Jazyky pre Dátovú Analytiku (JDA)

Prednáška č.5

Práca s dátami a vizualizácia

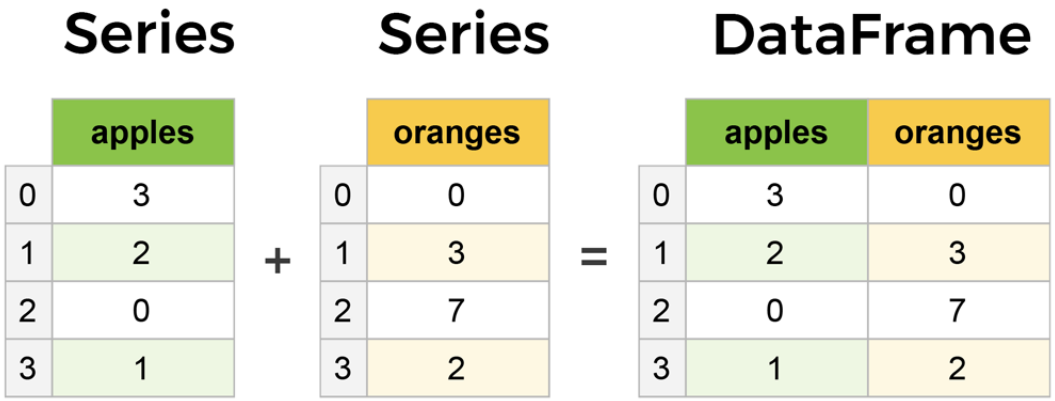
- Podobne ako v prípade R potrebujeme vedieť
  - načítať dáta do dátových tabuliek
  - vedieť s nimi pracovať
  - vizualizovať dáta, ich vlastnosti a vzťahy
  - realizovať zložitejšie analýzy
- Tak ako predtým – dátová tabuľka predstavuje typ objektu, kde máme množinu položiek = množina objektov / inštancií o ktoré sa zaujímame
  - Riadok = položka = objekt = inštancia = príklad = prípad (case)
  - Premenná = atribút = stĺpec = meranie alebo charakteristika objektu
    - Kvalitatívna – pohlavie, krajina pôvodu, ...
    - Kvantitatívna – výška, váha, ...
- Nástroje popísané v prednáške (samozrejme možností je viac)
  - Pandas
  - Seaborn

Pandas

- http://pandas.pydata.org
- Balík / nástroj pre ukladanie a prácu s tabuľkovými dátami vo forme dátových rámcov
- Postavený ako nadstavba NumPy knižnice
  - Predstavuje jeden zo základných nástrojov dátovej analytiky v Pythone
  - Má pomerne jednoduché a zrozumiteľné vysoko-úrovňové API
  - V podstate medzi R dataframe (a rozšíreniami) a Pandas je vo funkcionalite dualita – avšak názvy funkcií a spôsoby zápisu môžu byť iné (dosiahneme však v oboch prípadoch to čo potrebujeme)

Základné štruktúry pandas

- Series
  - v podstate usporiadaný slovník (mapa) / dict (s indexom), ktorý ale umožňuje opakovanie
  - Podtrieda numpy.ndarray
  - Dátovo môže byť ľubovoľný atomický typ (numerické hodnoty, reťazce, ...)
  - V podstate je to obdoba **vector** v R
- Dataframe
  - Dátová tabuľka, napríklad spojením viacerých Series vieme dostať dataframe (jedna Series = atribút)



Jednoduchý dataframe

```
df = pd.DataFrame({
'pocet': [2, 3, 1, 3, 4],
'typ': ['Chlieb', 'Maslo', 'Mlieko', 'Maslo', 'Mlieko'],
'vlastnik': ['Peter', 'Peter', 'Martin', 'Martin', 'Peter']
})
```

df

	pocet	typ	vlastnik
0	2	Chlieb	Peter
1	3	Maslo	Peter
2	1	Mlieko	Martin
3	3	Maslo	Martin
4	4	Mlieko	Peter

print(df)

```
   pocet     typ vlastnik
0      2  Chlieb    Peter
1      3   Maslo    Peter
2      1  Mlieko   Martin
3      3   Maslo   Martin
4      4  Mlieko    Peter
```

Výber prvkov / podmnožín (vybrané príklady – možností je viac)

- Výber stĺpca – názov stĺpca v []

```
df['typ']
```

```
0     Chlieb
1      Maslo
2     Mlieko
3      Maslo
4     Mlieko
Name: typ, dtype: object
```

- Výber riadka – použitie „loc“

```
df.loc[2]
```

```
pocet           1
typ        Mlieko
vlastnik   Martin
Name: 2, dtype: object
```

