



Vilniaus Universitetas

Dimensijos mažinimas klasterizavime

Darbą atliko:

Vainius Gataveckas, Matas Gaulia, Dovydas Martinkus

Duomenų Mokslas

3 kursas 2 gr.

Vilnius, 2022

Turinys

1	Tikslas ir uždaviniai	3
2	Duomenų aibė	4
3	Atliktos analizės aprašymas.....	5
3.1	Pradinis apdorojimas	5
3.2	Klasterizavimas k-means metodu	6
4	Išvados	16
	Priedas	17

1 Tikslas ir uždaviniai

Tikslas: Išskirti tiriamoje duomenų aibėje klasterius bei apibrėžti susidariusių klasterių specifiką.

Uždaviniai:

Pasirinkti duomenų aibę klasterizavimui.

Pasirinkti, pagal kokius požymių rinkinius bus atliekamas klasterizavimas.

Įvertinti optimalų klasterių skaičių ir suklasterizuoti duomenis naudojantis tam tikru klasterizavimo algoritmu.

Suklasterizuoti ir vizualizuoti duomenis, gautus panaudojus dimensijos mažinimo algoritmą.

Pateikti susidariusių klasterių aprašomasias statistikas ir palyginti kas pasikeitė klasterizavus originalius duomenis, ir sumažinus dimensiją

Rasti kokios tendencijos būdingos kiekvienam klasteriui ir kuo vienas klasteris skiriasi nuo kito.

2 Duomenų aibė

Spotify Past Decades Songs duomenų aibė

Duomenų aibės šaltinis: Kaggle

Nuoroda per internetą: <https://www.kaggle.com/cnic92/spotify-past-decades-songs-50s10s?select=1990.csv>

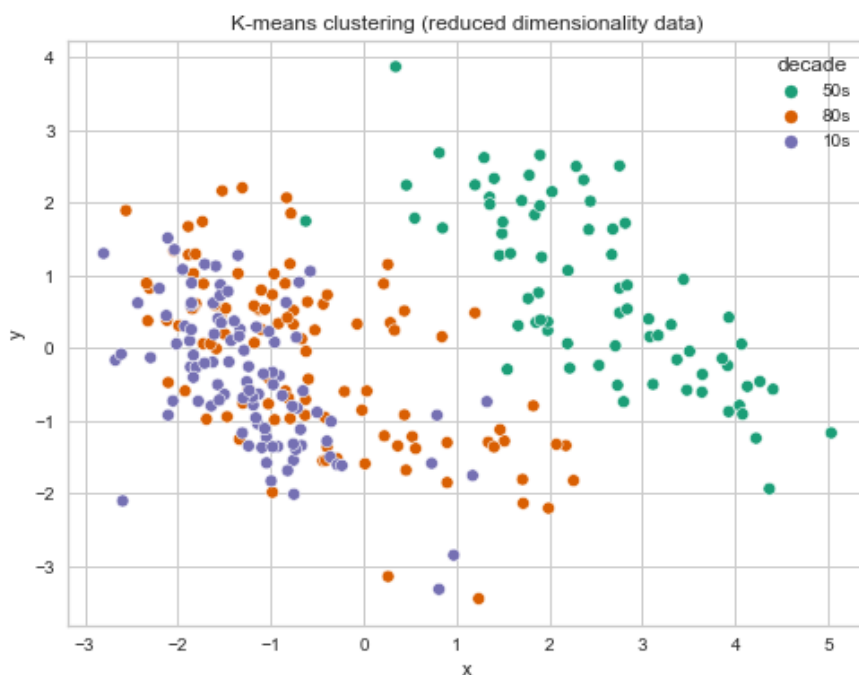
Duomenų aibę sudaro tokie požymiai:

- „Number“ – (kategorinis, nominalusis) dainą identifikuojantis kodas
- „Title“ – (kategorinis, nominalusis) dainos pavadinimas
- „Artist“ – (kategorinis, nominalusis) atlikėjas arba grupė
- „Top Genre“ – (kategorinis, nominalusis) dainos žanras
- „Year“ – (kiekybinis, diskretusis, intervalinė skalė) išleidimo metai
- „Decade“ – (kiekybinis, diskretusis, intervalinė skalė) išleidimo dešimtmetis
- „Tempo“ – (kiekybinis, tolydus, santykių skalė) dainos tempas
- „Loudness (dB)“ – (kiekybinis, tolydus, intervalų skalė) dainos garsumas
- „Duration“ – (kiekybinis, tolydus, santykių skalė) dainos trukmė
- „Energy“ – (kiekybinis, tolydus, santykių skalė) dainos energija
- „Danceability“ – (kiekybinis, tolydus, santykių skalė) lengvumas šokti pagal dainą
- „Liveness“ – (kiekybinis, tolydus, santykių skalė) kaip tikėtina, kad daina yra gyvas įrašas
- „Valence“ – (kiekybinis, tolydus, santykių skalė) dainos pozityvumas
- „Acousticness“ – (kiekybinis, tolydus, santykių skalė) dainos akustiškumas
- „Speechiness“ – (kiekybinis, tolydus, santykių skalė) kiek dainoje yra kalbama
- „Popularity“ – (kiekybinis, tolydus, santykių skalė) dainos populiarumas pagal perklausų skaičių

3 Atliktos analizės aprašymas

3.1 Pradinis apdorojimas

Duomenų aibę pasirinkta klasterizuoti pagal visus skaitinius požymius („Tempo“, „Loudness“, „Duration“, „Energy“, „Danceability“, „Liveness“, „Valence“, „Acousticness“, „Speechiness“, „Popularity“ n=10). Visų skaitinių požymių matavimo skalės suvienodintos standartizuojant juos pagal vidurkį ir dispersiją. Siekiant palyginti klasterizavimo rezultatus naudojant dimensijos mažinimo metodus ir jų nenaudojant, duomenų aibės požymių dimensija sumažinta iki 2 naudojantis pagrindinių komponentų analizės (angl. principal component analysis, toliau - PCA) metodu (žr. 1 pav.).



