi crvenih (protanomalija), zelenih (deuteranomalija) ili plavih (tritanomalija) tonova kao što je vidljivo na slici 1.

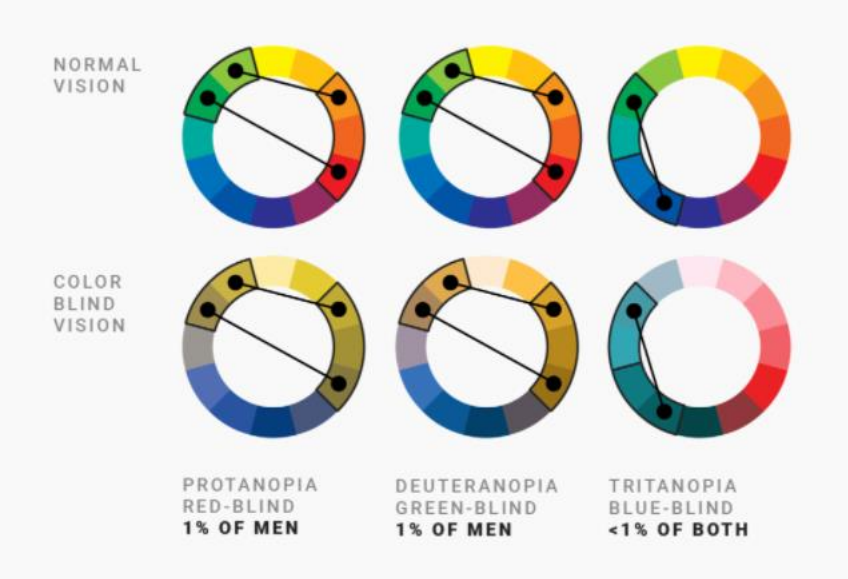

*Slika 1. Poremećaji u percepciji boja*

Ako se pak radi o daltonistima njihovi fotoreceptori uopće neće prepoznati gore spomenute boje. Ipak, želimo li da vizualizacija zadrži svoju prvotnu ulogu, dobra je ideja ne koristiti crveno-zelene kombinacije jer će one rezultirati nijansiranom smeđom vidljivoj na slici[2] 2.


*Slika 2. Daltonizam na spektralnom primjeru*

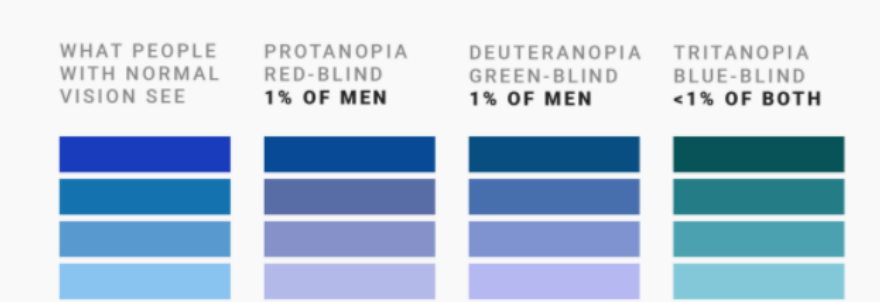
[2] Izvori slika: https://blog.datawrapper.de/colorblindness-part1/

U nastavku nešto o kontrastnim tonovima i kako spriječiti potencijalne misinterpretacije. Za vizualizacije koje bi trebale biti prilagođene daltonistima ne preporuča se kombinirati zelenu sa narančastom ili crvenom iste svjetline jer će takve boje samo izazvati ili dodatno pojačati konfuzije.




*Slika 3. Kombiniranje zelene sa krivim nijansama i tonovima*

Boja za koju se smatra da „stvara“ najmanje problema je plava. Neovisno o nijansi i zasićenosti najsigurnija je da će dočarati vizualizaciju onakvom kakva i uistinu jest.




*Slika 4. Nijansiranje plave*

Ukoliko nakon ovih savjeta i dalje postoji sumnja ili nesigurnost je li vizualizacija svima dobro vidljiva testirajte sa nekim od besplatnih alata dostupnih na Internetu, npr. Coblis ili Color Oracle.

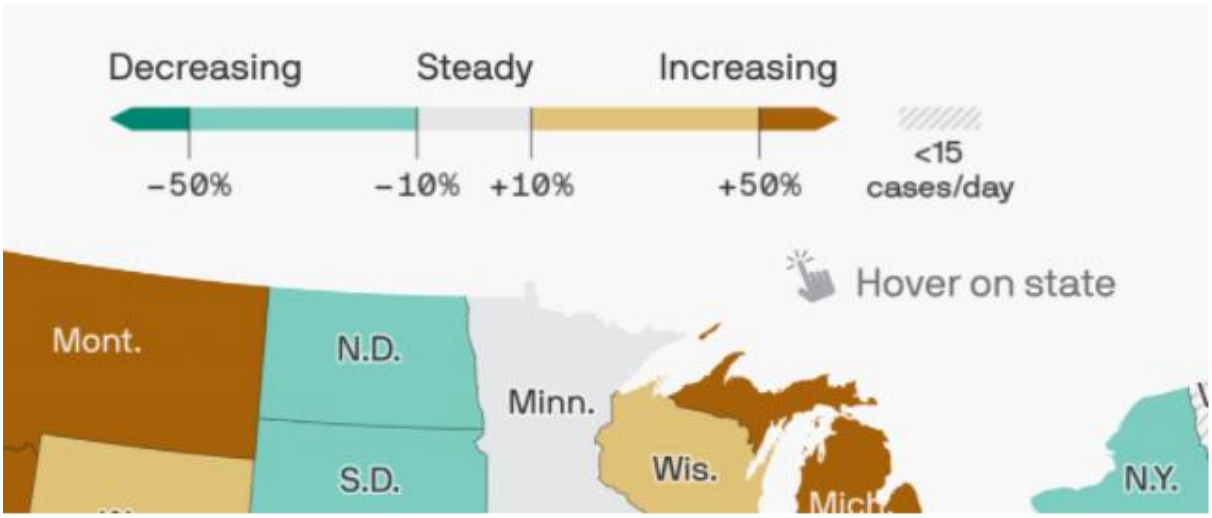
PSIHOLOGIJA BOJA

Nakon što smo izabrali podatke i uvjerili se da je s njima sve u redu te da će vizualizacija biti dobra dolazi, većini, lakši dio; izbor boja. Često se događa da već sam graf „diktira“ koje će se boje i gdje koristiti, a na nama ostaje izabrati iz ponuđenog.

Najbolji i najjednostavniji primjeri su kategoričke skale boja gdje razlikujemo širi raspon boja obzirom da svaka boja zastupa jednu kategoriju pri čemu je svaka kategorija jednako važna. Ako niste dični samostalnoj izradi paleta uvijek se nudi opcija izbora već postojećih materijala.

Za slučaj da se radi o numeričkim podacima koji se kreću od manjeg prema većem idealno je koristiti sekvencijalne skale boja. Detaljnije, to bi značilo da niže vrijednosti predstavljaju svjetliji tonovi, a više tamniji kao što je vidljivo na slici 32.

Usko povezane sa sekvencijalnim su divergentne ili bipolarne skale. Jedino što ih razlikuje je srednja vrijednost koja predstavlja „neutralno područje[3]“.




*Slika 5. Primjer bipolarne skale boja*

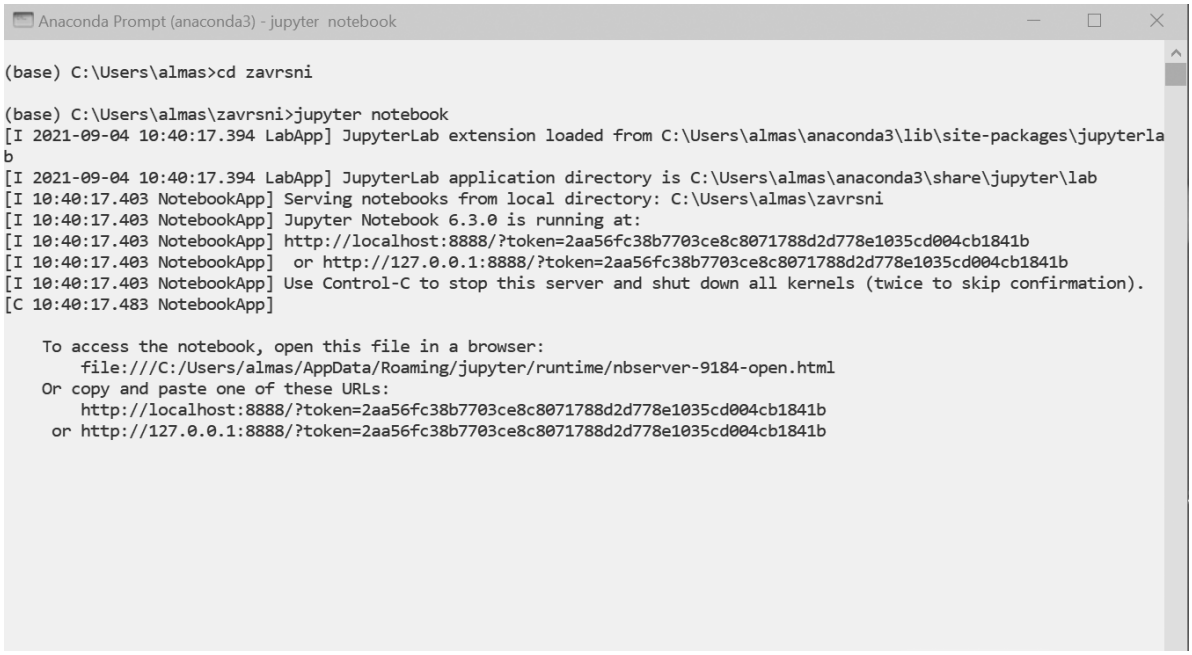
[3] Izvor slike: https://blog.datawrapper.de/which-color-scale-to-use-in-data-vis/

## SOFTWARE I TEHNIČKI UVJETI

## POSTAVLJANJE RADNOG OKRUŽENJA

Anaconda je znanstveno podatkovna platforma i distribucija za programske jezike Python i R. Fokusirana je na projekte sa većom količinom podataka i kao takva namijenjena pojednostavljenju upravljanja podacima i njihovoj implementaciji unutar spomenutih jezika.

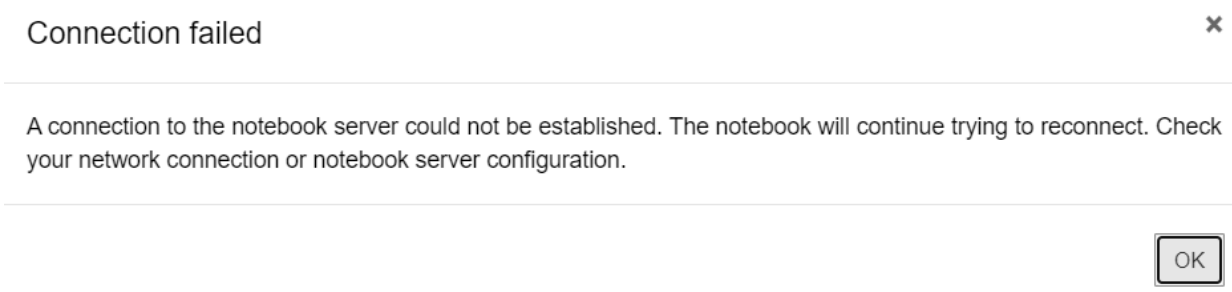
Online preuzimanjem platforme instanca projekta započinje pokretanjem Anaconda terminala u kojem se kreira radni folder, a zatim mu se pristupa.



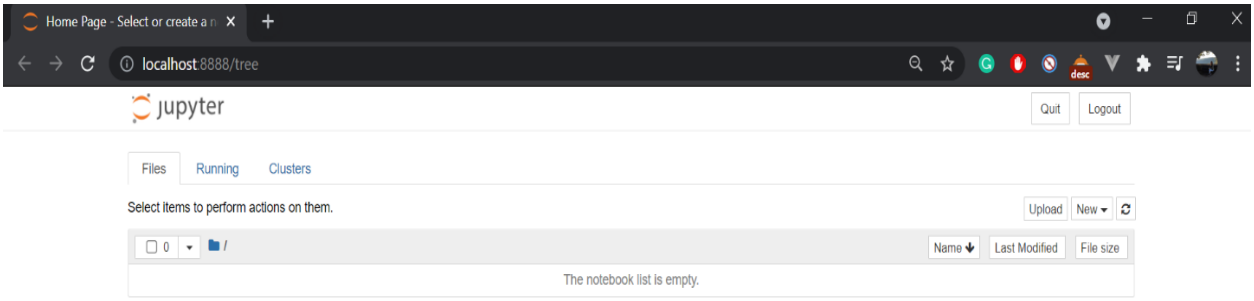

*Slika 6. Kreiranje instance projekta i naredba jupyter notebook*

Naredbom '*jupyter notebook*' pokreće se lokalni server na kojem se otvara prazan Jupyter notebook – *open-source* web aplikacija koja omogućuje stvaranje i dijeljenje dokumenata koji sadrže *live* ('živi') kod, jednadžbe, vizualizacije i narativni tekst.

Kako bi poslužitelj ostao cijelo vrijeme aktivan terminal mora nesmetano i dalje raditi, odnosno ostati uključen dokle god je korisnik u notebooku. Paralelno s tim terminal pamti svaki korisnikov unos i promjenu koju napravi na *kernel*u.




*Slika 7. Prikaz pogreške u trenutku kad se terminal ugasi*



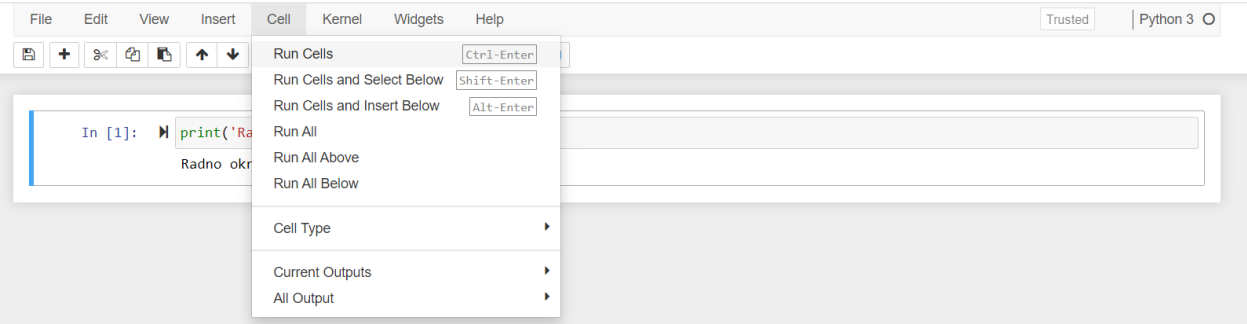

*Slika 8. Radno sučelje Jupyter Notebook*

Prva promjena je pokretanje *kernel*a sa *New* > *Python 3* - korisniku se otvara novi, u potpunosti prazan notebook koji će najprije u gornjem lijevom kutu klikom na `'Untitled'` preimenovati.

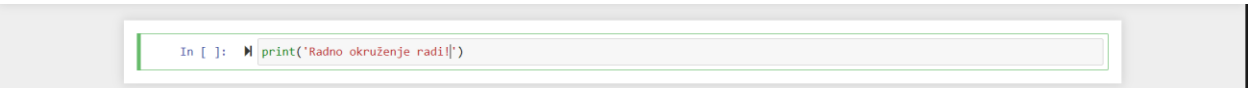
Ovo je samo testni notebook i nazvan je 'Postavljanje radnog okruženja' gdje mu je zadana naredba; ispis običnog teksta. Ovisno o tome je li korisnik u, *edit mode* – obrub mu se prikazuje zeleno ili u *command mode* – obrub mu se prikazuje plavo, ima određene ovlasti. Zeleno mu omogućuje unos teksta, *markdown*a[4], jednadžbi i koda, dok plavo pokretanje ili brisanje pojedinih stavki.

[4] Sintaksa formatiranja teksta

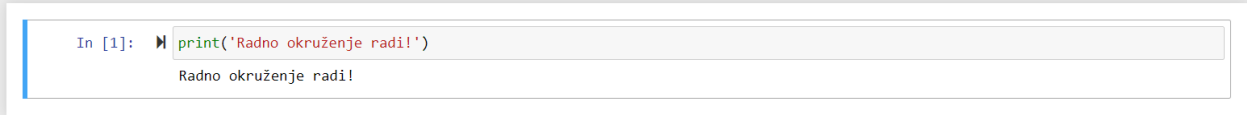
U jednostavnom primjeru korisnik pokreće liniju teksta preko naredbe '*Run Cells'* i dobiva očekivano; tekst se uspješno ispisuje. Kako ne bi svaki put ulazili u *Cell options* dovoljno je istu naredbu odraditi prečacima s tipkovnice; kombinacija tipki *Shift* + *Enter*.




*Slika 9. Primjer pokretanja koda*




*Slika 10. Funkcija print() - omogućuje ispis teksta*




*Slika 11. Uspješno kompiliranje funkcije print()*

Nakon što je osigurano da radno okruženje funkcionira, uvoze se biblioteke[5] ovisno o potrebama i tipu vizualizacije; *pandas, matplotlib, seaborn, plotly*, itd.

