Kryteria oceny jakości grupowania

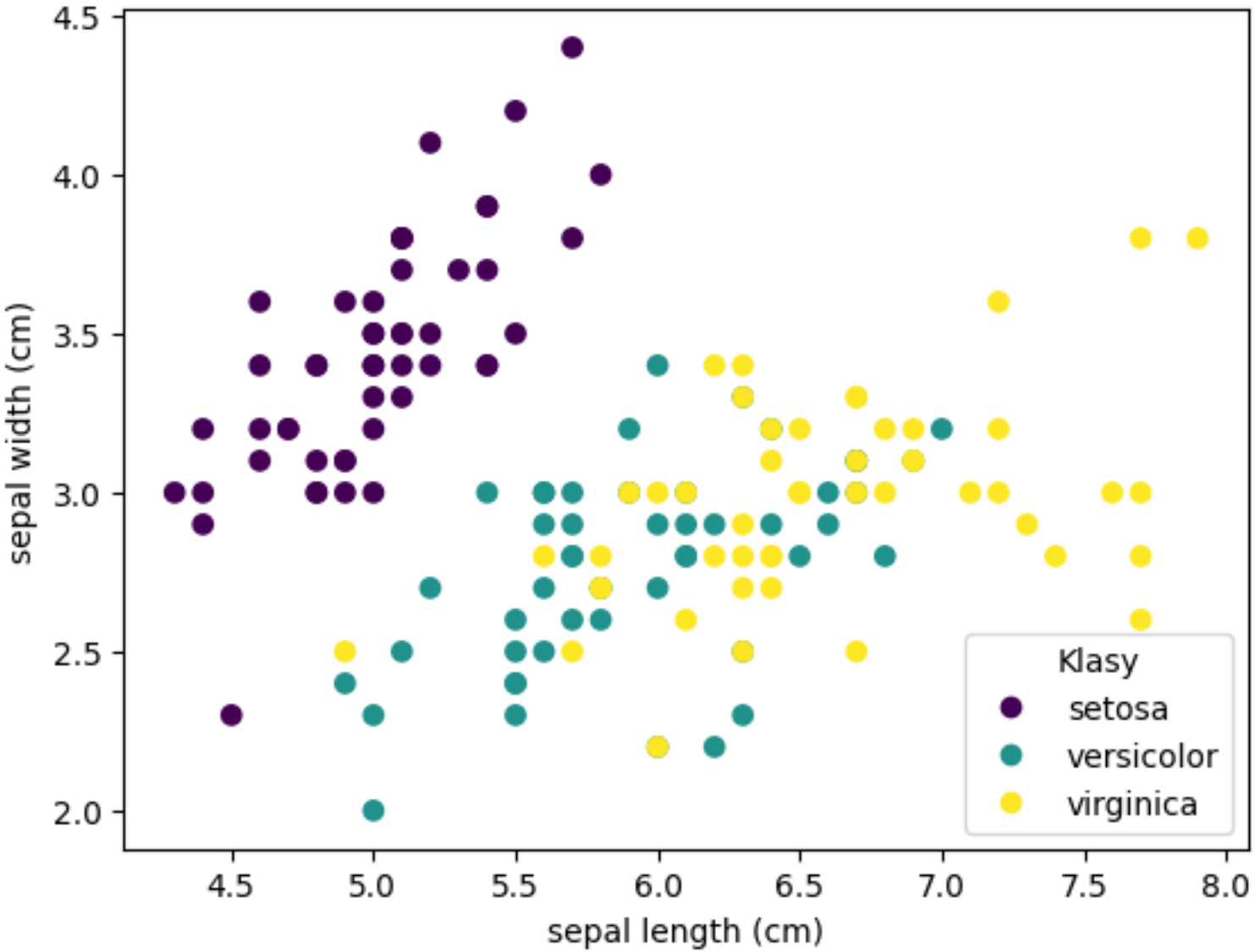
Adam Michalski

22 Kwietnia 2024

Problem oceny wyników klastrowania

Rozważmy pewne dane dla których dokonaliśmy podziału na k różnych grup zwanych klastrami. Stawiany problem polega na ocenie wyników klastrowania. W przypadku, gdy mamy dostęp do pożądanego podziału na klasy możemy próbować porównać otrzymane klastry z podziałem na klasy. Gdy jednak nie posiadamy wzorcowego podziału danych jesteśmy zmuszeni odwoływać się do pożądanych prawidłowości w klastrach.