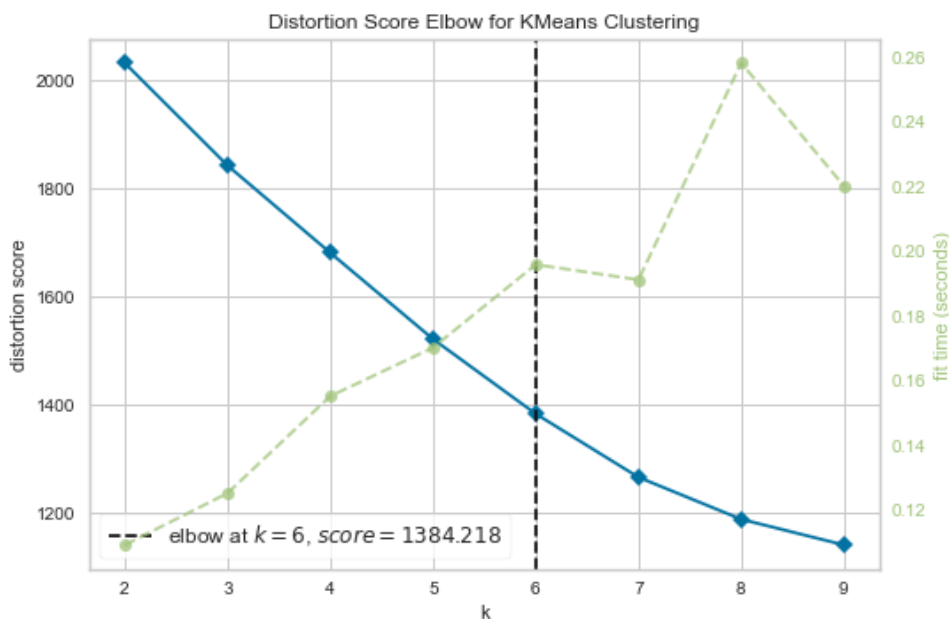
1 pav. Dimensijos sumažinimas naudojantis PCA metodu

3.2 Klasterizavimas k -means metodu

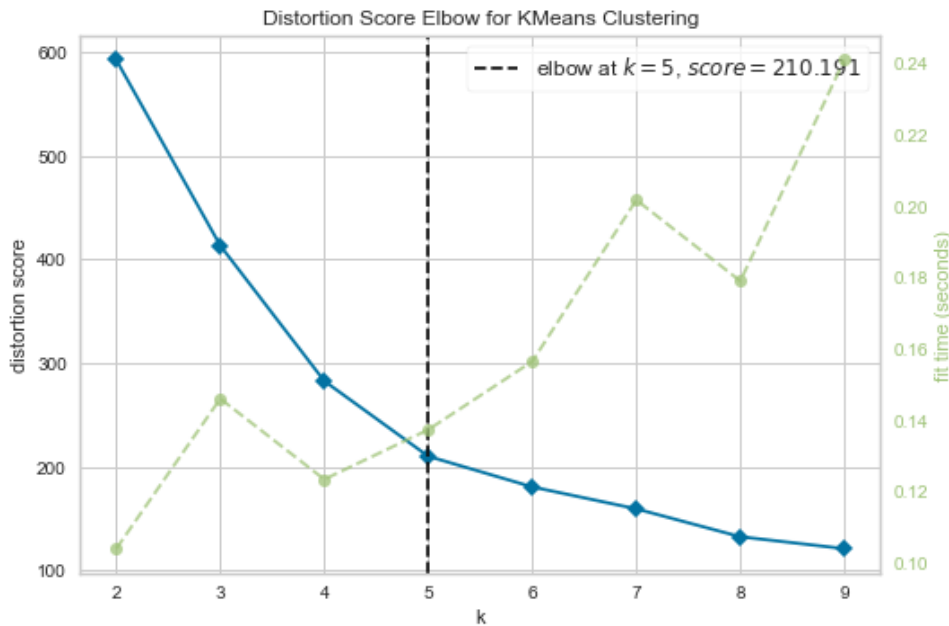
k -vidurkių metodas (angl. k -means) kiekvienam klasteriui apskaičiuoja centroidą (svorio centrą) ir kiekvieną objektą priskiria artimiausiam centroidui. Ši procedūra kartojama iki tol, kol tenkinamas sustojimo kriterijus. Centroidų (vadinasi ir klasterių) kiekis turi būti pasirenkamas iš anksto. Centroidai įprastai inicializuojami atsitiktinai. Šis klasterizavimo metodas lengvai interpretuojamas, gali būti taikomas dideliems duomenų kiekiams, tačiau yra netinkamas neiškilų formų klasteriams identifikuoti. Prasti klasterizavimo rezultatai gaunami ir turint skirtingo dydžio, tankio ar dispersijos tikruosius klasterius.

Optimaliam klasterių skaičiui rasti alkūnės metodu visiems klasteriams suskaičiuojamos ir į bendrą sumą sudedamos Euklidinių atstumų nuo klasterio vidurkio taško kvadratų sumos (angl. within cluster sum of squares). Siekiant surasti optimalų klasterių skaičių nubraižoma šio gauto dydžio kreivė pagal klasterių skaičių k ir ieškoma linkio taško (angl. elbow point).

Naudojant šį metodą, originalios dimensijos duomenyse nematoma aiški linkio vieta (žr. 2 pav.), tuo tarpu sumažintos dimensijos duomenyse pastebimas šiek tiek ryškesnis linkis ties $k=5$ (žr. 3 pav.). Dėl šios priežasties pasirinkta k -means metodu klasterizuoti duomenis į 5 klasterius tiek originalios, tiek sumažintos dimensijos duomenyse. Rezultatai vizualizuoti naudojant sumažintos dimensijos duomenis su klasterių žymomis, gautomis k -means algoritmui pateikiant atitinkamai originalios ir sumažintos dimensijos duomenis.