[5] Svaka od navedenih biblioteka može se instalirati i naredbom: `pip install imeBiblioteke`

## VIZUALIZACIJE I PRIKAZ PODATAKA

### TABLICA

Kako bi vizualizacija uopće dobila konačni izgled svakom grafu/dijagramu prethodi još jedan vid grafičkog prikaza podataka, a to je tablica. Unutar nje sve dostupne informacije su složene u „mrežu“ redova i stupaca koji korisnicima omogućuju brže skeniranje kroz podatke i traženje obrazaca među njima kako bi razvili uvid o dobivenim podacima. Principi[6] svake tablice su:

- organizacija – informacije su složene prema značajnosti, npr. abecedno ili hijerarhijski
- interaktivnost – u svakom trenutku korisnik mora imati mogućnost upravljanja tablicom prema načinu na koji on to želi
- intuitivnost – logička struktura tablice bi trebala jednostavno prikazivati sadržaj

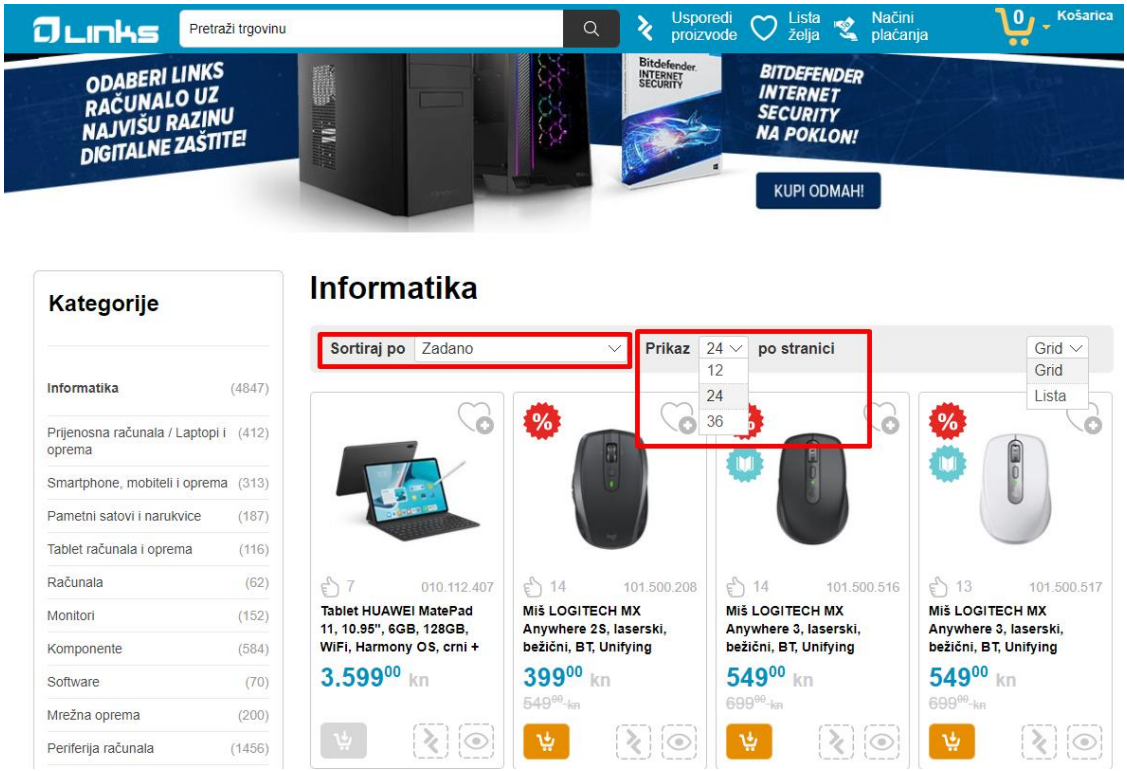
Međutim, nije uvijek slučaj da se svi pridržavaju ovih principa pa se često znaju vidjeti kaotične tablice iz kojih je teško dobiti korisne informacije. Prilikom odlučivanja o tome kako oblikovati tablicu, prednost treba dati čitljivosti i uklanjanju svih vizualnih „nereda“ koji potencijalno mogu odvratiti pogled. U nastavku slijede pravila tabličnog prikaza podataka[7]:

- izbor najboljeg stila reda – redovi uvelike pomažu korisnicima u skeniranju, čitanju i analizi podataka. Postavljanje redova u „mrežu“ preporuča se samo za tablice sa velikom količinom podataka, dok je za manje tablice preporučljivo koristiti horizontalne linije.
- održavanje kontrasta – uspostavljanje hijerarhije sa raznim stilovima teksta i bojama pozadine. Svaki *header* bi trebao biti reprezentativan i imati određenu težinu naspram cjelokupne tablice, a najlakši način za postizanje spomenutog je dodavanje tamnije pozadine u odnosu na tablicu.

[6] Ovisno o tipu tablice kojim raspolažemo postoje dodatna pravila, detaljnije na: https://material.io/components/data-tables

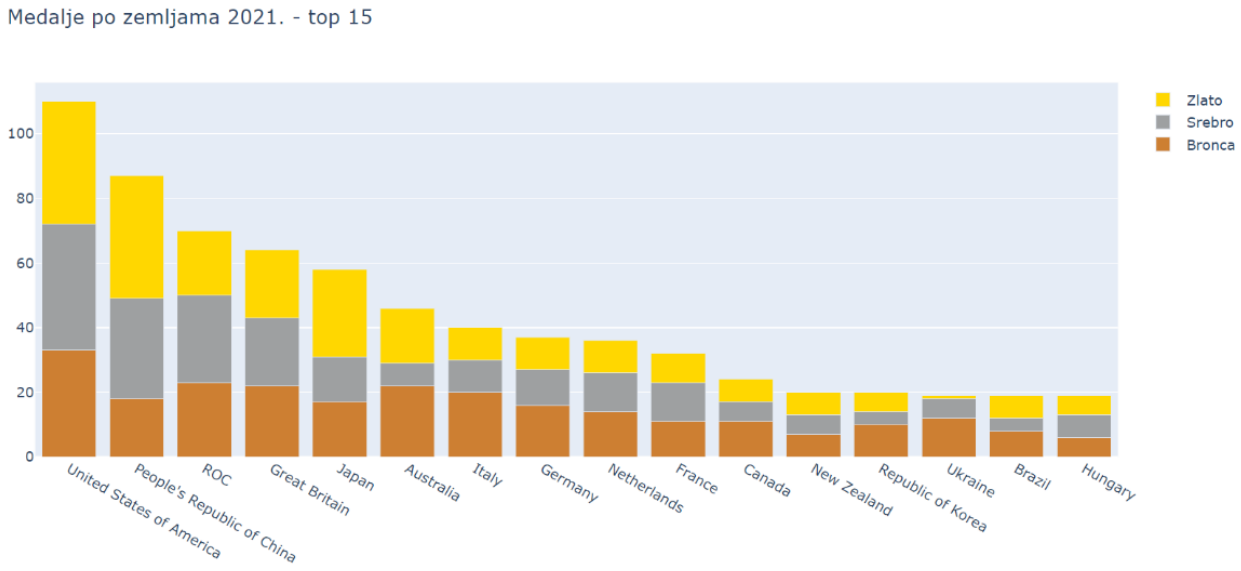
[7] Slikovne primjeri na: https://medium.com/design-with-figma/the-ultimate-guide-to-designing-data-tables-7db29713a85a

- ispravno poravnanje – predefinirano svi podaci u tablici imaju lijevo poravnanje što prati redoslijed čitanja s lijeva na desno. Nakon toga, sva imena stupaca moraju biti poravnata u skladu sa podacima koje predstavljaju.
- održavanje sadržaja – *header* i prvi stupac koji predstavlja *id* podatka bi trebali biti „usidreni“ kako bi korisnik u svakom trenutku znao kojem stupcu pripada promatrani podatak, odnosno na kojoj se poziciji trenutno nalazi
- filtriranje – korisno je dati mogućnost korisniku da dobivenu tablicu može filtrirati po njemu bitnim kategorijama, npr. želi izmijeniti predefinirano sortiranje po imenu u sortiranje po cijeni
- paginacija – za tablice s velikim količinama podataka savjetuje se „razbiti“ ih na više stranica od kojih će svaka imati isti broj redaka. Korisnik u svakom trenutku mora znati na kojoj je stranici i imati mogućnost navigacije između ostalih stranica, a za naprednije upite može mu se dati ovlast da sam manipulira brojem redova vidljivo na slici 13.




*Slika 12. Korisnička manipulacija uređivanjem tabličnog prikaza, Links portal (pristup: 11.9.2021.)*

## STUPČASTI DIJAGRAM




*Slika 13. Bar chart prikaz top 15 zemalja (OI, Tokio 2021.)*

Općenito, Kirk (2012.) navodi da su za svaki stupčasti dijagram (*bar chart*) potrebne dvije varijable; jedna kategorička (npr. olimpijac koji se natječe za Kanadu grafički pripada u kategoriju 'Kanada') i jedna numerička (npr. broj zlatnih medalja koje je kategorija 'Kanada' ostvarila).

Korisnost ovog dijagrama vidljiva je u jasnom prikazu usporedbi performansa pojedinih kategorija, dok su boje te koje će indicirati na isticanje određene vrijednosti. U većini grafova kategoričke varijable se postavljaju na apscisu, a ordinata je „rezervirana“ za brojčane vrijednosti koje predstavljaju frekvencije. Logika svakog stupčastog dijagrama je prema tome ista; što je više frekvencija, to je stupac duži.

```
df.head()
```

| | Country | Gold Medal | Silver Medal | Bronze Medal | Total | Rank By Total |
|---|---|---|---|---|---|---|
| **0** | United States of America | 38 | 39 | 33 | 110 | 1 |
| **1** | People's Republic of China | 38 | 31 | 18 | 87 | 2 |
| **2** | Japan | 27 | 14 | 17 | 58 | 5 |
| **3** | Great Britain | 21 | 21 | 22 | 64 | 4 |
| **4** | ROC | 20 | 27 | 23 | 70 | 3 |

*Slika 14. Pregled dostupnih podataka u csv datoteci*

Načela jasnog i preglednog dijagrama su:

- vrijednosti ordinate kreću od nule – osobi koja interpretira graf olakšana je usporedba dužine stupaca, tj. brži zaključak o vrijednostima omjera kojim graf rezultira
- ravni rubovi – naspram ostalih vrsta kao što su obli rubovi ili 3D prikazi, ravni rub najjasnije prikazuje granicu koju vrijednost postiže čime je osigurana točnost interpretacije
- postepeni pad/rast – vizualno je lakše pratiti od stupčastog dijagrama koji skokovito raste ili pada
- upotreba boja – načelo „manje je više“ vrijedi u potpunosti. Dijagram ne bi trebao obuhvaćati cijeli spektar boja, već samo one stupce koji su od interesa. Upravo bi zato paleta boja na takvim stupcima trebala biti istaknutija od ostalih.
- unutarnja anotacija – ukoliko se ne radi o složenom, naslaganom stupčastom dijagramu preporuča se prikazivanje maksimalne vrijednosti unutar stupca, a ako je problem preglednosti ili prostora korisno je postaviti takve anotacije s prikladnim bojama na *on-hover* opciju.

```
In [1]: import pandas as pd

In [2]: import plotly.graph_objs as go

In [3]: from plotly.offline import iplot

In [4]: df = pd.read_csv(r'\Users\almas\Desktop\Tokyo Medals 2021.csv')

In [5]: winner = df.sort_values(by="Rank By Total", ascending=True) [:16]
```

*Slika 15. Uvoz potrebnih biblioteka, čitanje iz csv-a i pohrana u varijablu winner*

Svaki kod neovisno o grafu/dijagramu započinje uvođenjem potrebnih biblioteka. Prva linija koda prikazuje uvođenje *pandas*-a koji predstavlja biblioteku i open-source alat za analizu i manipulaciju podataka. Kako bi se olakšao njegov poziv postavljen je alias `pd`.[8] Druga linija predstavlja uvođenje modula koji sadrži automatski generirane hijerarhije klasa (npr. `Figure`, `Bar`), instance tih klasa i metode za manipulaciju atributima (npr. `go.Figure()` ). Posljednja linija koja se tiče uvođenja

[8] Spomenuti princip sa aliasom je korišten za sve biblioteke i pakete.

biblioteka, treća, definira korištenje *Plotlya* u *offline* načinu pri čemu ključna riječ `iplot` generira interaktivne grafove direktno iz *pandas dataframe*-a.

U idućoj liniji koda *pandas* modul uz pomoć funkcije `read_csv()` očitava lokaciju i podatke zadane csv datoteke[9].

```
In [6]: winner
```

Out[6]:

| | Country | Gold Medal | Silver Medal | Bronze Medal | Total | Rank By Total |
|---|---|---|---|---|---|---|
| 0 | United States of America | 38 | 39 | 33 | 110 | 1 |
| 1 | People's Republic of China | 38 | 31 | 18 | 87 | 2 |
| 4 | ROC | 20 | 27 | 23 | 70 | 3 |
| 3 | Great Britain | 21 | 21 | 22 | 64 | 4 |
| 2 | Japan | 27 | 14 | 17 | 58 | 5 |
| 5 | Australia | 17 | 7 | 22 | 46 | 6 |
| 8 | Italy | 10 | 10 | 20 | 40 | 7 |
| 7 | Germany | 10 | 11 | 16 | 37 | 8 |
| 6 | Netherlands | 10 | 12 | 14 | 36 | 9 |
| 9 | France | 9 | 12 | 11 | 32 | 10 |
| 10 | Canada | 7 | 6 | 11 | 24 | 11 |
| 11 | New Zealand | 7 | 6 | 7 | 20 | 12 |
| 14 | Republic of Korea | 6 | 4 | 10 | 20 | 12 |
| 42 | Ukraine | 1 | 6 | 12 | 19 | 14 |
| 12 | Brazil | 7 | 4 | 8 | 19 | 14 |
| 13 | Hungary | 6 | 7 | 6 | 19 | 14 |

*Slika 16. Ispis vrijednosti varijable winner*

Varijabla u koju se pohranjuju pročitani podaci nazvana je `dataset`, iako može imati bilo koji drugi jednako deskriptivan naziv. Najčešće su to: *data, df, dataframe* i *dataset*. Dobiveni podaci se uzlazno sortiraju u varijablu `winner` funkcijom `sort_values()` prema rangu[10] po ukupnom broju medalja (`Rank By Total`) pri čemu 16 predstavlja prvih 15[11] država iz tablice.

Linija 6 definira da korisnik očekuje ispis podataka koji je pohranjen u toj varijabli što je vidljivo kao tablica na slici.

[9] Korišteni skup podataka: https://www.kaggle.com/berkayalan/2021-olympics-medals-in-tokyo
[10] Rang nije bio konačan u trenutku izrade dijagrama jer su OI još uvijek bile u tijeku. Prvi stupac s lijeva predstavlja poziciju koju država zauzima na rang listi ovisno o broju zlatnih medalja.
[11] Indeksiranje započinje od nule

Sedma linija najvećim dijelom utječe na konačni izgled, boje i raspored elemenata na vizualizaciji. U *trace1*, *trace2* i *trace3* koji predstavljaju rječnike podatkovnih parametara definirani su:

- `x` – parametar koji „na sebe“ pohranjuje vrijednost varijable *winner* koja očitava vrijednost iz csv datoteke, a ujedno i os na kojoj će se nalaziti kategoričke vrijednosti iz stupca *Country*
- `y` – os na kojoj će se nalaziti brojčane vrijednosti iz stupca *Gold Medal*
- `name` – parametar koji će pripadajući parametar prikazati na tumaču uz boju koja odgovara onoj na dijagramu
- `marker` – za parametre rječnika postavlja boju zadanu u HEX kodu[12]

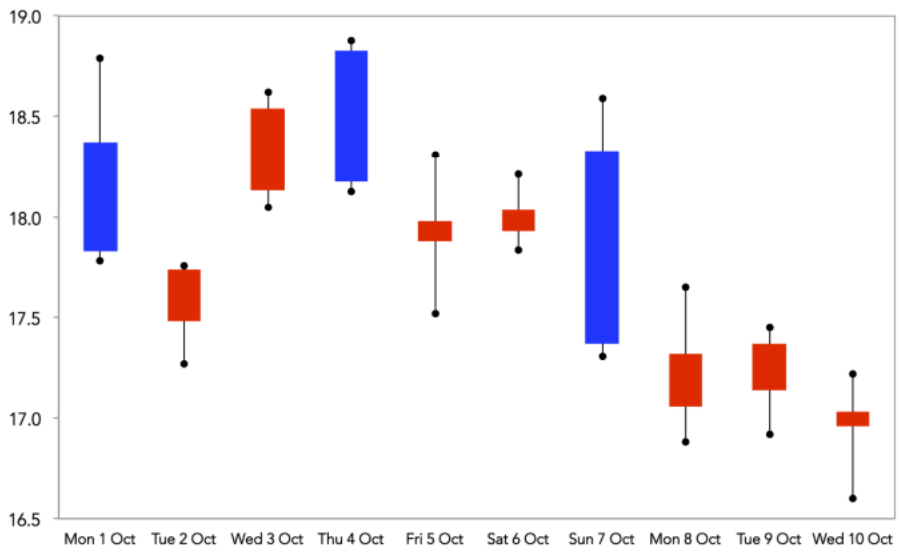
Svi *traces* su zatim pohranjeni u listu *data* gdje redoslijed kojim su upisani u polje determinira slaganje[13] na finalni dijagram (npr. *trace1* naveden je prvi što znači da će zlato biti prvo, a ostale boje, tj. *traces* će se slagati na nju). Iduća linija definira objekt *layout*, a funkciji *go.Layout()* kao parametar predaje *title* – varijablu koja definira naslov dijagrama. Pretposljednje, kompiliraju se *data* i oblik koristeći funkciju *go.Figure()* koja prosljeđuje te parametre varijabli *fig* koja postaje parametar funkcije *iplot* i crta prikaz vidljiv na slici 13.

In [7]:
```
trace1 = go.Bar(
    x=winner['Country'],
    y=winner['Gold Medal'],
    name = 'Zlato',
    marker=dict(color='#FFD700')
)
trace2 = go.Bar(
    x=winner['Country'],
    y=winner['Silver Medal'],
    name='Srebro',
    marker=dict(color='#9EA0A1')
)
trace3 = go.Bar(
    x=winner['Country'],
    y=winner['Bronze Medal'],
    name='Bronca',
    marker=dict(color='#CD7F32')
)
data = [trace3, trace2, trace1]
layout = go.Layout(
    title='Medalje po zemljama 2021. - top 15',barmode='stack'
)
fig = go.Figure(data=data, layout=layout)
iplot(fig)
```

*Slika 17. Rječnici trace1, trace2, trace3*

[12] HEX kod – integrirani HTML dio, tj. način za prikaz boje u RGB (red, green, blue) formatu
[13] *Stacking* – po čemu je i dijagram dobio naziv; *stacked bar chart*

KUTIJASTI DIJAGRAM




*Slika 18. Kutijasti dijagram prikaza burze za XY dionicu, Kirk (2012.)*

Najšira primjena kutijastog dijagrama je u kontekstu financija gdje potpomaže u otkrivanju ključnih statistika o burzovnom tržištu. Obzirom da je za sliku 19 promatrani period na dnevnoj bazi, primjenjuju se *OHLC* (`opening`, `highest`, `lowest`, `closing price`) mjerenja.

Generalno, *Box plot* poznat još kao *box whiskers* dijagram koristi „okvire“ i linije za prikaz distribucije jedne ili više grupa numeričkih podataka u ovisnosti o kategoriji. Granice okvira označavaju raspon centralnih 50% podataka, a središnja linija označava medijan. Linije („brkovi“[14]) koje su vidljive iz svakog dijagrama prikazuju raspon preostalih podataka pri čemu su točke indikatori stršećih vrijednosti.

Za razliku od klasičnog linijskog grafikona jasnije prikazuje distribucije numeričkih podataka, posebice ako se uspoređuju među grupama ili kategorijama (npr. broj epizoda ovisno o žanru kojem serija pripada). Pruža informacije na višoj razini zbog lako uočljivih simetrija podataka, iskrivljenosti, varijanci i već spomenutih stršećih vrijednosti. S druge strane, takvo promatranje uvelike ograničava detaljan pregled jer lako promaknu elementi poput broja „grba“ na grafu.

[14] whiskers, en. = brkovi, hrv.

Konstrukcija ovakvog dijagrama bazira se na kvartilima gdje vrijedi:

- prvi kvartil (Q1) - veći od 25%, a manji od 75% podataka
- drugi kvartil (Q2) – smješten na polovici seta podataka; naziva se još i medijan
- treći kvartil (Q3) – veći od 75%, a manji od 25% podataka
- interkvartilni raspon (IQR) – predstavlja razliku između prvog i trećeg kvartila čime određuje dužinu „brkova“ koji proizlaze iz kutije. U pravilu se svaki „brk“ proteže do najudaljenije točke podataka unutar 1.5x IQR-a, a sve vrijednosti preko toga su stršeće.

Za potrebe ove vizualizacije koristi se nova biblioteka *Seaborn* koja je bazirana na *Matplotu*, ali sa boljim performansom i korisničkim značajkama. Grafovi su vizualno privlačniji zbog većeg izbora paleta, a samim time i interaktivniji. Njezin import preko aliasa `sns` prikazan je u drugoj liniji koda, dok je u prvoj sada već poznati `pandas`. Treća linija postavlja jednu od 5 mogućih tema koju vizualizacija koristi, a `whitegrid` označava bijelu pozadinu dijagrama. Ovakav princip najčešće se koristi za grafove i dijagrame napravljene u *Seabornu* jer je takva boja najvidljivija. Četvrta linija prikazuje definiranje *dataseta* varijablom `df` koja očitava csv datoteku[15] sa desktopa.

```
In [1]: import pandas as pd

In [2]: import seaborn as sns

In [3]: sns.set_theme(style="whitegrid")

In [4]: df = pd.read_csv(r'\Users\almas\Desktop\netflix_titles.csv')
```

*Slika 19. Uvoz potrebnih biblioteka, postavljanje teme i čitanje potrebne csv datoteke*

[15] Korišteni skup podataka: Netflix TV Series Dataset | Kaggle

Prije početka crtanja grafa funkcijom `head()` pregledava se cijeli sadržaj csv datoteke.

