



به نام خدا

موضوع پروژه:

مروری بر خوشه بندی با استفاده از کتابخانه KMEANS و الگوریتم NUMPY

تهیه کننده:

معین حاج ملک

استاد مربوطه:

امین دهقان قنات کمان

خلاصه :

در زبان برنامه نویسی پایتون در زمانی که ما داده ای را بر روی حافظه ذخیره سازی میکنیم هر داده در حافظه ما برچسب مشخصی میگیرد که از طریق آن انواع نوع داده با همان برچسب میتوانند دسته بندی و مورد استفاده قرار بگیرند. اما در مواقعی داده هایی در حافظه ذخیره میشوند و برچسب ندارند و ما برای اینکه بخواهیم آن داده ها را دسته بندی کنیم و به آنها دسترسی داشته باشیم میتوانیم از الگوریتم `Kmeans` و همچنین کتابخانه `NumPy` استفاده کنیم .

`Kmeans` میتواند برای تایید فرضیات درباره اینکه چه نوع دسته ها و نظم هایی در داده ها وجود دارند یا برای شناسایی دسته های ناشناخته در مجموعه داده های پیچیده به کار برود .

در این پروژه من قصد دارم اول توضیحاتی ارائه دهم درباره الگوریتم `Kmeans` و کتابخانه `NumPy` و همچنین کتابخانه `Matplotlib` سپس مروری داشته باشیم بر نحوه پیاده سازی الگوریتم `Kmeans` و استفاده از کتابخانه `NumPy` و `Matplotlib` و با استفاده از آنها یک دسته را خوشه بندی کنیم.

فهرست مطالب

4.....	مقدمه
4.....	سوال اصلی
5.....	آشنایی با کتابخانه NumPy
6.....	نحوه نصب کتابخانه NumPy
7.....	نحوه کار با کتابخانه NumPy
10.....	آشنایی با کتابخانه Matplotlib
11.....	نحوه کار با کتابخانه Matplotlib
17.....	داده ها
17.....	کد ها
23.....	نتیجه کار
24.....	نتیجه گیری
25.....	منابع

مقدمه :

الگوریتم های خوشه بندی (Clustering) از جمله روش های یادگیری نظارت نشده هستند. از این الگوریتم ها زمانی استفاده میشود که داده های برچسب دار موجود نباشند. در الگوریتم Kmeans هدف پیدا کردن خوشه ها در مجموعه داده هایی است که به عنوان ورودی دریافت میکنند. این الگوریتم یکی از ساده ترین و همچنین محبوب و پرکاربرد ترین الگوریتم از جمله الگوریتم های خوشه بندی میباشد.

kmeans در کاربرد های گوناگون مورد استفاده قرار میگیرد. از جمله این کاربرد ها میتوان به مواردی همچون بخش بندی تصاویر - خوشه بندی مقالات خبری - خوشه بندی گونه ها - تشخیص ناهنجاری ها و خوشه بندی داده های بخش بندی ژن اشاره کرد.

سوال اصلی :

سوال اصلی ما در این پروژه این است که در مواقعی که ما در زبان برنامه نویسی پایتون داده هایی ذخیره کرده ایم که برجستگی ندارند آیا میتوان آنها را دسته بندی کرده و مرتبشان کنیم با چه کتابخانه های و چه الگوریتمی؟

کتابخانه ها :

NumPy : اگر به دنبال یادگیری علم داده توسط پایتون هستید چاره ای جز یادگیری نحوه کار با کتابخانه های پایتون که به طور تخصصی برای اینکار طراحی شده اند را نداریم. NumPy که مخفف Numerical Python است، یک کتابخانه برای کار با آرایه های عددی ارائه شده است. این کتابخانه امکان استفاده از آرایه ها و ماتریس های بزرگ چندبعدی را به ما میدهد. کتابخانه نامپای (NumPy) یکی از مهم ترین کتابخانه های پایتون برای کار در حوزه کامپیوترساینس است.

تفاوت های تعریف آرایه در NumPy و لیست در پایتون :

سرعت بالاتر : مهم ترین تفاوت آرایه تعریف شده در نامپای با آرایه ای که به صورت لیست ساخته شده است، سرعت آن هاست. دلیل سریع تر بودن NumPy ، نحوه ذخیره شدن داده ها در حافظه است. در یک لیست، برای هر آیتم، یک پوینتر در حافظه وجود دارد. هر کدام از پوینترها به نقطه ای از حافظه اشاره می کنند که اطلاعات مربوط به عنصر

متناظرشان در آنجا ذخیره شده است. این پوینترها پشت سر هم نیستند و ممکن است دو آیتم که در لیست کنار هم قرار گرفته‌اند، در دو نقطه متفاوت از حافظه ذخیره شوند.