- Výber viacerých riadkov – slicing

```
df.loc[2:4]
```

	pocet	typ	vlastnik
2	1	Mlieko	Martin
3	3	Maslo	Martin
4	4	Mlieko	Peter

- Výber viacerých stĺpcov – napr. cez loc

```
df.loc[:,'pocet':'typ‘]
```

	pocet	typ
0	2	Chlieb
1	3	Maslo
2	1	Mlieko
3	3	Maslo
4	4	Mlieko

- Výber viacerých riadkov a stĺpcov

```
df.loc[2:4,'pocet':'typ']
```

	pocet	typ
2	1	Mlieko
3	3	Maslo
4	4	Mlieko

Ďalšie operácie výpisu / otočenie

- Zobrazenie prvých „n“ riadkov
  - Funkcia head ... df.head(3)
- Zobrazenie posledných „n“ riadkov
  - Funkcia tail ... df.tail(2)
- Otočenie riadky <-> stĺpce – pivotná operácia (pivoting)
  - Pomocou loc .... df.loc[3:4].T

	pocet	typ	vlastnik
0	2	Chlieb	Peter
1	3	Maslo	Peter
2	1	Mlieko	Martin

	pocet	typ	vlastnik
3	3	Maslo	Martin
4	4	Mlieko	Peter

	3	4
pocet	3	4
typ	Maslo	Mlieko
vlastnik	Martin	Peter

Pridanie / odstránenie stĺpcov / riadkov

- Pridanie nového stĺpca
  - Môžeme si pripraviť list hodnôt a priradiť nový stĺpec

```
obchod = ['Billa','Billa','Tesco','Tesco','Tesco']
df['obchod'] = obchod
df
```

	pocet	typ	vlastnik	obchod
0	2	Chlieb	Peter	Billa
1	3	Maslo	Peter	Billa
2	1	Mlieko	Martin	Tesco
3	3	Maslo	Martin	Tesco
4	4	Mlieko	Peter	Tesco

- Odstránenie stĺpca
  - Pomocou funkcie drop

```
df.drop(columns=['obchod'])
```

- **Drop** sa dá použiť aj na odstránenie riadkov
- Pre pridanie riadkov sa dá použiť **append**

Načítanie dát do pandas dataframe

```
pocet,typ,vlastnik,obchod
2,Chlieb,Peter,Billa
3,Maslo,Peter,Billa
1,Mlieko,Martin,Tesco
3,Maslo,Martin,Tesco
4,Mlieko,Peter,Tesco
```

- Načítanie csv súboru
  - Funkcia read_csv
  - Povedzme že našu tabuľku mám v mytab.csv

```
df2 = pd.read_csv("mytab.csv")
df2
```

	pocet	typ	vlastnik	obchod
0	2	Chlieb	Peter	Billa
1	3	Maslo	Peter	Billa
2	1	Mlieko	Martin	Tesco
3	3	Maslo	Martin	Tesco
4	4	Mlieko	Peter	Tesco

- Existuje samozrejme viac nastavení
  - delimiter alebo sep (obidva sú aliasy pre separátor), header (default sa snaží použiť prvý riadok ako hlavičku – názvy stĺpcov), decimal (definícia znaku desatinnej „čiarky“), names (môžeme vložiť vlastné názvy stĺpcov), skiprows, nrows, na_filter, parse_dates, ... a mnoho ďalších
- Pre uloženie df do súboru ... napr. funkcia to_csv()

Popis dátovej množiny a jej atribútov

- Môžeme použiť shape na rýchle zistenie tvaru tabuľky (riadky x stĺpce) ... df.shape … (5,3)
- Podobne ako v R existuje „summary“ .... v pandas ale
  - potrebujeme describe pre numerické atribúty
  - funkciu value_counts pre diskrétne atribúty
- Describe – použije sa na všetky numerické

  ..... df.describe()
- Value_counts … početnosti...

  napr. pre atribút typ

  df['typ'].value_counts()

```
Maslo     2
Mlieko    2
Chlieb    1
Name: typ, dtype: int64
```

	pocet
count	5.000000
mean	2.600000
std	1.140175
min	1.000000
25%	2.000000
50%	3.000000
75%	3.000000
max	4.000000

Iris dataset

sepal_length,sepal_width,petal_length,petal_width,species
5.1,3.5,1.4,0.2,setosa
4.9,3,1.4,0.2,setosa
4.7,3.2,1.3,0.2,setosa
4.6,3.1,1.5,0.2,setosa
5,3.6,1.4,0.2,setosa
5.4,3.9,1.7,0.4,setosa
4.6,3.4,1.4,0.3,setosa
5,3.4,1.5,0.2,setosa
4.4,2.9,1.4,0.2,setosa
4.9,3.1,1.5,0.1,setosa
5.4,3.7,1.5,0.2,setosa
.....