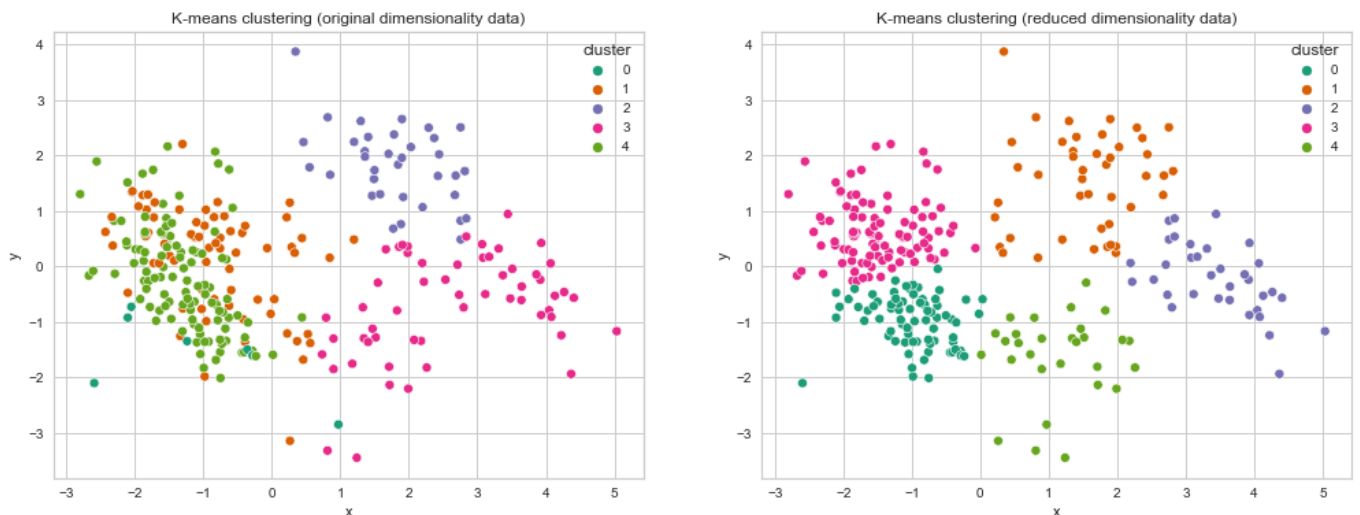


2 pav. Alkūnės (elbow) metodas originalios dimensijos duomenims



3 pav. Alkūnės (elbow) metodas sumažintos dimensijos duomenims

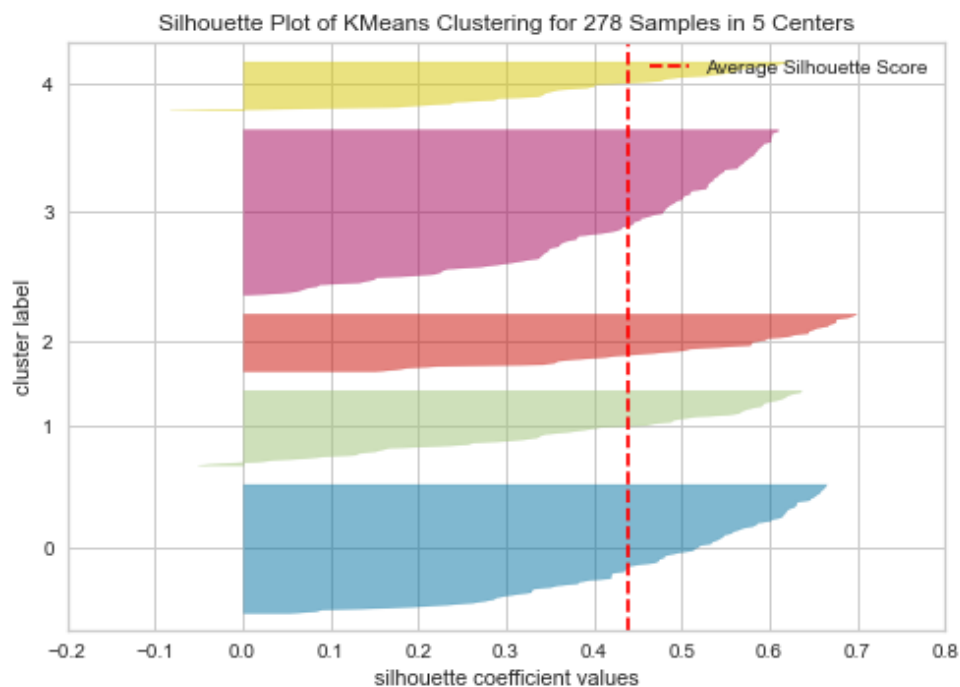
Originalios dimensijos duomenyse gauti klasteriai stipriai skiriasi nuo klasterių, gautų naudojant dimensijos mažinimo metodą, todėl klasteriai nėra stabilūs (žr. 4 pav.). Taip pat pastebėta, kad naudojant dimensijos mažinimą gaunami geresni klasterizavimo rezultatai - naudojant originalios dimensijos duomenis gaunamas itin mažas klasteris „0“, tačiau to yra išvengiama naudojant sumažintos dimensijos duomenis. Gautą geresnę klasterizavimo kokybę naudojant sumažintos dimensijos duomenis galime pamatyti ir silueto koeficiento reikšmių pasiskirstymo grafikuose (5 ir 6 pav.). Sumažintos dimensijos duomenyse klasterių dydžiai mažiau skiriasi tarpusavyje, silueto koeficiento reikšmės pasiskirsčiusios tolygiau.



4 pav. 5 klasteriai originalios ir sumažintos dimensijos duomenyse

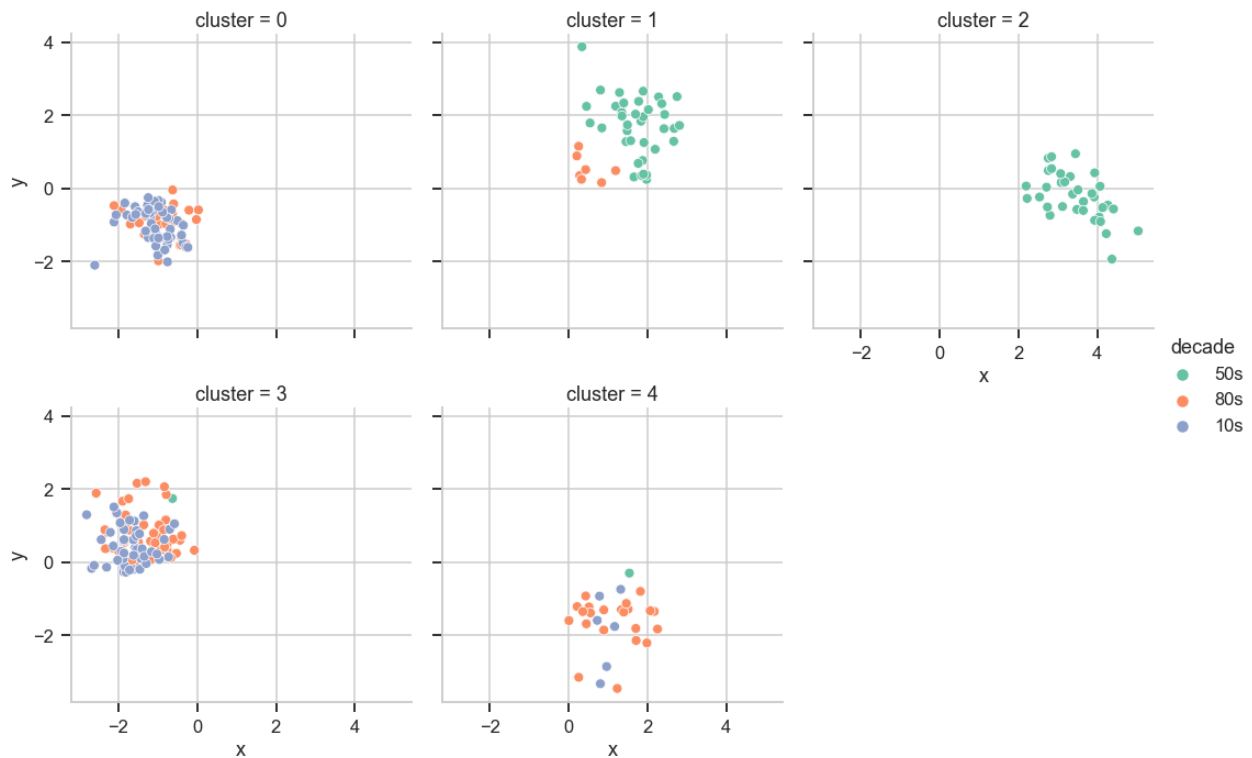


5 pav. Silueto koeficiento reikšmės originalios dimensijos duomenyse imant 5 klasterius



