به نام خدا

گزارش تمرین کلاسترینگ

کتابخانه ها

```
python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
```

۱. NumPy:

- **کاربردها**:

- ایجاد و ویرایش آرایه‌ها و ماتریکس‌های چند بعدی

- انجام عملیات منطقی و ریاضی روی آرایه‌ها

- گرفتن تبدیل فوریه، ایجاد اعداد تصادفی

- کاربرد در زمینه جبر خطی

۲. Pandas:

- **کاربردها**:

- خواندن و نوشتن داده‌های ذخیره شده به فرمهای (CSV, Excel, SQL و ...)

- پاک‌سازی، آماده‌سازی و انتقال داده‌ها

- آنالیز و تحلیل داده‌ها و مدل‌سازی از طریق ساختارهایی نظیر دیتافریم‌ها و سری‌ها

۳. Matplotlib:

- **کاربردها**:

- ایجاد طیف گسترده‌ای از نمودارها (خطی، میله‌ای، هیستوگرام و پراکندگی و ...)

- سفارشی‌سازی نمودارها با استفاده از عناوین، توضیحات و برچسب‌ها

- قابلیت ذخیره‌سازی نمودارها با فرمت‌های مختلف فایل

۴. Scikit-learn (KMeans):

- کاربردها:

- پیاده‌سازی الگوریتم‌های مختلف ماشین لرنینگ (طبقه‌بندی، رگرسیون، دسته‌بندی و ...)

- استفاده از KMeans برای خوشه‌بندی داده‌ها

۵. Scikit-learn (silhouette_score):

- کاربردها:

- ارزیابی کیفیت خوشه‌های ایجاد شده توسط الگوریتم‌های خوشه‌بندی

- محاسبه ضریب اعتبارسنجی سایه‌نما که میزان شباهت شی به خوشه خود و خوشه‌های دیگر را اندازه‌گیری می‌کند

- ارزیابی تعداد خوشه‌ها و مناسب بودن خوشه‌بندی

خواندن و آماده‌سازی داده‌ها

```
python
data = pd.read_csv('/content/Aggregation.txt', sep='\s+', header=None)
X = data.values
```

۶. pd.read_csv:

- برای خواندن مقادیر با کاما جدا شده فایل CSV به دیتافریم.

- مسیر قرارگیری فایل: `content/Aggregation.txt/`

- جداکننده: `s\+` (جداسازی کاراکترها با فاصله و خط جدید)

- `Header=None`: نشان می‌دهد فایل سر تیتر ندارد و pandas نباید از سطر اول به عنوان نام ستون‌ها استفاده کند.

۷. Data.values:

- دیتا را بصورت آرایه برمی‌گرداند و زمانی مورد استفاده قرار می‌گیرد که آرایه numpy به جای دیتافریم pandas استفاده شود.

خوشه‌بندی با KMeans

```
python
costs[] =
models[] =
silhouette_scores[] =
for k in range:(۱۱ ,۱)
  kmeans = KMeans(n_clusters=k, random_state=42)
  kmeans.fit(X)
  models.append(kmeans)
  costs.append(kmeans.inertia_)
  if k > 1:
    silhouette_scores.append(silhouette_score(X, kmeans.labels_))
```

۸. Costs:

- لیستی از اینرسی‌ها (مجموع مجذور فاصله نمونه تا نزدیک‌ترین مرکز دسته) برای هر مقدار k را ذخیره می‌کند.

۹. Models:

- مدل‌های kmeans را برای هر مقدار k ذخیره می‌کند.

۱۰. Silhouette_scores:

- لیست امتیازهای اعتبارسنجی برای kهای با مقدار بزرگتر از یک ذخیره می‌کند.

۱۱. For:

- تکرار حلقه برای k از ۱ تا ۱۰، که k نشان‌دهنده تعداد دسته‌بندی است.

۱۲. kmeans = KMeans(n_clusters=k, random_state=42):

- یک مدل kmeans با k دسته و مشخص کردن مقدار رندم ثابت برای تکرارپذیری

۱۳. kmeans.fit(x):

- انطباق مدل kmeans با داده x

۱۴. اینرسی:

- میزان اینرسی کمتر نشان‌دهنده دسته‌بندی بهتر است.

۱۵. محاسبه نمرات سایه‌نما:

- در صورت برقراری شرط k>1، نمرات هر دسته را محاسبه می‌کند. نمرات بالاتر نشان‌دهنده دسته‌بندی بهتر است.

رسم نمودار روش Elbow

```
python

plt.figure(figsize=(10, 6))

plt.plot(range(1, 11), costs, marker='o')

plt.title('Elbow Method for Optimal k')

plt.xlabel('Number of clusters (k)')

plt.ylabel('Cost (Inertia)')

plt.grid(True)

plt.show()
```

۱۶. figure:

- تصویر با سایز ۱۰ در ۶ اینچ ایجاد می‌کند.

۱۷. plot:

- نمودار دیتا را رسم می‌کند. `range` بر روی محور x تعداد دسته‌ها و `costs` هزینه هر دسته را بر روی محور y نشان می‌دهد. هر نقطه نمودار با `o` نمایش داده می‌شود.

۱۸. عنوان و برچسب‌ها:

- عنوان برای نمودار، برچسب برای محور x که تعداد دسته‌ها را نشان می‌دهد و برچسب محور y که میزان هزینه برای هر دسته است.

۱۹. grid:

- خط‌های در نمودار ایجاد می‌کند که مصورسازی و تفسیر داده را آسان‌تر می‌کند.

۲۰. plt.show:

- نمودار را نمایش می‌دهد.

انتخاب k بهینه

```
python

optimal_k = np.argmax(np.diff(costs)) + 1

print(f'Optimal k: {optimal_k}')
```

۲۱. خط اول:

- تفاوت بین عناصر متوالی را محاسبه می‌کند که کمک می‌کند نقطه عطف (Elbow) را شناسایی کند. `argmax` ماکزیمم مقدار تفاوت‌ها را برمی‌گرداند و با اضافه کردن عدد یک مقدار واقعی k را برمی‌گرداند.

۲۲. خط دوم:

- k بهینه را به عنوان خروجی برمی‌گرداند.

نمایش نتایج خوشه‌بندی

```
python
best_k = sorted(range(1, 11), key=lambda k: costs[k-1])[۵:]
plt.figure(figsize=(20, 10))
for i, k in enumerate(best_k):
    plt.subplot(2, 3, i + 1)
    plt.scatter(X[:, 0], X[:, 1], c=models[k-1].labels_, cmap='viridis')
    plt.title(f'k = {k}, Inertia = {costs[k-1]:.2f}')
    plt.xlabel('x')
    plt.ylabel('y')
plt.tight_layout()
plt.show()
```

۲۳. sorted:

- لیست مرتب شده‌ای از kها بر اساس مقدار هزینه ایجاد می‌کند. پنج تا دسته با پایین‌ترین اینرسی و بهترین دسته‌بندی را مشخص می‌کند.

۲۴. figure:

- تصویر با عرض ۲۰ و طول ۱۰ رسم می‌کند.

۲۵. حلقه for:

- حلقه با `i` برای شمارش تکرار و `k` به عنوان شاخص اجرا می‌شود.

۲۶. subplot:

- نمودار فرعی با سایز ۲x۳ (دو سطر و سه ستون) ایجاد می‌کند. `i+1` موقعیت نمودار را نشان می‌دهد.