	sepal_length	sepal_width	petal_length	petal_width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

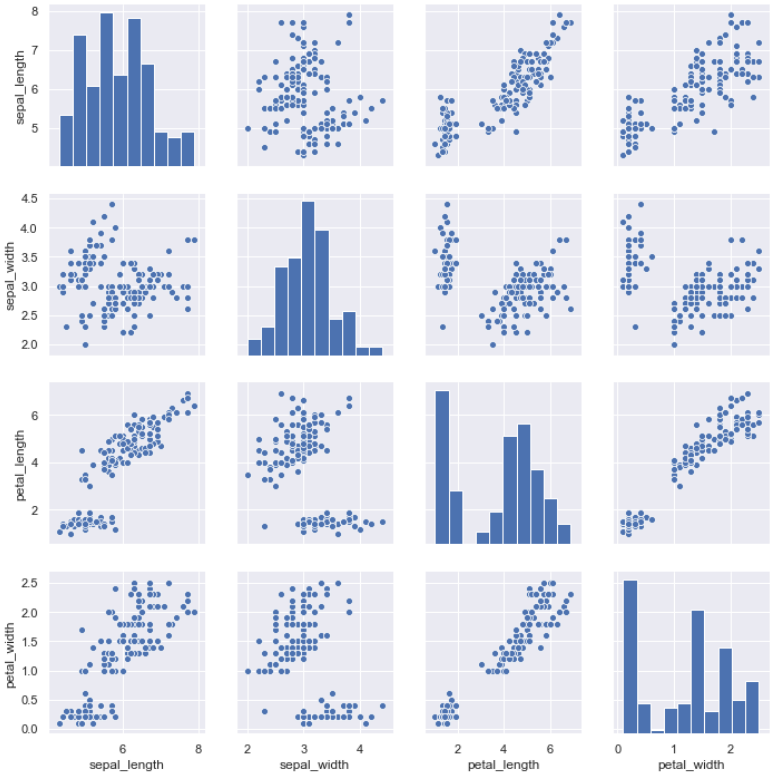
Popis vzťahov medzi atribútmi

- Vzťah dvojíc numerických atribútov – korelácie
  - Matica korelačných koeficientov – corr funkcia

  Nech data je dataframe s iris dátami

  data.corr() .... vypíše korelačnú maticu pre vš. numerické atribúty
  - Grafické zobrazenie bodových grafov medzi všetkými dvojicami numerických atribútov ... pair plot ... sns.pairplot(data)
- V R alternatívy
  - cor() funkcia pre korelačnú maticu
  - pairs() funkcia pre pairplot

	sepal_length	sepal_width	petal_length	petal_width
sepal_length	1.000000	-0.109369	0.871754	0.817954
sepal_width	-0.109369	1.000000	-0.420516	-0.356544
petal_length	0.871754	-0.420516	1.000000	0.962757
petal_width	0.817954	-0.356544	0.962757	1.000000



Kontingenčné tabuľky / pivotné tabuľky

- Ak chceme zobrazovať početnosti alebo štatistiky riadkov zoskupených podľa nejakých faktorových atribútov
  - Napr. podľa jednotlivých tried v iris (setosa,virginica, versicolor), alebo podľa pohlavia (titanic)
- Môžeme použiť
  - crosstab ... všeobecný postup na početnosti rôznych kombinácií hodnôt atribútov
  - pivot_table .... dobré ak chceme určiť aj odvodené hodnoty
- Príklad1 – df, chceme tabuľku ktorý vlastník koľko ktorých typov má

```
pd.crosstab(index=df["typ"], columns=df["vlastnik"])
```

vlastnik	Martin	Peter
typ		
Chlieb	0	1
Maslo	1	1
Mlieko	1	1

- Príklad 2 – iris, chceme vypočítať priemernú hodnotu pre atribúty sepal.length a sepal.width zoskupných podľa species

```
pd.pivot_table(data, index="species",
values=["sepal_length", "sepal_width"], aggfunc=["mean"])
```

	mean	
	sepal_length	sepal_width
species		
setosa	5.006	3.418
versicolor	5.936	2.770
virginica	6.588	2.974

NA hodnoty a diskretizácia

- Ak máme NA hodnoty, môžeme ich nahradiť pomocou funkcie **fillna**
  - Napr. máme atribút „vyska“ v datasete „dt“ a rozhodneme sa nahradiť všetky NA hodnoty mediánom výšky

```
data["vyska"].fillna(data["vyska"].median(), inplace=True)
```