6 pav. Silueto koeficiento reikšmės sumažintos dimensijos duomenyse imant 5 klasterius

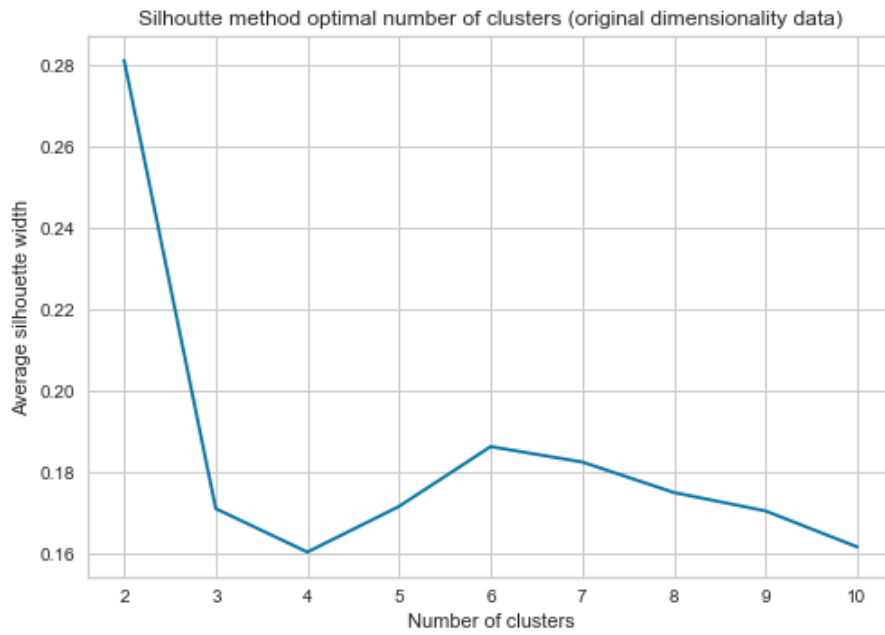
Lyginant sumažintos dimensijos duomenimis gautus klasterius pagal dainų išleidimo dešimtmetį rastos gana ryškios tendencijos (žr. 7 pav.). Klasteriams „1“ ir „2“ daugiausiai priklauso 50-ųjų dainos, tuo tarpu likusiems klasteriams priklauso 80-ųjų ir 2010-ųjų dainos.



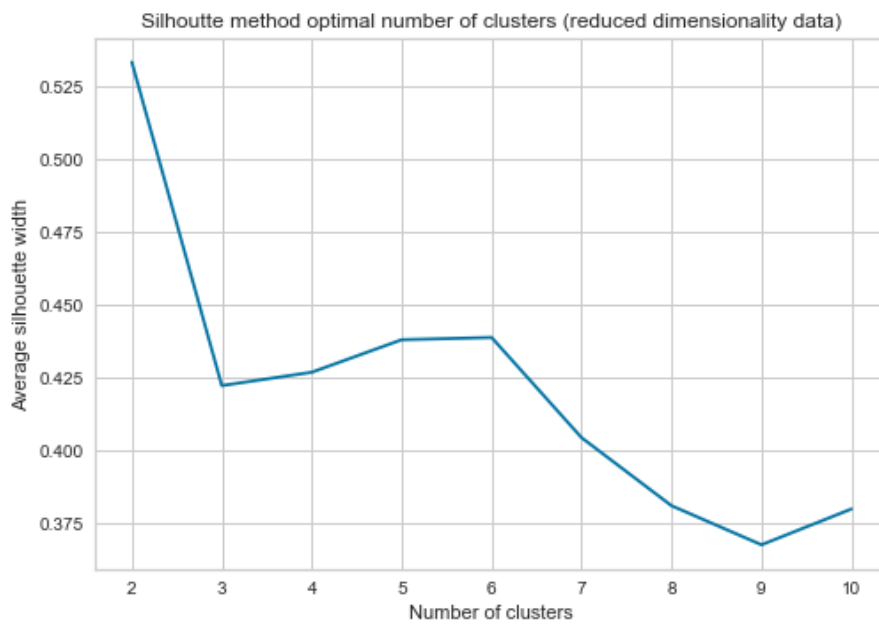
7 pav. Dainų dešimtmečiai pagal klasterį naudojant 5 klasterius

Silueto koeficientas lygina kiekvieno objekto panašumą su savo paties klasterio objektais lyginant su panašiausiu kito klasterio objektu. Optimalaus klasterių skaičiaus radimo metode, paremtame silueto koeficientais, kiekvienam klasterių skaičiui k skaičiuojamas vidutinis silueto koeficientas ir kaip optimalus klasterių skaičius pasirenkamas toks k , su kurio gaunama didžiausia šio dydžio reikšmė.

Tiek originalios, tiek sumažintos dimensijos duomenyse optimalus klasterių skaičius pagal vidutinį silueto koeficientą $k = 2$ (žr. 8 ir 9 pav.). Toks pat klasterių skaičius būtų pasirenkamas ir naudojantis empiriniu metodu, pagal kurį $k \approx \sqrt{\frac{n}{2}}$ (nes turima $n=10$, todėl $k \approx 2.23$).