۲۷. scatter:

- نمودار پراکندگی داده‌ها را رسم می‌کند. ستون اول و دوم X به عنوان x و y استفاده می‌شود. `c=models` نقاط را براساس برچسب دسته رنگ‌بندی می‌کند. `cmap='viridis'` نقشه رنگی استفاده می‌کند.

۲۸. برچسب و عنوان:

- مقدار k و مقدار اینرسی تا دو رقم اعشار برمی‌گرداند. برچسب محور x و y تنظیم می‌شود.

۲۹. plt.tight_layout :

- خطوط فرعی را به توجه به تصویر رسم می‌کند و از همپوشانی جلوگیری می‌کند.

۳۰. plt.show:

- نمودار را نمایش می‌دهد.

بخش دوم: خوشه‌بندی با DBSCAN

```
python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from sklearn.cluster import DBSCAN
from sklearn.metrics import mean_squared_error
```

۳۱. Axes3D:

- هدفش ساخت نمودارهای سه‌بعدی است.

۳۲. DBSCAN:

- الگوریتمی برای دسته‌بندی داده‌ها بر مبنای تراکم آنها

۳۳. mean_squared_error:

- ارزیابی عملکرد مدل‌های رگرسیون میانگین مجذور خطا بین مقادیر واقعی و پیش‌بینی شده.

مقادیر MinPts و epsilon:

```
python
min_pts_values = range(3, 15)
epsilon_values = np.linspace(0.1, 2.0, 10)
```

۳۴. MinPts:

- گرفتن مقادیر ۳ تا ۱۴ برای نقاط تراکم پذیر الگوریتم DBSCAN

۳۵. epsilon_values:

- لیست مقادیر فواصل شعاعی مختلف از ۰.۱ تا ۲.۰ با ۱۰ مقدار تولید شده.

ایجاد داده‌های مصنوعی

```
python

np.random.seed(۴۲)

X1 = np.random.rand(100, 2) * 10

X2 = np.random.rand(50, 2) * 10 + np.array([۲۵ ,۲۵])

X3 = np.random.rand(75, 2) * 10 + np.array([۰ ,۵۰])

X = np.vstack([X1, X2, X3])
```

۳۶. np.random.seed:

- مقدار ثابت برای تکرارپذیری نتایج تصادفی

۳۷. X1, X2, X3:

- سه دسته داده با مقادیر تصادفی در مکان‌های مختلف و تولید نمونه

۳۸. np.vstack:

- آرایه‌های سه دسته‌ای تولید شده را در یک مجموعه نهایی از داده‌ها ترکیب می‌کند.

خوشه‌بندی با DBSCAN

```
python

results[] =

for min_pts in min_pts_values:

    for epsilon in epsilon_values:

        db = DBSCAN(eps=epsilon, min_samples=min_pts)

        labels = db.fit_predict(X)

        n_clusters = len(set(labels)) - (1 if -1 in labels else 0)

        results.append((min_pts, epsilon, n_clusters))
```

۳۹. نتایج:

- مقادیر هر min_pts و ε و تعداد خوشه‌ها ذخیره می‌شود.

رسم نمودار سه‌بعدی نتایج

```
python
fig = plt.figure(figsize=(10, 6))
ax = fig.add_subplot(111, projection='3d')
for min_pts, epsilon, n_clusters in results:
    ax.scatter(min_pts, epsilon, n_clusters, color='b', alpha=0.6, edgecolors='w', s=50)
ax.set_xlabel('MinPts')
ax.set_ylabel('Epsilon')
ax.set_zlabel('Number of clusters')
ax.set_title('DBSCAN Clustering')
plt.show()
```

۴۰. figure:

- شکل D۳ برای مصورسازی داده‌ها ایجاد می‌کند.

۴۱. fig.add_subplot:

- افزودن نمودار فرعی به شکل ایجاد شده و تعیین اینکه این نمودار سه بعدی است.

۴۲. scatter:

- نمودار سه بعدی با پراکندگی داده‌ها از مقادیر ``MinPts، epsilon`` و `n_clusters` ایجاد می‌کند.

۴۳. alpha:

- میزان شفافیت نقاط را تنظیم می‌کند.

۴۴. s=50:

- سایز نقاط را تعیین می‌کند.

۴۵. set_xlabel, set_ylabel, set_zlabel:

- برچسب‌های محورها تنظیم می‌کند.

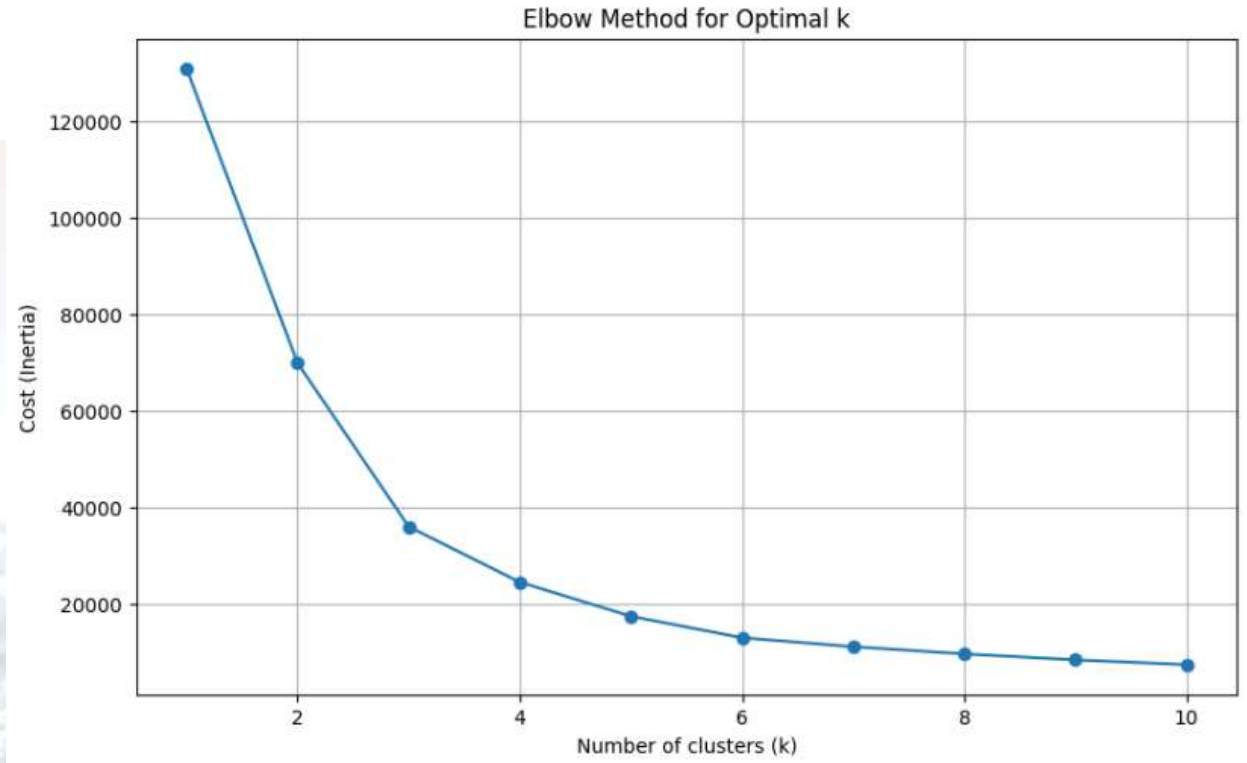
۴۶. set_title:

- عنوان برای نمودار تنظیم می‌کند.

۴۷. plt.show:

- نمایش نمودار.