PRZYPADEK ZE ZNANYM PODZIAŁEM NA KLASY

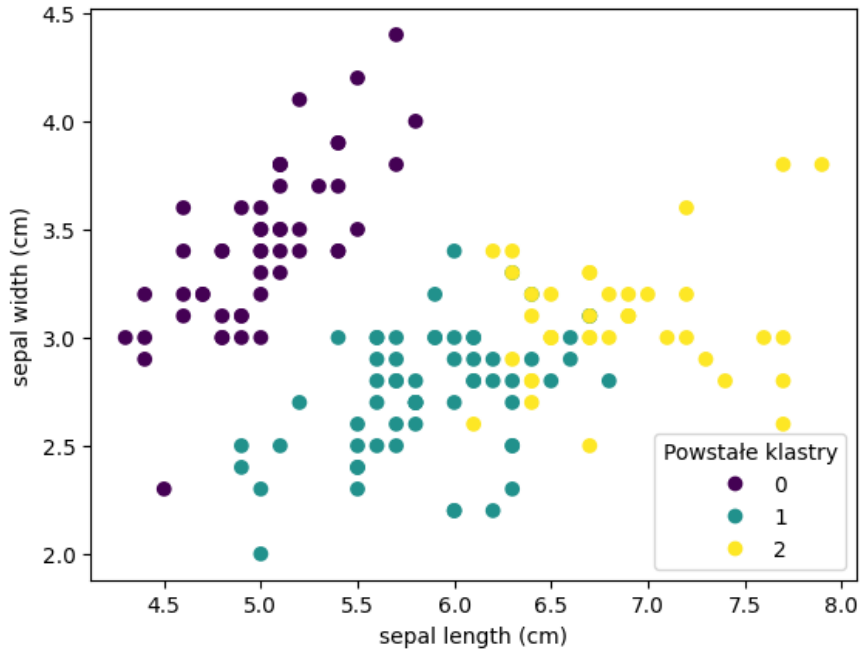


Przykładowe klastrowanie danych ze znanymi klasami o irysach:

```
df = pd.DataFrame(irysy.data, columns = irysy.feature_names)
df.head()
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