اما در NumPy تمامی آیتم‌های یک آرایه در کنار همدیگر ذخیره میشوند به همین دلیل باعث میشود دسترسی به آیتم‌های یک آرایه در این کتابخانه با سرعت بیشتری انجام شود.

حلقه‌های کمتر : نامپای به ما این امکان را میدهد حلقه‌های کمتری استفاده کنیم به چه صورت. نامپای در پایتون شامل توابع ریاضی زیادی است که کمک میکند یکسری عملیات ریاضی به صورت ماتریسی انجام شوند. طبیعتاً انجام برخی محاسبات مانند ضرب دو آرایه اگر به صورت ماتریسی انجام شوند زمان کمتری طول میکشند.

کدهای تمیزتر : وقتی تعداد حلقه‌های کمتری استفاده شود قطعاً تعداد خطوط کد ما کمتر شده و تمیز تر و خوانا تر میشوند.

نحوه نصب و فراخوانی کتابخانه NumPy در پایتون :

برای استفاده از هر کتابخانه‌ای در پایتون ابتدا باید آن را بر روی سیستم خود نصب و سپس برای استفاده از آن باید آن را در برنامه خود وارد کنیم.

نصب : برای نصب کتابخانه NumPy ابتدا باید چک کنیم که بر روی سیستم خود پایتون را داریم یا خیر بعد از آن میتوان در cmd ویندوز با استفاده از کد دستوری زیر کتابخانه NumPy را بر روی سیستم خود نصب کنیم.

```
pip install numpy
```

با اجرای کد بالا کتابخانه بر روی سیستم نصب خواهد شد و آماده استفاده است. اکنون برای استفاده از آن در پایتون باید آن را فراخوانی کنیم که به صورت زیر است :

```
import numpy
```

با وارد کردن کد بالا در ابتدا برنامه خود میتوانیم از کتابخانه NumPy در برنامه خود استفاده کنیم . برای استفاده از کتابخانه در خط کد های خود میتوانیم از روش زیر استفاده کنیم :

```
Arr_5 = numpy.array([1, 2, 3])
```

همچنین میتوانیم بعد از نوشتن اسم کتابخانه هنگام فراخوانی یک اسم داخلی به کتابخانه بدهیم که سریع تر بتوانیم به کتابخانه دسترسی داشته باشیم به صورت زیر :

```
import numpy as np
```

تعریف آرایه در NumPy :

تعریف یک آرایه در نامپای بسیار ساده میباشد و با استفاده از متد `np.array()` انجام میشود و سپس کافی است داخل پرانتز آرایه خود را به شکل یک لیست وارد کنیم مثال زیر نمونه ای از تعریف آرایه میباشد :

```
arr_1 = np.array([1, 2, 3])
```

اندیس دهی به آرایه در NumPy :

اندیس دهی در پایتون برای دسترسی به یک المان (یا یک درایه) از یک آرایه استفاده میشود . برای این کار کافی است که آدرس درایه مورد نظر را داشته باشیم . آدرس یک درایه شماره سطر و ستون آن است . به روش زیر به طور مثال می توانید درایه ای که در سطر دوم و ستون اول از `arr_5` وجود دارد را استخراج کنید :

```
arr_5[1, 0]
```

در پایتون اندیس ها از صفر شروع میشوند . یعنی اگر درایه ای در ستون اول باشد، شماره آن در پایتون صفر است! برای سطرها هم همینطور، اگر در سطر دوم باشد، شماره آن در پایتون، یک خواهد بود.

Slicing آرایه در NumPy :

تکه برداری یا اسلایسینگ بدین معنا است که یک بخش از آرایه را برداریم . برای چنین عملیاتی در پایتون از علامت کولن (:) استفاده میکنیم. علامت : در پایتون معادل با «تا» است. یعنی یک بازه را انتخاب می کند.مانند مثال زیر :

```
arr_3[0:2, 0]
```

ستون اول و سطر اول و دوم را برداشتیم.

محاسبه ابعاد یک آرایه در NumPy :

یکی از پر استفاده ترین آتریبیوت های داخل کتابخانه NumPy دستور Shape میباشد . با استفاده از این دستور میتوان ابعاد یک آرایه را در NumPy محاسبه کرد. مثال زیر نمونه ای از آن است :

```
arr_3.shape
```

عدد اولی که در خروجی به ما نمایش میدهد تعداد سطر ها و عدد دوم تعداد ستون ها میباشد .