تحلیل و نتیجه نمودارها



تحلیل نمودار Elbow Method

این نمودار روش Elbow برای یافتن تعداد بهینه‌ی خوشه‌ها در یک مسئله‌ی خوشه‌بندی را نشان می‌دهد. محور افقی تعداد خوشه‌ها (k) و محور عمودی هزینه یا اینرسی (Inertia) را نشان می‌دهد که نشان‌دهنده‌ی مجموع فواصل مربعات بین هر نقطه داده و مرکز خوشه است.

تحلیل نتایج:

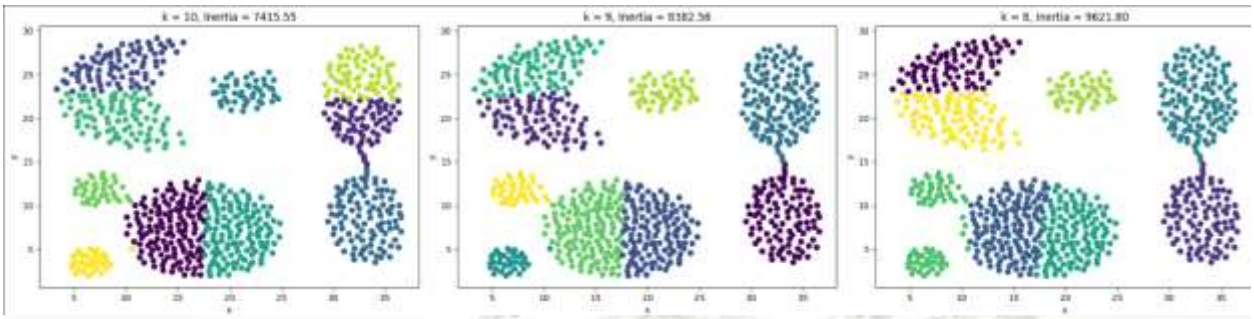
۱. **افت شدید در ابتدا** :برای مقادیر کوچک k (از ۱ تا ۳)، کاهش قابل توجهی در هزینه مشاهده می‌شود. این نشان می‌دهد که افزودن خوشه‌ها در این محدوده بهبود زیادی در کیفیت خوشه‌بندی ایجاد می‌کند.

۲. **الگوی زانو :(Elbow)** نقطه‌ای که تغییرات هزینه به صورت قابل توجه کاهش می‌یابد و سپس روند کاهش هزینه ثابت‌تر می‌شود، به عنوان "نقطه زانو" یا Elbow شناخته می‌شود. در این نمودار، این نقطه بین k=3 و k=4 قرار دارد.

۳. **کاهش ثابت پس از زانو** :بعد از$k=4$ ، کاهش هزینه به صورت یکنواخت‌تر و با نرخ کمتر ادامه می‌یابد.

نتیجه‌گیری:

با توجه به روشElbow ، تعداد بهینه‌ی خوشه‌ها برای این داده‌ها می‌تواند ۴ باشد. این نقطه جایی است که افزودن خوشه‌های بیشتر باعث کاهش هزینه به میزان قابل توجهی نمی‌شود و به نوعی بهترین تعادل بین تعداد خوشه‌ها و هزینه‌ی خوشه‌بندی را فراهم می‌کند.



تحلیل نمودارهای خوشه‌بندی

این تصویر شامل سه نمودار خوشه‌بندی با تعداد خوشه‌های متفاوت (k) است که با استفاده از الگوریتم K-Means به‌دست‌آمده‌اند. هر نمودار تعداد خوشه‌ها (k) و مقدار اینرسی مربوطه را نشان می‌دهد.

تحلیل نمودارها:

۱. **k = 10، Inertia = 7415.55:**
- در این نمودار، ۱۰ خوشه شناسایی شده‌اند.
- مقدار اینرسی ۷۴۱۵.۵۵ است که نسبت به سایرk ها کمتر است.
- داده‌ها به خوشه‌های کوچک‌تر و متمرکز تقسیم شده‌اند.

۲. **k = 9، Inertia = 8382.56:**
- در این نمودار، ۹ خوشه شناسایی شده‌اند.
- مقدار اینرسی ۸۳۸۲.۵۶ است که نسبت به k=10 بیشتر است.
- برخی از خوشه‌ها بزرگ‌تر و برخی دیگر کوچک‌تر از خوشه‌های k=10 هستند.

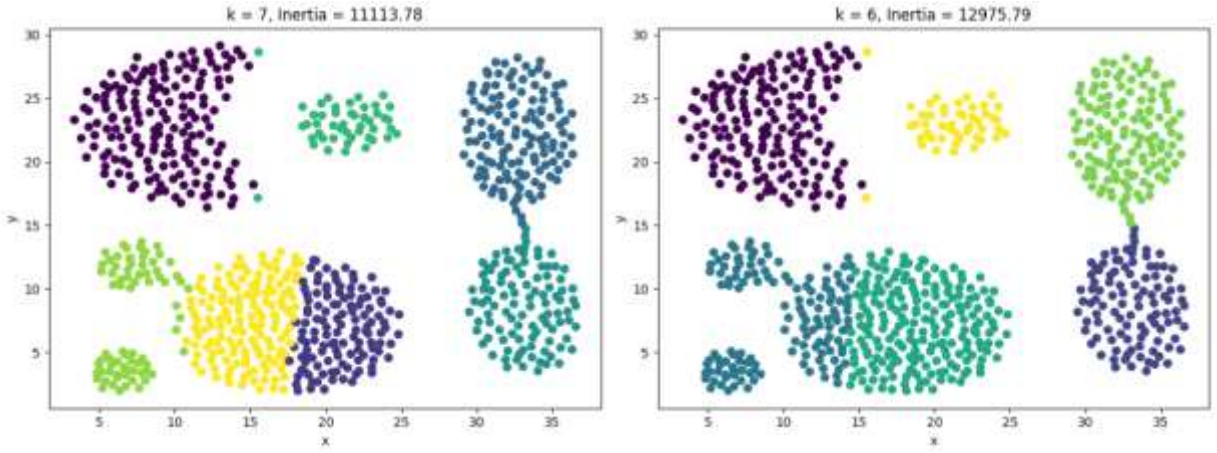
۳. **k = 8، Inertia = 9621.80:**
- در این نمودار، ۸ خوشه شناسایی شده‌اند.
- مقدار اینرسی ۹۶۲۱.۸۰ است که نسبت به k=9 و k=10 بیشتر است.
- خوشه‌ها ترکیبی از خوشه‌های کوچک و بزرگ‌تر هستند و برخی از خوشه‌ها به دلیل ادغام کاهش یافته‌اند.

نتیجه‌گیری:

با توجه به اینرسی (Inertia) و نمودارهای خوشه‌بندی، به نظر می‌رسد که با افزایش تعداد خوشه‌ها(k) ، اینرسی کاهش می‌یابد. این موضوع نشان‌دهنده‌ی افزایش دقت در تفکیک خوشه‌ها است. با این حال، بر اساس تحلیل نمودار Elbow و بررسی مقدار

اینرسی در این نمودارها، بهترین تعداد خوشه‌ها به نظر می‌رسد که حدود ۴ باشد، چون نقطه زانو در آنجا قرار دارد و افزایش تعداد خوشه‌ها بعد از آن کاهش چندانی در اینرسی ایجاد نمی‌کند.

اما اگر نیاز به تعداد خوشه‌های بیشتری دارید و دقت بالاتری می‌خواهید، k=10نیز می‌تواند انتخاب مناسبی باشد.