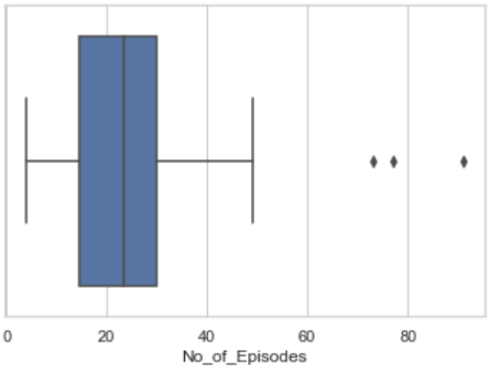
```
df.head()
```

| | Title | Genre | Premiere | No_of_Seasons | No_of_Episodes | Watched |
|---|---|---|---|---|---|---|
| 0 | Stranger Things | Science Fiction Horror | July 15, 2016 | 3 | 25 | True |
| 1 | The Crown | Historical Drama | November 4, 2016 | 4 | 40 | False |
| 2 | Ozark | Crime Drama | July 21, 2017 | 3 | 30 | True |
| 3 | Lost in Space | Science Fiction | April 13, 2018 | 2 | 20 | False |
| 4 | The Umbrella Academy | Superhero Action | February 15, 2019 | 2 | 20 | False |

*Slika 20. Pregled sadržaja csv datoteke*

Novo naspram prošlog grafa je korištenje podataka. *Seaborn* kao biblioteka ne zahtjeva od korisnika deklaraciju objekata, već se svi parametri koje korisnik želi prikazati dodijele funkciji `boxplot()`. Linija 5 prikazuje parametar *x* kojem je korisnik dodijelio vrijednost broja epizoda (`No_of_Episodes`) iz `dataseta df`.

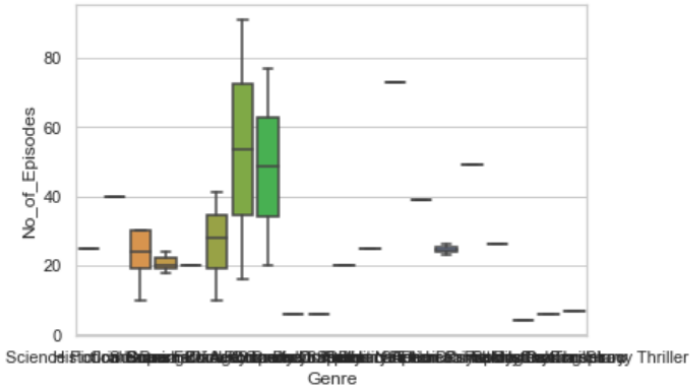
```
In [5]: sns.boxplot(x=df["No_of_Episodes"])

Out[5]: <AxesSubplot:xlabel='No_of_Episodes'>
```




*Slika 21. Prikaz broja epizoda pojedinih serija kutijastim dijagramom*

Najveći problem predstavljaju kategorije na x osi (*Genre*) kojih je previše neovisno postavi li se graf horizontalno ili vertikalno, stoga slika 22 prikazuje neuspjeli pokušaj kutijastog dijagrama.

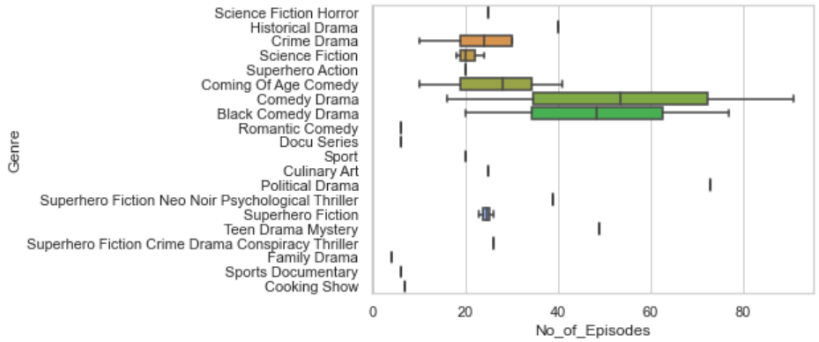
```
In [8]: ## failed
box_plot= sns.boxplot(x="Genre", y="No_of_Episodes", data=df)
```




*Slika 22. Zasićeni kutijasti dijagram*

Nakon uspješnog čišćenja datoteke, zadržane su sve korisniku relevantne kategorije i graf je postavljen horizontalno.

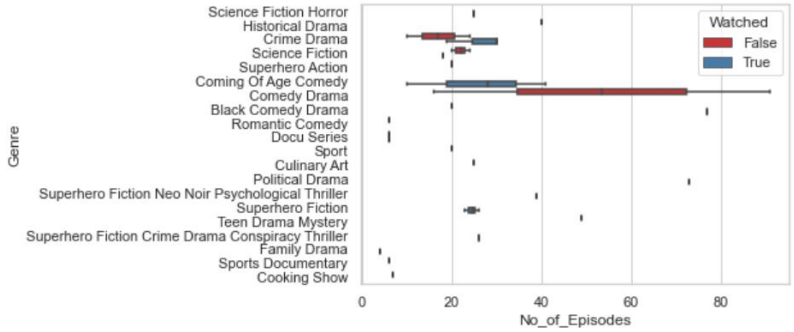
```
In [9]: box_plot= sns.boxplot(x="No_of_Episodes", y="Genre", data=df)
```




*Slika 23. Pročišćeni kutijasti dijagram*

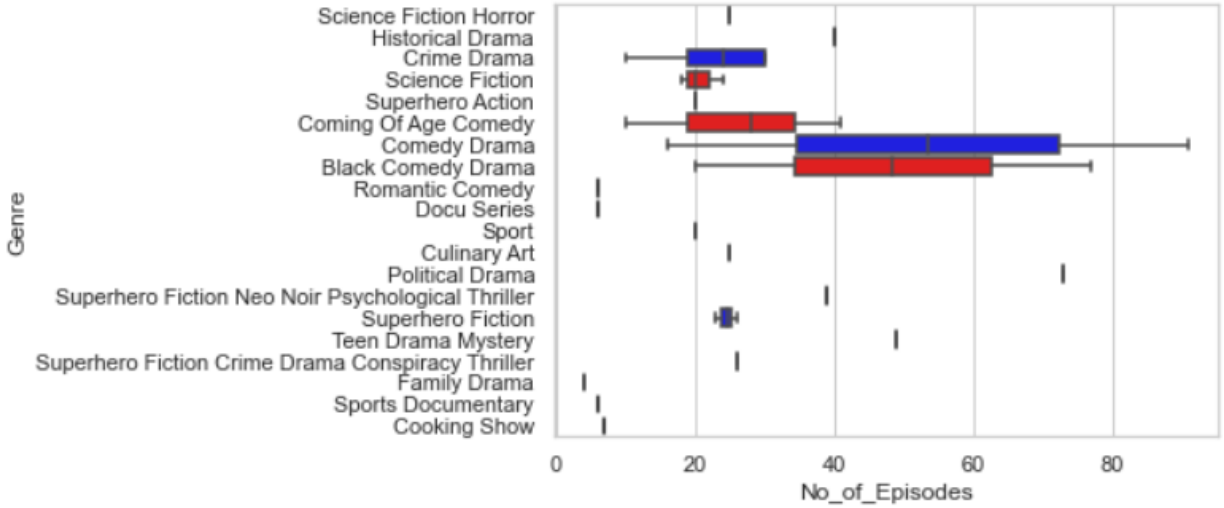
Dodatni parametri u posljednjoj liniji kao što su *hue* – definira je li korisnik pogledao seriju koja pripada spomenutom žanru i *palette* – definira koji set boja će graf koristiti.

```
In [10]: box_plot= sns.boxplot(x="No_of_Episodes", y="Genre", hue="Watched",data=df, palette="Set1")
```




*Slika 24. Finalni kutijasti dijagram sa tumačem*

Kao dodatnu mogućnost, isti graf može biti sortiran po medijanu. Kutija u tom grafu prikazuje interkvartilni raspon, a radi li se o iskrivljenim podacima medijan neće biti na sredini IQR-a. Konkretno na primjeru sa slike 25 brojevi su različitih redova veličina i nisu sortirani te bi ih trebalo logaritmirati.

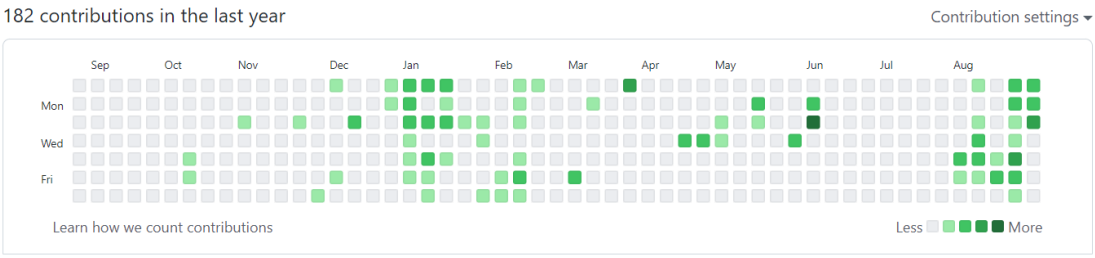



*Slika 25. Sortiranje po medijanu*

Graf sa slike 23 daje zaključiti da korisnika ne zanimaju serije koje imaju manje od 20 epizoda ako se radi o krimić i povijesnim dramama ili više od 30ak epizoda ukoliko je riječ o dramskim komedijama. Saznanja poput ovih pomažu *streaming* servisima[16] kao što su *Netflix* ili *Amazon Prime* da bolje generiraju potencijalne preporuke svojim korisnicima i time ih duže zadrže u sustavu korištenja naplatnih usluga.

[16] Mrežni pružatelj (*online provider*) koji isporučuje multimedijalni sadržaj – film, serija, glazba, itd.

## TOPLINSKA KARTA




*Slika 26. Prikaz doprinosa na Githubu za 2020./2021., autorski profil*

Za Kirka (2012.) toplinska karta (*heat map*) predstavlja grafički prikaz sadržan od dvije varijable; jedna je kategorička (npr. o kojem je alergenu riječ) , a druga kvantitativni omjer (npr. koncentracija alergena na pojedini datum).

Spomenuti grafički prikaz kao najveću prednost ima brz pronalazak podudarnosti uzoraka za otkrivanje redoslijeda i hijerarhije različitih kvantitativnih vrijednosti u matrici kategorijskih kombinacija. Jednostavnije, svaka ćelija predstavlja jednu vrijednost varijable pri čemu ćelija ima predodređenu boju ovisno o rangu u koji varijabla „upada“.

Da bi vizualizacija bila uspješna važno je slijediti pravila:

- upotreba sekvencijalnih paleta – preporuča se korištenje jedne boje i njenih tonova (od svjetlijih za manje, do tamnijih za veće vrijednosti)
- legenda – prikazuje se uz kartu kao vrsta skale na kojoj je prikazan raspon vrijednosti unutar kojeg se podaci kreću (također prati izabranu paletu boja)
- prikaz vrijednosti – mogu odmah biti vidljive ili *on-hover* opcijom

U IT svijetu najpoznatija toplinska karta jest ona sa Githuba; aktivnost korisnika i njegovi doprinosi[17] u posljednjih godinu dana.

[17] *Contributions*

Kako bi se izbjeglo ponavljanje objašnjenja već spomenutih biblioteka prelazi se na novu; *numpy*. Biblioteka je uvedena zbog izvođenja funkcije nad poljem `pollen_concentration`.

```
In [1]: import plotly

In [2]: from plotly.offline import iplot

In [3]: import plotly.graph_objs as go

In [4]: import numpy as np
```

*Slika 27. Uvoz potrebnih biblioteka*

U linijama 5 i 6 prikazana je inicijalizacija polja `allergens` – sadrži podatke o biljkama čija se alergena svojstva promatraju i `date` – datum kada su podaci preuzeti[18].

```
In [5]: allergens = [
            "crkvina",
            "trave",
            "hrast",
            "borovi",
            "cempresi",
            "trputac",
            "pelin",
            "loboda"
        ]

In [6]: date = [
            "August 5th",
            "August 6th",
            "August 7th",
            "August 8th",
            "August 9th",
            "August 10th",
            "August 11th",
            "August 12th",
        ]
```

*Slika 28. Inicijalizacija polja allergens i date*

[18] Promatrani podaci se odnose za Pulu, Istarska županija (5.-12.kolovoz, 2021.)

U liniji 7 u polje *pollen_concentration* funkcijom *np.array()* dodjeljuju se objekti sa vrijednostima svakodnevnih koncentracija.

```
In [7]:
pollen_concentration = np.array(
    [
        [5.8, 3.6, 1.3, 0.5, 0.4, 0.4, 0, 0 ],
        [6.0, 3.3, 1.5, 0.5, 0.5, 0.2, 0, 0],
        [6.1, 2.5, 0.9, 0.9, 0.5, 0.6, 0, 0],
        [6.2, 3.8, 1.6, 1.0, 0.9, 0.4, 0, 0],
        [4.8, 2.1, 1.2, 1.0, 0.7, 0.2, 0, 0],
        [3.9, 2.5, 1.0, 1.0, 0, 0, 0.6, 0.6],
        [3.9, 2.5, 1.0, 1.0, 0, 0, 0.6, 0.4],
        [4.4, 3.0, 0.6, 1.5, 0, 0, 0.4, 0.8]
    ]
)
```

*Slika 29. Funkcija np.array()*

Linija 8 objekt *trace* dodjeljuje vrijednosti iz *defaultne* funkcije *go.Heatmap()* čiji su parametri:

- *x, y, z* – predstavljaju koordinate grafa
- *type* – predstavlja tip grafa
- *colorscale* – definira koja paleta boja je korištena.

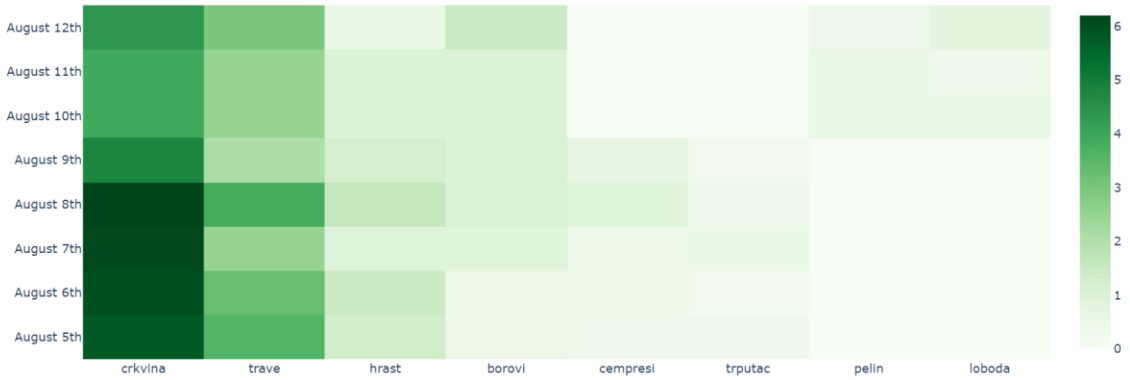
U 9. liniji definira se varijabla *data* kao lista koja sadrži objekt toplinske karte (*trace*) koji je prethodno opisan, a zatim je ta varijabla proslijeđena kao vrijednost *defaultnog* parametra *data.*

Funkcija *go.Figure()* iz linije 10 će u varijabli *fig* pohraniti očekivani grafički prikaz, dok će u sljedećoj pozvati prikaz grafa koji Plotly zatim iscrtava.

```
In [8]:
trace = go.Heatmap(
    x = allergens,
    y = date,
    z = pollen_concentration,
    type = 'heatmap',
    colorscale = 'greens'
)

In [9]:
data = [trace]

In [10]:
fig = go.Figure (data = data)

In [11]:
iplot(fig)
```

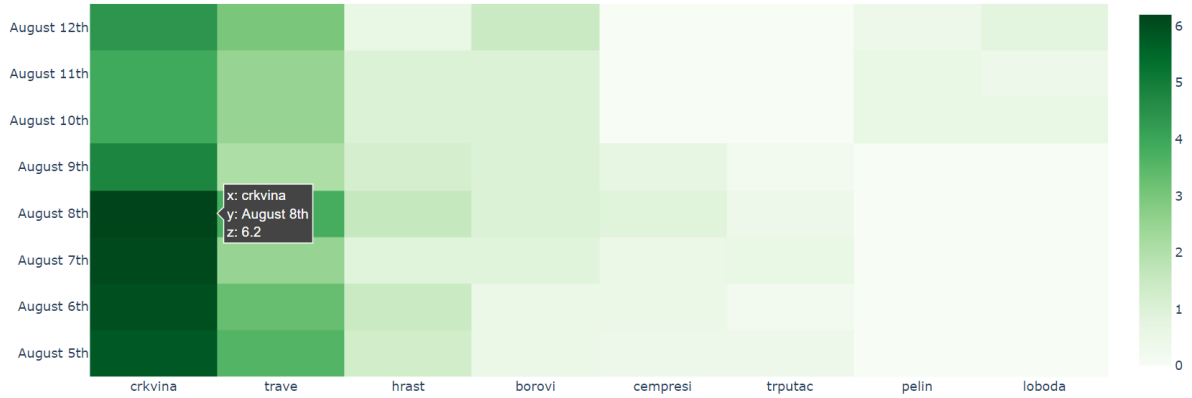
*Slika 30. Objekt trace i iscrtavanje grafa Plotlyem*

Nakon završnog pokretanja koda, grafički prikaz izgleda ovako:




*Slika 31. Toplinska karta - koncentracija peludi u Puli, kolovoz 2021.*

Pređe li se mišem preko ćelije ispisuju se sve koordinate i njihove vrijednosti (ime biljke, datum mjerenja, koncentracija):

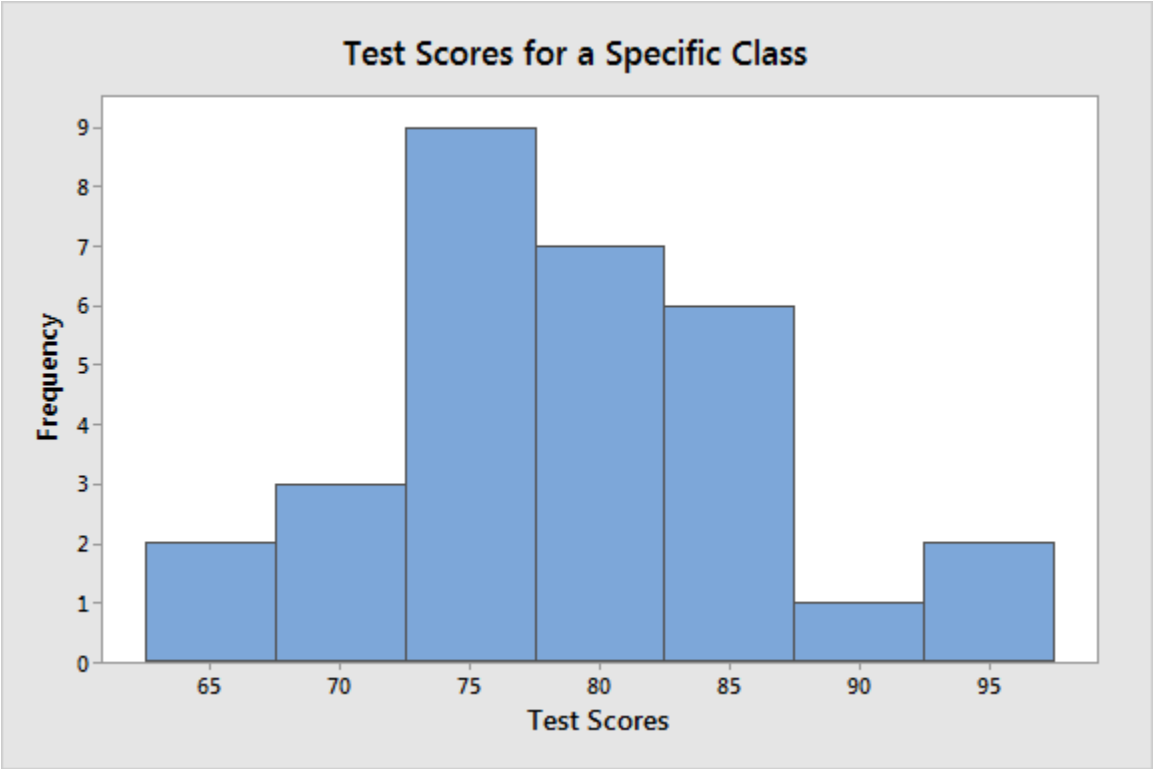



*Slika 32. Toplinska karta s prikazom on-hover opcije*

Dodatni podatak je da se sve koncentracije od 0 do 1.9 smatraju niskim, od 2.0 do 5.9 umjerenim, od 6.0 do 11.9 visokim i sve >= 12 vrlo viskom koncentracijama.[19]

[19] Za pojedine dane ne postoje podaci o mjerenju ili nema peludi pa su tim biljkama koncentracije peludi postavljene na 0 (izvor: https://www.plivazdravlje.hr/alergije/prognoza/20/Pula.html )

HISTOGRAM




*Slika 33. Rezultati ispita prikazani pomoću histograma*

Najčešći primjer ovog grafa ujedno vidljiv i na slici 33 je prikaz rezultata ispita[20] pri čemu apscisa predstavlja postotak riješenosti, a ordinata frekvenciju studenata koji ostvaruju spomenuti rezultat.