```
from sklearn.cluster import KMeans
klastrofikator = KMeans(n_clusters = 3, random_state=11)
model = klastrofikator.fit_predict(df, irysy.target)
_, ax = plt.subplots()
scatter = ax.scatter(irysy.data[:, 0], irysy.data[:, 1], c=model)
ax.set(xlabel=irysy.feature_names[0], ylabel=irysy.feature_names[1])
_ = ax.legend(scatter.legend_elements()[0], [0, 1, 2], loc="lower right", title="Powstałe klastry")
```



Dokładność dla metod klasyfikacji

Dokładnością klasyfikacji na podstawie danych o n wierszach wartości kolumny y o skali nominalnej zwracającej predykcję $\hat{y}$ nazywamy liczbę ze zbioru $[0, 1]$ daną wzorem

$$acc(y, \hat{y}) := \frac{1}{n} \sum_{i=1}^{n-1} \mathbb{1}(\hat{y}_i = y_i),$$

gdzie $\mathbb{1}$ jest indykatorem zbioru. Im większa dokładność tym lepiej.

Dokładność dla klastrowania

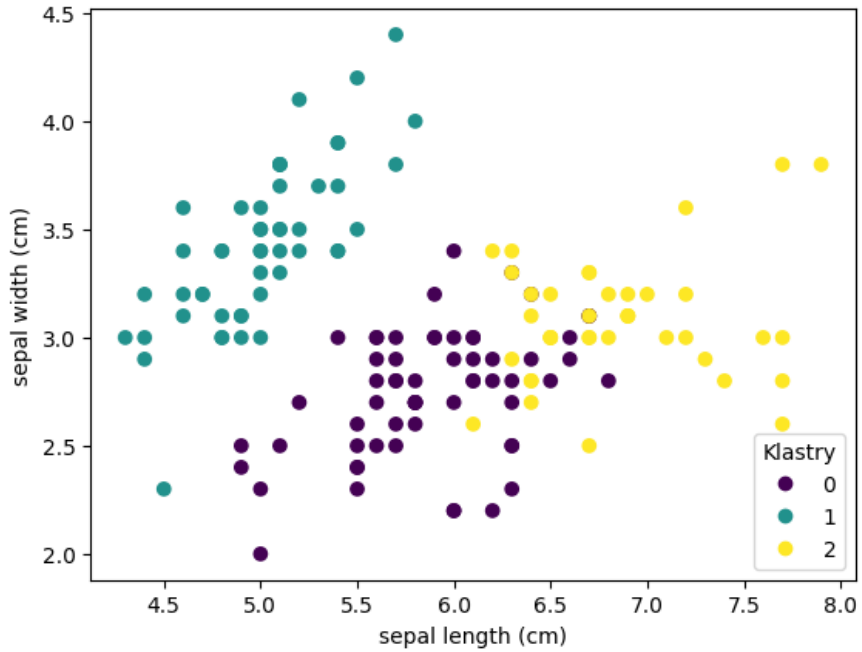
Dokładnością podziału na k grup na podstawie danych o n wierszach przy pożądanym podziale na klasy y zwracającej klastry $\hat{y}$ nazywamy liczbę ze zbioru $[0, 1]$ daną wzorem

$$acc(y, \hat{y}) := \max_{\sigma \in S_k} \frac{1}{n} \sum_{i=1}^{n-1} \mathbb{1}(\sigma(\hat{y}_i) = y_i),$$

gdzie $\mathbb{1}$ jest indykatorem zbioru oraz S_k jest grupą permutacji k elementowych. Im większa dokładność tym lepiej.

Wartość dokładności klastrowania może się różnić od dokładności klasyfikacji przy wyborze innej konwencji nazywania klastrów, co zilustrowałem poniżej zmieniając punkt początkowy algorytmu K średnich.

```
from sklearn.cluster import KMeans
klastrofikator = KMeans(n_clusters = 3, random_state=1)
klastry = klastrofikator.fit_predict(df, irysy.target)
_, ax = plt.subplots()
scatter = ax.scatter(irysy.data[:, 0], irysy.data[:, 1], c=klastry)
ax.set(xlabel=irysy.feature_names[0], ylabel=irysy.feature_names[1])
_ = ax.legend(scatter.legend_elements()[0], [0, 1, 2], loc="lower right", title="Klastry")
```




```
acc_1 = np.sum(irysy.target==klastry)/len(irysy.target)
acc_1
```

```
0.24
```

```
from sklearn.metrics import accuracy_score
acc_2 = accuracy_score(klastry, irysy.target)
acc_2
```