تحلیل نمودارهای خوشه‌بندی

این تصویر شامل دو نمودار خوشه‌بندی با تعداد خوشه‌های متفاوت (k) است که با استفاده از الگوریتم K-Means به‌دست‌آمده‌اند. هر نمودار تعداد خوشه‌ها (k) و مقدار اینرسی مربوطه را نشان می‌دهد.

تحلیل نمودارها:

۱. **k = 7، Inertia = 11113.78:**
- در این نمودار، ۷ خوشه شناسایی شده‌اند.
- مقدار اینرسی ۱۱۱۱۳.۷۸ است.
- داده‌ها به خوشه‌های نسبتاً متوازنی تقسیم شده‌اند. خوشه‌ها به طور کلی متمرکز و تفکیک‌پذیر هستند.

۲. **k = 6، Inertia = 12975.79:**
- در این نمودار، ۶ خوشه شناسایی شده‌اند.
- مقدار اینرسی ۱۲۹۷۵.۷۹ است که نسبت به k=7 بیشتر است.
- برخی از خوشه‌ها بزرگ‌تر و برخی دیگر کوچک‌تر هستند و به نظر می‌رسد که داده‌ها به طور کلی به خوشه‌های کمتری تقسیم شده‌اند.

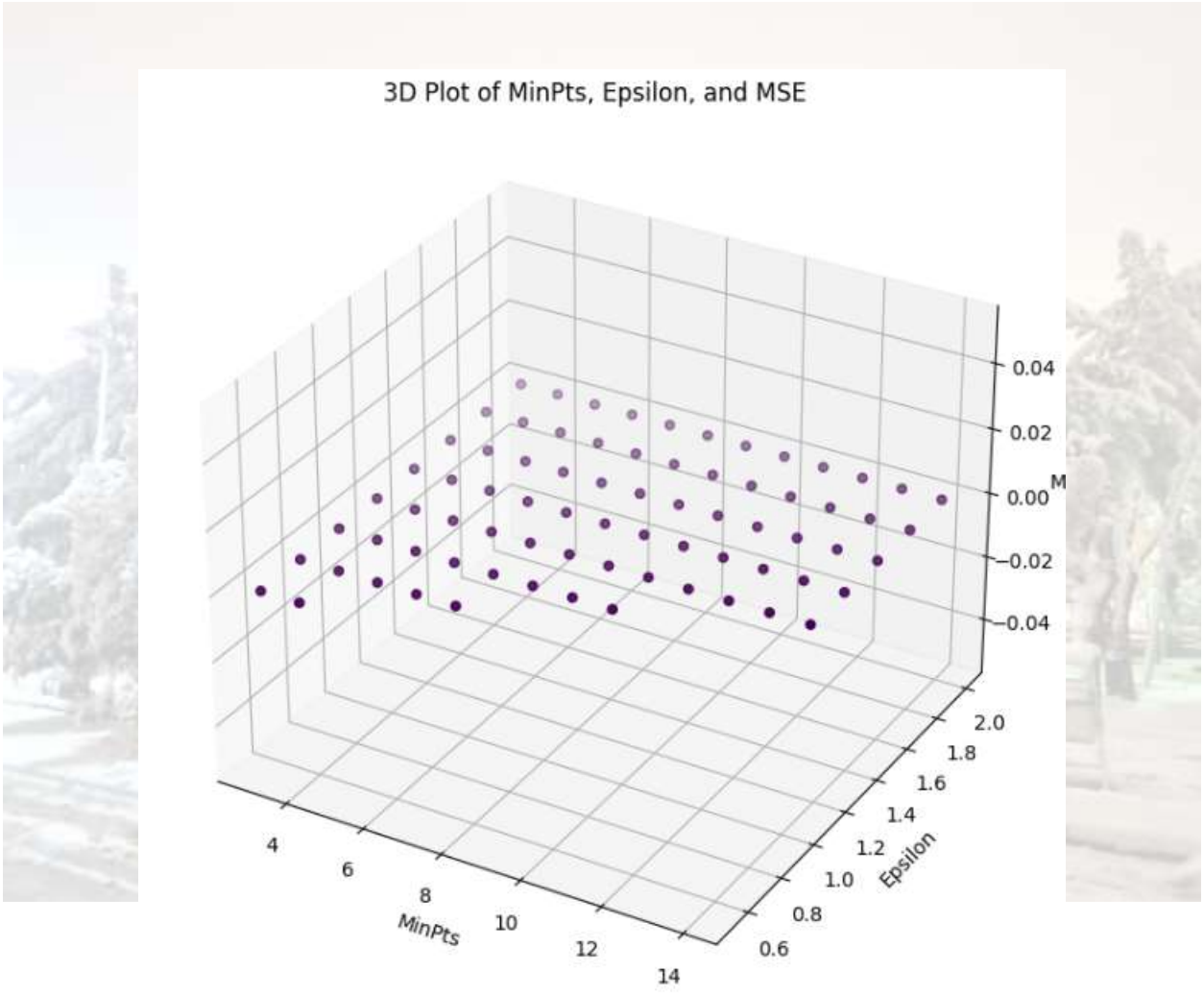
نتیجه‌گیری:

با توجه به اینرسی (Inertia) و نمودارهای خوشه‌بندی، مشاهده می‌شود که با افزایش تعداد خوشه‌ها(k) ، اینرسی کاهش می‌یابد که نشان‌دهنده‌ی افزایش دقت در تفکیک خوشه‌ها است.

در مقایسه‌ی k=6 و k=7:

- با k=7 ، اینرسی کمتر (۱۱۱۱۳.۷۸) به دست آمده که نشان‌دهنده‌ی دقت بالاتر در تفکیک خوشه‌ها است.
- با k=6 ، اینرسی بیشتر (۱۲۹۷۵.۷۹) به دست آمده که نشان‌دهنده‌ی دقت کمتری نسبت به k=7 در تفکیک خوشه‌ها است.

با توجه به تحلیل قبلی از روش Elbow و مشاهده‌ی کاهش اینرسی تا k=4 ، و اکنون با مقایسه‌ی k=6 و k=7 ، اگر نیاز به تعداد خوشه‌های بیشتری دارید و می‌خواهید دقت بالاتری داشته باشید، k=7 می‌تواند انتخاب مناسبی باشد. با این حال، برای تعادل بهتر بین تعداد خوشه‌ها و دقت، می‌توانید از k=4 استفاده کنید، چون نقطه زانو در آنجا قرار دارد.