„Grafikon koji prikazuje raspodjelu vrijednosti numeričke varijable kao niz stupaca. Svaki stupac pokriva niz numeričkih vrijednosti koji se nazivaju klasa, a visina stupca označava učestalost podataka u njoj.“ riječi su kojima Yi[21] (2019.) definira histogram.

Općenito, histogrami su korisni za prikaz općih distribucijskih značajki skupa podataka. Daju uvid u tip raspodjele – iskrivljena ili simetrična, njene vrhove i odstupanja, a za njegovo korištenje potrebna je samo varijabla s kontinuiranim numeričkim vrijednostima.

[20] Izvor: https://statisticsbyjim.com/basics/descriptive-inferential-statistics/
[21] Mike Yi, autor članka na tutorial portalu chartio

Kao i svaki dijagram do sad, histogram također ima pravila:

- nulta vrijednost – najbitnija značajka svakog histograma. Graf koji se prikazuje mora započeti s vrijednosti 0 budući da je frekvencija podataka u svakoj klasi prikazana visinom stupca. U protivnom, distribucija podataka i percepcija će biti iskrivljene.
- širina klase – procjenjuje se količinom podataka. Ukoliko je dostupnih podataka premalo treba imati na umu da bi histogramu mogli nedostajati detalji potrebni za očitavanje korisnosti. S druge strane, zasićenje podacima može rezultirati grubom distribucijom iz koje je teško shvatljiva poruka grafa. Savjet je testiranje i odvajanje vremena na probne grafove kako bi se izabrao što vjerodostojniji grafikon.
- grafičke oznake – ako je riječ o manje podataka oznaka se može postaviti na svaku vrijednost (npr. 0, 1, 2, 3..itd.), dok je za rad s većim podacima preporuka koristiti skalu djeljivu sa 5 (npr. prethodno spomenuti prikaz rezultata ispita), 10 ili 20.
- multimodalnost – nije isključivo da histogram može imati samo jednu maksimalnu lokalnu frekvenciju
- ne miješati sa stupčastim dijagramom – vizualno najuočljivija razlika je u razmacima između stupaca koji kod histograma nisu vidljivi jer su stupci međusobno povezani. Nadalje, za razliku od stupčastog dijagrama koji koristi diskretne numeričke ili kategoričke podatke, histogram koristi nediskretne kvantitativne podatke. Grafički gledano to znači da se stupci histograma pružaju iznad kvantitativnih varijabli - brojevni rang, dok u stupčastom dijagramu to vrijedi za kvalitativne varijable, tj. kategorije.
- *binning*, tj. odabir granica „kanti“ – što je širi raspon kante, to je manje stupaca na histogramu. Širine kanti bi trebale biti jednake i koristiti samo cijele vrijednosti poput 1,2,5,10,20,25,50,100 itd. kako bi interpretacija grafa bila lakša. Ukoliko su kante preuske ili preširoke graf bi mogao sakriti potencijalno važne detalje. Za histograme poput ovog na slici 35 idealno je korištenje širih kanti kako bi se eliminirala „buka“.

MATEMATIČKI PRIKAZ BINNING-a

Prije samog koda važno je prikazati kako se matematički bira koliko je optimalno „kanti“ potrebno za histogram[22]. Izvedena formula dobivena je iz binomne distribucije po kojoj slijedi izračun da je:

$$k = \lceil log_2 n \rceil + 1$$

Pri čemu vrijedi da je:

- n – ukupan broj opservacija u skupu podataka
- k – broj razreda koji se najčešće kreće između 5 i 15

Generalno gledano, histogram je najlakši za prikazati pomoću *matplotlib* biblioteke i dovoljno je svega nekoliko linija koda. Kako bi fokus ostao na samom kodu potrebnom za graf, preskočen je dio gdje se objašnjava dio uvođenja biblioteka i postavljanja varijable koja čita iz csv-a, ali je prikazano što csv datoteka sve sadrži.

```
df.head()
```

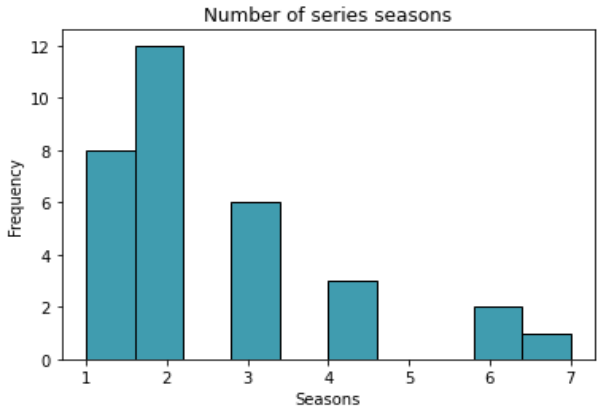
| | Title | Genre | Premiere | No_of_Seasons | No_of_Episodes | Watched |
|---|---|---|---|---|---|---|
| 0 | Stranger Things | Science Fiction Horror | July 15, 2016 | 3 | 25 | True |
| 1 | The Crown | Historical Drama | November 4, 2016 | 4 | 40 | False |
| 2 | Ozark | Crime Drama | July 21, 2017 | 3 | 30 | True |
| 3 | Lost in Space | Science Fiction | April 13, 2018 | 2 | 20 | False |
| 4 | The Umbrella Academy | Superhero Action | February 15, 2019 | 2 | 20 | False |

*Slika 34. Prikaz podataka iz csv datoteke*

[22] Detaljnije objašnjenje: https://www.statology.org/sturges-rule/

Funkcija *plt.hist()* kao parametre uzima: vrijednost koja se prikazuje na x osi (*Seasons*, tj. broj sezona koje ima serija) jer je y-os namijenjena za frekvencije, *edgecolor* – boju obruba stupca i *color* – boju stupca. Za preostale funkcije koje se tiču oznaka i naslova nije potrebno podrobnije objašnjenje.

```
In [4]: plt.hist(df['No_of_Seasons'], edgecolor='black', color='#409caf')
        plt.title('Number of series seasons')
        plt.xlabel('Seasons')
        plt.ylabel('Frequency')

Out[4]: Text(0, 0.5, 'Frequency')
```

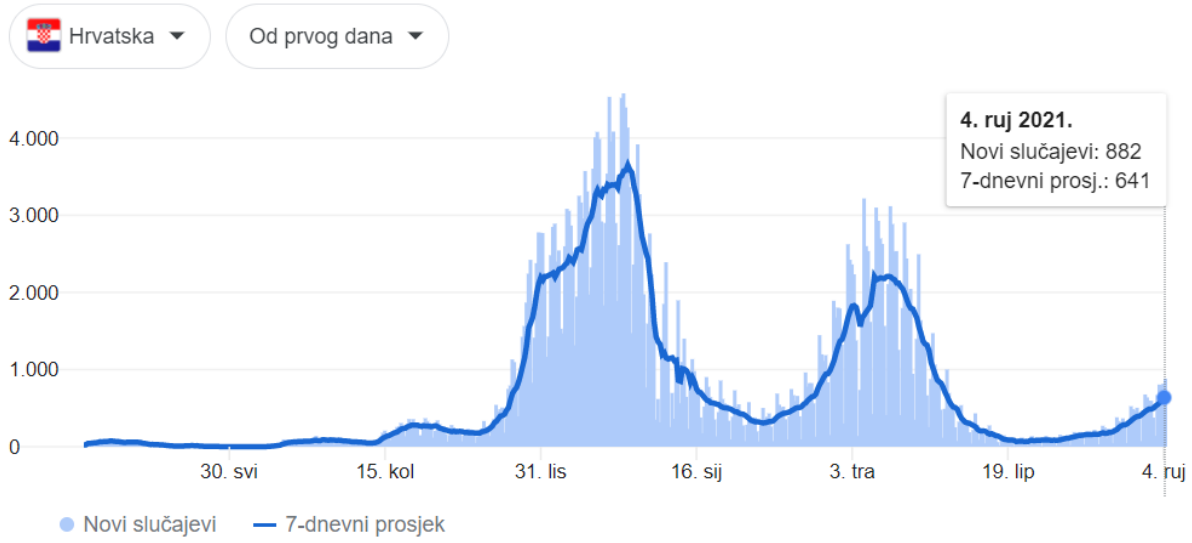



*Slika 35. Finalni histogram*

Prikazani histogram daje uvid o omjeru broja sezona i njihovoj frekvenciji iz kojeg je jasno da prevladavaju serije koje imaju po jednu ili dvije sezone.[23]

[23] Izvor (modificiranog broja podataka): https://www.kaggle.com/harshitshankhdhar/netflix-and-amazon-prime-tv-series-dataset

LINIJSKI DIJAGRAM




*Slika 36. Novi slučajevi COVID-19 u Hrvatskoj, izvor: JHU CSSE COVID-19 Data*

Linijski dijagram je uvjerljivo najpoznatiji graf u posljednje dvije godine obzirom na aktualnosti i pandemiju COVID-19 koja je pogodila cijeli svijet. Na slici 36 vidljiv je linijski dijagram koji prikazuje broj novozaraženih od 19.ožujka 2020. do 4.rujna 2021.

Teorija koja stoji iza ovakvog grafa definira ga kao točke povezane linijama s lijeva na desno kako bi pokazale promjene u vrijednosti. Vodoravna os prikazuje kontinuirani napredak, najčešće vremenski, dok okomita os daje informaciju o vrijednostima za mjerni podatak koji je od interesa. Odluku oko izbora pravilne veličine intervala donosi analitičar pri čemu je važno imati na umu učestalost generiranja relevantnih podataka. Primjerice, graf sa slike 37 ažurira se na dnevnoj bazi jer je nepotrebno takve podatke ažurirati iz minute u minutu ili iz sata u sat.

Glavna prednost ovog grafa nad primjerice histogramom koji također na svojoj y osi prikazuje frekvencije jest dobar način usporedbe. Više histograma na istom skupu osi bilo bi preteško za postaviti, a još teže iščitati stoga se linijski grafikoni nerijetko nazivaju i frekvencijskim poligonima.

Dodatno, kako bi se izbjegli potencijalni problemi oko iščitavanja grafa slijedi par savjeta:

- odgovarajući interval mjerenja – ukoliko se radi o vremenskim podacima preširok interval može utjecati na predugo shvaćanje u kojem smjeru ide linija trenda, dok prekratak može navesti na krive zaključke – „otkriven je samo šum, ali ne i signal“
- količina redaka – u principu se koristi pet ili manje redaka uz uvjet da su linije dovoljno dobro razdvojene. Za slučaj da je ipak potrebno više redaka savjetuje se izrada mreže manjih linijskih grafikona koji će biti razvrstani prema važnim karakteristikama, npr. prosjek
- pozicija nule – iako je osnova za stupčaste grafikone i histograme, linijski dijagram ne mora uključivati nultu osnovicu. Njegov glavni cilj je naglasiti promjene vrijednosti, a ne veličinu samih vrijednosti pa se u slučajevima gdje je nula besmislena preporuča zumirati raspon okomite osi.
- interpolacija krivulje – u pokušaju „glađeg“ povezivanja svih točaka često se viđa krivulja koja prolazi kroz sve točke odjednom. Takav oblik će samo narušiti percepciju trendova u podacima jer su upravo smjer i strmina linije trebali biti pokazatelji promjene vrijednosti.

Za početak, prikupljeni podaci su kombinacija modificirane csv datoteke i izvora sa stranice WHO[24]-a, za dane koji nisu dostupni u csv-u; *croatia_stats_covid19*. Funkcijom *head()* provjereno je stanje za prvih 5 mjeseci 2020.

```
df.head()
```

| | month | cumulative_total_cases | cumulative_total_deaths | year |
|---|---|---|---|---|
| 0 | January | 0 | 0 | 2020 |
| 1 | February | 6 | 0 | 2020 |
| 2 | March | 867 | 6 | 2020 |
| 3 | April | 2076 | 67 | 2020 |
| 4 | May | 2246 | 103 | 2020 |

*Slika 37. Prvih 5 mjeseci COVID-19 slučajeva, Hrvatska, 2020.*

Na slici 39 prikazan je eksponencijalni graf koji predstavlja porast broja novozaraženih u 2020. – definirano kroz *query()* parametar. Sljedećom linijom definira se kako će na apscisi biti prikazani

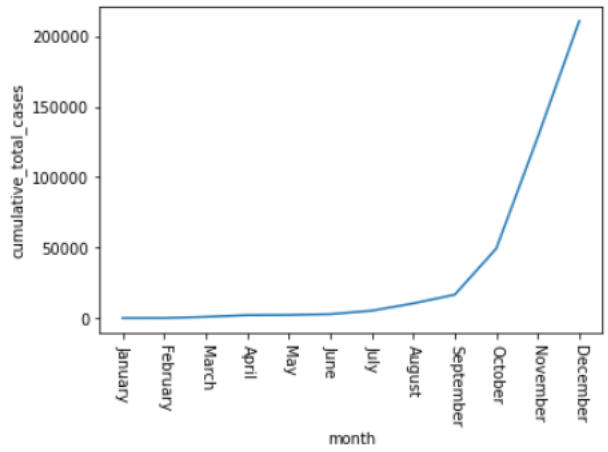
[24] World Health Organization, zadnji dan pregleda 5.rujna 2021. (izvor: https://covid19.who.int/region/euro/country/hr )

mjeseci, a na ordinati brojevi slučajeva u tisućama. Kako bi graf kasnije bio smisleniji varijabli *January* dodijeljena je vrijednost 0,0 iako je u stvarnosti prvi slučaj, odnosno prvih 6 zabilježeno u veljači iste godine.

Linija u kojoj se spominju *xticks* postavljena je sa svrhom okretanja riječi na x osi za 270° kako bi se riječi razdvojile i bile vidljivije.

```
plt.xticks(rotation = 270)
last_year = df.query("year==2020")
sns.lineplot(data=last_year, x="month", y="cumulative_total_cases")
```

```
]: <AxesSubplot:xlabel='month', ylabel='cumulative_total_cases'>
```

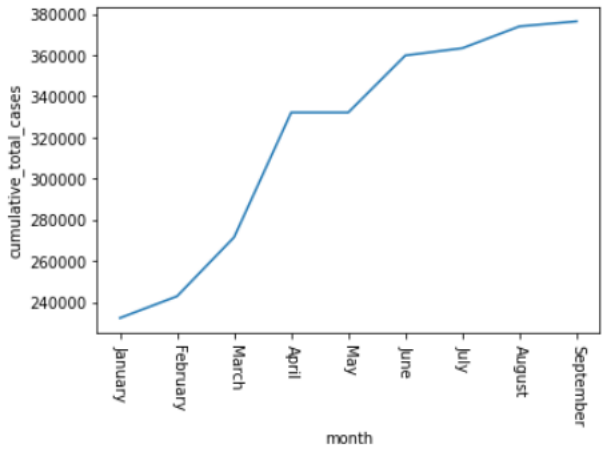



*Slika 38. Prikaz broja novozaraženih COVIDom-19 , Hrvatska, 2020.*

Na slici 40 prikazan je isti scenarij, ali je razlika u parametru koji prima `query()` te je on sada postavljen na trenutnu godinu: 2021. Odmah se da primijetiti da graf više nije u potpunosti eksponencijalnog rasta već postoje i podaci o padu broja novozaraženih.

```
plt.xticks(rotation = 270)
last_year = df.query("year==2021")
sns.lineplot(data=last_year, x="month", y="cumulative_total_cases")
```

```
<AxesSubplot:xlabel='month', ylabel='cumulative_total_cases'>
```

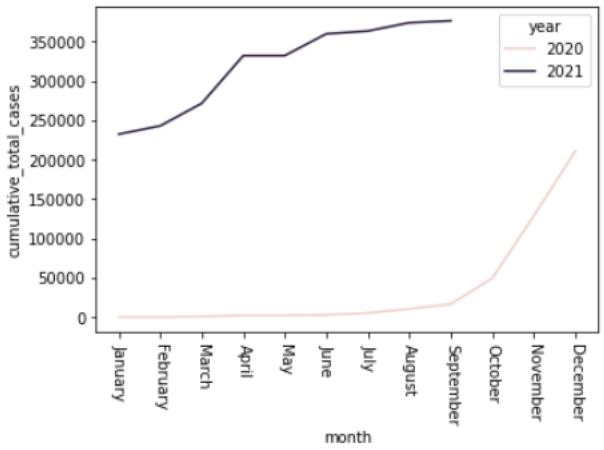



*Slika 39. Prikaz broja novozaraženih COVIDom-19, Hrvatska, 2021.*

Posljednje dodano, na slici 41 prikazani su usporedni grafovi o ukupnom broju novozaraženih i umrlih kroz obje godine. Ovdje dolazi do izražaja važnost varijable `January` jer iako virus nije bio prisutan u 2020. ove godine jest pa se te vrijednosti moraju prikazati. Ono što možda ovdje zbunjuje je kako graf za 2021. ne kreće od nule, a razlog tome je što se broj novozaraženih konstantno prebacuje iz mjeseca u mjesec. Dakle, nakon prosinca 2020. podaci od 210837 novozaraženih se nisu nulirali već su nastavili dalje i došli do konačnih 232426 na kraju siječnja 2021.

```
plt.xticks(rotation = 270)
sns.lineplot(data=comparison, x="month", y="cumulative_total_cases", hue="year")
```

```
<AxesSubplot:xlabel='month', ylabel='cumulative_total_cases'>
```

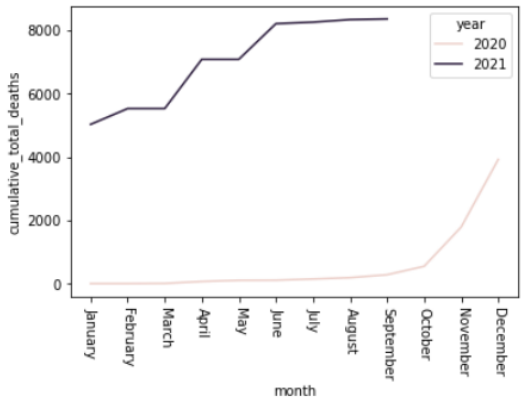



*Slika 40. Usporedni grafovi novozaraženih COVIDom-19 u Hrvatskoj za period 2020.-2021.*

Posljednji graf ove vrste na slici 42 prikazuje usporedni graf ukupnog broja preminulih kroz obje godine.

```
plt.xticks(rotation = 270)
sns.lineplot(data=comparison, x="month", y="cumulative_total_deaths", hue="year")
```

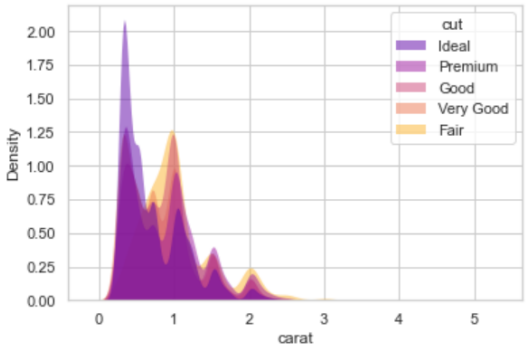
```
]: <AxesSubplot:xlabel='month', ylabel='cumulative_total_deaths'>
```




*Slika 41. Usporedni grafovi preminulih od COVIDa-19 u Hrvatskoj za period 2020.-2021.*

## KDE GRAF – GRAF PROCJENE FUNKCIJE GUSTOĆE DISTRIBUCIJE

```
In [5]: sns.kdeplot(
            data=df, x="carat",hue="cut",
            fill=True, common_norm=False, palette="plasma",
            alpha=.5, linewidth=0,
        )

Out[5]: <AxesSubplot:xlabel='carat', ylabel='Density'>
```




*Slika 42. KDE dijagram omjera karata u dijamantu i njegove gustoće*

KDE (*Kernel Density Estimation*) dijagram na slici 43 prikazuje funkciju gustoće vjerojatnosti kontinuiranih varijabli i omogućava iscrtavanje više varijabli u cjelini.