Broadcasting یا انتشار همگانی در NumPy :

واژه ی Broadcasting به چگونگی رفتار NumPy با آرایه هایی با Shape متفاوت در خلال عملگرهای محاسباتی اشاره دارد. به طور خلاصه، آرایه ی کوچکتر به اندازه ی بزرگتر پخش می شود تا به شکل و Shape یکسانی با آن تبدیل شود. اینکه از Broadcast استفاده بکنیم یا خیر؛ به شرایط داده و الگوریتم بستگی دارد و می تواند با توجه به آن ها به کارگیری این مفهوم در کارایی برنامه تأثیر مثبت یا منفی داشته باشد Broadcasting در NumPy با توجه به محدودیت های مسأله و داده، کارایی را کنترل می کند.

قانون Broadcasting : باید اندازه محور نهایی یا آخر (یا همان مقدار آخرین Shape) در هر دو آرایه داده یکسان باشد یا مقدار بعد آخر حداقل یکی از آرایه ها برابر با یک باشد.

ایجاد آرایه :

چند روش برای ایجاد آرایه وجود دارند. برای مثال، می توان با استفاده از تابع array یک آرایه را از فهرست معمولی پایتون یا چندتایی ها ایجاد کرد. نوع آرایه حاصل، برابر با نوع عناصر موجود در دنباله های تشکیل دهنده آن خواهد بود.


```
>>> from numpy import *
>>> a = array( [2,3,4] )
>>> a
array([2, 3, 4])
>>> a.dtype
dtype('int32')
>>> b = array([1.2, 3.5, 5.1])
>>> b.dtype
dtype('float64')
```

پرینت کردن آرایه :

زمانی که یک آرایه را پرینت می‌کنید NumPy آن را به صورت یک فهرست تودرتو نمایش می‌دهد که طرح کلی آن به صورت زیر است:

- آخرین محور از چپ به راست پرینت میشود.
- محور ما قبل آخر از بالا به پایین پرینت میشود.
- باقی محور ها نیز از بالا به پایین پرینت و هر کدام با یک خط خالی از قبلی جدا میشوند.

بدین ترتیب آرایه‌های تک‌بعدی به صورت ردیفی، آرایه‌های دوبعدی به صورت ماتریس و آرایه‌های سه‌بعدی به صورت فهرستی از ماتریس‌ها پرینت می‌شوند.

```
>>> a = arange(6) # 1d آرایه
```

```
>>> print a
```

```
[5 4 3 2 1 0]
```

```
///
```

```
>>> b = arange(12).reshape(4,3) # 2d آرایه
```

```
>>> print b
```

```
[2 1 0 ]
```

```
[5 4 3 ]
```

```
[8 7 6 ]
[[11 10 9 ]
///
>>>c = arange(24).reshape(2,3,4) # 3d آرایه
>>>print c
[3 2 1 0 ]]]
[7 6 5 4 ]
[[11 10 9 8 ]
[15 14 13 12]]
[19 18 17 16]
[[[23 22 21 20]
```

: Matplotlib

نخستین و پرکاربردترین کتابخانه در زبان پایتون است.

در سال 2003 متولد شده است .

با توجه به قدیمی بودن خود کاربرد ها و دوست داران زیادی را دارد و این مسئله باعث شده که مثال های زیادی در دنیای اینترنت در مورد آن وجود داشته باشد.

● گراف های مختلفی دارد.

● انواع شخصی سازی روی نمودار ها را می توان انجام داد.

● یادگیری آن بسیار آسان است.

● می توان با فرمت های مختلفی خروجی نمودار ها را گرفت.

● رابط آن شباهت زیادی به MATLAB دارد.

● بسیاری از کتابخانه های تصویرسازی از دل این کتابخانه ارتقاء یافته اند.

با آرایه های NumPy میتواند کار می کند.

کتابخانه Matplotlib برای چه کار هایی استفاده میشود :

رسم نمودار : به ما این اجازه را می دهد تا با استفاده از داده های عددی و ماتریسی، انواع نمودار به شکل هایی هم چون نمودار خطی، میله ای، دایره ای، پراکندگی و نمودار های سه بعدی و را رسم کنیم.

تنظیمات نمودار : با استفاده از Matplotlib این امکان وجود دارد که تنظیمات مربوط به نمودار را به شکل دیگری اعمال کنیم.

مثال : می توان تغییرات را روی اندازه، عنوان، محورها و رنگ های نمودار انجام داد.

رسم تصاویر و نمودار های پویا : می توان تصاویر و نمودار های پویا را بر اساس و با استفاده از داده های رسم کرد که با گذشت زمان تغییر کند.