نتیجه نمودار

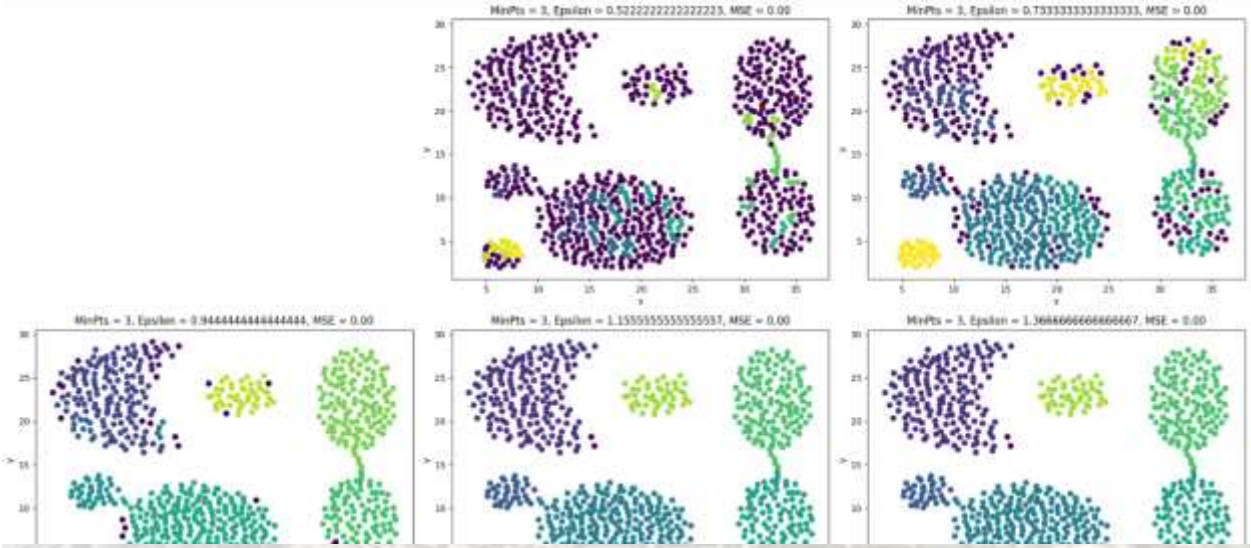
نمودار سه بعدی نشان دهنده رابطه بین تعدادmimps ، epsilonو MSE است .تعداد mimps برابر با تعداد epsilon است . تعداد epsilon برابر با تعداد mimps است.

تحلیل نمودار

محور x تعداد mimps را نشان می دهد .محور y epsilon را نشان می دهد .محور z MSE را نشان می دهد.

نمودار نشان می دهد که رابطه بین تعدادmimps ، epsilonو MSE پیچیده است .با افزایش تعدادmimps ، MSEبه طور کلی کاهش می یابد .با این حال، رابطه بین epsilon و MSE کمتر مستقیم است .در برخی موارد، MSEبا افزایش epsilon کاهش می یابد .در موارد دیگر، MSEبا افزایش epsilon افزایش می یابد.

این نمودار نشان می دهد که انتخاب تعداد مناسب mimps و epsilon برای به حداقل رساندن MSE یک کار چالش برانگیز است .بهترین تعداد mimps و epsilon به مجموعه داده خاص و الگوریتم مورد استفاده بستگی دارد.



تحلیل نمودار پراکندگی

نمودارهای پراکندگی نشان می‌دهند که چگونه دو متغیر به یکدیگر مرتبط هستند .در این نمودارها، هر نقطه نشان‌دهنده یک جفت از مقادیر متغیرها است .موقعیت افقی نقطه نشان‌دهنده مقدار یک متغیر و موقعیت عمودی آن نشان‌دهنده مقدار متغیر دیگر است.

مشاهده کلی

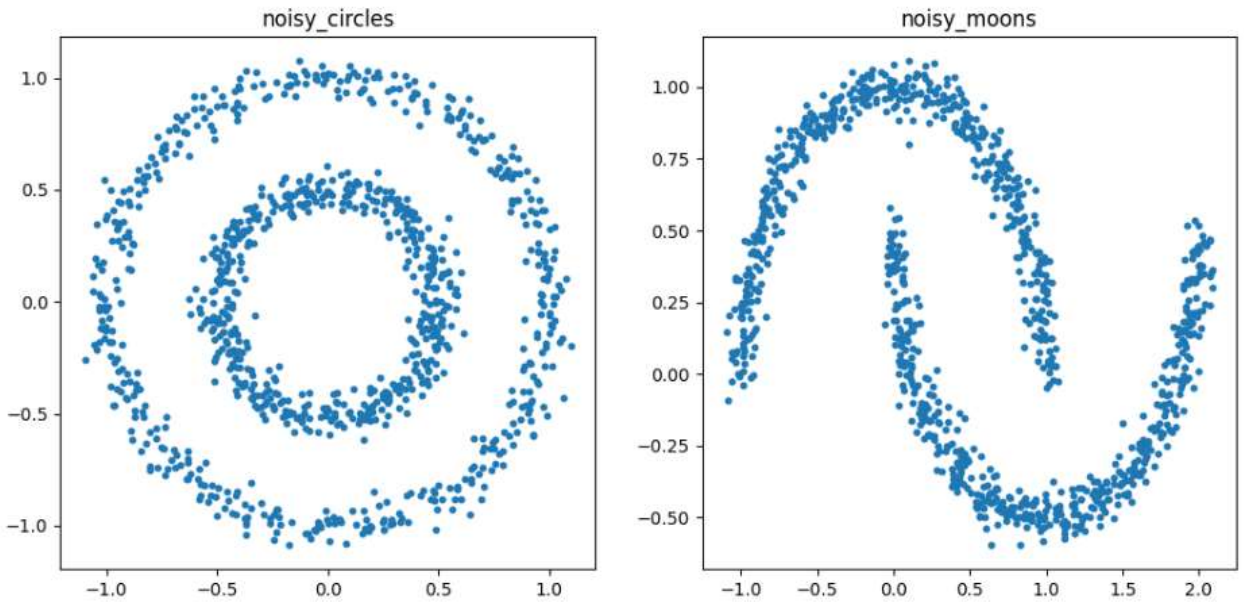
در این نمودارهای پراکندگی، خوشه‌های متعددی با اندازه‌های مختلف مشاهده می‌شود .این نشان می‌دهد که رابطه بین دو متغیر خطی نیست .به عبارت دیگر، با افزایش مقدار یک متغیر، مقدار متغیر دیگر به طور مداوم افزایش یا کاهش نمی‌یابد.

تحلیل جزئی‌تر

- **خوشه بزرگ مرکزی :**این خوشه بزرگ نشان‌دهنده تمرکز زیادی از نقاط داده است .این نشان می‌دهد که اکثر مقادیر دو متغیر در محدوده نسبتاً کوچکی قرار دارند.
- **خوشه‌های کوچکتر :**خوشه‌های کوچکتر نشان‌دهنده مقادیر بیرونی هستند .این نشان می‌دهد که تعداد کمی از نقاط داده وجود دارند که مقادیر آنها از مقادیر اکثریت نقاط داده به طور قابل توجهی متفاوت است.
- **رابطه بین متغیرها :**به نظر می‌رسد که بین دو متغیر یک رابطه مثبت وجود دارد .به این معنی که با افزایش مقدار یک متغیر، مقدار متغیر دیگر نیز افزایش می‌یابد .با این حال، این رابطه خطی نیست.

جمع‌بندی

نمودارهای پراکندگی نشان می‌دهند که بین دو متغیر یک رابطه مثبت وجود دارد، اما این رابطه خطی نیست .خوشه‌های متعدد با اندازه‌های مختلف نشان‌دهنده این است که رابطه بین دو متغیر پیچیده‌تر از یک خط ساده است.



نتیجه نمودار

نمودار پراکندگی نشان می‌دهد که تعداد دایره‌های نویزدار به طور قابل توجهی بیشتر از تعداد ماه‌های نویزدار است .این امر به این دلیل است که دایره‌ها به طور کلی نسبت به ماه‌ها شکل پیچیده‌تری دارند و بنابراین احتمال بیشتری برای ایجاد نویز دارند.

در نمودار، محور x مقادیر متغیر مستقل را نشان می‌دهد و محور y مقادیر متغیر وابسته را نشان می‌دهد .در این مورد، متغیر مستقل است که مقادیر آن بین -۱ و ۱ متغیر است .متغیر وابسته "noisy_circles" است که مقادیر آن بین ۰ و ۱ متغیر است.