```
0.8866666666666667
```

Jednorodność klastrowania

Klastrowanie nazywamy jednorodnym, jeśli w każdym klastrze znajdują się jedynie elementy tej samej klasy. Trywialnym przykładem tego klastrowania jest podział n danych na n jednoelementowych klastrów, ponieważ w żadnym z klastrów nie ma elementów różnych klas (innych elementów wogóle nie ma).

Zupełność klastrowania

Klastrowanie nazywamy zupełnym, jeśli wszystkie elementy klasy należą do tego samego klastra. Trywialnym przykładem takiego klastrowania jest zawsze podział na 1 grupę, gdyż element żadnej z klas nie może trafić do innego klastra niż reszta elementów tej samej klasy (innych klastrów nie ma).

Miarą jednorodności μ_{jedn} oraz miarą zupełności μ_{zup} dla dowolnego klastrowania są liczby z przedziału $[0, 1]$ wyznaczone przy pomocy ilorazu entropii warunkowej między klasami, a klastrami oraz entropii w całych danych. Klastrowania jednorodne mają miarę jednorodności 1, a klastrowania zupełne mają miarę zupełności 1.

V- miara

V-miarą podziału na k grup na podstawie danych o n wierszach przy pożądanym podziale na klasy y zwracającej klastry $\hat{y}$ nazywamy liczbę ze zbioru $[0, 1]$ stanowiącą średnią harmoniczną z parametrem β miar jednorodności i zupełności, czyli

$$v = \frac{(1+\beta) \cdot \mu_{jedn} \cdot \mu_{zup}}{\beta \cdot \mu_{jedn} + \mu_{zup}}.$$

Im większa wartość V-miary tym lepsze klastrowanie. Parametr β pozwala na decydowanie w jakim stopniu uwzględniana ma być jednorodność, a na ile zupełność klastrowania (wartość standardowa to $\beta = 1$ przypisująca równe znaczenie tym miarom).

Wartość V-miary dla rozważanego klastrowania danych o irysach wynosi:

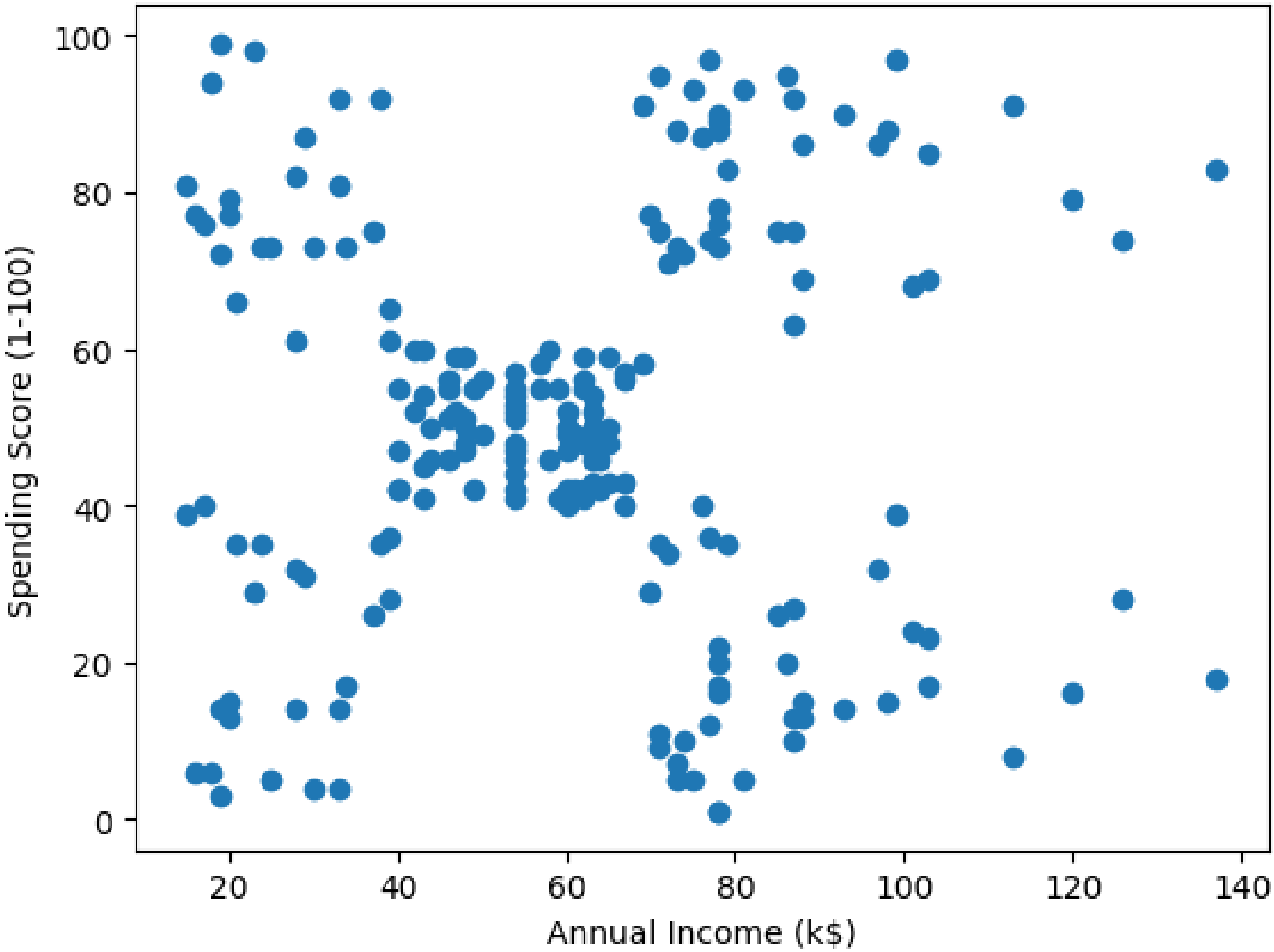
```
from sklearn.metrics import homogeneity_score, completeness_score
mu_jedn = homogeneity_score(klastry, irysy.target)
mu_zup = completeness_score(klastry, irysy.target)
v_1= 2*mu_jedn*mu_zup/(mu_jedn+mu_zup)
v_1
```