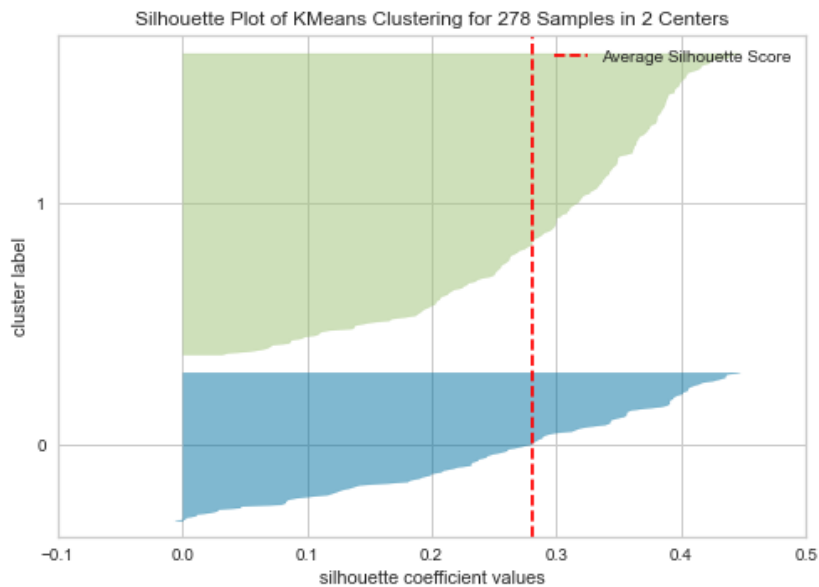


8 pav. Silueto metodas originalios dimensijos duomenims

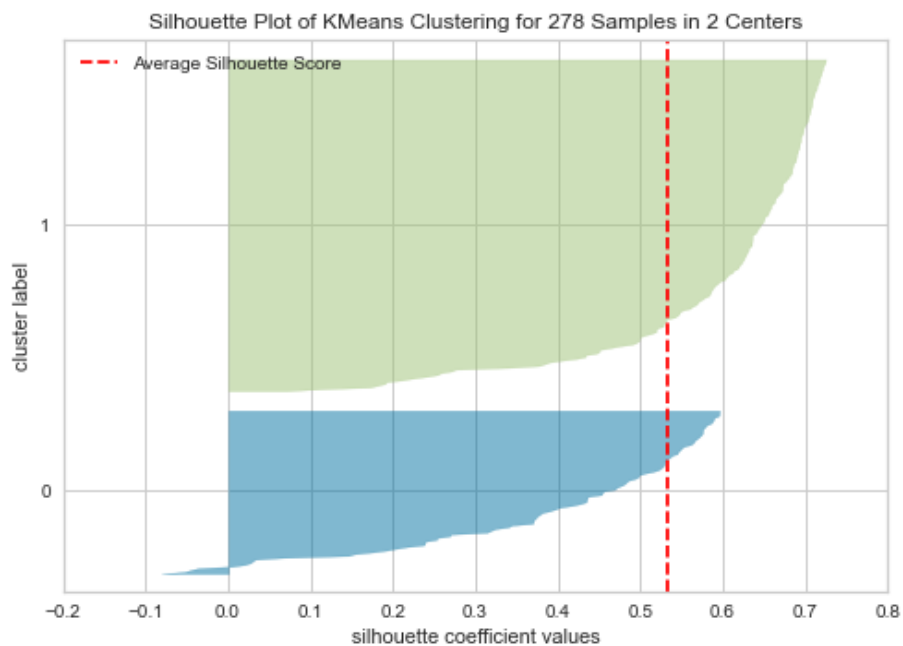


9 pav. Silueto metodas sumažintos dimensijos duomenims

Klasterizavimo kokybė vizualizuota vaizduojant individualių taškų silueto koeficiento reikšmių pasiskirstymą pagal klasterius (žr. 10 ir 11 pav.). Matoma, kad tik labai maža dalis taškų turi neigiamas silueto koeficiento reikšmes (yra išskirtys).

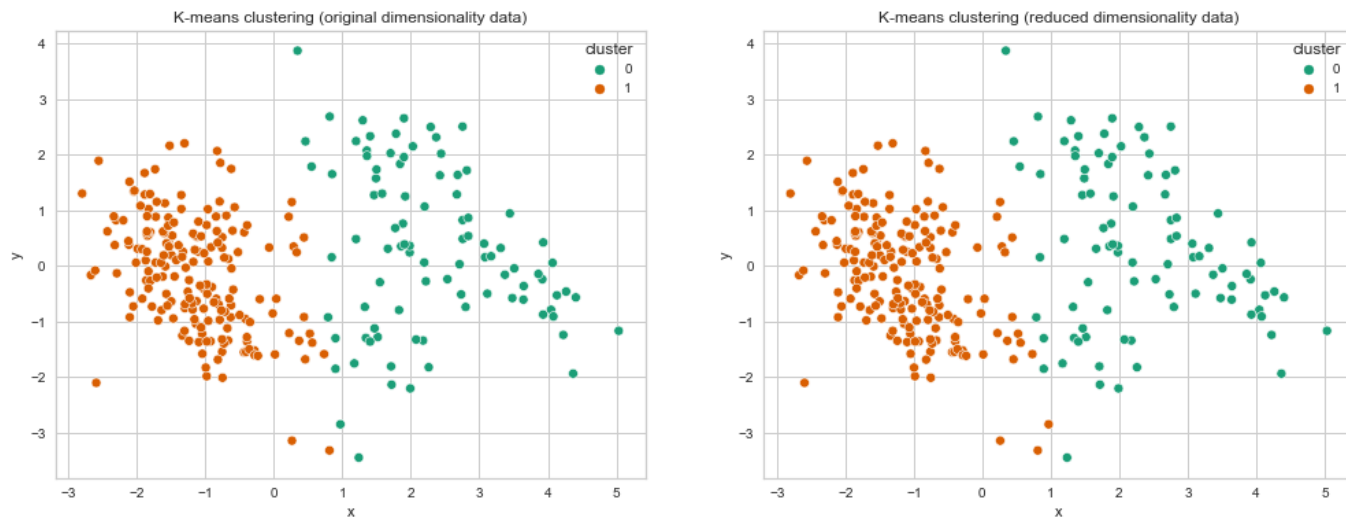


10 pav. Silueto koeficiento reikšmės originalios dimensijos duomenyse imant 2 klasterius



11 pav. Silueto koeficiento reikšmės sumažintos dimensijos duomenyse imant 2 klasterius

Gauti klasteriai stabilūs tarp originalios ir sumažintos dimensijos duomenų (skirtingas klasteris priskirtas tik vienam objektui) (žr. 12 pav.). Kadangi imant $k=2$ gaunami stabilūs klasteriai (be to šis klasterių skaičius gautas kaip optimalus naudojantis 2 iš 3 pasirinktų metodų), pasirinkta detaliau ištirti klasteriuose pasireiškiančias tendencijas imant šį klasterių skaičių.



12 pav. 2 klasteriai originalios ir sumažintos dimensijos duomenyse

Skirtingų požymių reikšmių vidurkių ir medianos reikšmės pagal gautus klasterius pateiktos lentelėse (žr. 1 ir 2 lentelės).