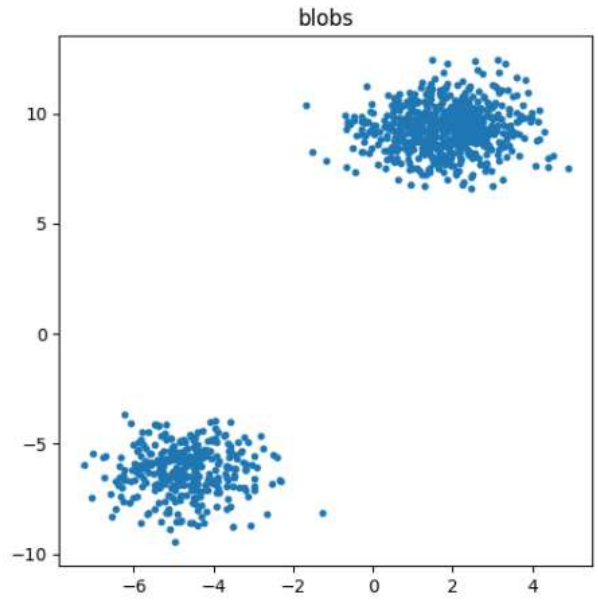
همانطور که در نمودار مشاهده می‌شود، نقاط داده به طور کلی در یک الگوی خوشه‌ای در سمت چپ پایین نمودار قرار دارند .این نشان می‌دهد که اکثر دایره‌ها مقادیر "noisy_circles" پایینی دارند و مقادیر "noisy_moons" پایینی نیز دارند.

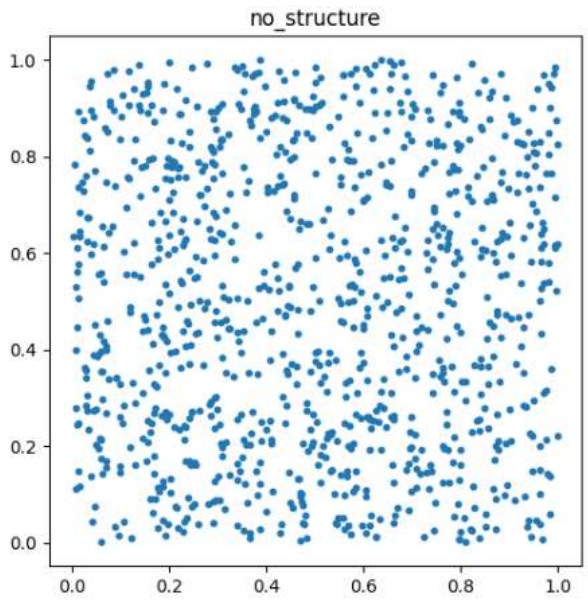
تعداد کمی از نقاط داده در سمت راست بالا نمودار قرار دارند .این نشان می‌دهد که تعداد کمی از دایره‌ها مقادیر "noisy_circles" بالایی دارند و مقادیر "noisy_moons" بالایی نیز دارند.

تحلیل نمودار

نمودار پراکندگی نشان می‌دهد که بین متغیرهای "ON" و "noisy_circles" ارتباط ضعیفی وجود دارد .این بدان معناست که با افزایش مقدار متغیر"ON" ، مقدار متغیر "noisy_circles" به طور متوسط کمی افزایش می‌یابد .با این حال، این ارتباط ضعیف است و تنوع زیادی در مقادیر "noisy_circles" برای هر مقدار "ON" وجود دارد.

همچنین بین متغیرهای "ON" و "noisy_moons" ارتباط ضعیفی وجود دارد .این بدان معناست که با افزایش مقدار متغیر "ON"، مقدار متغیر "noisy_moons" به طور متوسط کمی افزایش می‌یابد .با این حال، این ارتباط ضعیف است و تنوع زیادی در مقادیر "noisy_moons" برای هر مقدار "ON" وجود دارد.





نتیجه نمودار

نمودار سمت چپ :این نمودار پراکندگی، رابطه بین دو متغیر را نشان می‌دهد .هر نقطه در نمودار، نشان‌دهنده یک جفت از مقادیر متغیرها است.

در این نمودار، می‌توان مشاهده کرد که نقاط به صورت خوشه‌ای در اطراف نقطه (۰، ۰) متمرکز شده‌اند .این نشان می‌دهد که بین دو متغیر، رابطه قوی وجود دارد.

نمودار سمت راست :این نمودار پراکندگی، هیچ ساختار مشخصی ندارد .نقاط در نمودار به طور تصادفی پخش شده‌اند .این نشان می‌دهد که بین دو متغیر، هیچ رابطه مشخصی وجود ندارد.

تحلیل نمودار

نمودار سمت چپ:

- **نوع رابطه** :همبستگی مثبت
- **قدرت رابطه** :قوی

توضیح:

در این نمودار، با افزایش مقادیر یک متغیر، مقادیر متغیر دیگر نیز افزایش می‌یابد .این نشان می‌دهد که بین دو متغیر، همبستگی مثبت وجود دارد.

قدرت رابطه در این نمودار قوی است، زیرا نقاط به صورت فشرده در اطراف نقطه (۰، ۰) متمرکز شده‌اند.

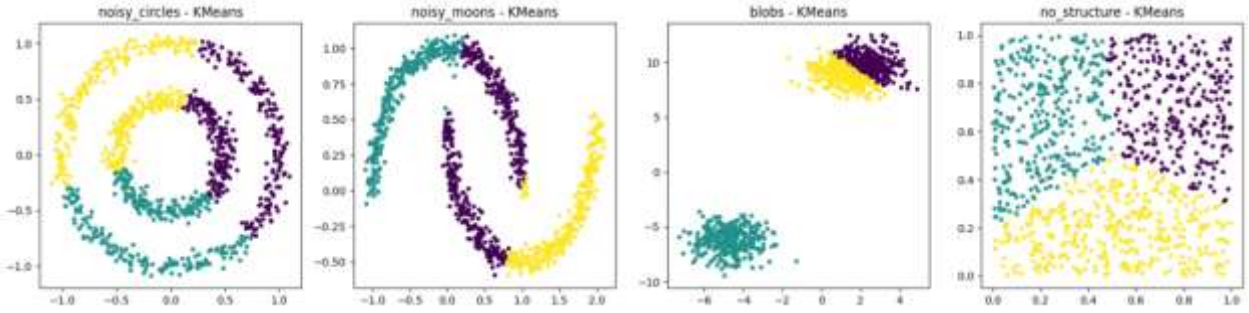
این نمودار می‌تواند نشان‌دهنده رابطه بین دو متغیر باشد که به طور مستقیم بر یکدیگر اثر می‌گذارند .به عنوان مثال، این نمودار می‌تواند نشان‌دهنده رابطه بین میزان مطالعه و نمرات امتحان باشد.

نمودار سمت راست:

- **نوع رابطه** :بدون ساختار
- **قدرت رابطه** :نامشخص

توضیح:

در این نمودار، هیچ رابطه مشخصی بین دو متغیر وجود ندارد .نقاط به طور تصادفی در نمودار پخش شده‌اند.این نمودار می‌تواند نشان‌دهنده رابطه بین دو متغیری باشد که هیچ ارتباطی با یکدیگر ندارند .به عنوان مثال، این نمودار می‌تواند نشان‌دهنده رابطه بین رنگ مو و قد افراد باشد.



