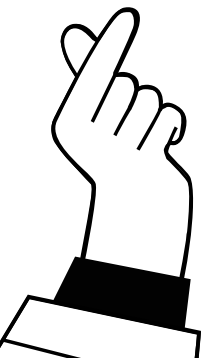




# PPZD

## semestrální práce

EMa Fantová



# What the data?!

Databáze Kaggle: Dataset „K-Pop Hits Through The Years“

90's – 2021 -> 2001 – 2020

top K-pop hity z každého roku

název písně, umělec + 11 kategorií -> 5 ponecháno

Danceability: Jak se skladba vhodná k taneční choreografii

závisí na kombinaci hudebních prvků, jako je tempo, stabilita rytmu...

0,0 je nejméně vhodná k tanci / 1,0 je nejvhodnější k tanci



Key: /tónina - Soubor všech tónů určité stupnice

Loudness: /hlasitost - Celková hlasitost skladby v decibelech (průměr)

Duration\_ms: /délka v ms - Doba trvání stopy v milisekundách

title	artist/s	danceability	key	loudness	tempo	duration_ms
I'm Coming (Feat. Tablo)	Rain	0.855000	6	-8.443000	125.486000	223893
O -正.反.合.	TVXQ!	0.741000	0	-2.259000	115.039000	255587
U	SUPER JUNIOR	0.791000	6	-3.769000	115.025000	226747





# Co s daty?

Pro hudební program „M countdown“

hudební show- živá vystoupení, hudební řebříčky...

Segment poohlédnutí do historie

analýza dat – jak se proměnily Hity v posledních 20 letech

oblíbení umělci, preferované parametry skladeb...



# Příprava dat

Stažení a nahrání dat na github  
Rozdělení do seznamů po 5 letech

```
urls1 = [  
    "https://raw.githubusercontent.com/emememinem/PPzadek/main/KPopHits2001.csv",  
    "https://raw.githubusercontent.com/emememinem/PPzadek/main/KPopHits2002.csv",  
    "https://raw.githubusercontent.com/emememinem/PPzadek/main/KPopHits2003.csv",  
    "https://raw.githubusercontent.com/emememinem/PPzadek/main/KPopHits2004.csv",  
    "https://raw.githubusercontent.com/emememinem/PPzadek/main/KPopHits2005.csv"]
```

Definice funkce „KPHurls“ na upravení dat  
nahrání do df, smazání sloupců, spojení  
„koke“ neobsahuje parametry, „VA“ neobsahuje jména umělců

```
def KPHurls(urls):  
    dfpole = []  
    for url in urls:  
        df = pd.read_csv(url)  
        df_bez = df.drop(["nub", "energy", "mode", "speechiness", "acousticness", "valence", "time_signature"], axis=1)  
        df_bez = df_bez[~df_bez["artist/s"].str.contains("코케", "Various Artists")]  
        dfpole.append(df_bez)  
    merged_dfpole = pd.concat(dfpole, ignore_index=True)  
    return merged_dfpole
```



# Top 10

Další funkce slouží pro vypsání top 10 umělců

V souboru se v jednom řádku vyskytuje více umělců

`str.split(",")` – rozdělí a výsledek je přiřadí do sloupce "artists\_list"

Následně se spočítá výskyt jednotlivých umělců

Funkce vrací pouze prvních deset výskytů

```
def top10(kph):  
    kph["artists_list"] = kph["artist/s"].str.split(",")  
    artist_counts = kph.explode("artists_list")["artists_list"].value_counts()  
    artist_countsTOP = artist_counts.head(10)  
    return artist_countsTOP
```



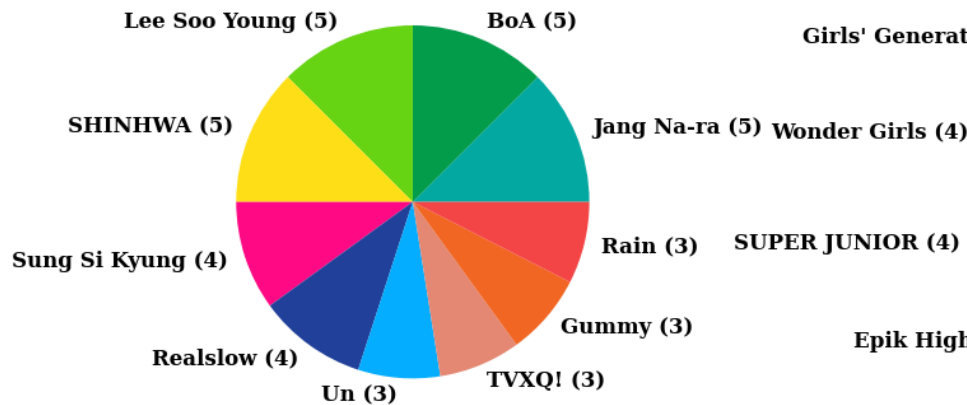
# grafy

Mřížka čtyř koláčových grafů  
nastavené vlastní barvy  
for/in prochází současně názvy a počty výskytů a páruje je