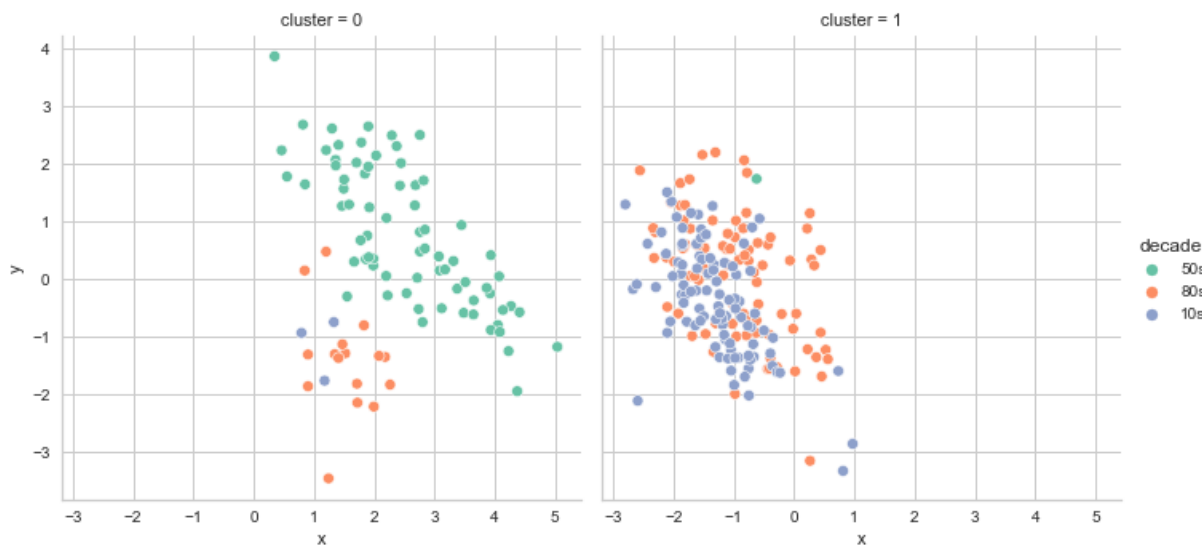
1 lentelė Požymių reikšmių vidurkiai skirtinguose klasteriuose

„0“ originalios dimensijos duomenyse	„0“ sumažintos dimensijos duomenyse	„1“ originalios dimensijos duomenyse	„1“ sumažintos dimensijos duomenyse	Požymis
113.2	113.42	120.68	120.53	tempo
34.01	34.08	70.05	69.83	energy
51.24	51.26	64.98	64.9	danceability
-11.99	-11.98	-6.72	-6.75	loudness
16.66	16.74	17.79	17.75	liveness
51.96	52.33	57.46	57.25	valence
170.92	170.04	232.78	232.88	duration
70.6	70.62	15.52	15.8	acousticness
4.3	4.02	6.55	6.67	speechiness
46.93	46.56	71.89	71.94	popularity

2 lentelė Požymių reikšmių medianos reikšmės skirtinguose klasteriuose

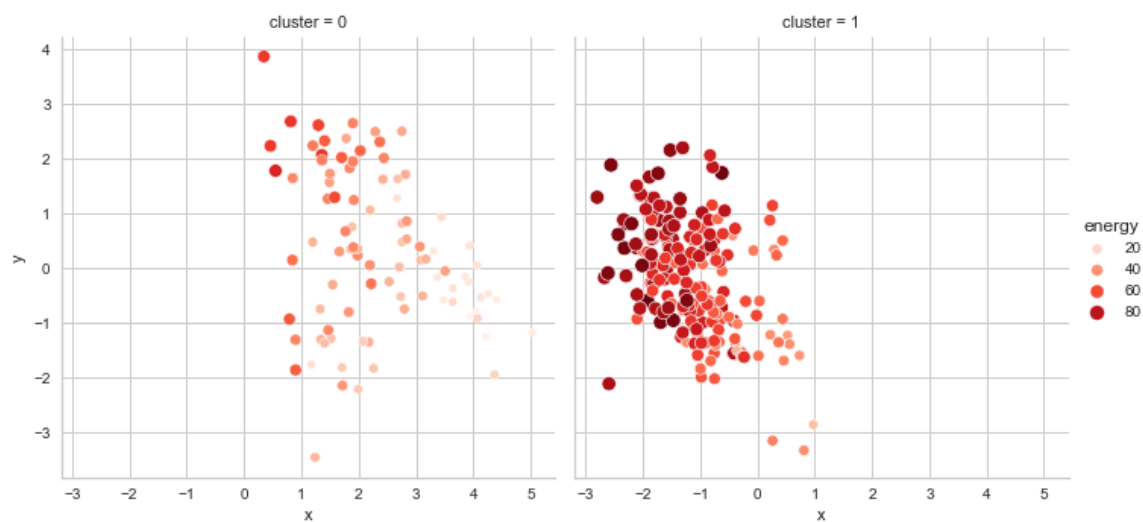
„0“ originalios dimensijos duomenyse	„0“ sumažintos dimensijos duomenyse	„1“ originalios dimensijos duomenyse	„1“ sumažintos dimensijos duomenyse	Požymis
108.5	109.0	120.5	120.0	tempo
32.0	32.0	71.0	71.0	energy
51.0	51.0	67.0	67.0	danceability
-11.5	-11.0	-6.0	-6.0	loudness
13.0	13.0	12.0	12.0	liveness
49.0	50.0	56.0	56.0	valence
155.5	155.0	222.5	223.0	duration
74.0	75.0	11.0	11.0	acousticness
3.0	3.0	4.0	4.0	speechiness
44.0	44.0	72.0	72.0	popularity

Tiriant dainų išleidimo dešimtmečio pasiskirstymą pagal klasterius rasta, kad pirmajame klasteryje dominuoja 50-ųjų dainos, tuo tarpu antrajame – 80-ųjų ir 2010-ųjų dainos (žr. 13 pav.).