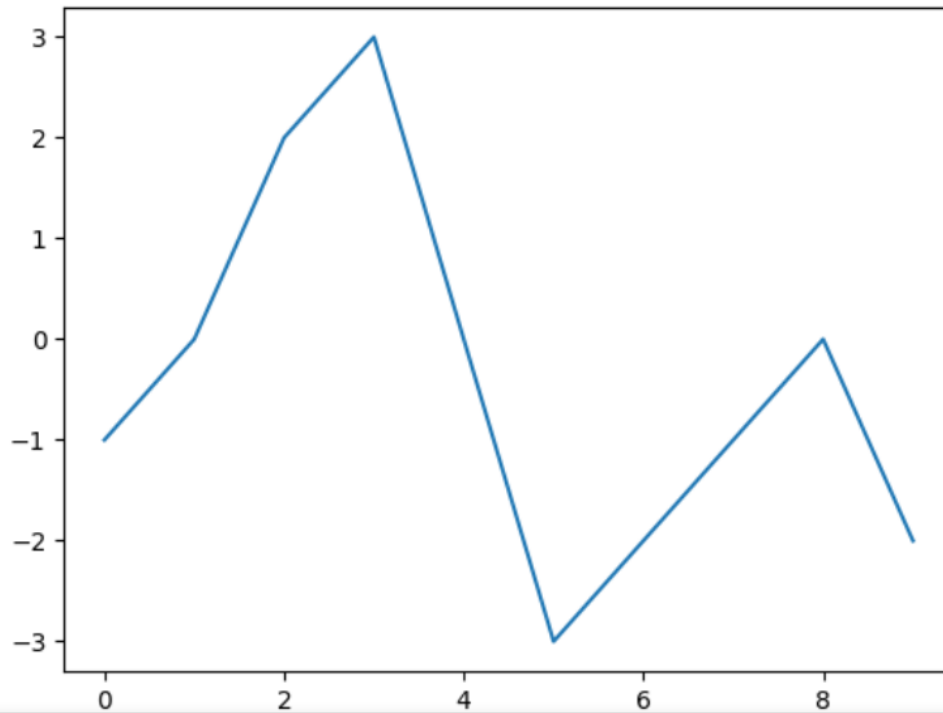
ترسیم تصاویر دوبعدی و سه بعدی : Matplotlib این اجازه را میدهد تا تصاویر دوبعدی و سه بعدی را ترسیم کنیم که این کار امکان مفیدی برای نمودار های پیچیده و تصاویر علمی است.

رسم نمودار توزیع و احتمال : با این کتابخانه می توان نمودار های توزیع احتمال مختلفی را رسم کرد.

رسم نمودار های شبکه ای : با این نمودار های توزیع احتمال مختلفی را رسم کرد.

نمودار خطی :

```
[2]: import matplotlib.pyplot as plt
x = [0, 1, 2, 3, 4, 5, 8, 9]
y = [-1, 0, 2, 3, 0, -3, 0, -2]
plt.plot(x, y)
plt.show()
```

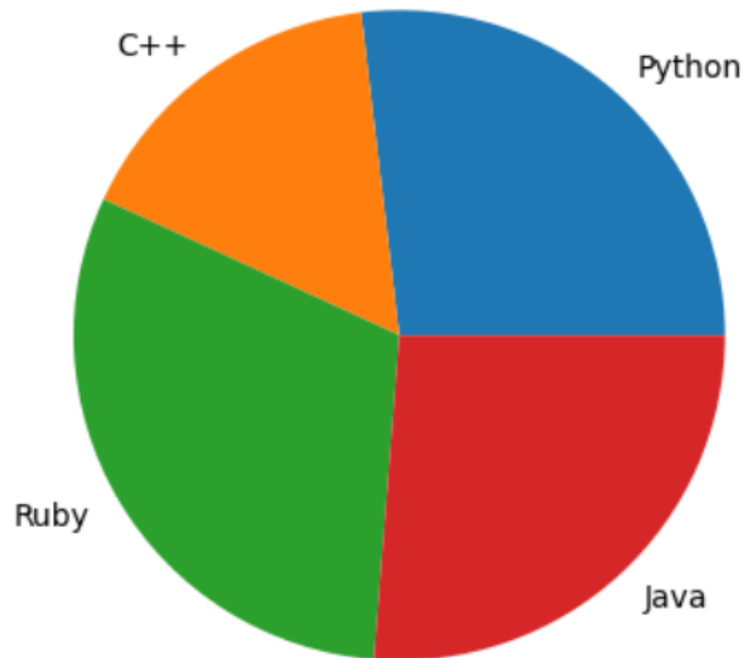


نمودار دایره ای :

```
[3]: import matplotlib.pyplot as plt

labels = 'Python', 'C++', 'Ruby', 'Java'
sizes = [215, 130, 245, 210]
plt.pie(sizes, labels=labels)

plt.show()
```