- Diskretizácia – transformácia numerického atribútu na diskrétny (ako faktor v R) – podobne ako v R funkcia cut()
  - Napr. máme ten istý atribút výška, ale zmeníme ho tak že namiesto konkrétnych hodnôt v cm budú hodnoty pretransformované na nízky, normálny, vysoký (hranice sú jednotlivé hranice binov, čiže normálny je medzi 160 a 190)

```
data["vyska_ordinal"] = pd.cut(data["vyska"], bins=[0, 160, 190, 210], include_lowest=True, labels=["nizky", "normalny", "vysoky"])
```

Vizualizácia dát v Python

- Existujú základné systémy ako Matplotlib
- Nad nimi vzniklo viacero rozšírení, ktoré zjednodušujú syntax a prístup k funcionalitám - ako napr. Seaborn
- Existuje samozrejme veľa ďalších knižníc, ktoré sú často používané (najmä JS knižnice) aj v R aj V Pythone
  - Plotly
  - Bokeh
  - D3
  - ...

Seaborn

- Seaborn je python knižnica určená pre vizualizácie, resp. vykresľovanie
- Umožňuje používať mnoho rôznych štýlov vizualizácií
- Na rozdiel od matplotlib umožňuje jak jednoduché, tak pestrejšie metódy vizualizácie
- Veľmi dobre integrovaná s Pandas a dátovými rámcami
  - Seaborn vyžaduje mať nainštalované knižnice numpy, scipy, pandas a matplotlib
- Zdroje (dokumentácia a príklady)
  - https://seaborn.pydata.org
  - https://github.com/mwaskom/seaborn

Matplotlib vs Seaborn

- Hlavné rozdiely
  - matplotlib API je API na relatívne nízkej úrovni
  - Zložitejšie vizualizácie pomocou matplotlib sú komplikované, vyžadujú množstvo kódu, ktorý sa netýka priamo vizualizácie
  - matplotlib nie je stavaný pre prácu s Pandas dátovými rámcami – ak chceme vizualizovať dáta v Pandas, musíme extrahovať stĺpce a konvertovať do formátu vhodného pre matplotlib

Matplotlib vs Seaborn (2)

```
fig = plt.figure(figsize=(10,4))
title = fig.suptitle("Wine Type - Quality", fontsize=14)
fig.subplots_adjust(top=0.85, wspace=0.3)

ax1 = fig.add_subplot(1,2,1)
ax1.set_title("Red Wine")
ax1.set_xlabel("Quality")
ax1.set_ylabel("Frequency")
rw_q = red_wine['quality'].value_counts()
rw_q = (list(rw_q.index), list(rw_q.values))
ax1.set_ylim([0,2500])
ax1.tick_params(axis='both', which='major', labelsize=8.5)
bar1 = ax1.bar(rw_q[0], rw_q[1], color='red',
               edgecolor='black', linewidth=1)

ax2 = fig.add_subplot(1,2,2)
ax2.set_title("White Wine")
ax2.set_xlabel("Quality")
ax2.set_ylabel("Frequency")
ww_q = white_wine['quality'].value_counts()
ww_q = (list(ww_q.index), list(ww_q.values))
ax2.set_ylim([0,2500])
ax2.tick_params(axis='both', which='major', labelsize=8.5)
bar2 = ax2.bar(ww_q[0], ww_q[1], color='white',
               edgecolor='black', linewidth=1)
```

Matplotlib kód (hore) vs Seaborn kód (dole)

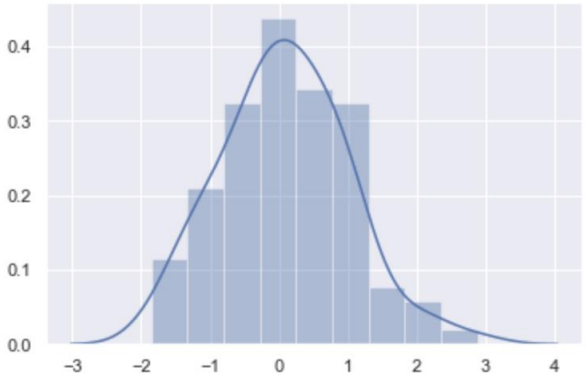
```
fig = plt.figure(figsize=(10, 7))

cp = sns.countplot(data=wines,
                   x="quality",
                   hue="wine_type",
                   palette={"red": "#FF9999", "white": "#FFE888"})
```