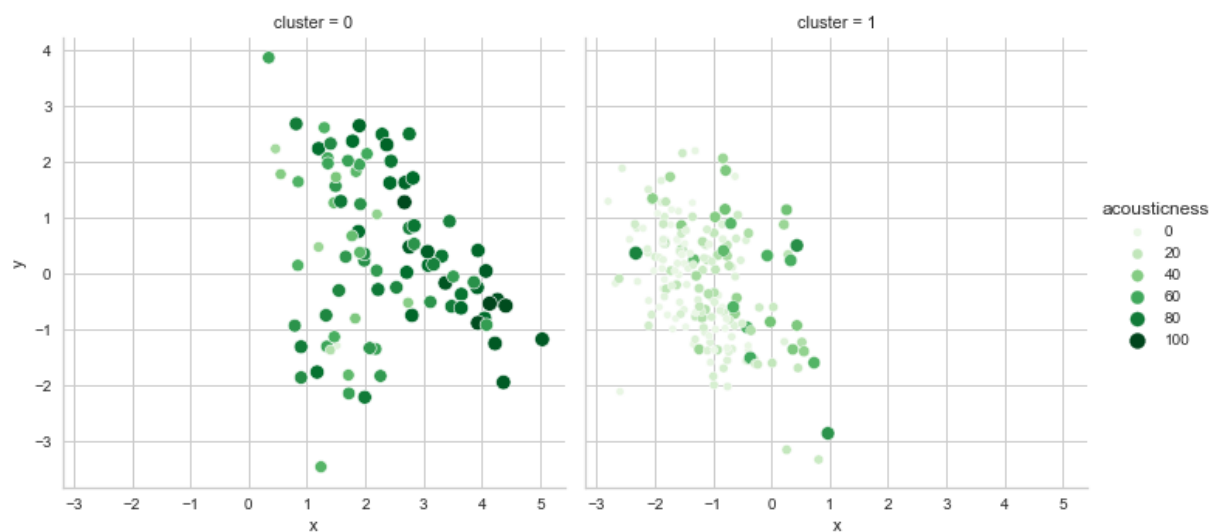


13 pav. Dainų dešimtmečiai pagal klasterį naudojant 2 klasterius

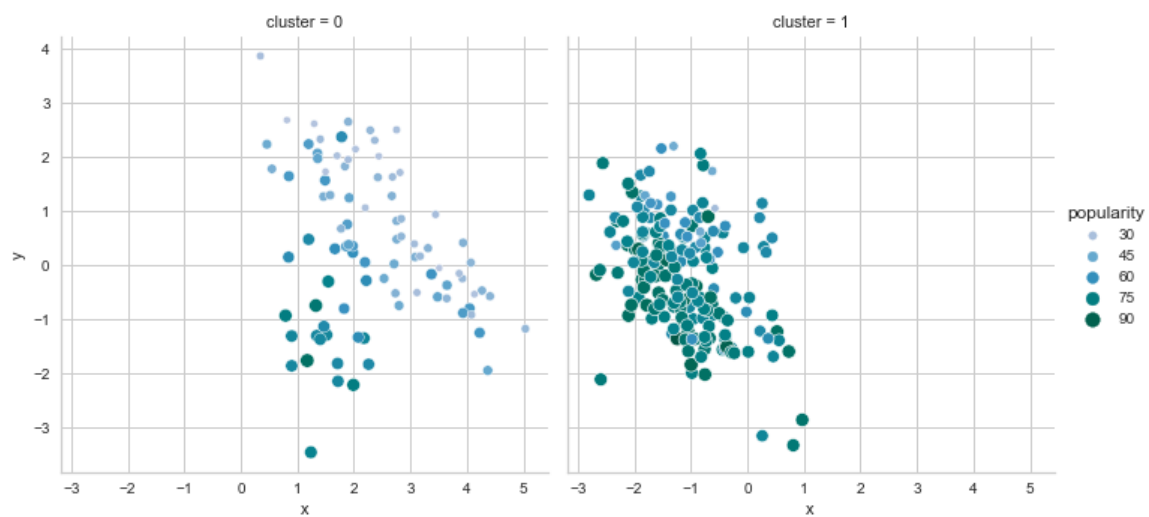
Lyginant klasterius pagal skaitinius požymius rasta, kad tiek pagal medianą, tiek pagal vidurkį, antrajam klasteriui priklauso žemo akustiškumo, didelės energijos ir didelio populiarumo dainos lyginant su pirmuoju klasteriu (žr. 14, 15 ir 16 pav.).



14 pav. Dainų energija pagal klasterį



15 pav. Dainų akustiškumas pagal klasterį



16 pav. Dainų populiarumas pagal klasterį

4 Išvados

Duomenų aibę sudaro skirtingų dešimtmečių dainos su skaitiniais požymiais apie šias dainas. Duomenys standartizuoti ir pasirinkta klasterizavimui naudoti visus skaitinius požymius ($n=10$). Siekiant palyginti klasterizavimo rezultatus gautus naudojant dimensijos mažino metodus ir jų nenaudojant, klasterizavimui naudojamų požymių dimensija sumažinta iki 2 naudojant PCA metodą.

Klasterizavimui pasirinkta naudoti k -vidurkių (angl. k -means) algoritmą. Optimalus klasterių skaičius naudojantis alkūnės metodu negautas vienareikšmiškai aiškus. Naudojant vieną iš galimų reikšmių $k=5$ gauti nestabilūs klasteriai – duomenų aibės objektų priskyrimas klasteriams stipriai skiriasi naudojant sumažintos dimensijos duomenis. Tarta, kad jeigu pasirenkamas šis klasterių skaičius, tai sumažinus požymių dimensiją gaunami geresni klasterizavimo rezultatai. Sumažintos dimensijos duomenimis gautuose klasteriuose rastos tendencijos objektams priklausyti tam tikram klasteriui pagal dainų išleidimo dešimtmečius.

Optimalus klasterių skaičius naudojantis empiriniu ir vidutinio silueto metodais gautas $k=2$. Šie klasteriai išlieka stipriai stabilūs imant originalios ir sumažintos dimensijos duomenis (skirtingas klasteris priskirtas tik vienam duomenų aibės objektui), todėl pasireiškiančios požymių tendencijos šiuose klasteriuose ištirtos detaliau.

Lyginant kokios tendencijos išryškėja sudarytuose klasteriuose rasta, kad didžioji dalis pirmajam klasteriui priklausančių dainų yra išleistos 50-aisias, tuo tarpu antrajam klasteriui priklauso beveik vien tik 80-ųjų ir 2010-ųjų dainos.

Pastebėta, kad antrajam klasteriui priklausančios dainos yra stipriai mažiau akustiškos, tačiau didesnės energijos ir didesnio populiarumo negu dainos, priklausančios pirmajam klasteriui.

Antrajam klasteriui priklausančios dainos taip pat yra vidutiniškai ilgesnės, garsesnės, labiau tinkamos šokti, greitesnio tempo, tačiau šios tendencijos nėra tokios ryškios kaip minėtos prieš tai.