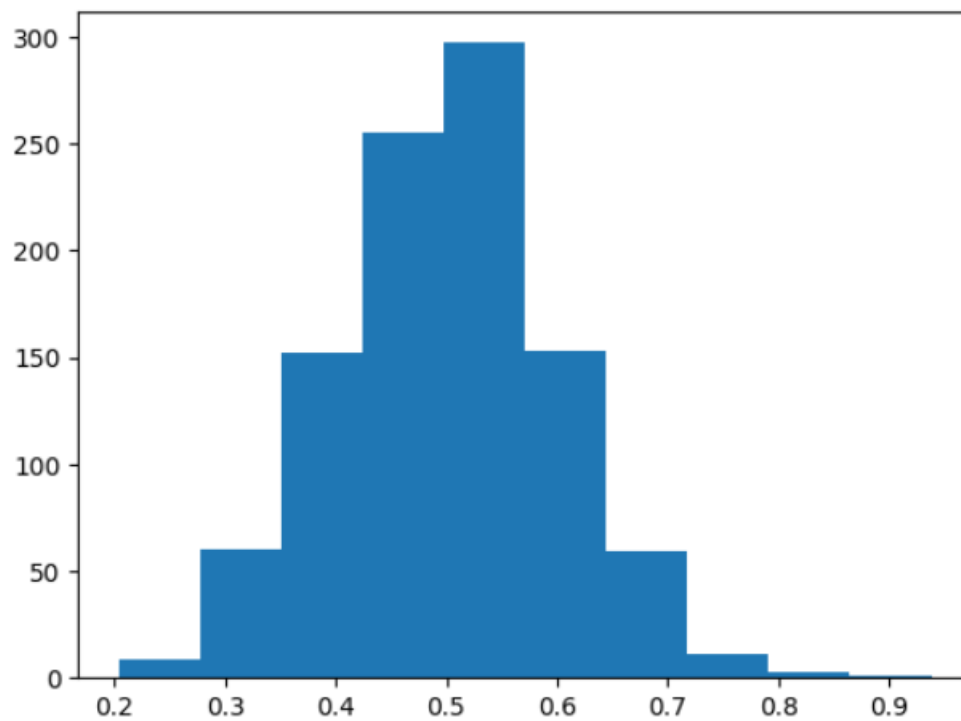
نمودار هیستوگرام :

```
[4]: import matplotlib.pyplot as plt
import numpy as np

Mean = 0.5 #  $\mu$ 
STD = 0.1 #  $\sigma$ 
x = np.random.normal(Mean, STD, 1000)

x = plt.hist(x)

plt.show()
```



رسم چند نمودار در یک قاب :

```

import matplotlib.pyplot as plt
import numpy as np

x = np.arange(-5, 5, 0.01)
y = np.sin(2 * np.pi * x)
y2 = np.cos(2 * np.pi * x)

axes1 = plt.subplot2grid((4, 4), (0, 0), rowspan = 2)
axes1.set_facecolor('y')
axes1.plot(x, y)

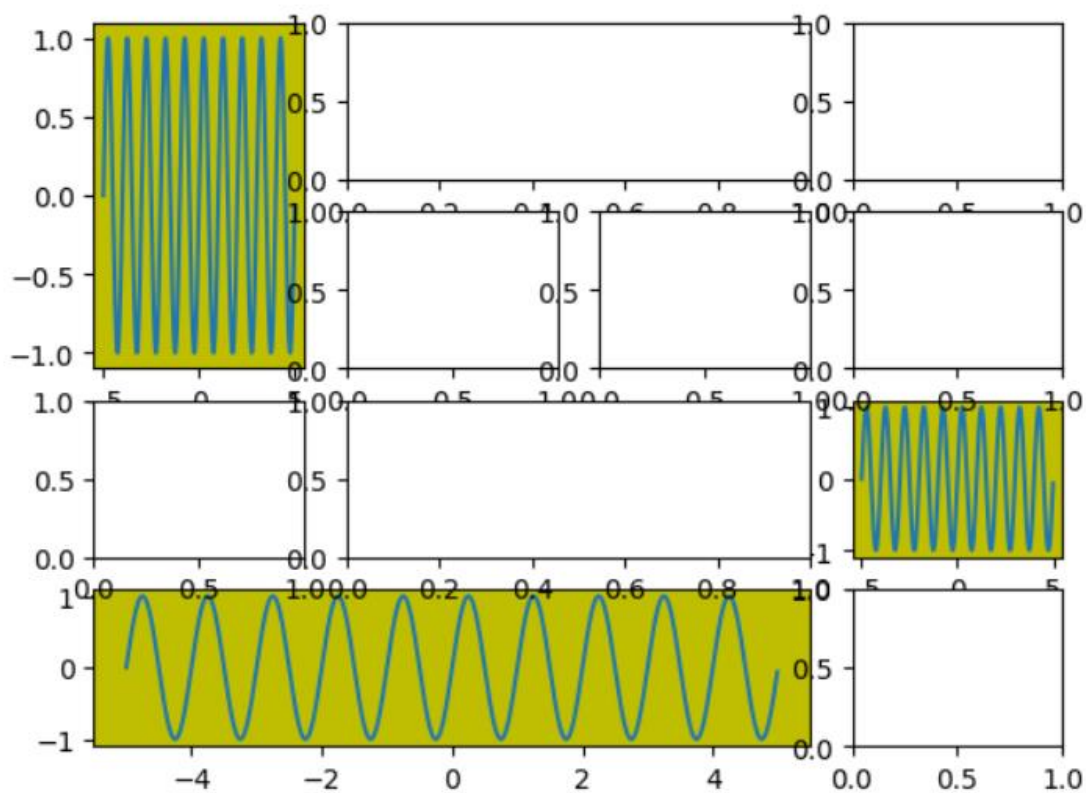
axes2 = plt.subplot2grid((4, 4), (2, 3))
axes2.set_facecolor('y')
axes2.plot(x, y)

axes3 = plt.subplot2grid((4, 4), (3, 0), colspan = 3)
axes3.set_facecolor('y')
axes3.plot(x, y)

plt.subplot2grid((4, 4), (0, 1), colspan = 2)
plt.subplot2grid((4, 4), (0, 3))
plt.subplot2grid((4, 4), (1, 1))
plt.subplot2grid((4, 4), (1, 2))
plt.subplot2grid((4, 4), (1, 3))
plt.subplot2grid((4, 4), (2, 0))
plt.subplot2grid((4, 4), (2, 1), colspan = 2)
plt.subplot2grid((4, 4), (3, 3))

plt.show()