Za kreiranje spomenutog dijagrama najlakše je voditi se odgovorima na sljedeća pitanja:

1. Koji raspon pokriva promatrač?
   Ovisi o količini podataka koji su dostupni u *datasetu* ili preferencijama promatrača (izbor podataka kojima će se dati prioritet u crtanju).
2. Koje su središnje mjere tendencije podataka?
   Najvažnije su vrhunac i iskrivljenost grafa jer pokazuju najveće koncentracije podataka, odnosno tendenciju.
3. Jesu li podaci iskrivljeni u jednom smjeru?
   Za korišteni *diamonds.csv* distribucija je izrazito iskrivljena na više mjesta. Ukoliko se korisnik želi riješiti toga dovoljan je poziv funkcije *log()*[25] nad željenom kolumnom.

[25] U sklopu biblioteke *Numpy*

4. Bimodalnost/multimodalnost podataka?

   Silvermanovim testom provjerava se broj modova u spomenutom grafu. Kako je najčešće nemoguće izravno koristiti procjenu gustoće za identifikaciju odstupanja jer procijenjena distribucija većinom je multimodalna. Dokaz je u čestim promjenama distribucije iz konveksne u konkavnu.

5. Jesu li stršeće vrijednosti značajne?

   Kako bi se otkrile stršeće vrijednosti koristi se usporedba procijenjene gustoće u danim točkama podataka sa prosječnom gustoćom njenih susjeda. Takva vrijednost najčešće odstupa od opažanja dovoljno da izazove sumnju da je generirana drugačijim mehanizmom od onog kojim je tretiran uzorak. Slijedom toga, svi uzorci za koje se tako čini mogu se smatrati potencijalnim odstupanjima do trenutka dok funkcija evaluacije ne pokaže drugačije.

Prije crtanja grafa funkcijom `head()` provjerava se prvih pet podataka koji se nalaze u csv datoteci.

```
df.head()
```

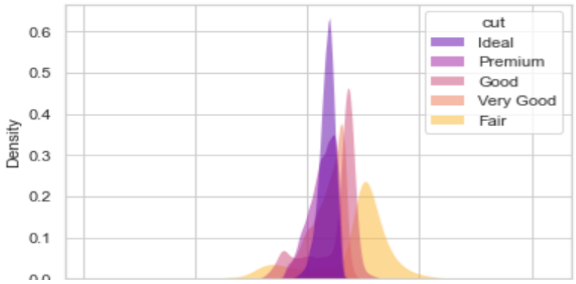
| | Unnamed: 0 | carat | cut | color | clarity | depth | table | price | x | y | z |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 0.23 | Ideal | E | SI2 | 61.5 | 55.0 | 326 | 3.95 | 3.98 | 2.43 |
| **1** | 2 | 0.21 | Premium | E | SI1 | 59.8 | 61.0 | 326 | 3.89 | 3.84 | 2.31 |
| **2** | 3 | 0.23 | Good | E | VS1 | 56.9 | 65.0 | 327 | 4.05 | 4.07 | 2.31 |
| **3** | 4 | 0.29 | Premium | I | VS2 | 62.4 | 58.0 | 334 | 4.20 | 4.23 | 2.63 |
| **4** | 5 | 0.31 | Good | J | SI2 | 63.3 | 58.0 | 335 | 4.34 | 4.35 | 2.75 |

*Slika 43. Podaci vezani uz csv datoteku o dijamantima*

Funkcija *seaborn.kdeplot()* predstavlja raspodjelu vjerojatnosti vrijednosti podataka kao područje ispod iscrtane krivulje. Jedini novi parametri su: *fill* – postavljen na *True* označava da će se područje ispod krivulje ispuniti i *common_norm* – postavljen na *False* znači da će svaka distribucija provesti normalizaciju neovisno jedna o drugoj.

26

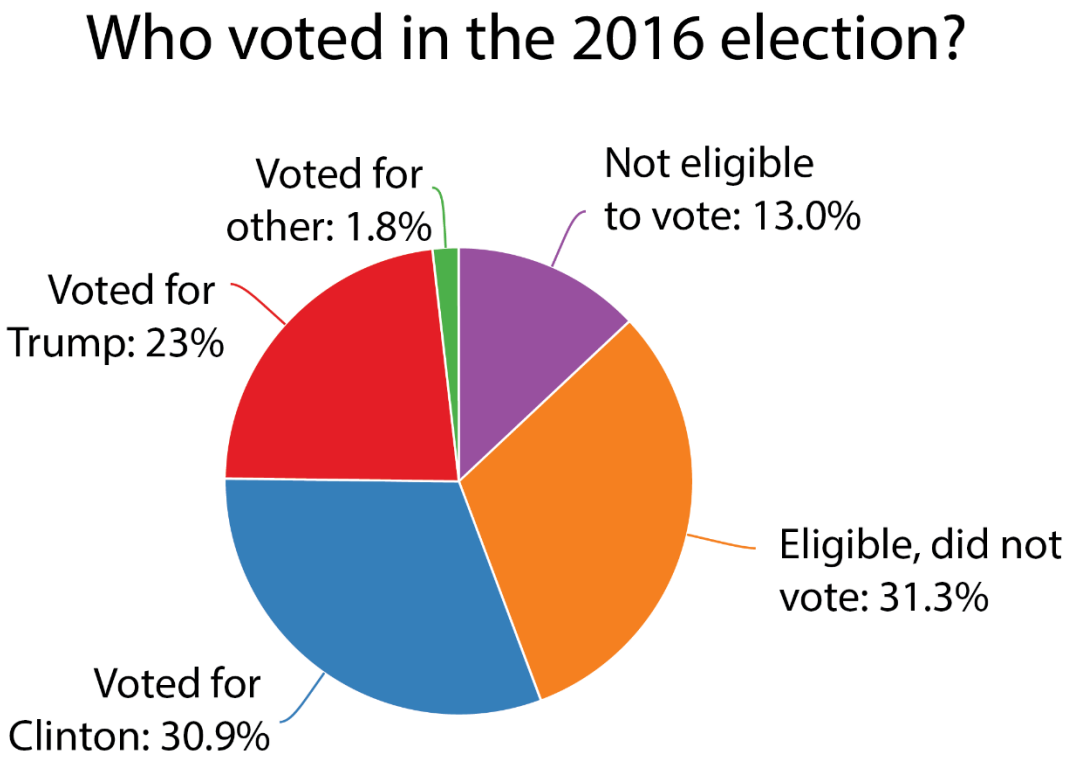
```
In [6]: sns.kdeplot(
            data=df, x="depth", hue="cut",
            fill=True, common_norm=False, palette="plasma",
            alpha=.5, linewidth=0, ##moze i alpha=0.5
        )

Out[6]: <AxesSubplot:xlabel='depth', ylabel='Density'>
```




*Slika 44. KDE dijagram omjera dubine dijamanta i gustoće*

[26] Izvor: Diamonds | Kaggle

PITA DIJAGRAM




*Slika 45. Pita dijagram nakon izbora u SAD-u, 2016.*

Dijagrami su najčešće viđeni nakon izbora jer bi trebali jasno prikazati podijeljenost prema stranci ili kandidatu. Neovisno o znanju osobe koja interpretira graf, pita dijagram slovi za jedan od najjednostavnijih grafova za interpretaciju. Slika 46 prikazuje pita dijagram za rezultate izbora[27] u Americi 2016.

*Pie chart*, poznat kao kružni ili tortni dijagram, prikazuje kako je ukupna količina podataka podijeljena između razina kategorijske varijable uz uvjet da je krug podijeljen na kriške. Svaka kategorička vrijednost odgovara jednom presjeku kruga, a veličina kriške označava udio cjeline koju zauzima svaka razina kategorije.

[27] Izvor: https://www.nj.com/news/2017/02/the_real_winner_of_the_presidential_election_in_nj_apathy.html

Pita dijagram, zbog svoje specifičnosti sa podacima, ima vrlo usku primjenu. Prema Kirku (2012.) jedna varijabla je kategorija, a druga označava frekvenciju te kategorije pri čemu je važno naglasiti da se koristi samo za cijeli iznos koji je najčešće jednak 100. Primarni cilj takvog dijagrama je uspoređivanje doprinosa svake grupe u sklopu cjeline umjesto međusobnog uspoređivanja.

Kako bi dijagram bio jasniji spominju se određeni trikovi i savjeti:

- princip proporcionalne tinte – zasjenjeno područje grafa koristi se za predstavljanje numeričke vrijednosti, a količina tinte koja ga prikazuje trebala bi biti proporcionalna samoj vrijednosti
- uključiti napomene – osobama koje interpretiraju graf teško je razaznati točne omjere iz grafikona osim ako se ne radi o polovinama, trećinama ili četvrtinama. Korisno je napisati postotke da bi se izbjeglo procjenjivanje vrijednosti.
- redoslijed kriški – tipično se prati uzorak da vrijednosti idu od veće prema manjoj ukoliko se radi o kategorijama sa sličnim vrijednostima. Što se tiče kutova kretnje većina alata kreće od desne strane, odnosno od vrha u smjeru kazaljke na satu što je jednako intuiciji čitanja; od vrha prema dnu.
- ograničen broj kriški – kružni dijagrami s puno kriški mogu biti teški za čitanje, a samim time i interpretaciju. Općenita preporuka je korištenje do 5 kriški, a ostatak - gomila manjih kriški se spaja u jednu, veću koja će biti neutralna i predstavljati kategoriju „Ostalo“.
- 2D prikaz – nepotrebno je i estetski neprihvatljivo postavljati ovakvu vrstu dijagrama u trodimenzionalan prikaz jer praznine mogu utjecati na otežano mjerenje podataka.
- prilagodba kompatibilnim podacima – ukoliko se koriste podaci koji ne predstavljaju usporedbu dijelova u cjelini može doći do velike zabune kod rada s postotcima (npr. zbrojeni omjeri mogu doći preko 100%).

Kako bi se pojednostavio prikaz, za potrebe kružnog dijagrama podaci su preuzeti sa deklaracije namirnica koje nas okružuju, umjesto online dostupnih *datasetova*. U

prvom primjeru prikazan je sastav Cedevite[28], dok drugi prikazuje odnos nutritivnih vrijednosti u *Grandino Triple Chocolate* keksima.[29]


*Slika 46. Cedevita limun, 250g*


*Slika 47. Grandino Triple Chocolate keksi*

Dijagram vezan za Cedevitu namjerno ne poštuje treći savjet, vezan za ograničen broj kriški, kako bi se ukazalo na nepreglednost grafa. Dodatno, u ovom prikazu se u prvoj liniji uvodi nova biblioteka `Plotly` – specifična po interaktivnosti koju pruža na svojim grafovima.

U iduće dvije linije deklarirana su polja: `nutrients` – sadrži sve sastoje koji se mogu pronaći u Cedeviti i `nutrition_value` – sadrži količinu promatranog sastojka u odnosu na 100g smjese.

```
In [1]: import plotly
        from plotly.offline import iplot
        import plotly.graph_objs as go
```

*Slika 48. Uvoz plotly-a i njegovih dodatnih biblioteka*

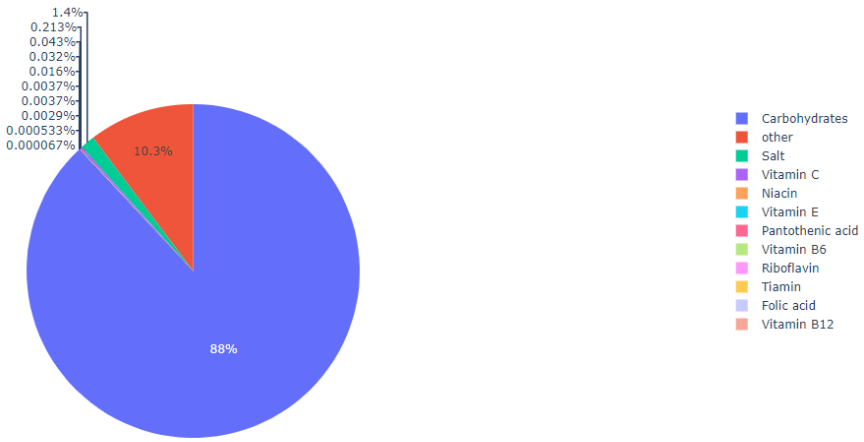
[28] Okus limuna, pakiranje 250g (izvor: vlastito očitanje sa nutritivnog sastava proizvodnog pakiranja)
[29] Lidl, pakiranje 200g (izvor: vlastito očitanje sa nutritivnog sastava proizvodnog pakiranja)

Ostatak koda podudara se sa korištenim u stupčastom dijagramu uz iznimku imena funkcije; ovdje se radi o `go.Pie()`, a ne o `go.Bar()`.

```
In [2]: nutrients = ['Carbohydrates','Salt','Vitamin C','Niacin', 'Vitamin E',
                     'Pantothenic acid','Vitamin B6','Riboflavin','Tiamin','Folic acid','Vitamin B12','other']

In [3]: nutrition_value = [88,1.4,0.213,0.043,0.032,0.016,0.0037,0.0037,0.0029,0.000533,0.000067,10.259]

In [4]: trace = go.Pie(labels = nutrients, values = nutrition_value)

In [5]: data = [trace]

In [6]: fig = go.Figure( data = data)

In [7]: iplot(fig)
```

*Slika 49. Deklaracija polja i funkcije potrebne crtanje dijagrama*

Kako su manji postotci navedeni po sastojcima[30] umjesto da su odvojeni kao „Ostalo“ grafički prikaz nažalost nije uspio prikazati sve boje, već su one vidljive samo u tumaču s desne strane.



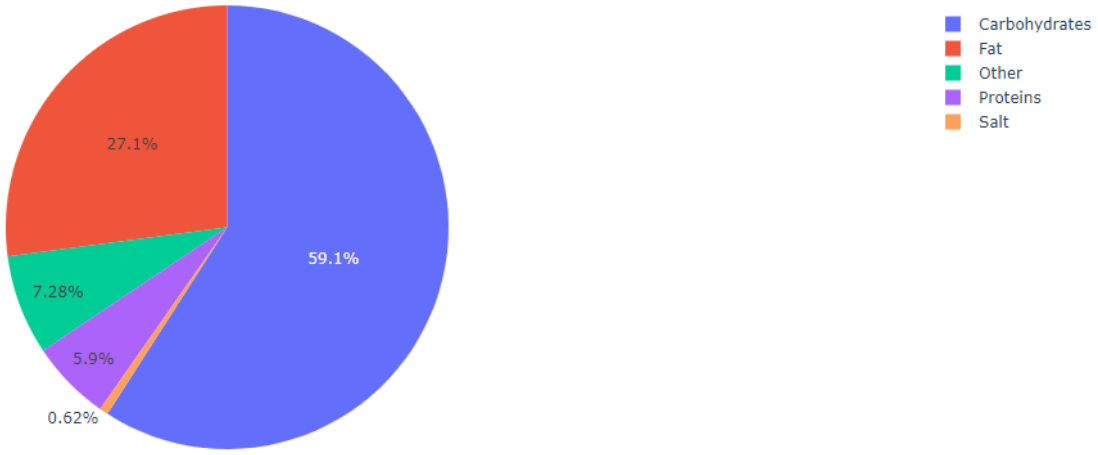

*Slika 50. Pita dijagram za Cedevitu limun, 200 grama*

[30] Nisu navedeni podaci o mastima i bjelančevinama jer su im vrijednosti jednake nuli pa se grafički niti ne bi prikazale.

U drugom prikazu dostupno je manje sastojaka što je drastično utjecalo na vidljivost postotaka, a samim time i dijagrama. Kod je identičan prethodnom uz sitnije izmjene, ali rezultat je puno lakši za interpretaciju.