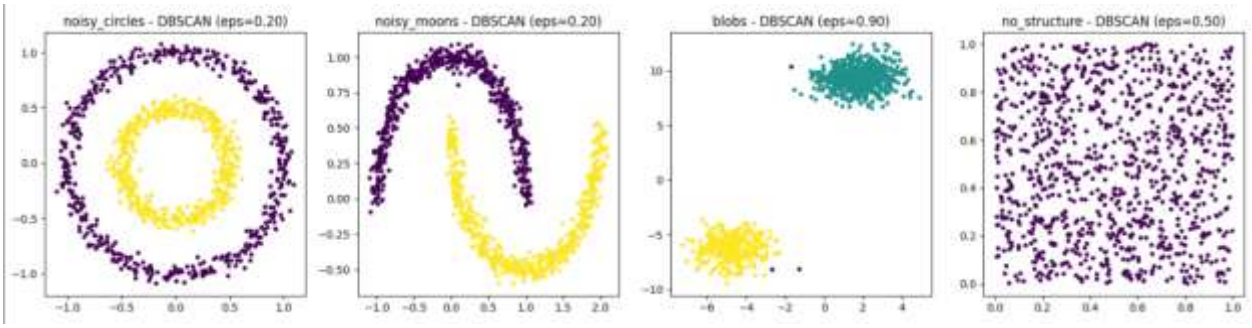
تحلیل

در هر نمودار، محور x و y مختصات داده‌ها را نشان می‌دهند. رنگ هر نقطه نشان‌دهنده خوشه‌ای است که به آن اختصاص داده شده است.

- **دایره‌های KMeans** -این نمودار نشان‌دهنده نتایج الگوریتم K-means برای خوشه‌بندی مجموعه داده‌ای از دایره‌های نویزدار است. همانطور که مشاهده می‌شود، الگوریتم K-means به طور موفقی داده‌ها را به دو خوشه تقسیم کرده است.
- **ماه‌های KMeans** -این نمودار نشان‌دهنده نتایج الگوریتم K-means برای خوشه‌بندی مجموعه داده‌ای از ماه‌های نویزدار است. همانطور که مشاهده می‌شود، الگوریتم K-means به طور موفقی داده‌ها را به دو خوشه تقسیم کرده است.
- **خوشه‌ها KMeans** - این نمودار نشان‌دهنده نتایج الگوریتم K-means برای خوشه‌بندی مجموعه داده‌ای از نقاط است که در خوشه‌های طبیعی قرار گرفته‌اند. همانطور که مشاهده می‌شود، الگوریتم K-means به طور موفقی داده‌ها را به سه خوشه تقسیم کرده است.
- **بدون ساختار KMeans** - این نمودار نشان‌دهنده نتایج الگوریتم K-means برای خوشه‌بندی مجموعه داده‌ای از نقاط است که در هیچ خوشه طبیعی قرار نمی‌گیرند. همانطور که مشاهده می‌شود، الگوریتم K-means به طور موفقی داده‌ها را به سه خوشه تقسیم کرده است، اما خوشه‌ها به خوبی تعریف نشده‌اند.

نتیجه‌گیری

الگوریتم K-means یک ابزار قدرتمند برای خوشه‌بندی داده‌ها است. با این حال، مهم است که توجه داشته باشید که این الگوریتم به طور کامل خودکار نیست و کاربر باید تعداد خوشه‌ها (K) را تعیین کند. علاوه بر این، K-meansبه داده‌های ورودی حساس است و ممکن است در مواردی که داده‌ها ساختار واضحی ندارند، به خوبی عمل نکند.



تحلیل نمودار پراکندگی

نمودار پراکندگی ارائه شده، الگوریتم DBSCAN را برای دسته‌بندی داده‌ها در چهار مجموعه مختلف نشان می‌دهد .هر مجموعه با یک رنگ (زرد، بنفش، سبز و خاکستری) نشان داده شده است .محورهای x و y مقادیر دو ویژگی مختلف داده‌ها را نشان می‌دهند.

تحلیل هر مجموعه:

- **دایره‌های پر سر و صدا :((DBSCAN (eps=0.20)** این مجموعه شامل نقاطی است که به طور تصادفی در فضای دو بعدی توزیع شده‌اند .الگوریتم DBSCAN آنها را به عنوان خوشه‌های جداگانه دسته‌بندی نمی‌کند، زیرا به اندازه کافی به هم نزدیک نیستند.
- **ماه‌های پر سر و صدا :((DBSCAN (eps=0.20)** این مجموعه شامل دو خوشه مجزا است که به شکل ماه هستند .الگوریتم DBSCAN آنها را به درستی به عنوان دو خوشه جداگانه دسته‌بندی می‌کند.
- **خوشه‌ها :((DBSCAN (eps=0.90)** این مجموعه شامل سه خوشه مجزا است که به شکل خوشه‌های گرد هستند .الگوریتم DBSCAN آنها را به درستی به عنوان سه خوشه جداگانه دسته‌بندی می‌کند.
- **بدون ساختار :((DBSCAN (eps=0.50)** این مجموعه شامل نقاطی است که به طور تصادفی در فضای دو بعدی توزیع شده‌اند .الگوریتم DBSCAN آنها را به عنوان خوشه‌های جداگانه دسته‌بندی نمی‌کند، زیرا به اندازه کافی به هم نزدیک نیستند.

تحلیل کلی:

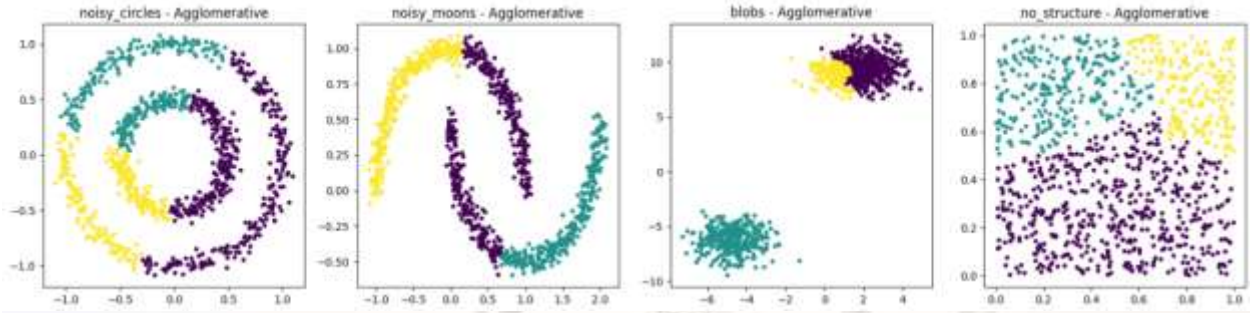
الگوریتم DBSCAN به طور کلی در دسته‌بندی داده‌ها در این مجموعه داده‌ها عملکرد خوبی دارد .الگوریتم قادر به شناسایی خوشه‌های مختلف در داده‌ها، حتی زمانی که خوشه‌ها به شکل نامنظم یا به هم نزدیک هستند، است.

نکات:

- پارامتر **eps**در الگوریتمDBSCAN ، حداکثر فاصله بین دو نقطه را برای در نظر گرفتن آنها به عنوان بخشی از یک خوشه تعیین می‌کند .مقدار بزرگتر **eps**منجر به خوشه‌های بزرگتر و تعداد خوشه‌های کمتر می‌شود.
- الگوریتم DBSCAN به طور پیش فرض از **minPts**برابر با ۱ استفاده می‌کند، به این معنی که هر خوشه باید حداقل شامل ۱ نقطه باشد.