```

داده های ما: داده های ما در این برنامه مجموعه ای داده از اعداد دست نوشته از کتابخانه Sklearn میباشد.

کدها:

1- اول با وارد کردن کتابخانه ها شروع میکنیم.

```
import matplotlib.pyplot as plt

import numpy as np

from sklearn.cluster import KMeans

from sklearn.datasets import load_digits

from sklearn.decomposition import PCA
```

1. خط اول: این خط کتابخانه Matplotlib را وارد میکنیم.

2. خط دوم: این خط کتابخانه NumPy را اضافه میکنیم .

3. خط سوم: این خط کلاس Kmeans را از کتابخانه sklearn.cluster وارد میکنیم که پیاده سازی الگوریتم خوشه بندی Kmeans را در اختیار قرار میدهد.

4. خط چهارم: این خط تابع load_digits را از کتابخانه sklearn.datasets وارد میکنیم که یک مجموعه داده از اعداد دست نوشته را در اختیار ما قرار میدهد.

5. خط پنجم: این خط کلاس PCA را از کتابخانه sklearn.decomposition وارد میکنیم که استفاده از روش تجزیه تفضیلی برای کاهش ابعاد داده ها را فراهم میکند.

2 — در ادامه بعد از وارد کردن کتابخانه ها حالا شروع به کد نویسی کرده :

```
data, labels = load_digits(return_X_y=True)
```

در این خط مجموعه داده ها و برچسب های مربوط به آن ها را با استفاده از تابع `load_digits` بارگیری می کنیم و با استفاده از `return_x_y=True` داده ها و برچسب ها را همزمان برمیگردانیم.

```
(n_samples, n_features), n_digits = data.shape, np.unique(labels).size
```

در این خط تعداد نمونه ها، تعداد ویژگی ها، تعداد دسته ها را از مجموعه داده ها و برچسب ها استخراج می کنیم.

```
reduced_data = PCA(n_components=2).fit_transform(data)
```

در این خط داده ها را با استفاده از روش تجزیه به فضای دو بعدی تبدیل می کنیم.

```
kmeans = KMeans(init="random", n_clusters=10, n_init=4)
```

اینجا یک شی از کلاس `Kmeans` را با پارامترهای مشخص شده ایجاد می کنیم که از آن برای انجام خوشه بندی استفاده کنیم.

```
kmeans.fit(reduced_data)
```

در این خط الگوریتم خوشه بندی `Kmeans` را روی داده های کاهش یافته اجرا می کنیم و مرکز خوشه ها را یافته و ذخیره می کنیم.

```
u = 0.02
```

در این خط یک مقدار ثابت را تعریف می کنیم که در مراحل بعد از آن استفاده کنیم.

```
x_min, x_max = reduced_data[:, 0].min() - 1, reduced_data[:, 0].max() + 1
```

