دوره جامع پایتون:

بخش یادگیری ماشین

جلسه بیستم و دوم

دکتر ذبیح اله ذبیحی

جنگل تصادفی Random forest

- جنگل تصادفی یک الگوریتم یادگیری تحت نظارت است که از آن هم برای طبقه بندی و هم رگرسیون استفاده می شود.

عملکرد الگوریتم جنگل تصادفی:

- با کمک مراحل زیر می توانیم متوجه چگونگی عملکرد الگوریتم جنگل تصادفی شویم.
- مرحله ۱ :
- ابتدا، از مجموعه داده فراهم شده نمونه های تصادفی را انتخاب کنید.
- مرحله ۲ :
- سپس، این الگوریتم برای هر نمونه، یک درخت تصمیم گیری خواهد ساخت و در ادامه از هر درخت تصمیم گیری، نتیجه پیش بینی را خواهد گرفت.
- مرحله ۳ :
- در این مرحله، برای هر نتیجه پیش بینی، رای گیری انجام می شود.
- مرحله ۴ :
- در انتها، آن نتیجه پیش بینی که بیشترین تعداد رای را داشته باشد به عنوان نتیجه پیش بینی نهایی انتخاب می شود.

مثال

```
import pandas as pd
candidates = {'gmat':
[780,750,690,710,680,730,690,720,740,690,610,690,710,680,770,610,580,650,540,590,620,600,55
0,550,570,670,660,580,650,660,640,620,660,660,680,650,670,580,590,690],
'gpa':
[4,3.9,3.3,3.7,3.9,3.7,2.3,3.3,3.3,1.7,2.7,3.7,3.7,3.3,3.3,3,2.7,3.7,2.7,2.3,3.3,2,2.3,2.7,3,3.3,3.7,2.3,3.
7,3.3,3,2.7,4,3.3,3.3,2.3,2.7,3.3,1.7,3.7],
'work_experience': [3,4,3,5,4,6,1,4,5,1,3,5,6,4,3,1,4,6,2,3,2,1,4,1,2,6,4,2,6,5,1,2,4,6,5,1,2,1,4,5],
'admitted': [1,1,0,1,0,1,0,1,1,0,0,1,1,0,1,0,0,1,0,0,1,0,0,1,0,0,0,0,1,1,0,1,1,0,0,1,1,1,0,0,0,0,1]
}
df= pd.DataFrame(candidates,columns= ['gmat', 'gpa','work_experience','admitted'])
X = df[['gmat', 'gpa','work_experience']]
y = df['admitted']
```