```
0.7419116631817836
```

```
from sklearn.metrics import v_measure_score
v_2= v_measure_score(klastry, irysy.target)
v_2
```

```
0.7419116631817836
```

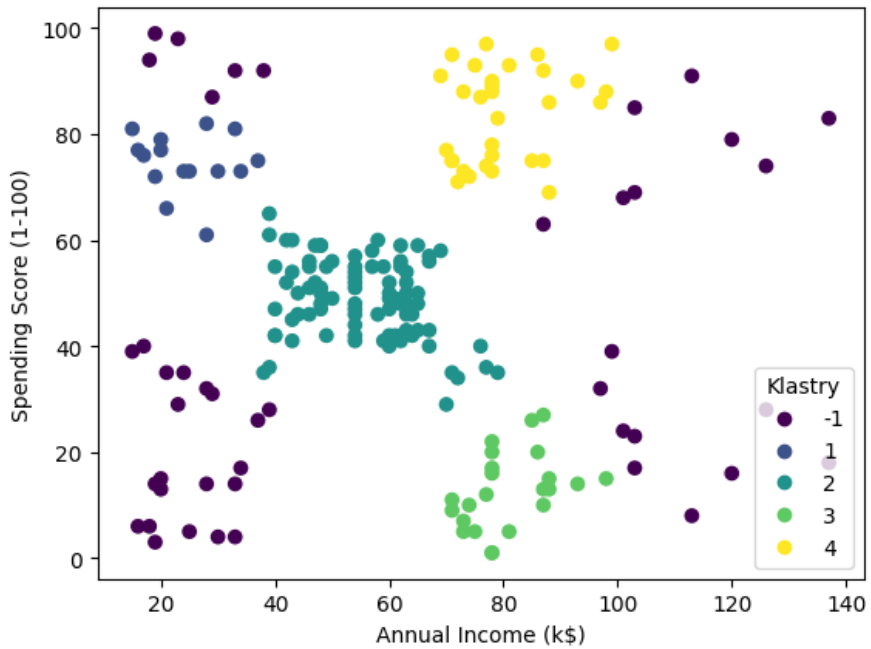
PRZYPADEK BEZ ZNANEGO PODZIAŁU NA KLASY



Przykładowe klastrowanie dla danych o zarobkach i wydatkach klientów pewnej firmy (Otrzymany klaster o numerze -1 można traktować jako klaster obserwacji odstających):

CustomerID	Annual Income (k$)	Spending Score (1-100)
1	15	39
2	15	81
3	16	6
4	16	77