Typy vizualizácií v Seaborn

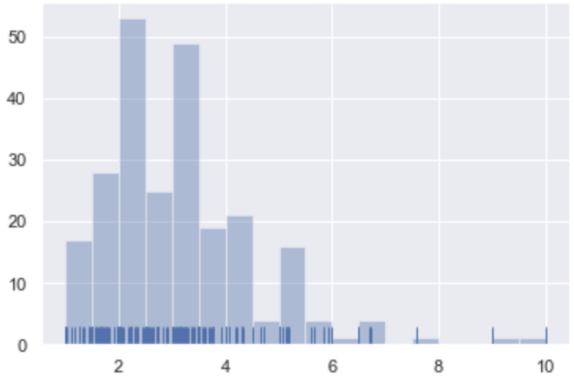
- Rôzne typy vizualizácií
  - Vizualizácie distribúcií hodnôt premenných (tzv. „univariate plots“ - histogramy, density plot)
  - Vizualizácie vzájomných závislostí 2 a viac atribútov spojitých premenných (bodové grafy, regresné grafy)
  - Vizualizácie vzájomných závislostí 2 a viac atribútov kategorických a spojitých (stĺpcové grafy, krabicové grafy)
  - Iné a kombinované vizualizácie
- V podstate ku každému poznáte ekvivalent v R, resp. ekvivalent existuje v niektorej z knižníc
  - preto nebudem rozoberať detaily vizualizácie a kód podrobne v prednáške – viac viď. Prednáška 2 + dosť podrobné cvičenia z Pythonu a im príslušné Jupyter notebooky

Vizualizácie distribúcií jednej premennej

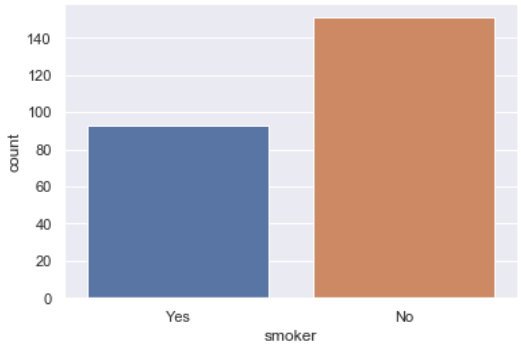
- Vizualizácie distribúcie hodnôt premenných – histogramy a ich rozšírenia
- distplot() pre numerické atribúty – histogram
  - používa hist() funkciu z matplotlib a rozširuje ju
  - KDE (Kernel Density Estimation) – odhad distribúcie hodnôt v dátach – density plot
  - "rug" plot – vykresľovanie dátových bodov na jednej z osí
- boxplot() – pre jednu premennú - boxplot pre zobrazenie sumarizácie 5 čísel
- countplot() – pre kategorické atribúty, početnosti hodnôt atribútu
- Binning – pre spojité premenné automatická/voliteľná diskretizácia na ordinálnu premennú
- Rôzne modifikácie parametrov vykresľovania a kombinácie premenných



Histogram + density (KDE)