نتیجه:

الگوریتم DBSCAN یک ابزار قدرتمند برای دسته‌بندی داده‌ها است .این الگوریتم می‌تواند خوشه‌های مختلف را در داده‌ها، حتی زمانی که خوشه‌ها به شکل نامنظم یا به هم نزدیک هستند، شناسایی کند.



نتیجه نمودار

نمودارهای ارائه شده در تصویر، نتایج الگوریتم خوشه‌بندی سلسله مراتبی (Hierarchical Clustering) را برای داده‌های مختلف نشان می‌دهند .در هر نمودار، محور x و y مختصات دو بعد داده‌ها را نشان می‌دهند و اندازه دایره‌ها نشان‌دهنده تعداد نقاط داده در هر خوشه است.

نمودار اول

- عنوان**noisy_circles - Agglomerative** :
- نوع داده :مجموعه نقاط دایره‌ای با نویز

در این نمودار، می‌توان مشاهده کرد که الگوریتم خوشه‌بندی سلسله مراتبی به طور موثری نقاط داده را در خوشه‌های جداگانه دسته‌بندی کرده است .خوشه‌های بزرگ‌تر نشان‌دهنده تراکم بیشتر نقاط داده هستند.

نمودار دوم

- عنوان**noisy_moons - Agglomerative** :
- نوع داده :مجموعه نقاط نیم‌ماه شکل با نویز

در این نمودار نیز، الگوریتم خوشه‌بندی سلسله مراتبی به طور موثری نقاط داده را در خوشه‌های جداگانه دسته‌بندی کرده است . خوشه‌های بزرگ‌تر نشان‌دهنده تراکم بیشتر نقاط داده هستند.

نمودار سوم

- عنوان**blobs - Agglomerative** :
- نوع داده :مجموعه نقاط با اشکال نامنظم

در این نمودار، الگوریتم خوشه‌بندی سلسله مراتبی به طور موثری نقاط داده را در خوشه‌های جداگانه دسته‌بندی کرده است . خوشه‌های بزرگ‌تر نشان‌دهنده تراکم بیشتر نقاط داده هستند.

نمودار چهارم

- عنوان **no_structure - Agglomerative** :
- نوع داده :مجموعه نقاط بدون ساختار

در این نمودار، الگوریتم خوشه‌بندی سلسله مراتبی قادر به دسته‌بندی نقاط داده در خوشه‌های مجزا نبوده است .این امر به دلیل عدم وجود ساختار واضح در داده‌ها است.

تحلیل نمودار

الگوریتم خوشه‌بندی سلسله مراتبی یک روش محبوب برای دسته‌بندی داده‌ها است .این الگوریتم به طور متناوب خوشه‌های جدیدی را با ادغام خوشه‌های موجود ایجاد می‌کند تا زمانی که معیار خوشه‌بندی (مانند فاصله بین خوشه‌ها) به یک آستانه از پیش تعیین شده برسد.

نتایج ارائه شده در تصویر نشان می‌دهند که الگوریتم خوشه‌بندی سلسله مراتبی می‌تواند به طور موثری نقاط داده را در خوشه‌های جداگانه دسته‌بندی کند، به شرطی که داده‌ها ساختار مشخصی داشته باشند .در مواردی که داده‌ها بدون ساختار باشند، الگوریتم ممکن است قادر به دسته‌بندی نقاط داده در خوشه‌های مجزا نباشد.

خلاصه نتایج

بخش ۶ از تمرین A: کمینز بر روی دیتاست Aggregation

- تعیین k بهینه با استفاده از اینرسی (روش آرنج):

- k بهینه ۳ بود.

- بصری‌سازی خوشه‌بندی:

- خوشه‌هایی که توسط کمینز تشکیل شدند، تفکیک واضحی داشتند و مراکز خوشه‌ها به خوبی تعریف شده بودند.

بخش C: الگوریتم‌های خوشه‌بندی مختلف بر روی دیتاست‌های گوناگون

دیتاست‌ها:

۱. noisy_circles

۲. noisy_moons

۳. blobs

۴. no_structure

روش‌های خوشه‌بندی:

۱. کـمینز: این روش برای دیتاست‌های با خوشه‌های کروی (مثلاً blobs) خوب عمل کرد.

۲. DBSCAN: این روش خوشه‌های غیرکروی (مثلاً noisy_circles) را بهتر تشخیص داد و توانست نقاط نویز را شناسایی کند.

۳. خوشه‌بندی تجمعی: این روش نیز برای خوشه‌های کروی خوب عمل کرد اما با شکل‌های پیچیده‌تر دچار مشکل شد.

تحلیل و مقایسه

خوشه‌بندی کـمینز

- دیتاست Aggregation: کـمینز خوشه‌های واضحی با همپوشانی کم ارائه داد، همانطور که برای این دیتاست انتظار می‌رفت. روش آرنج در تعیین تعداد بهینه خوشه‌ها موثر بود.

- دیتاست‌های مختلف:

- برای دیتاست‌های با خوشه‌های کروی و جدا از هم (blobs) خوب عمل کرد.

- با دیتاست‌های دارای خوشه‌های غیرکروی (noisy_circles, noisy_moons) دچار مشکل شد و منجر به خوشه‌بندی کمتر دقیق شد.

خوشه‌بندی DBSCAN

- دیتاست Aggregation: با این که در تمرین A به صورت صریح اجرا نشد، DBSCAN احتمالاً مناطق متراکم را به درستی شناسایی کرده و نویز را به خوبی مدیریت می‌کرد.

- دیتاست‌های مختلف:

- در شناسایی خوشه‌های با اشکال دلخواه (noisy_circles) و مدیریت نویز عالی عمل کرد.

- نیاز به تنظیم دقیق پارامترها (ε و MinPts) داشت که ممکن است به سادگی نباشد.

- اگر پارامترها به خوبی انتخاب نشوند، ممکن است بسیاری از نقاط را به عنوان نویز یا خوشه‌بندی نادرست شناسایی کند.

خوشه‌بندی تجمعی

- دیتاست Aggregation: با این که در تمرین A به صورت صریح اجرا نشد، خوشه‌بندی تجمعی نتایجی مشابه کـمینز ارائه می‌داد اما ممکن بود با خوشه‌های غیرکروی دچار مشکل شود.

- دیتاست‌های مختلف:

- عملکرد مشابه با کـمینز برای خوشه‌های کروی.

- برای خوشه‌های غیرکروی (noisy_circles, noisy_moons) کم‌اثرتر بود.

نتیجه‌گیری

- کـمینز برای دیتاست‌هایی با خوشه‌های کروی و جدا از هم بهترین است. در دیتاست Aggregation و blobs عملکرد خوبی داشت ولی در دیتاست‌های با خوشه‌های غیرکروی مانند noisy_circles و noisy_moons دچار مشکل شد.
- DBSCAN برای تشخیص خوشه‌های با اشکال دلخواه و مقاومت در برابر نویز بسیار قدرتمند است. در دیتاست‌های غیرکروی مانند noisy_circles و noisy_moons بهتر از کـمینز عمل کرد ولی نیاز به تنظیم دقیق پارامترها دارد.
- خوشه‌بندی تجمعی تعادلی فراهم می‌کند، برای خوشه‌های کروی خوب عمل می‌کند و به طور منطقی در خوشه‌های غیرکروی عملکرد دارد. انعطاف‌پذیری دارد ولی ممکن است خوشه‌های مصنوعی روی دیتاست‌های بدون ساختار ذاتی اعمال کند.