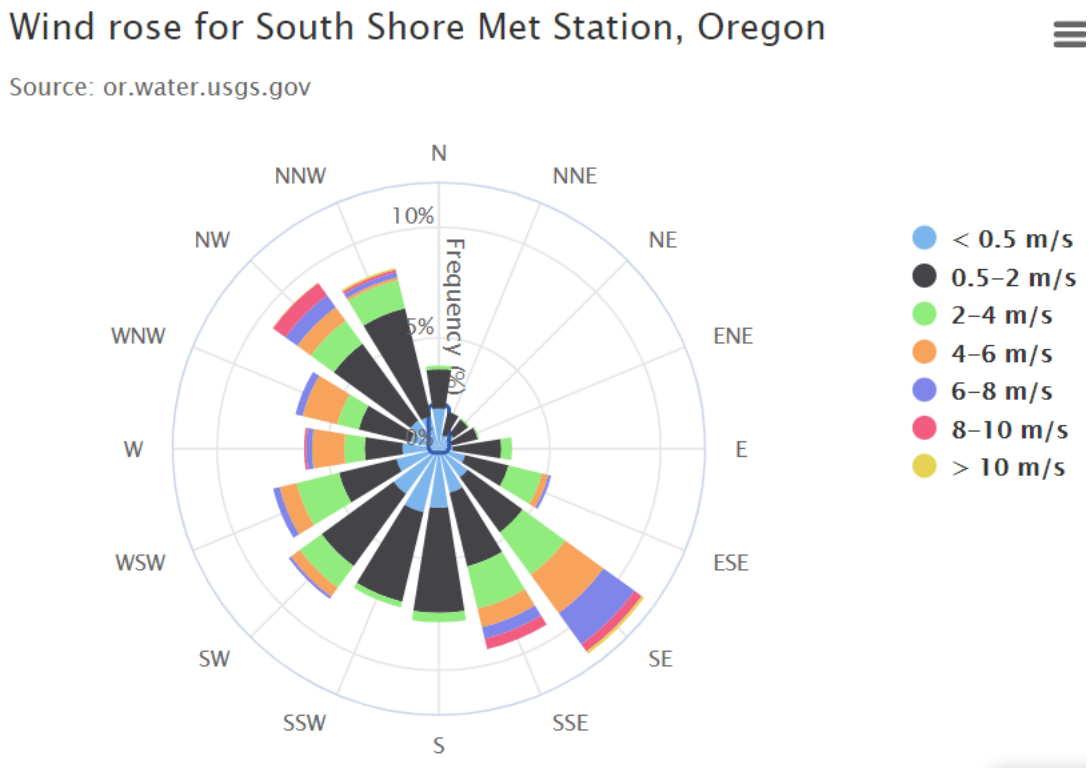
```
In [8]: cookie_nutrients = ['Fat', 'Carbohydrates','Proteins','Salt','Other']

In [9]: cookie_nutr_value = [27.1, 59.1, 5.9, 0.62, 7.28]

In [10]: trace = go.Pie(labels = cookie_nutrients, values = cookie_nutr_value)

In [11]: data = [trace]

In [12]: fig = go.Figure(data = data)

In [13]: iplot(fig)
```

*Slika 51. Kod potreban za crtanje kružnog dijagrama - Grandino keksi*




*Slika 52. Pita dijagram - Grandino Triple Choco keksi*

POLARNI DIJAGRAM / PAUKOV DIJAGRAM

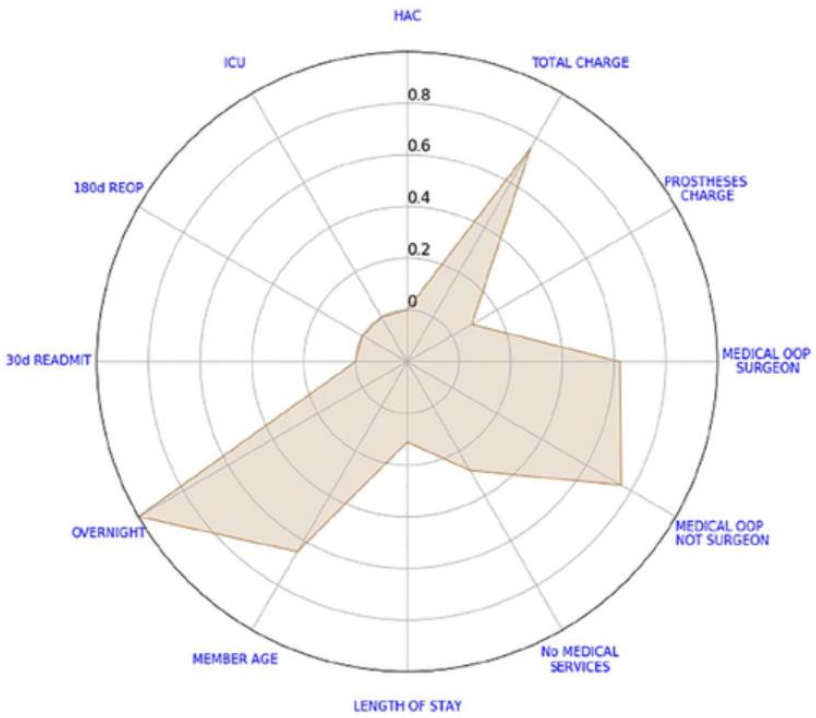



*Slika 53. Naslagani polarni dijagram, podaci o jačini vjetra u Oregonu*

Primjer polarnog grafikona je ruža vjetrova[31] koja prikazuje smjer i jačinu puhanja vjetra na nekom mjestu vidljivom na slici 54.

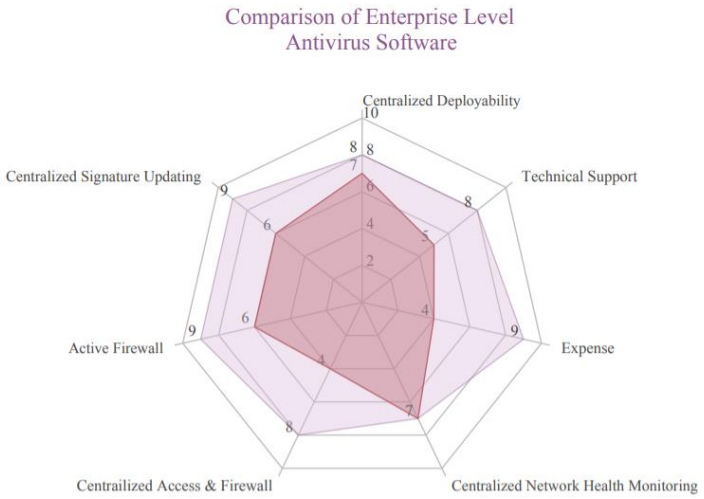
Korisnost polarnog/paukovog dijagrama sveprisutna je pa tako primjer sa slike 55 pokazuje njegovu primjenu u medicini. Konkretnije, radi se o izvještaju mjerenja performansi prijema pacijenta u bolnicu radi ortopedskog kirurškog zahvata. Iako sadrži podatke za 10 operacija, vizualizacija je sažeta i jasno prikazuje širenje podataka što omogućuje jednostavno razvrstavanje „usluga“ prema visini performansa. U konačnici, dobiven je učinkovit usporedni alat za razlikovanje ishoda u odnosu na vrijednosti prijema pacijenta[32].

[31] Izvor: https://www.highcharts.com/docs/chart-and-series-types/polar-chart
[32] Detaljnije o istraživanju: https://journals.sagepub.com/doi/full/10.1177/1460458219895190




*Slika 54. Polarni/paukov dijagram bolničkih usluga u slučaju kirurškog zahvata*

U drugoj sferi, npr. u poslovnom i IT svijetu ovakav dijagram uvelike koristi konzultantima za upravljanje kako bi pokazali doprinos i prednosti njihove organizacije nad konkurentima. Na slici 56 vidljiva je usporedba ponude dvaju poznatih antivirusnih softvera u odnosu na ocjenu koju daju prema zadanoj kategoriji.




*Slika 55. Usporedba antivirusnih softvera Kaspersky i Norton*

Po svojoj definiciji, polarni grafikon je varijanta kružnog dijagrama koja se koristi za izvođenje više varijabilnih usporedbi ili analiza u kojem se podaci jedne varijable preklapaju s podacima druge. Niz podataka je predstavljen zatvorenom krivuljom koja povezuje točke u tzv. polarni koordinatni sustavu pri čemu je važno da podaci pokrivaju više vremenskih ciklusa ili jedan duži. U protivnom, kratki ciklusi ne mogu dati relevantne vrijednosti te će se točke „stisnuti“ na vanjske rubove grafikona. Važno je napomenuti da svaka točka podatka, neovisno o položaju, je određena udaljenošću od pola (radijalna koordinata) i kutom iz fiksnog smjera (kutna koordinata) što kasnije uvelike olakšava identifikaciju anomalija.

Imajte na umu da kod ovakvog grafikona je preporučeno korištenje samo dvije vrste strukturiranih podataka, odnosno 2 različita mnogokuta. U protivnom, karta će rezultirati zbunjujućom i prenatrpanom.

```
import matplotlib.pyplot as plt
```

```
import numpy as np
```

```
import pandas as pd
```

```
data = pd.read_csv(r'\Users\almas\Desktop\pokemons.csv')
```

```
data.head()
```

]:

| | # | Name | Type 1 | Type 2 | Total | HP | Attack | Defense | Sp. Atk | Sp. Def | Speed | Generation | Legendary |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | Bulbasaur | Grass | Poison | 318 | 45 | 49 | 49 | 65 | 65 | 45 | 1 | False |
| 1 | 2 | Ivysaur | Grass | Poison | 405 | 60 | 62 | 63 | 80 | 80 | 60 | 1 | False |
| 2 | 3 | Venusaur | Grass | Poison | 525 | 80 | 82 | 83 | 100 | 100 | 80 | 1 | False |
| 3 | 3 | VenusaurMega Venusaur | Grass | Poison | 625 | 80 | 100 | 123 | 122 | 120 | 80 | 1 | False |
| 4 | 4 | Charmander | Fire | NaN | 309 | 39 | 52 | 43 | 60 | 50 | 65 | 1 | False |

*Slika 56. Uvoz potrebnih biblioteka i očitavanje prvih 5 podataka*

Nova biblioteka koja se uvodi je *NumPy* pod aliasom `np`, a svoju prednost pokazuje u jačini rada s brojčanim podacima.

Obzirom na veliku količinu podataka u csv datoteci[33] ovaj graf je ograničen na samo dio polja, ali je ujedno i jedini interaktivan jer od korisnika zahtjeva unos kao što je prikazano na slici 58.

```
value_columns=np.array(['HP','Attack', 'Defense', 'Sp. Atk', 'Sp. Def', 'Speed'])

pokemon_No=int(input("The pokemon's data to plot(Input a number of row here, as each row represents a pokemeon's data): "))
```

```
The pokemon's data to plot(Input a number of row here, as each row represents a pokemeon's data): 4
```

*Slika 57. Funkcija koja očekuje unos korisnika*

Nakon unosa zapis se pohranjuje u varijablu vidljivu na slici 59, dok se linije u idućem redu tiču se konkatenacije – pridruživanja vrijednosti novoj varijabli *θ.*

```
selected_pokemon_stats=data.loc[pokemon_No,value_columns].values

theta=np.linspace(0, 2*np.pi, len(value_columns), endpoint=False)
selected_pokemon_stats=np.concatenate((selected_pokemon_stats,[selected_pokemon_stats[0]]))
theta=np.concatenate((theta,[theta[0]]))
```

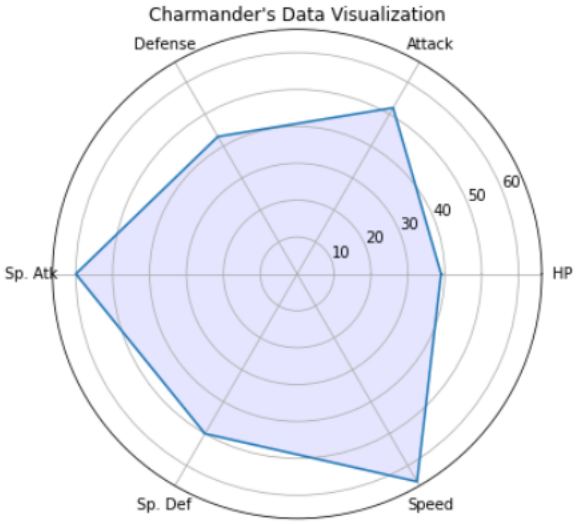
*Slika 58. Konkatenacija*

Kako bi graf ostao istovremeno vizualno prihvatljiv i funkcionalan, uvodi se spomenuta varijabla *θ*[34] – kreira vektor od vrijednosti 0 do 2π sa brojem koraka koji su jednaki dužini varijable `selected_pokemon_stats` . Idući red prikazuje kako se raspoređuje mreža naspram predodređenih vrijednosti polja, dok se ostale linije tiču uređivanja grafa i dodavanja tumača te boja – parametar `'b'` označava *blue*, tj. plavu boju.

Pretposljednja linija dohvaća ime indeksiranog pokemona pri čemu temeljem njegove pozicije piše naslov, a posljednja linija `plt.show()` prikazuje graf na ekranu.

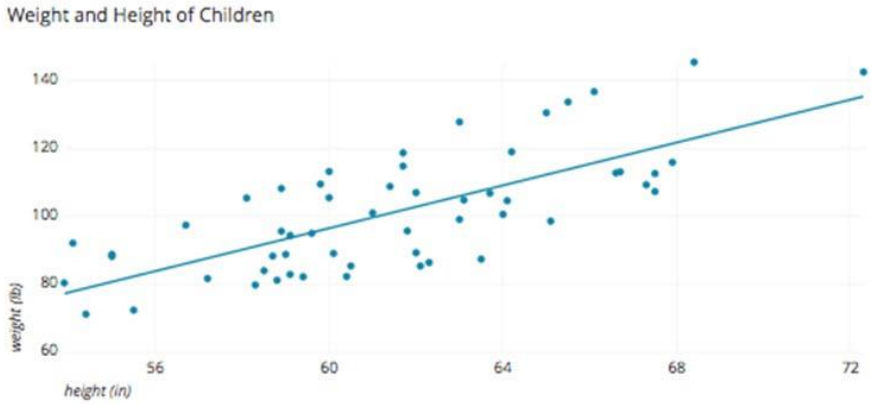
[33] Izvor: https://www.kaggle.com/abcsds/pokemon
[34] theta

```
plt.figure(figsize=(10, 6))
plt.subplot(polar=True)

theta = np.linspace(0, 2 * np.pi, len(selected_pokemon_stats))

lines, labels = plt.thetagrids(range(0, 360, int(360/len(value_columns))), (value_columns))

plt.plot(theta, selected_pokemon_stats)
plt.fill(theta, selected_pokemon_stats, 'b', alpha=0.1)

plt.title(str([data.loc[pokemon_No,"Name"]][0])+"'s Data Visualization")

plt.show()
```




*Slika 59. Finalni polarni dijagram izabranog pokemona*

## DIJAGRAM RASPRŠENJA




*Slika 60. Dijagram raspršenja na primjeru odnosa visina-težina djeteta*

Dijagram sa slike 60 prikazuje raspršenje podataka u kojima se s linijom trenda u omjer stavljaju visina i težina što je često viđen primjer. Ukoliko se radi o manjem setu podataka preporuka je dodati liniju trenda koja će u konačnici održati konzistentnost dijagrama.

Po Kirku (2012.), svaki dijagram raspršenja (*scatter plot*) ima minimalno dvije kvantitativne varijable. Koristi se kako bismo otkrili uzorak korelacija između prethodno spomenutih varijabli, a prikazuje se u koordinatnom sustavu sa x i y osima. Njegova važnost je u jasnom prikazu stršećih vrijednosti.

Za poboljšanje uspješnosti  izrade ovakvog dijagrama postoji set pravila:

- Izbjegnite *overplotting* – ukoliko je moguće ne koristiti pretjerane količine podataka jer će dovesti do preklapanja točaka što će rezultirati poteškoćama u interpretaciji (npr. dijagram raspršenja se pretvori u toplinsku kartu)
- *Correlation ≠ causation* – indirektno vezano za izradu, ali povezano sa tumačenjem dijagrama. Vezu između dvije varijable ne bi trebalo gledati kao uzročno-posljedičnu jer postoji mogućnost da na promatrano utječe treća varijabla. (npr. bolja ocjena određene epizode ne ovisi jer je epizoda bila kraća, već o ocjeni publike i broju glasova.)

- Treća, numerička varijabla – dijagram je puno lakše pratiti ako je sa strane vođen varijablom koja ga detaljnije objašnjava (npr. omjer ocjene naspram broja glasova u odnosu na 3. varijablu; gledanost u milijunima)
- Nijansiranje iste boje - svjetliji tonovi će indicirati na manje, a tamniji na veće vrijednosti. Korištena paleta `Blues` korisna je za sve skupine korisnika pa čak i daltoniste, s posebnim naglaskom na ciljanu skupinu koja pati od tritanomalije[35].

Odmah su prikazane dvije dodatne funkcije koje `dataset`[36] vrši nad sobom; `head()` i `tail()`. Ovisno o funkciji moguće je provjeriti prvih, odnosno posljednjih pet[37] vrijednosti csv-a.

```
In [3]: dataset.head()
```

Out[3]:

| | Unnamed: 0 | Season | EpisodeTitle | About | Ratings | Votes | Viewership | Duration | Date | GuestStars | Director | Writers |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | Pilot | The premiere episode introduces the boss and s... | 7.5 | 4936 | 11.2 | 23 | 24 March 2005 | NaN | Ken Kwapis | Ricky Gervais \|Stephen Merchant and Greg Daniels |
| 1 | 1 | 1 | Diversity Day | Michael's off color remark puts a sensitivity ... | 8.3 | 4801 | 6.0 | 23 | 29 March 2005 | NaN | Ken Kwapis | B. J. Novak |
| 2 | 2 | 1 | Health Care | Michael leaves Dwight in charge of picking the... | 7.8 | 4024 | 5.8 | 22 | 5 April 2005 | NaN | Ken Whittingham | Paul Lieberstein |
| 3 | 3 | 1 | The Alliance | Just for a laugh, Jim agrees to an alliance wi... | 8.1 | 3915 | 5.4 | 23 | 12 April 2005 | NaN | Bryan Gordon | Michael Schur |
| 4 | 4 | 1 | Basketball | Michael and his staff challenge the warehouse ... | 8.4 | 4294 | 5.0 | 23 | 19 April 2005 | NaN | Greg Daniels | Greg Daniels |

```
In [4]: dataset.tail()
```

Out[4]:

| | Unnamed: 0 | Season | EpisodeTitle | About | Ratings | Votes | Viewership | Duration | Date | GuestStars | Director | Writers |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 183 | 183 | 9 | Stairmageddon | Dwight shoots Stanley with a bull tranquilizer... | 8.0 | 1985 | 3.83 | 22 | 11 April 2013 | NaN | Matt Sohn | Dan Sterling |
| 184 | 184 | 9 | Paper Airplane | The employees hold a paper airplane competitio... | 8.0 | 2007 | 3.25 | 22 | 25 April 2013 | NaN | Jesse Peretz | Halsted Sullivan \| Warren Lieberstein |
| 185 | 185 | 9 | Livin' the Dream | Dwight becomes regional manager after Andy qui... | 9.0 | 2831 | 3.51 | 42 | 2 May 2013 | Michael Imperioli | Jeffrey Blitz | Niki Schwartz-Wright |
| 186 | 186 | 9 | A.A.R.M. | Dwight prepares for a marriage proposal and hi... | 9.5 | 3914 | 4.56 | 43 | 9 May 2013 | NaN | David Rogers | Brent Forrester |
| 187 | 187 | 9 | Finale | One year later, Dunder Mifflin employees past ... | 9.8 | 10515 | 5.69 | 51 | 16 May 2013 | Joan Cusack, Ed Begley Jr, Rachel Harris, Nanc... | Ken Kwapis | Greg Daniels |

*Slika 61. Funkcije head() i tail()*

[35] Anomalija pri kojoj osoba pokazuje slabost u raspoznavanju plave boje, javlja se veoma rijetko. (https://medical-dictionary.thefreedictionary.com/tritanomaly )

[36] Izvor: https://www.kaggle.com/nehaprabhavalkar/the-office-dataset

[37] 5 je *defaultna* vrijednost, a ukoliko korisnik želi više ili manje vrijednosti mora željeni broj predati kao parametar funkcijama

Linija 7 prikazuje inicijalizaciju varijabli koje će se koristiti na grafu u odnosu na njihove nazive u csv-u.