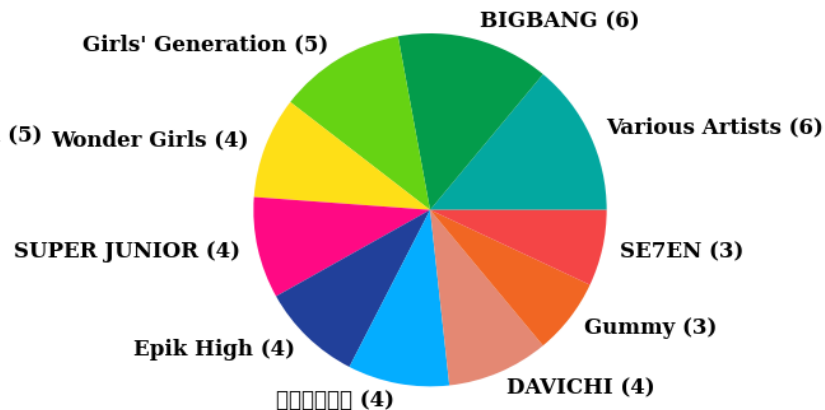
```
barvy = ["#03a8a0", "#039c4b", "#66d313", "#fedf17", "#ff0984", "#21409a", "#04adff", "#e48873", "#f16623", "#f44546"]
fig, axs = plt.subplots(2, 2, figsize=(10, 8)) # 2x2 mřížka grafu
axs[0, 0].pie(kph1top, labels=[f"{label} ({count})" for label, count in zip(kph1top.index, kph1top)], colors=barvy)
axs[0, 0].set_title("top 10 umělců z období 2001-2005")
axs[0, 1].pie(kph2top, labels=[f"{label} ({count})" for label, count in zip(kph2top.index, kph2top)], colors=barvy)
axs[0, 1].set_title("top 10 umělců z období 2006-2010")
axs[1, 0].pie(kph3top, labels=[f"{label} ({count})" for label, count in zip(kph3top.index, kph3top)], colors=barvy)
axs[1, 0].set_title("top 10 umělců z období 2011-2015")
axs[1, 1].pie(kph4top, labels=[f"{label} ({count})" for label, count in zip(kph4top.index, kph4top)], colors=barvy)
axs[1, 1].set_title("top 10 umělců z období 2016-2020")
```



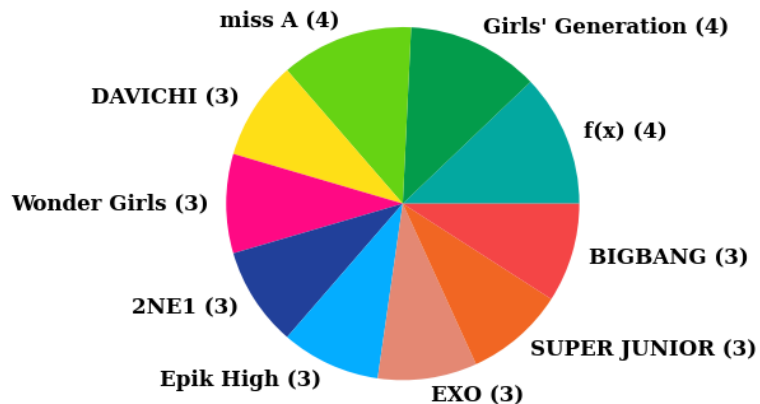
top 10 umělců z období 2001-2005



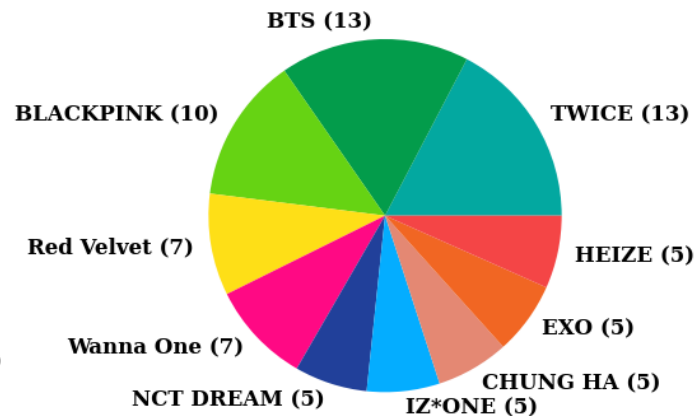
top 10 umělců z období 2006-2010



top 10 umělců z období 2011-2015



top 10 umělců z období 2016-2020





# Mediány/průměry

Aritmetický průměr: součet hodnot vydělený jejich počtem  
je ovlivněn extrémními hodnotami

Medián: prostřední hodnota ze seřazené posloupnosti hodnot  
není ovlivněn extrémními hodnotami

Funkce na spočítání průměrů a mediánů:  
výsledek: slovník „kategorie“ : „med“/“pru“


```
def mediany(kph):  
    kategorie = ["key", "danceability", "loudness", "tempo", "duration_ms"]  
    med = {} #slovník  
    for sloup in kategorie:  
        med[sloup] = kph[sloup].median()  
    return med  
  
def prumery(kph):  
    kategorie = ["key", "danceability", "loudness", "tempo", "duration_ms"]  
    pru = {} #slovník  
    for sloup in kategorie:  
        pru[sloup] = kph[sloup].mean()  
    return pru
```






# Extrémní hodnoty??

Cyklus na porovnání med a pru slovníků  
zjistit zda ne/má kategorie extrémní hodnoty  
->vhodné použít medián, nikoli průměr



```
def porovnaX(kph_med, kph_pru):  
    kategorie = ["key", "danceability", "loudness", "tempo", "duration_ms"] #co hodnotim  
  
    for kat in kategorie:  
        rozdil = abs(kph_med[kat] - kph_pru[kat]) #záporný hodnoty  
  
        if kat == "key": #if key==key  
            if rozdil > 0.2:  
                print("{} má extrémní hodnoty".format(kat)) #vytiskne ti tu kategorii  
            else:  
                print("{} nemá extrémní hodnoty".format(kat))  
        elif kat == "danceability":
```





~toť vše~