با استفاده از این خط حداقل و حداکثر مقادیر ویژگی اول (درایه های ستون اول) داده های کاهش یافته محاسبه می کنیم و در متغیرهای `x_min` و `x_max` ذخیره می کنیم.

```
y_min, y_max = reduced_data[:, 1].min() - 1, reduced_data[:, 1].max() + 1
```

با استفاده از این خط حداقل و حداکثر مقادیر ویژگی دوم (درایه های ستون دوم) داده های کاهش یافته محاسبه می کنیم و در متغیرهای `y_min` و `y_max` ذخیره می کنیم.

```
xx, yy = np.meshgrid(np.arange(x_min, x_max, u), np.arange(y_min, y_max, u))
```

در این خط شبکه ای از نقاط را در فضای دو بعدی ایجاد می کنیم که از آن برای ترسیم مرز خوشه ها استفاده شود. این شبکه توسط تابع `meshgrid` و `arange` ایجاد میشود.

```
b = kmeans.predict(np.c_[xx.ravel(), yy.ravel()])
```

این خط برچسب های خوشه ها را برای نقاط شبکه روی داده های کاهش یافته پیش بینی میکند. برای اینکار ابتدا نقاط شبکه را تراز و سپس تابع `predict` را روی آن اجرا میکند.

```
b = b.reshape(xx.shape)
```

در این خط برچسب های خوشه ها را به شکل شبکه تغییر میدهد تا بتوانیم آن را به عنوان نقشه رنگی نمایش دهیم.

```
plt.figure
```

```
plt.clf()
```

با استفاده از دو خط فوق یک شکل جدید برای نمایش نمودار ایجاد میکنیم و سپس با استفاده از خط کد دوم تمام محتویات شکل فعلی را پاک میکنیم.

```
plt.imshow(
```

```
    b,
```

```
    interpolation="nearest",
```

```
    extent=(xx.min(), xx.max(), yy.min(), yy.max()),
```

```
    cmap=plt.cm.Paired,
```

```
    aspect="auto",
```

```
    origin="lower",
```

```
)
```

در این خطوط کد نقشه رنگی برچسب های خوشه ها را با استفاده از تابع `imshow` رسم میکنیم. همچنین پارامترهای مختلفی مانند تغییر اندازه تصویر و همچنین نحوه نمایش آن نیز قابل تنظیم میباشد.

```
plt.plot(reduced_data[:, 0], reduced_data[:, 1], "k.", markersize=2)
```

با استفاده از داده های ذخیره شده در آرایه «داده های کاهش شده»، یک نمودار پراکنده ترسیم می کنیم. به طور خاص مقادیر ستون اول "داده کاهش" را در محور `x` و مقادیر ستون دوم را در محور `y` ترسیم می کند. آرگومان `"k."` مشخص می کند که نشانگرها باید نقاط سیاه باشند و آرگومان `"markersize=2"` اندازه نشانگرها را 2 پیکسل تعیین می کند.

```
centroids = kmeans.cluster_centers_
```

با استفاده از این خط کد مرکز های خوشه بندی را در الگوریتم `kmeans` استخراج میکنیم. بعد از مساوی متغیری است که مرکز های خوشه ها را در خروجی الگوریتم `kmeans` ذخیره میکند.

```
Plt.scatter(  
    Centroids[:, 0],  
    Centroids[:, 1],  
    Marker="x",  
    S=169,  
    Linewidths=3,  
    Color="w",  
    Zorder=10,  
)
```

این خط کد مرکز های خوشه بندی را در نمودار نشان می دهد. با استفاده از `plt.scatter`` ، ما می توانیم مرکز های خوشه ها را با استفاده از مختصات آنها (در این حالت، مختصات ستون اول و دوم) روی نمودار نشان دهیم ``marker="x"` . مشخص می کند که نماد مرکزها از نوع `"x"` باشد ``s=169`` . تعیین می کند که اندازه نماد مرکزها 169 پیکسل باشد . ``linewidths=3`` تعیین می کند که ضخامت خطوط نماد مرکزها 3 پیکسل باشد ``color="w"` . مشخص می کند که رنگ خطوط نماد مرکزها سفید باشد ``zorder=10`` . تعیین می کند که نماد مرکزها در لایه 10 قرار بگیرد و روی سایر عناصر نمودار قرار بگیرد.

```
Plt.title(  
    "K-means clustering\n"  
    "MoeinHajmalek"  
)
```

```
Plt.xlim(x_min, x_max)  
Plt.ylim(y_min, y_max)  
Plt.xticks()  
Plt.yticks()  
Plt.show()
```