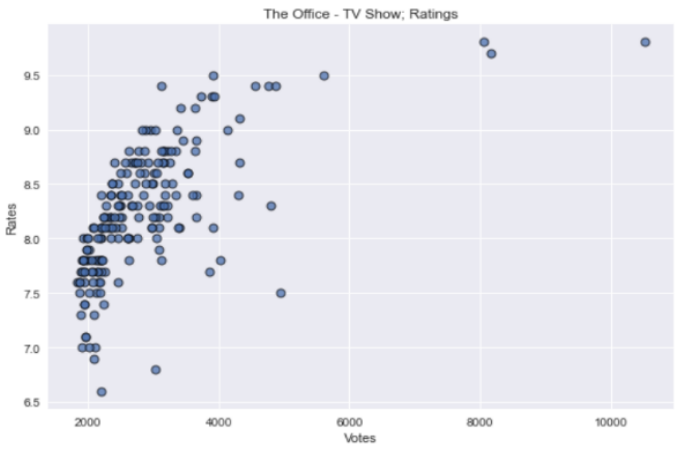
```
In [7]: view_count = dataset['Viewership']
        rates = dataset['Ratings']
        votes = dataset['Votes']
```

*Slika 62. Inicijalizacija varijabli*

Osma linija definira izgled grafa pa su tako:

- `Votes` i `Rates` - položaji podataka obzirom na osi x i y
- `edgecolor` – rubna boja oznake, u ovom slučaju crna
- `linewidth` – debljina ruba oznake
- `alpha` – vrijednost zakrivljenosti (raspon je vrijednost između 0 i 1)
- `plt.title()` – naslov grafa
- `plt.xlabel()` – oznaka x osi
- `plt.ylabel()` – oznaka y osi
- `plt.tight_layout()` – prilagođava sve navedene parametre da stanu u područje slike

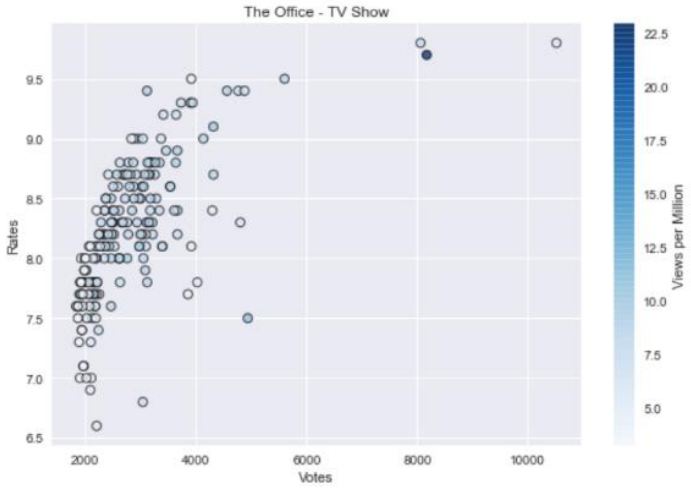
```
In [8]: plt.scatter(votes, rates, edgecolor='black',linewidth=1, alpha=0.75)

        plt.title('The Office - TV Show; Ratings')

        plt.xlabel('Votes')
        plt.ylabel('Rates')

        plt.tight_layout()
```




*Slika 63. Osnovni dijagram raspršenja*

Posljednje, kako bi graf izgledao kao na slici 65 uvedena su dva dodatna parametra: `c` – koji predstavlja skalu pregleda u milijunima; vidljiva desno od dijagrama i `cmap` – parametar kojem je pridružena vrijednost seta boja pod nazivom `Blues`.

```
In [12]: plt.scatter(votes, rates, c=view_count, cmap='Blues', edgecolor='black',linewidth=1, alpha=0.75)

         cbar= plt.colorbar()
         cbar.set_label('Views per Million')

         plt.title('The Office - TV Show')

         plt.xlabel('Votes')
         plt.ylabel('Rates')

         plt.tight_layout()
```



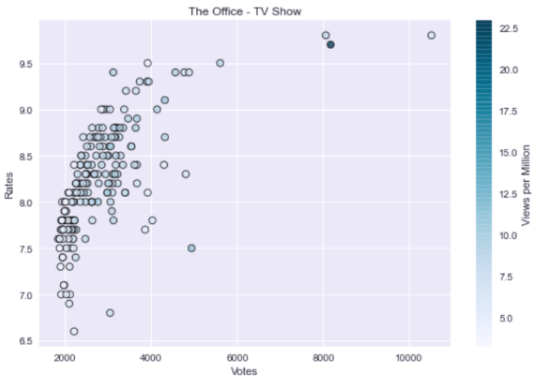

*Slika 64. Uređivanje podataka i konačan dijagram*

Dobiveni dijagram uspoređuje broj glasova[38] u tisućama sa ocjenom[39], a svaka od 'točkica' predstavlja jednu, samostalnu vrijednost koja se u ovom slučaju odnosi na epizodu. Preuzeti skup podataka za seriju „U uredu“, američka verzija, sastoji se od: 9 sezona, 187 epizoda, naslova epizoda, kratkog sadržaja, ocjene publike, broja glasova publike, broja pregleda, trajanja pojedine epizode, datuma objavljivanja, gostujućih glumaca te direktora i pisaca pojedine epizode. Na prvi pogled vidljiva su odstupanja i stršeće vrijednosti. U desnom kutu grafa vidi se jedna epizoda koja daleko odstupa od

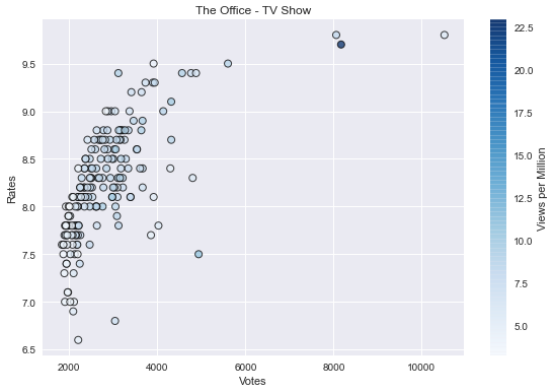
[38] *Votes*
[39] *Rates*

ostalih ocjenom i brojem glasova. Radi se o finalu serije koje je dobilo ocjenu 9.8 temeljem 10.515 glasova.

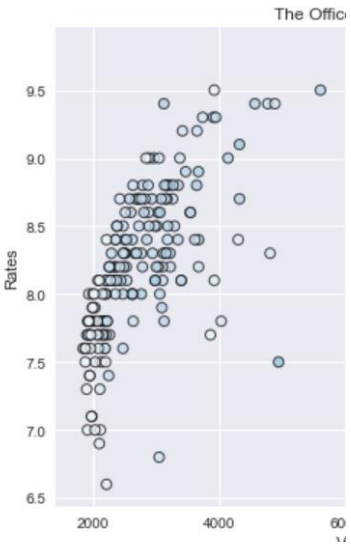
U svrhu shvaćanja što je stršeća vrijednost prikazana tamnijom bojom uvedena je treća varijabla kategoričkog tipa koja predstavlja pregled epizode u milijunima. Ustanovljeno je da se pregledi kreću do približno 22 milijuna gdje se zapravo nalazi i *Stress Relief* epizoda s rekordnih 22.91 milijun pregleda.




*Slika 66. Dijagram kakav vide osobe s tritanomalijom*

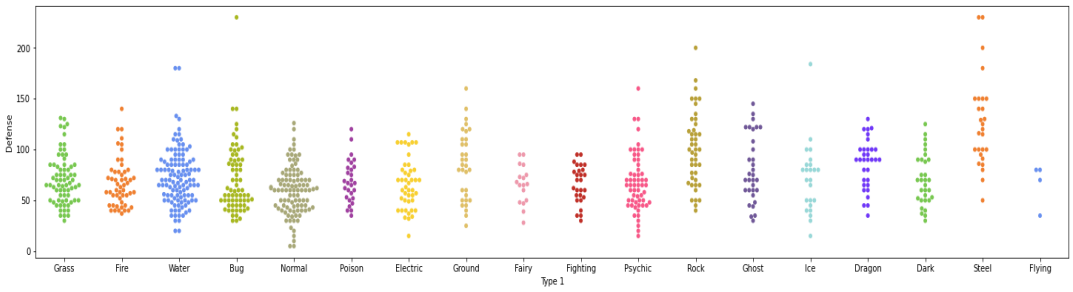



*Slika 65. Dijagram raspršenja*




*Slika 67. Zoomirani prikaz vrijednosti bez outliera*

ROJASTI DIJAGRAM




*Slika 68. Rojasti dijagram odnosa tipa pokemona i snazi obrane*

Rojasti dijagram podvrsta je dijagrama raspršenja i predstavlja kategoričke vrijednosti. Sličan je trakastom dijagramu, ali najbitnija razlika je što izbjegava preklapanja u podacima.

Za ovakav tip dijagrama savjetuje se koristiti manje *datasetove* kako bi se izbjegla pretrpanost. Funkcija `head()` sa slike 70 ispisuje prvih 5 pokemona, gledano abecedno.

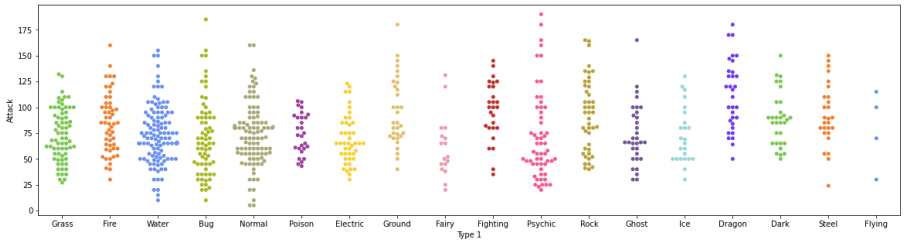
```
df.head()
```

| | # | Name | Type 1 | Type 2 | Total | HP | Attack | Defense | Sp. Atk | Sp. Def | Speed | Generation | Legendary |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | Bulbasaur | Grass | Poison | 318 | 45 | 49 | 49 | 65 | 65 | 45 | 1 | False |
| **1** | 2 | Ivysaur | Grass | Poison | 405 | 60 | 62 | 63 | 80 | 80 | 60 | 1 | False |
| **2** | 3 | Venusaur | Grass | Poison | 525 | 80 | 82 | 83 | 100 | 100 | 80 | 1 | False |
| **3** | 3 | VenusaurMega Venusaur | Grass | Poison | 625 | 80 | 100 | 123 | 122 | 120 | 80 | 1 | False |
| **4** | 4 | Charmander | Fire | NaN | 309 | 39 | 52 | 43 | 60 | 50 | 65 | 1 | False |

*Slika 69. Funkcija head()*

Objašnjenje koda počinje grafičkim prikazom od kojih je prvi sa funkcijom `sns.lmplot()`. Cilj ove funkcije je demonstracija koliko su podaci teški za interpretaciju ako se samo „postave“ na graf i dodijeli im se linija trenda.

```
In [9]: pokemon_type_colors = ['#78C850',  # Grass
                               '#F08030',  # Fire
                               '#6890F0',  # Water
                               '#A8B820',  # Bug
                               '#A8A878',  # Normal
                               '#A040A0',  # Poison
                               '#F8D030',  # Electric
                               '#E0C068',  # Ground
                               '#EE99AC',  # Fairy
                               '#C03028',  # Fighting
                               '#F85888',  # Psychic
                               '#B8A038',  # Rock
                               '#705898',  # Ghost
                               '#98D8D8',  # Ice
                               '#7038F8',  # Dragon
                              ]

In [10]: plt.figure(figsize=(20,5))
         sns.swarmplot(x='Type 1', y='Attack', data=df,
                       palette=pokemon_type_colors)

Out[10]: <AxesSubplot:xlabel='Type 1', ylabel='Attack'>
```




*Slika 70. Definiranje palete boja i prikaz prvog grafa*

U devetoj liniji stoga se definira polje `pokemon_type_colors` u kojem se nalazi cijela paleta boja kako bi graf bio što vjerodostojniji. Prvi prikazan graf definira omjer tipa[40] Pokemona i snage napada (`Attack`), a drugi umjesto napada prikazuje silinu obrane (`Defense`).

U obje linije `figsize`=(x, y) predstavlja varijablu čiji je prvi atribut - x povećanje grafa u širinu, a drugi atribut – y povećanje grafa u visinu. Vrijednosti varijabli izražene su u inčima.

PRETVORBA INČA U PIKSELE[41]

Prethodno spomenuta (`figsize`) „daje“ količinu prostora koju osi imaju unutar figure, dok dpi (*dots per inch*) determinira koliko piksela figura komprimira. Po *matplotlib* predefiniranim pravilima ono iznosi 100.

Formula za $figsize = (w, h)$ je:

$$p_{x_1}, p_y = w * dpi, h * dpi$$

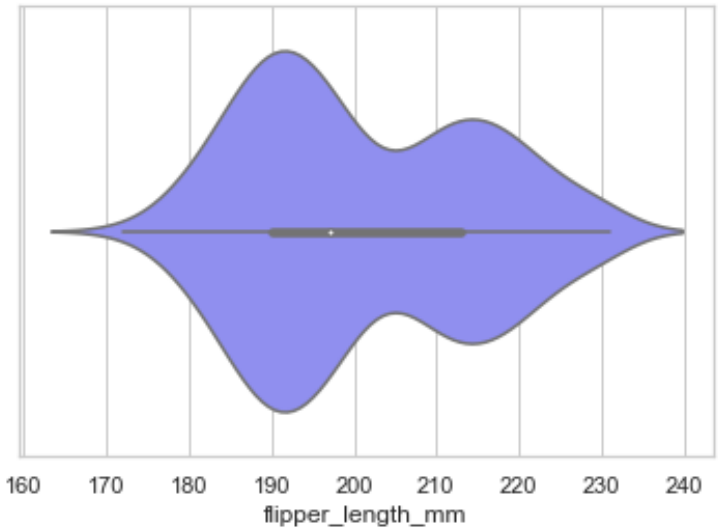
[40] Pripada li vodenim, vatrenim, električnim, klasičnim i sl. kategorijama (izvor: https://www.kaggle.com/abcsds/pokemon )

[41] Detaljniji primjer: https://stackoverflow.com/questions/47633546/relationship-between-dpi-and-figure-size

U prijevodu, ako se promijeni veličina figure u inčima točke se ne mijenjaju što znači da su elementi i dalje iste veličine. S druge strane, promjena dpi-ja utječe na skaliranje elemenata u odnosu:

72 dpi = 1 px jačine

VIOLINSKI DIJAGRAM




*Slika 71. Horizontalni violinski graf (duljina peraje pingvina u mm)*

Violinski dijagram (*Violin plot*) prikazuje distribuciju brojčanih podataka jedne ili više grupa koristeći krivulje gustoće. Njihova gustoća ovisi o obliku i iskrivljenosti podatka, dok širina odgovara približnoj učestalosti pojavljivanja podataka. U pravilu veći broj preciznih podataka znači i pouzdaniji graf. Imajući na umu količinu podataka potrebno je pravodobno utjecati i na položaj dijagrama. Primjerice, radi li se sa manjim grupama podataka koje imaju kraća imena preporuča se korištenje vertikalnog violinskog dijagrama.

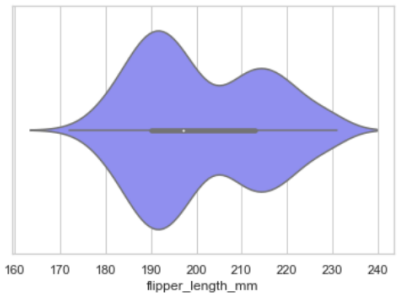
Općenito, lakše je proširiti graf/dijagram po vertikalnoj osi nego po horizontalnoj čime je znatno poboljšana i čitljivost u slučaju novo dodanih podataka.

Ovakav tip dijagrama nije čest zbog svog kompleksnijeg postavljanja koje rezultira slabijim shvaćanjem od strane osobe koja interpretira graf, stoga mu se dodaje pomoćni graf kao što je *boxplot*. Rubovi *boxplot*-a predstavljaju krajeve prvog i trećeg kvartila, dok je bijela točka vrijednost medijana; vidljivo na slici 71.

Kako bi uopće došlo do prikazanog dijagrama potrebno je uvesti do sad već dobro poznatu biblioteku *seaborn* pod aliasom `sns` (linija 1). U drugoj liniji se navodi tema koju vizualizacija koristi; *`whitegrid`*. Ovakav princip i boja su izabrani zbog povećane vidljivosti za sve korisnike.

Za razliku od ostalih skupova podataka koji se čitaju sa vlastitog računala, ovaj se čita direktno iz Github repozitorija[42]. U varijablu *`penguins`* spremaju se svi dostupni podaci za istoimeni csv file, dok je funkcija *`load_dataset()`* predefinirana da dohvaća podatke online. U idućem koraku testira se najjednostavniji primjer violinskog dijagrama, a to je po jednoj osi. Zadano je da os-x prihvaća parametre *`flipper_length_mm`* i prikazuje ih u paleti izabranoj po vlastitom izboru[43].

```
In [1]: import seaborn as sns
In [2]: sns.set_theme(style="whitegrid")
In [3]: penguins = sns.load_dataset("penguins")
In [4]: plot = sns.violinplot(x=penguins["flipper_length_mm"],palette="cool")
```

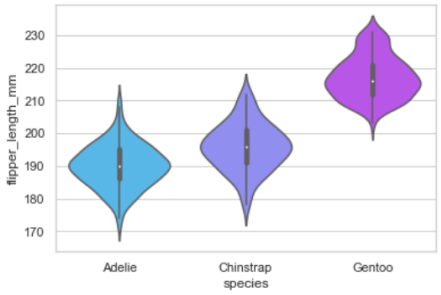



*Slika 72. Violinski dijagram po osi x*

[42] https://github.com/mwaskom/seaborn-data , naveden i u literaturi
[43] Uzeto je u obzir da paleta boja odgovara svim korisnicima bez obzira na očne i vidne poremećaje te percepcije boja

Ukoliko se doda os-y unutar istog paketa i palete, sa logičnije postavljenom osi x dobiva se pregledniji i značajniji graf od prethodnog. Vizualno je zanimljiviji jer istovremeno uspoređuje više kategorija dok je prethodni prikupio podatke od svih i prikazao u jednom.