Priedas

Žemiau pateiktas naudotas programinis kodas:

```
# #### Read-in the data

import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import seaborn as sns

def read_clean_data(filename):
    df = pd.read_csv(filename)[['title', 'artist', 'year', 'bpm', 'nrgy', 'dnce', 'dB', 'live', 'val',
    'dur', 'acous', 'spch', 'pop']]
    df =
df.rename({'bpm': 'tempo', 'nrgy': 'energy', 'dnce': 'danceability', 'dB': 'loudness', 'live': 'liveness',
    'val': 'valence', 'dur': 'duration', 'acous': 'acousticness', 'spch': 'speechiness', 'pop': 'popularity'},
            axis = 1)
    df['decade'] = filename[2:4] + 's'
    return df

from sklearn.preprocessing import StandardScaler

filenames = ['1950.csv', '1980.csv', '2010.csv']

df = pd.concat([read_clean_data(i) for i in filenames]).reset_index()
df_id = df[["title", "artist", "decade"]]
df = df.iloc[:, 4: len(df.columns) - 1]

# pozymiai, pagal kuriuos klasteriuojama
df.describe().T.drop("count", axis=1)

x = StandardScaler().fit_transform(df)

from sklearn.cluster import KMeans, AgglomerativeClustering, DBSCAN
from sklearn.manifold import MDS
from sklearn.decomposition import PCA

def do_mds(x, n_components=2, **kwargs):
    mds = MDS(n_components, **kwargs)
    x_trans = mds.fit_transform(x)

    return x_trans

def do_pca(x, n_components=2, **kwargs):
    pca = PCA(n_components, **kwargs)
    x_trans = pca.fit_transform(x)

    return x_trans
```

```

# klasterizavimas originaliems duomenims ir sumazinus dimensija
x_small = do_pca(x,n_components=2)

df_plot = pd.DataFrame(do_pca(x))
df_plot.columns = ["x","y"]


fig, ax = plt.subplots(1,1,figsize=(8, 6))
plot = sns.scatterplot(x="x",y="y",hue=df_id["decade"],data=df_plot,palette="Dark2",ax=ax)
plot.set_title("K-means clustering (reduced dimensionality data)")


# ##### KMeans


def do_kmeans(x, standartize = True,**kwargs):

    model = KMeans(**kwargs)
    pred = model.fit_predict(x)

    return pred, model


from yellowbrick.cluster import SilhouetteVisualizer, KElbowVisualizer

visualizer = KElbowVisualizer(KMeans(), k=(2,10))

visualizer.fit(x)
visualizer.show()


visualizer = KElbowVisualizer(KMeans(), k=(2,10))

visualizer.fit(x_small)
visualizer.show()


silhouette_scores = []
for i in list(range(2,11)):
    model = KMeans(i)
    visualizer = SilhouetteVisualizer(model, colors='yellowbrick')
    visualizer.fit(x)
    silhouette_scores.append(visualizer.silhouette_score_)

plt.clf()
plt.plot(range(2,11),silhouette_scores)
plt.title("Silhoutte method optimal number of clusters (original dimensionality data)")
plt.xlabel("Number of clusters")
plt.ylabel("Average silhouette width")


silhouette_scores = []
for i in list(range(2,11)):
    model = KMeans(i)
    visualizer = SilhouetteVisualizer(model, colors='yellowbrick')

```

```

visualizer.fit(x_small)
silhouette_scores.append(visualizer.silhouette_score_)

plt.clf()
plt.plot(range(2,11),silhouette_scores)
plt.title("Silhoutte method optimal number of clusters (reduced dimensionality data)")
plt.xlabel("Number of clusters")
plt.ylabel("Average silhouette width")

for i in [2,5]:
    model = KMeans(i,random_state=123)
    visualizer = SilhouetteVisualizer(model, colors='yellowbrick')

    visualizer.fit(x)
    visualizer.show()

for i in [2,5]:
    model = KMeans(i,random_state=123)
    visualizer = SilhouetteVisualizer(model, colors='yellowbrick')

    visualizer.fit(x_small)
    visualizer.show()

# elbow metodos
fig, ax = plt.subplots(1,2,figsize=(17, 6))
ax = ax.flatten()
df_plot["cluster"], _ = do_kmeans(x,n_clusters=5,random_state=123)
plot = sns.scatterplot(x="x",y="y",hue="cluster",data=df_plot,palette="Dark2",ax=ax[0])
plot.set_title("K-means clustering (original dimensionality data)")

df_plot["cluster"], _ = do_kmeans(x_small,n_clusters=5,random_state=123)
plot = sns.scatterplot(x="x",y="y",hue="cluster",data=df_plot,palette="Dark2",ax=ax[1])
plot.set_title("K-means clustering (reduced dimensionality data)")

# kuo skiriasi klasteriai pagal desimtmecius
sns.set_context("talk")
df_plot["decade"] = df_id["decade"]
sns.relplot(x="x",y="y",kind="scatter",data=df_plot,col="cluster",hue="decade",palette="Set2",col_wrap=
3)

# silhouette metodos
sns.set_context("notebook")
fig, ax = plt.subplots(1,2,figsize=(17, 6))
ax = ax.flatten()
df_plot["cluster"], _ = do_kmeans(x,n_clusters=2,random_state=123)
df["cluster_original"] = df_plot["cluster"]
plot = sns.scatterplot(x="x",y="y",hue="cluster",data=df_plot,palette="Dark2",ax=ax[0])
plot.set_title("K-means clustering (original dimensionality data)")

```

```
df_plot["cluster"], _ = do_kmeans(x_small, n_clusters=2, random_state=123)
df["cluster_reduced"] = df_plot["cluster"]
plot = sns.scatterplot(x="x", y="y", hue="cluster", data=df_plot, palette="Dark2", ax=ax[1])
plot.set_title("K-means clustering (reduced dimensionality data)")
```

```
def statistics(df):
    statistics_1 = df.groupby("cluster_original").describe().T
    statistics_1 = statistics_1.reset_index()[statistics_1.reset_index()["level_1"].isin(["50%"])]
    columns = list(statistics_1.columns)
    columns[0] = "variable"
    columns[1] = "statistic"
    for i in range(2, len(columns)):
        columns[i] = str(columns[i]) + "_original"
    statistics_1.columns = columns

    statistics_2 = df.groupby("cluster_reduced").describe().T
    statistics_2 = statistics_2.reset_index()[statistics_2.reset_index()["level_1"].isin(["50%"])]
    columns = list(statistics_2.columns)
    columns[0] = "variable"
    columns[1] = "statistic"
    for i in range(2, len(columns)):
        columns[i] = str(columns[i]) + "_reduced"
    statistics_2.columns = columns

    statistics = statistics_1.merge(statistics_2, how="inner")

    statistics = statistics.reindex(sorted(statistics.columns), axis=1)

    return statistics
```

```
# klasteriu palyginimas
print(df["cluster_original"].value_counts())
print(df["cluster_reduced"].value_counts())
```

```
statistics(df)
```

```
# kuo skiriasi klasteriai pagal desimtmečius
df_plot["decade"] = df_id["decade"]
sns.relplot(x="x", y="y", kind="scatter", data=df_plot, col="cluster", hue="decade", palette="Set2")
```

```
df_plot["energy"] = df["energy"]
sns.relplot(x="x", y="y", kind="scatter", hue="energy", size="energy", data=df_plot, col="cluster", palette="Reds")
```

```
df_plot["acousticness"] = df["acousticness"]
sns.relplot(x="x", y="y", kind="scatter", hue="acousticness", size="acousticness", data=df_plot, col="cluster", palette="Greens", hue_norm = (-10, 100))
```

```
df_plot["popularity"] = df["popularity"]
sns.relplot(x="x",y="y",kind="scatter",hue="popularity",size="popularity",data=df_plot,col="cluster",pa
lette="PuBuGn",
            hue_norm = (-10,100))
```