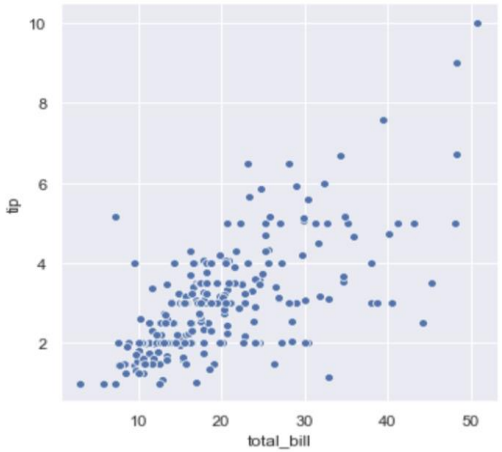
Histogram + rug



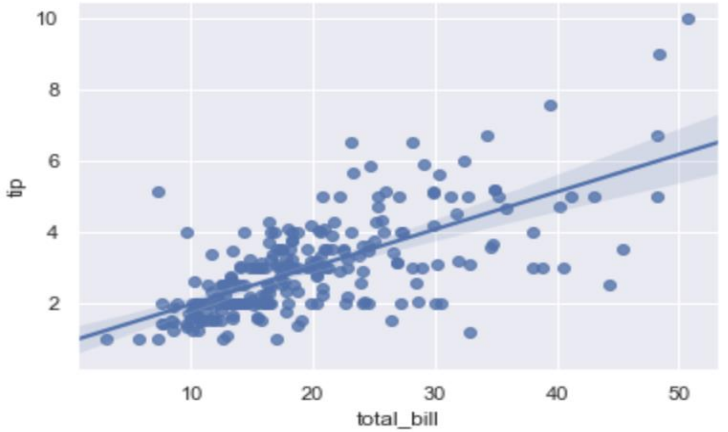
countplot

Bodové (a regresné) grafy

- Bodový graf (scatter plot) – funkcia scatterplot()
  - zobrazuje vzájomnú distribúciu hodnôt dvoch numerických premenných
  - osi x a y – premenné
  - každý bod zodpovedá jednému príkladu v dátach
- Regresný graf (regression plot) – funkcia regplot() + funckia lmplot()
  - pri vizualizácii vzájomnej závislosti hodnôt dvoch numerických premenných môže byť užitočné vizualizovať aj odhadnúť funkciu charakterizujúcu ich vzájomnú súvislosť
  - dopĺňa scatter plot o vykreslenie (napr. lineárnej) závislosti dvoch premenných
- Rôzne spôsoby kombinácie vykresľovania hodnôt atribútov aj vzhľadom na hodnoty iných a kategorických atribútov – kombinované grafy (viď. jointplot())



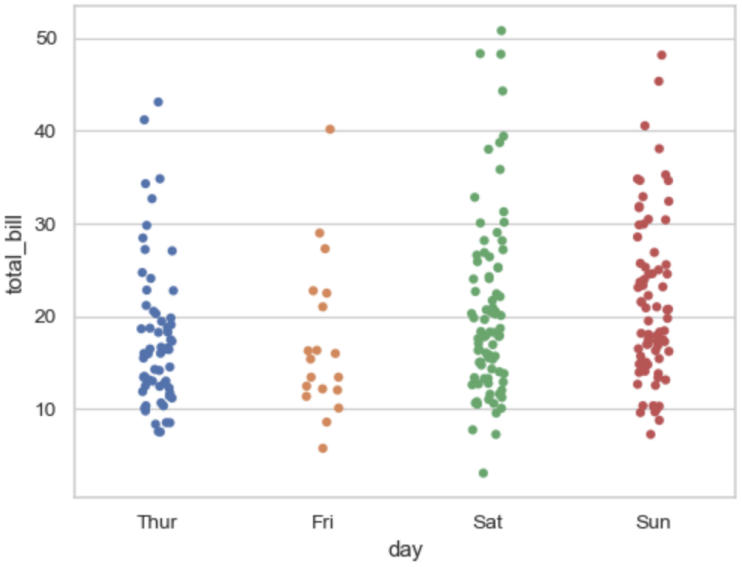
Scatter plot



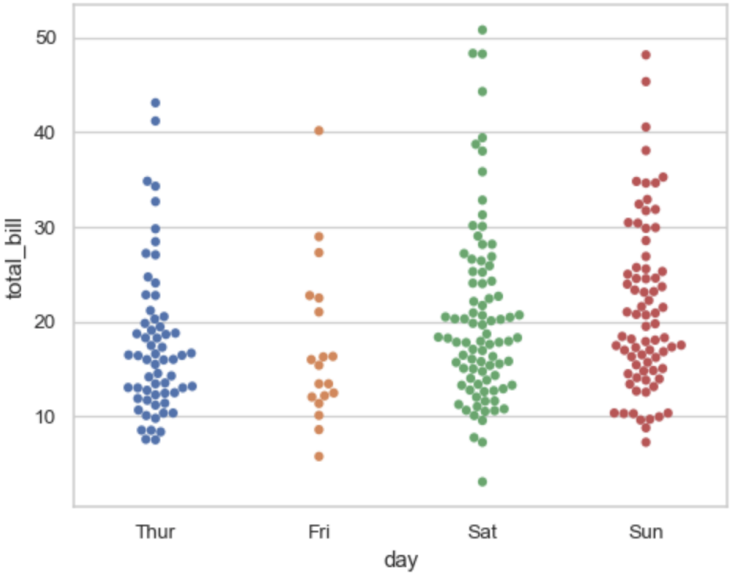
Regression plot

Bodové grafy pre kategorické premenné

- Vizualizáciu závislosti hodnôt dvoch atribútov, ak jeden z atribútov je kategorický
- Viacero spôsobov ako takéto dáta vizualizovať
- Strip plot – funkcia stripplot()
  - zobrazuje dáta podľa kombinácie hodnôt 2 atribútov
  - možné odhadnúť hustotu rozdelenia v rámci kombinácii hodnôt
- Swarm plot – funckia swarmplot() – body sa neprekrývajú, lepšia predstava o distribúcií hodnôt