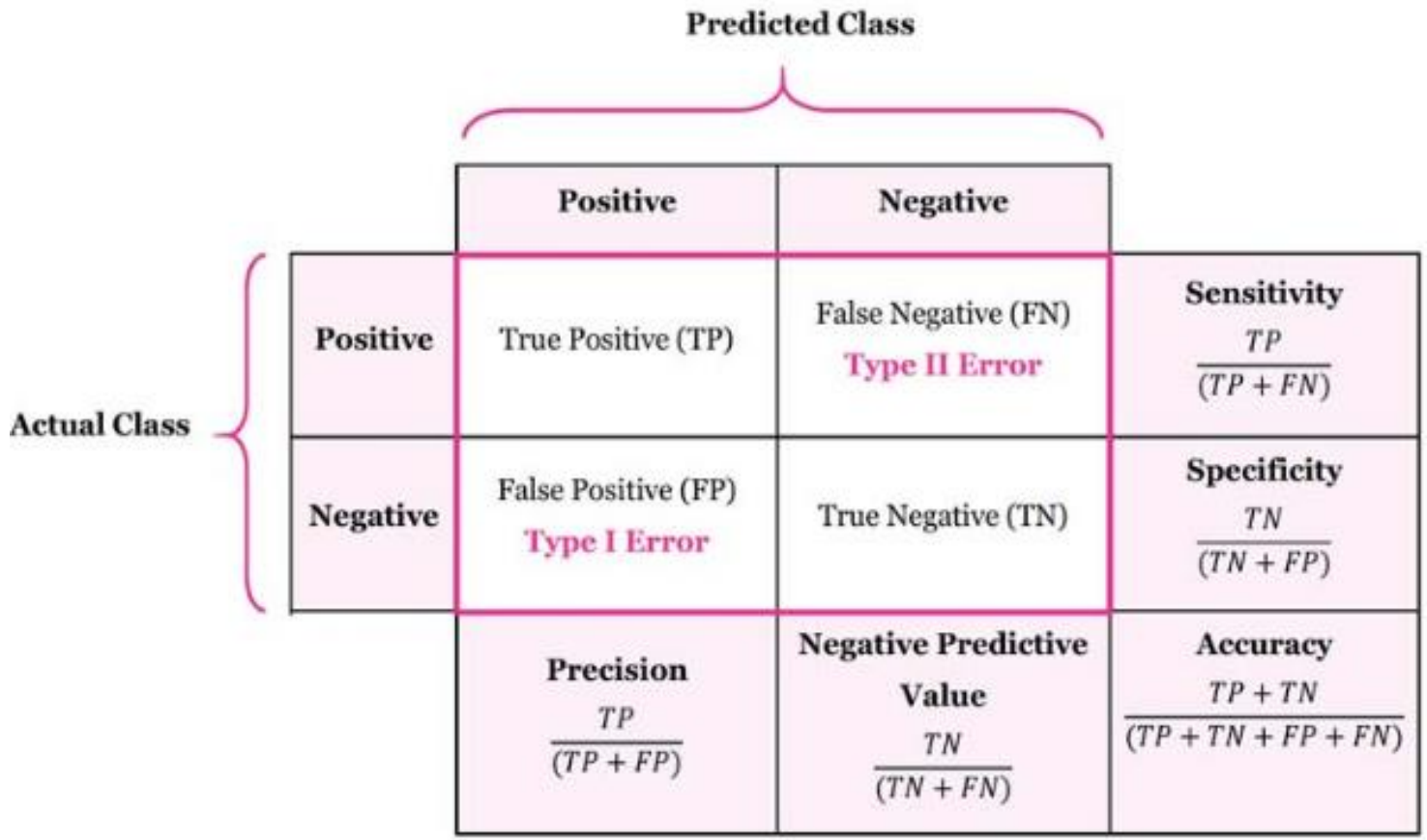
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.30)
```

داده را به دو قسمت یادگیری و تست تقسیم می کنیم. کد زیر مجموعه داده را به ۷۰ درصد داده یادگیری و ۳۰ درصد داده تست تقسیم می کند

```
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier(n_estimators = 50)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)