با دستور `plt.title` عنوانی را برای نمودار ایجاد میکنیم

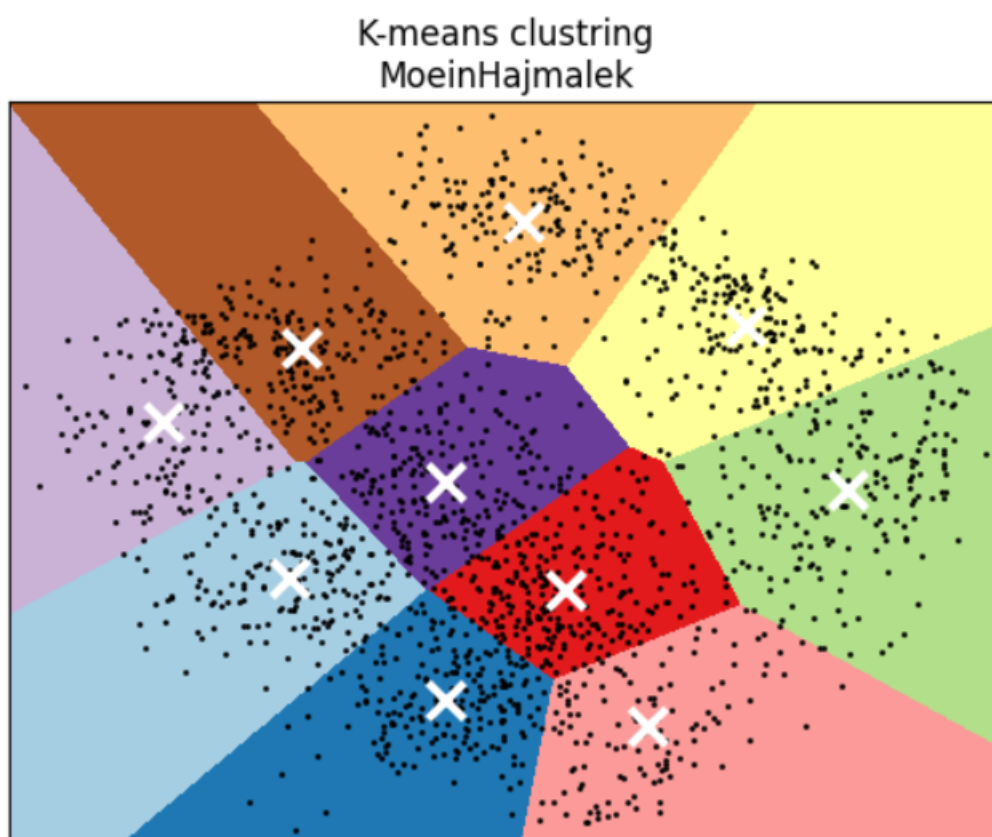
خط بعد محدوده محور x را در نمودار تنظیم می کند `plt.xlim(x_min, x_max)`. با استفاده از دو پارامتر `'x_max'` و `'x_min'`، محدوده مقادیر محور x را تعیین می کند. این کد می تواند استفاده شود تا محدوده نمایش داده شده در محور x را تغییر دهد و بخشی خاص از داده ها را در نمودار برجسته کند.

در خط بعد محدوده محور y را در نمودار تنظیم می کند `plt.ylim(y_min, y_max)`. با استفاده از دو پارامتر `'y_max'` و `'y_min'`، محدوده مقادیر محور y را تعیین می کند. این کد می تواند استفاده شود تا محدوده نمایش داده شده در محور y را تغییر دهد و بخشی خاص از داده ها را در نمودار برجسته کند.

دو خط بعد محورهای x و y را بدون تیکها (`tick`) و برچسبها (`label`) نشان می دهد.

و در خط آخر نمودار را نمایش می دهد. اجرای این خط باعث نمایش تمامی تغییرات و تنظیمات قبلی در نمودار می شود.

نتیجه کار :



نتیجه گیری :

با استفاده از الگوریتم Kmeans و کتابخانه پایتون ما توانستیم مثالی از خوشه بندی را حل کنیم ولی به ما نتیجه خوبی نمیدهد. چرا چون با یه صحیح و غلط کردن و بررسی کردن میشه متوجه شد که خیلی از دیتا های ما بد دسته بندی می شنود شاید این الگوریتم توانسته باشه این داده ها را دسته بندی کند ولی لزوما دسته بندی که انجام میشود یک دسته بندی خوب نیست در اینجا است که ما میفهمیم که چرا ما به شبکه های عصبی نیاز داریم با استناد به اینکه خواهیم یک همچنین نمونه از داده هارا دسته بندی و خوشه بندی کنیم بهترین روش استفاده از شبکه های عصبی است.

منابع :

7learn.com

Blog.faradars.org

Howsaw.org

Tosinso.com

Hooshio.com

Matlabsite.com

Xrio.ir

Owjtech.ir

Scikit-learn.org