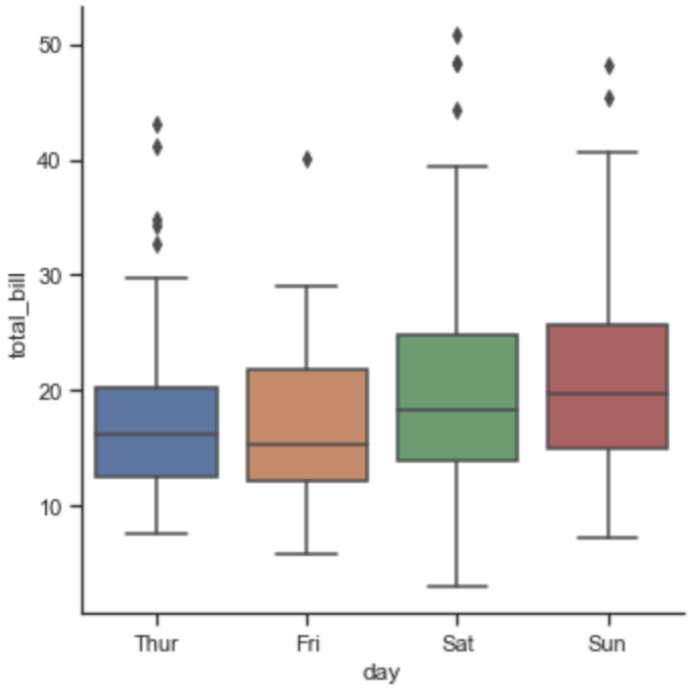
Strip plot

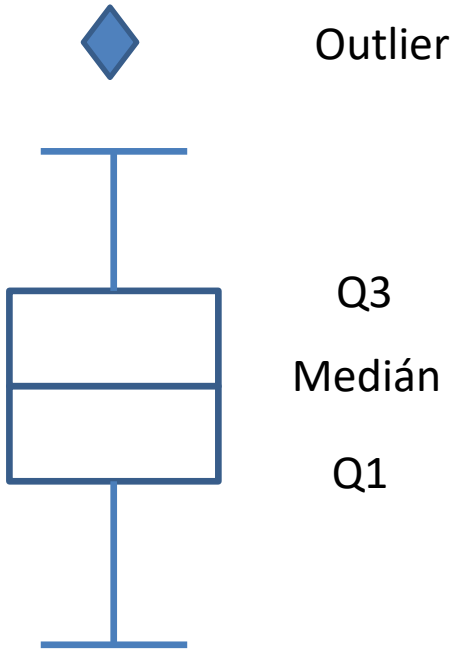


Swarm plot

Boxplot – pre viacrozmerný prípad

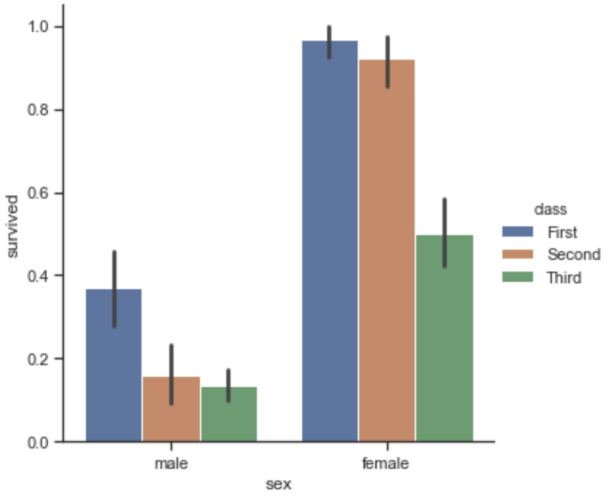
- Box plot (box and whisker plot) - krabicové grafy pre vizualizácie distribúcie hodnôt pre jednotlivé hodnoty kategorickej premennej