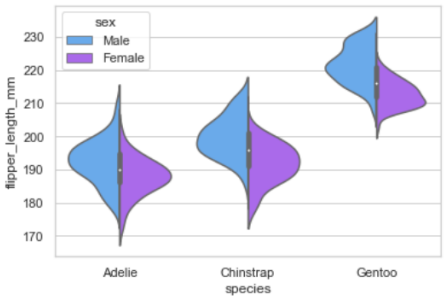
```
In [5]: plot = sns.violinplot(x="species", y="flipper_length_mm",
                             data=penguins, palette="cool")
```




*Slika 73. Violinski dijagram ovisnosti duljine peraje pingvina u mm naspram vrste kojoj pripada*

Najsloženiji tip violinskog dijagrama je sa dodatnim tumačem koji je treći argument definiran kao `hue`. Općenito, `hue` može sadržavati samo dvije vrijednosti koje su najčešće isključive jedna spram druge (npr. muško-žensko, istina-laž). U protivnom, dolazi do greške, ali program se pobrine za to i korisniku odmah javlja što je krivo.

```
In [6]: plot = sns.violinplot(x="species", y="flipper_length_mm", hue="sex",
                             data=penguins, palette="cool", split=True,
                             scale="count")
```




*Slika 74. Složeni violinski dijagram sa hue parametrom*

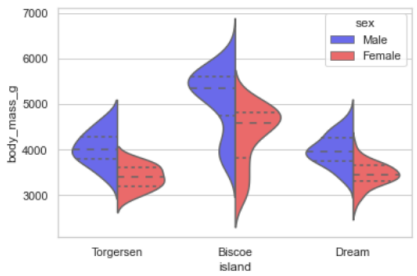
U priloženom primjeru *hue* je postavljen na vrijednost koja u csv fileu ima predefinirane 3 vrijednosti što se kosi s prethodno spomenutim pravilima. Rješenje greške vidljivo je na dnu koda pod dijelom *ValueError*.

```
In [9]:  ##ovaj ne radi due to Value error da hue moze imat samo 2 vrijednosti
         plot = sns.violinplot(x="species", y="flipper_length_mm", hue="island",
                               data=penguins, palette="Set2", split=True,
                               scale="count")

---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-9-0fa5cf9fea5b> in <module>
      1 ##ovaj ne radi due to Value error da hue moze imat samo 2 vrijednosti
----> 2 plot = sns.violinplot(x="species", y="flipper_length_mm", hue="island",
      3                       data=penguins, palette="Set2", split=True,
      4                       scale="count")

~\anaconda3\lib\site-packages\seaborn\_decorators.py in inner_f(*args, **kwargs)
     44             )
     45         kwargs.update({k: arg for k, arg in zip(sig.parameters, args)})
---> 46         return f(**kwargs)
     47     return inner_f
     48

~\anaconda3\lib\site-packages\seaborn\categorical.py in violinplot(x, y, hue, data, order, hue_order, bw, cut, scale, scale_
hue, gridsize, width, inner, split, dodge, orient, linewidth, color, palette, saturation, ax, **kwargs)
   2395 ):
   2396
-> 2397     plotter = _ViolinPlotter(x, y, hue, data, order, hue_order,
   2398                              bw, cut, scale, scale_hue, gridsize,
   2399                              width, inner, split, dodge, orient, linewidth,

~\anaconda3\lib\site-packages\seaborn\categorical.py in __init__(self, x, y, hue, data, order, hue_order, bw, cut, scale, sc
ale_hue, gridsize, width, inner, split, dodge, orient, linewidth, color, palette, saturation)
    539         if split and self.hue_names is not None and len(self.hue_names) != 2:
    540             msg = "There must be exactly two hue levels to use `split`.'"
--> 541             raise ValueError(msg)
    542         self.split = split
    543

ValueError: There must be exactly two hue levels to use `split`.'
```

*Slika 75. ValueError koji očekuje 2, a dobiva 3 parametra*

Ukoliko postoji potreba za vizualizacijom kvartila na violinskom dijagramu dovoljno je dodati novi parametar *inner="quartile"* koji će iscrtati sve potrebno. Ostali parametri kao što su *scale="count"* – postavlja širinu violina u odnosu na broj promatranja u uzorku, a *split=True* – nacrtat će polovinu violine za svaku razinu.

```
In [10]: plot = sns.violinplot(x="island", y="body_mass_g", hue="sex",
                               data=penguins, palette="seismic", split=True,
                               scale="count", inner="quartile")
```




*Slika 76. Violinski dijagram sa kvartilima*

## ŠTO (NE) S GRAFOVIMA?

Nastavno na definiciju vizualizacije iz uvoda „(…) *vizualizacija je jezik donošenja odluka. Dobri grafovi učinkovito prenose informacije. Odlični grafovi omogućuju, informiraju i poboljšavaju donošenje odluka.*“ (Vitagliano, 2020.). Na pojedinim grafičkim prikazima dalo se vidjeti da nije uvijek tako. Razlog tome su manji nedostatci vizualizacija.

Prvi takav je pristranost osobe koja provodi vizualizaciju. Na vlastitom primjeru to bi bila toplinska karta koja prikazuje koncentraciju peludi u Puli za kolovoz. Pristranost ovdje ne podrazumijeva izbor lokacije (jednak je mjestu stanovanja) već korištenje samo osobno važnih podataka. Konkretnije kroz primjer, isključeni su podaci od ostalih dana u mjesecu kolovozu ili podaci o prethodnim mjesecima pa čak i podaci o prethodnoj godini. Graf je isključiv zbog svoje usredotočenosti na kraći period što bi moglo utjecati na osobu koja interpretira isti. Vizualizacija ne pruža brojne zaključke u smislu usporedbe podataka ili predviđanja obzirom na svoju ograničenost, ali je dobar primjer za prikaz izrade glede grafičkog izgleda.

Drugi nedostatak je također vezan uz podatke, a tiče se njihove istinitosti i točnosti. Terminom *corrupted data*[44] najbolje je obuhvaćeno o čemu je riječ s posebnim naglaskom na prijenos i obradu. Dakle, ustanovi li se obradom da je podatak lažiran ili dupliciran vizualizacija može pretrpjeti veliku štetu. Ona može biti tehnički točna i napraviti očekivano (npr. Nacrtati kružni dijagram, ali će zasićenost podataka biti enormna) što će utjecati na preglednost grafa.

Treći nedostatak, uz napomenu da su izuzeti ekonomski nedostaci (preskup i kompleksan proces, neshvaćanje važnosti upravljanja raspoloživim podacima i nestručnost pojedinaca) tiče se sortiranja podataka. Brojni *datasetovi* koji su dostupni *online* imaju velika pitanja upotrebljivosti[45] zbog svoje nelogično složene strukture ili u dodatnom slučaju, nepotpune. Srećom, program za takve csv i ostale datoteke izbaci upozorenje i odmah navede gdje nedostaje podatak. (npr. pri izradi stupčastog dijagrama za olimpijske medalje nedostajao je podatak kojeg je korisnik kasnije unio, ali pogreška

[44] Oštećenje podataka - odnosi se na sve greške u kompjuterskim podacima koje mogu nastati prilikom pisanja, čitanja, pohrane, prijenosa ili obrade, a uvode nenamjerne promjene u izvornim podacima (*Techopedia*, 2017.)
[45] Ponašanje koje sprječava dovršetak zadataka do pogrešnog tumačenja sadržaja

koja je prethodila tome izgledala je ovako *ParserError: Error tokenizing dana. C error: Expected 1 fields in line 40, saw 2*).

Pretposljednji nedostatak povezan je uz pojam laganja statistikom. Prvi primjer tiče se uzorka s ugrađenom pristranošću i svakodnevno se u medijima može čuti minimalno jedan takav. Primjerice, „Nakon završenog učilišta 96% naših polaznika odmah pronađe posao!“ zvuči primamljivo, ali i problematično jer kao analitičar ne znamo način provođenja ankete ni veličinu uzorka. U istraživanju su mogli sudjelovati npr. samo polaznici koji se žele nečim pohvaliti ili nešto istaknuti. Drugi primjer laganja statistikom je usko vezan uz nerazlikovanje korištenja prosjeka pa se tako često čuje podatak kako je „prosječna neto plaća porasla za 2% na području RH“. Nitko ne ulazi u detalje radi li se pritom o aritmetičkom prosjeku, medijanu - polovini od broja ili modu kao najčešćem broju. Promatraju li se medijan i mod, vrijednost same plaće drastično opada. Ovakve činjenice su korisne ukoliko pretražujete neki posao za kojim idete samo zbog plaće.

Konačno, posljednji nedostatak veže se uz samo laganje grafovima sveobuhvatno prikazano na slici 77. Najčešći problemi proizlaze iz:

- lošeg skaliranja – analitičar grafa može biti potencijalno odvučen da se bavi beznačajnom kategorijom ili graf jednostavno prikazuje nerealne oscilacije između usporednih podataka[46]
- neadekvatnog izbora grafikona obzirom na dostupne podatke – podrobnije objašnjenje nije potrebno
- neusporedivih podataka – odgovori koji nisu relevantni u odnosu na pitanje[47]

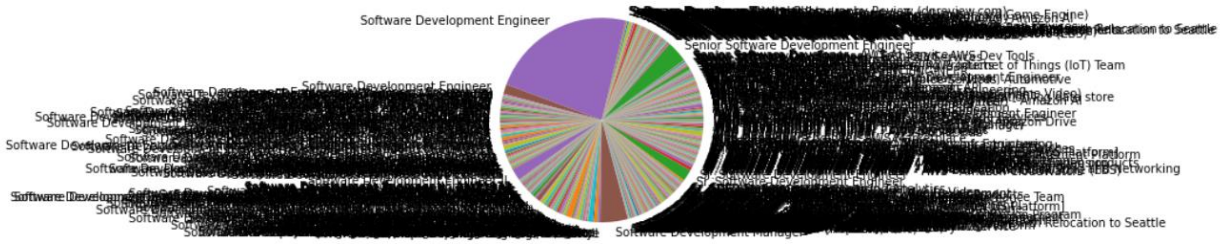
[46] Izvor: https://www.pinterest.com/pin/415527503099030015/

[47] Izvor: https://www.reddit.com/r/dataisugly/comments/2h8nlb/milk_or_gas_yes_or_no_xpost_rfunny/


*Slika 77. Sveobuhvatni primjeri laganja grafovima*

Većina ovih nedostataka može biti riješena ukoliko pojedinci ne odluče „učiniti nešto na svoju ruku“. Primjerice, ukoliko postoji podatak koji nedostaje, upotpuniti njegovo mjesto nečim relevantnim ili istinitim umjesto lažiranim. Nadalje, preuzimati samo podatke koji su jamčeno pouzdani i točni te dovoljno istraženi. Imajući na umu ove nedostatke i savjete kako ih suzbiti vizualizacije će postati preglednije, jasnije, ali prije svega korisnije kako nama koji ih izrađujemo tako i krajnjim korisnicima koji ih interpretiraju.

```
In [131]: y.groupby("Title").size().plot.pie(y="Title",ylabel="LABELA")
Out[131]: <AxesSubplot:ylabel='LABELA'>
```




*Slika 78. Prezasićeni kružni dijagram*

# ZAKLJUČAK

Vizualizacija i prikaz podataka ulaze u novu eru. Papir i nekadašnji grafički prikazi iz Excela zamijenjeni su programskim jezicima poput Pythona. Umjetna inteligencija, dostupnost informacija i podataka, a samim time razvoj i napredak tehnologija povećavaju vrijednost koju analitika može pružiti.

Dok načela učinkovitosti ostaju nepromijenjena, biblioteke poput *Matplot*a, *Seaborn*a i *Plotly*a pomiču vizualizaciju iz sfere umjetnosti u znanost koja je danas poznata kao *data science*[48]*.* Ovisno o razini znanja i količini snalažljivosti svaka od spomenutih biblioteka korisna je u nekom od već spomenutih načela: *matplotlib* – izvorno *low level* i otvorenog tipa daje dosta slobode korisnicima za izradu jednostavnijih grafičkih prikaza kao što su histogrami, stupčasti i kružni dijagrami. *Seaborn* – baziran na *Matplotu,* u klasi je *high level* sučelja sa raznim *defaultnim* stilovima. Najveća prednost nad *Matplotom* je pisanje koda u obliku *one linera*[49] gdje bi bili potrebni deseci. Posljednje navedena biblioteka, *Plotly,* svoju specifičnost daje u mogućnosti izrade interaktivnih grafova koji se najbolje vide na trodimenzionalnim primjerima.

U konačnici, samo pravilnim korištenjem biblioteka i ispravnom manipulacijom podataka održat će se rast i popularnost ovih alata, a vizualizacija će ostati dosljedna, jasna te korisna u donošenju potrebnih odluka.

[48] Znanost o podacima
[49] One liner – kod koji karakterizira samo jedna linija koda

LITERATURA


1. Huff, D. (2001.) *Kako lagati statistikom*, [posuđeno: 21.4.2021., GKČ Pula]
2. Dokumentacija platforme Anaconda: https://anaconda.en.softonic.com/ [pristup: 5.8.2021.]
3. Dokumentacija Jupyter Notebook (*Starting the notebook server*): https://jupyter-notebook.readthedocs.io/en/stable/notebook.html [pristup: 5.8.2021.]
4. https://chartio.com/learn/charts/what-is-a-scatter-plot/ , teorija za Scatter plot (dijagram raspršenja) [pristup: 5.8.2021.]
5. https://www.kaggle.com/nehaprabhavalkar/the-office-dataset , csv korišten za izradu *Scatter* plota (dijagram raspršenja) [preuzeto: 5.8.2021.]
6. https://matplotlib.org/stable/tutorials/colors/colormaps.html , dokumentacija za korištenje boja u sklopu Matplotlib biblioteke [pristup: 5.8.2021.]
7. https://www.plivazdravlje.hr/alergije/prognoza/20/Pula.html , Peludna prognoza za Pula, Hrvatska [pristup: svakodnevno 5.8.2021.-12.8.2021.]
8. https://www.kaggle.com/abcsds/pokemon korišteni dataset za izradu Swarm plota [pristup: 6.8.2021.]
9. https://blog.datawrapper.de/which-color-scale-to-use-in-data-vis/ , Savjeti za korištenje boja ovisno o vizualizaciji [pristup: 6.8.2021.]
10. https://blog.datawrapper.de/colorblindness-part2/ , Savjeti za korištenje boja ovisno o vizualizaciji – prilagodba daltonistima [pristup: 6.8.2021.]
11. https://www.delftstack.com/howto/seaborn/seaborn-swarm-plot-python/ teorija za Swarm plot (rojasti dijagram) [pristup: 6.8.2021.]
12. Kirk, A. (2012.) *Data Visualization: a successful design process.* Birmingham: Packt Publishing Ltd.
13. Nelli, F. (2018.) *Python Dana Analytics: With Pandas, NumPy, and Matplotlib*: https://www.researchgate.net/publication/327921438_Python_Data_Analytics_With_Pandas_NumPy_and_Matplotlib [preuzeto 7.8.2021.]
14. Johansson, R. (2019.) *Scientific Computing and Dana Science: Applications with Numpy, SciPy and Matplotlib.* Second Edition: https://www.researchgate.net/publication/330067748_Numerical_Python_Scientif

ic_Computing_and_Data_Science_Applications_with_Numpy_SciPy_and_Matplotlib [preuzeto: 7.8.2021.]

15. Sievert, C. (2020.) *Interactive Web-Based Data Visualization with R, Plotly, and Shiny*: https://www.researchgate.net/publication/340147564_Interactive_Web-Based_Data_Visualization_with_R_plotly_and_shiny [preuzeto: 7.8.2021.]
16. https://www.color-blindness.com/coblis-color-blindness-simulator/ , simulator boja za daltoniste; prilagođeno na tritanomaliju [pristup: 8.8.2021.]
17. https://medical-dictionary.thefreedictionary.com/tritanomaly , medicinska definicija tritanomalije [pristup: 8.8.2021.]
18. https://chartio.com/learn/charts/violin-plot-complete-guide/ , teorija potrebna za Violin plot [pristup: 8.8.2021.]
19. https://github.com/mwaskom/seaborn-data/blob/master/penguins.csv , korišteni dataset za izradu Violin plota [pristup: 8.8.2021.]
20. https://www.kaggle.com/harshitshankhdhar/netflix-and-amazon-prime-tv-series-dataset , korišteni dataset za izradu Box plota [pristup: 8.8.2021.]
21. https://seaborn.pydata.org/generated/seaborn.boxplot.html , teorija potrebna za Box plot [pristup: 8.8.2021.]
22. https://cis.temple.edu/~latecki/Papers/mldm07.pdf , teorija potrebna za KDE plot [pristup: 10.8.2021.]
23. https://seaborn.pydata.org/generated/seaborn.kdeplot.html , KDE plot primjeri i dokumentacija [pristup: 10.8.2021.]
24. https://www.kaggle.com/shivam2503/diamonds korišteni dataset za izradu KDE plota [pristup: 10.8.2021.]
25. https://www.fusioncharts.com/resources/chart-primers/heat-map-chart , teorija za Heatmap Chart (toplinska karta) [pristup: 11.8.2021.]
26. https://datagy.io/histogram-python/ , teorija za histogram [pristup: 11.8.2021.]
27. https://www.tutorialspoint.com/plotly/plotly_bar_and_pie_chart.htm , teorija za kružni dijagram [pristup: 11.8.2021.]
28. https://www.highcharts.com/docs/chart-and-series-types/polar-chart , teorija za polarni dijagram [pristup: 12.8.2021.]

29. https://chartio.com/learn/charts/bar-chart-complete-guide/ , teorija potrebna za Bar chart [pristup: 13.8.2021.]
30. https://www.kaggle.com/berkayalan/2021-olympics-medals-in-tokyo , korišteni dataset za izradu Bar charta [pristup: 13.8.2021.]
31. https://plotly.com/python/graph-objects/ , Dokumentacija grafičkih objekata za izradu *Stacked Bar Charta* [pristup: 13.8.2021.]
32. https://www.tutorialspoint.com/plotly/plotly_bar_and_pie_chart.htm , Grafički objekti za izradu *Stacked Bar Charta* [pristup: 13.8.2021.]
33. https://chartio.com/learn/charts/line-chart-complete-guide/ , teorija potrebna za Line chart [pristup: 4.9.2021.]
34. https://covid19.who.int/region/euro/country/hr , ažurni podaci vezani uz COVID-19 za područje RH [pristup: 4.9.2021.]
35. https://www.kaggle.com/josephassaker/covid19-global-dataset , korišteni skup podataka za izradu line plota [pristup: 4.9.2021.]
36. https://www.callingbullshit.org/tools/tools_proportional_ink.html , [pristup: 9.9.2021.]
37. https://skeptric.com/dip-statistic/ [pristup: 9.9.2021.]
38. https://blog.datawrapper.de/colorblindness-part1/ [pristup: 9.9.2021.]
39. https://material.io/components/data-tables [pristup: 11.9.2021.]
40. https://medium.com/design-with-figma/the-ultimate-guide-to-designing-data-tables-7db29713a85a [pristup: 11.9.2021.]

# POPIS SLIKA

SAŽETAK I KLJUČNE RIJEČI

SAŽETAK:

Svjesno ili ne, oduvijek i konstantno smo izloženi podacima. Za one koji su nam poznatiji ili srodniji struci klasificiramo u informacije koje zatim plasiramo dalje. Ipak, nije uvijek tako jer ponekad imamo mnoštvo podataka za koje ni sami ne znamo kako iz njih dobiti nešto jasno, a samim time i korisno. U prilog tome, razvojem programskih jezika i pripadajućih biblioteka vizualizacija postaje ključna pri radu s podacima. Prikladno potrebama ovog rada koriste se platforma Anaconda i programski jezik Python (sa bibliotekama: Matplotlib, Plotly, Seaborn i Numpy).

Sve vizualizacije sadrže popratni kod i teorijsko objašnjenje, odnosno tumač.

KLJUČNE RIJEČI: podaci, prikaz podataka, tehnike vizualizacije, Python, Jupyter Notebook, Anaconda, Matplotlib, Seaborn, Plotly, Numpy

ABSTRACT:

Consciously or not, always and constantly, we are exposed to ample data. For those which are familiar to us or more related to the chosen profession, we have accurately classified them into valuable information that we then place further. However, this is not inevitably the case. Every so often we retain data for which we ourselves do not know how to get something clear from them, and in time valuable. In addition, with the gradual development of modern programming languages and associated libraries, visualization becomes crucial when properly working with data. Appropriately, the needs of this academic paper are adequately managed by the leading Anaconda platform and the Python programming language (With libraries: Matplotlib, Plotly, Seaborn, and Numpy).

All visualizations precisely contain an accompanying code and a theoretical explanation.

KEYWORDS: data, data visualization, visualization techniques, Python, Jupyter Notebook, Anaconda, Matplotlib, Seaborn, Plotly, Numpy