```
from sklearn.cluster import DBSCAN
silhouette_score(df, klastry)
klastrofikator = DBSCAN(eps=13, min_samples=15)
klastry = klastrofikator.fit_predict(df)
_, ax = plt.subplots()
scatter = ax.scatter(df['Annual Income (k$)'], df['Spending Score (1-100)'], c=klastry)
ax.set(xlabel=df.columns[0], ylabel=df.columns[1])
_ = ax.legend(scatter.legend_elements()[0], [-1, 1, 2, 3, 4], loc="lower right", title="Klastry")
```



Aby wprowadzić pojęcie sylwetki potrzebujemy wprowadzić pewną metrykę d na zbiorze danych.

Sylwetka klastrów

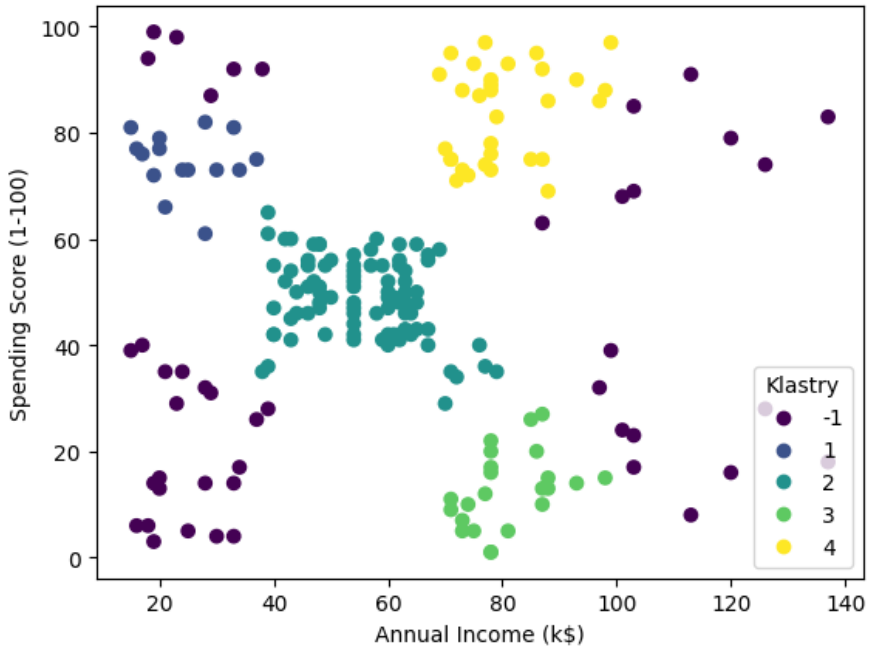
Rozważmy klastrowanie danych na k klastrów. Niech dla i-tego klastra ($i \in \{1, \ldots, k\}$) liczba a_i oznacza średnią odległość względem metryki d między elementami tego klastra oraz niech b_i oznacza minimalną odległość między pewnym elementem i-tego klastra, a elementem z innego klastra minimalizującymi wartość metryki d. Oznaczmy przez a średnią wartość liczb a_i oraz przez b średnią wartość liczb b_i. Wówczas sylwetką klastrowania dla próbki z danych nazywamy liczbę z przedziału $[-1, 1]$ daną wzorem

$$\frac{b - a}{max(b, a)}.$$

Rozważywszy średnią wartość sylwetki dla wielu próbek z danych S możemy wnioskować, iż gdy wartość ta jest bliska 1 klastrowanie jest skuteczne, ponieważ elementy leżące blisko siebie trafiają do tych samych klastrów. Jeśli wartość S jest bliska elementy różnych klastrów są blisko siebie, a gdy S jest liczbą bliską -1 elementy nawet elementy tych samych klastrów są od siebie znacząco oddalone.

Wartość sylwetki dla rozważanego klastrowania

Można podejrzewać, że ze względu na dużą odległość między elementami klastra -1 oraz małą odległość między elementami pozostałych klastrów wartość średniej sylwetki dla rozważanego klastrowania będzie liczbą większą od zera, ale znacząco mniejszą niż 1. Przy użyciu Jupytera otrzymujemy dokładną wartość oraz wykres sylwetek:




```
from sklearn.metrics import silhouette_score
sylwetka = silhouette_score(df, klastry)
sylwetka
```

```
0.367676415477805
```

Bibliografia


Evaluation of clustering.
https://www.youtube.com/watch?v=-jfHDMb7Ioc&t=1062s.


Silhouette score.
https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html#sklearn-metrics-silhouette-score.


V-measure score.
https://scikit-learn.org/stable/modules/generated/sklearn.metrics.v_measure_score.html#sklearn.metrics.v_measure_score.