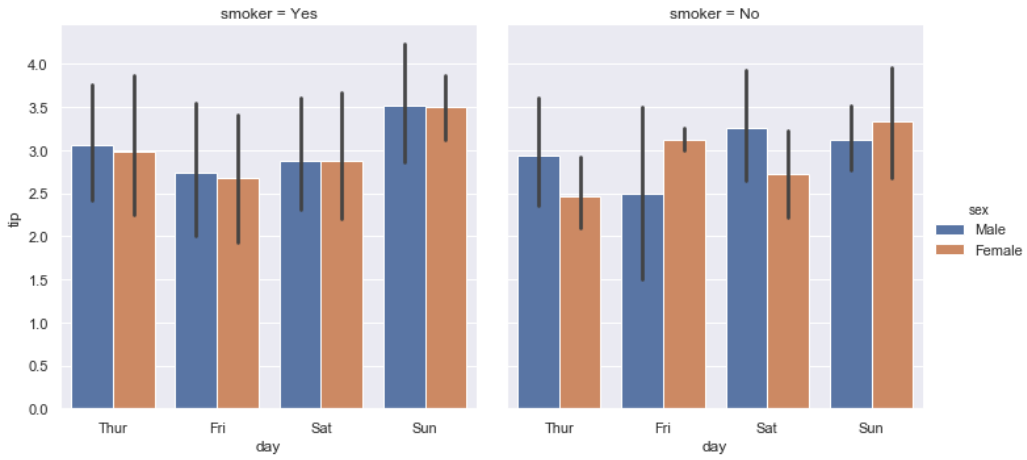


Barploty pre viacrozmerné prípady

- Bar plot – funkcia barplot() – stĺpcové grafy pre vizualizácie distribúcie hodnôt atribútu pre jednotlivé hodnoty kategorickej premennej, vrátane odhadu rozsahov hodnôt pre jednotlivé hodnoty kategorickej premennej
- Štandardne používa počítanie priemeru ako estimátor a štandardnú odchýlku
- Ak chceme ešte barploty podľa ďalšej kategorickej premennej zoskupiť – existuje kombinovaný graf – funkcia catplot()



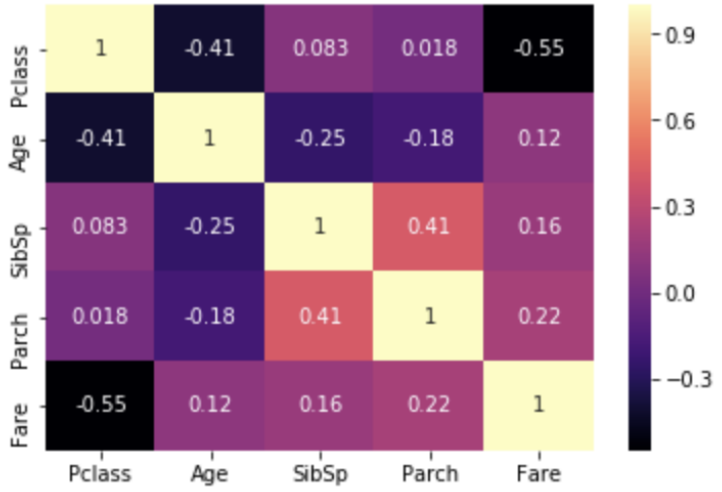
barplot

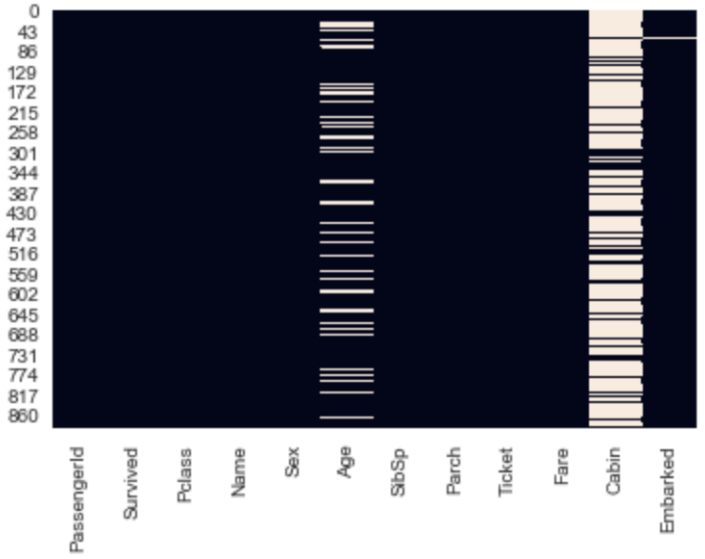


catplot

Heatmapy

- Funkcia heatmap() - využitie na rôzne vizualizácie, kde vieme dodatočnou informáciou (jasnosť alebo farba) vyjadriť rozdiely / porovnania medzi rôznymi prvkami datasetu, veľmi vhodne sa kombinuje aj s geografickými mapami
- Príklady
  - (vľavo) vizualizácia vzájomnej závislosti hodnôt premenných – vhodné pre vizualizáciu korelácií, kontigenčných tabuliek
  - (vpravo) možné využitie na iné účely, napr. vizualizácia chýbajúcich hodnôt v datasete





Kombinované vizualizácie

- Kombinácie viacerých premenných a aspektov v rámci zloženej vizualizácie
- Kombinácie rôznych typov vizualizácií
- Príklady
  - (vľavo) pair plot s rozdelením bodov podľa triedy v iris datasete
  - (vpravo) jointplot – okrem bodového grafu sú na okraji aj distribúcie hodnôt jednotlivých atribútov (scatterplot + distplot)