print(y_pred)
```

```
from sklearn.metrics import classification_report, confusion_matrix,
accuracy_score
result = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:")
print(result)
result1 = classification_report(y_test, y_pred)
print("Classification Report:",)
print (result1)
result2 = accuracy_score(y_test,y_pred)
print("Accuracy:",result2)
```


Predicted Class
Positive
Negative
Actual Class
Positive
True Positive (TP)
False Negative (FN)
Type II Error
Sensitivity
$\frac{TP}{(TP + FN)}$
Negative
False Positive (FP)
Type I Error
True Negative (TN)
Specificity
$\frac{TN}{(TN + FP)}$
Precision
$\frac{TP}{(TP + FP)}$
Negative Predictive Value
$\frac{TN}{(TN + FN)}$
Accuracy
$\frac{TP + TN}{(TP + TN + FP + FN)}$

یادگیری بدون نظارت

- در بحث یادگیری بدون ناظر هم معمولا با دو مساله مواجه می شویم: کلاسترینگ (خوشه بندی) و کاهش بعد. خوشه بندی در واقع همان جداسازی میوه های متفاوت بدون حضور ناظر است. کاهش بعد بیشتر در بحث پیش پردازش داده مطرح است و زمانی کاربرد دارد که پیچیدگی دیتاست زیاد است یا به بیان دیگر دارای ابعاد زیادی است که برای مدل یادگیری ماشین قابل فهم نیست. کاهش بعد کمک می کند تا تعداد ویژگی ها (یا بعد) داده کم شود.
- خوشه‌بندی، وظیفه گروه‌بندی یک مجموعه از نمونه‌ها به صورتی است که نمونه‌های داخل یک خوشه دارای بیشترین «مشابهت» (Similarity) با یکدیگر و کمترین مشابهت با داده‌های خارج از خوشه خودشان باشند. مشابهت، سنجه‌ای است که قدرت ارتباط بین دو نمونه داده را منعکس می‌کند.

خوشه بندی

خوشه‌بندی، اساسا برای «داده‌کاوی اکتشافی» Exploratory Data Miningمورد استفاده قرار می‌گیرد و کاربردهای گسترده‌ای در بسیاری از زمینه‌ها شامل یادگیری ماشین، «بازشناسی الگو» Pattern Recognition، «تحلیل تصویر» Image Analysis، «بازیابی اطلاعات» Information Retrieval، «بیوانفورماتیک» Bio-informatics، «فشرده‌سازی داده‌ها» Data Compressionو گرافیک کامپیوتری دارد.

بخش‌بندی تصاویر

خوشه‌بندی داده‌های بخش‌بندی ژن

خوشه‌بندی مقالات خبری

خوشه‌بندی گونه‌ها

تشخیص ناهنجاری

- معمولا در حالت چند متغیره، باید از ویژگی‌های مختلف اشیا به منظور طبقه‌بندی و خوشه کردن آن‌ها استفاده کرد. به این ترتیب با داده‌های چند بعدی سروکار داریم که معمولا به هر بعد از آن، ویژگی یا خصوصیت گفته می‌شود. با توجه به این موضوع، استفاده از توابع فاصله مختلف در این جا مطرح می‌شود. ممکن است بعضی از ویژگی‌های اشیا کمی و بعضی دیگر کیفی باشند. به هر حال آنچه اهمیت دارد روشی برای اندازه‌گیری میزان شباهت یا عدم شباهت بین اشیاء است که باید در روش‌های خوشه‌بندی لحاظ شود.

انواع الگوریتم های خوشه بندی

- در مجموع، پنج نوع الگوریتم خوشه بندی مجزا وجود دارد که به شرح زیر هستند :

- خوشه بندی مبتنی بر پارتیشن بندی (بخش بندی)
- خوشه بندی سلسله مراتبی
- خوشه بندی مبتنی بر مدل
- خوشه بندی مبتنی بر تراکم
- خوشه بندی فازی

- خوشه بندی مبتنی بر پارتیشن بندی (بخش بندی)
- در این نوع خوشه بندی ، الگوریتم، داده ها را به زیرمجموعه ای از kگروه، تقسیم بندی می کند. این kگروه یا خوشه باید از قبل تعریف شده باشند. این الگوریتم، داده ها را بر اساس این دو شرط تقسیم بندی می کند – اول، هر گروه باید حداقل یک نقطه (عضو) داشته باشد. دوم اینکه هر نقطه باید تنها به یک گروه تعلق داشته باشد. خوشه بندی K-Meansرایج ترین نوع روش خوشه بندی مبتنی بر پارتیشن بندی است.

- خوشه بندی سلسله مراتبی
- ایده اصلی این نوع خوشه بندی ، ایجاد سلسله ای از خوشه ها است. برخلاف خوشه بندی مبتنی بر پارتیشن بندی ، نیازی نیست داده ها از پیش تعریف شده باشند. دو روش برای انجام خوشه بندی سلسله مراتبی وجود دارد. رویکرد اول، رویکرد پایین به بالا است که به روش تجمعی نیز شناخته می شود و رویکرد دوم روش تجزیه ای است که سلسله ای از خوشه ها را در یک رویکرد بالا به پایین تجزیه می کند. در نتیجه این نوع خوشه بندی ، ما یک نمودار درختی به نام Dendogramبه دست می آوریم.

- خوشه بندی مبتنی بر تراکم
- در این نوع خوشه ها، مناطق متراکمی در فضای داده وجود دارند که توسط مناطق پراکنده از یکدیگر جدا می شوند. این نوع از الگوریتم های خوشه بندی نقش مهمی در ارزیابی و پیدا کردن ساختار های اشکال غیر خطی براساس تراکم دارند. الگوریتم پرطرفدار مبتنی بر تراکم، DBSCAnاست که امکان خوشه بندی مکانی داده ها دارای نویز را فراهم می آورد. این روش از دو مفهوم استفاده می کند – دسترسی داده ها و اتصال داده ها.

- خوشه بندی مبتنی بر مدل
- در این نوع روش خوشه بندی ، داده های مشاهده شده از یک توزیع متشکل از ترکیبی از دو یا چند مولفه خوشه حاصل می شود. علاوه بر این، هر خوشه مولفه، یک تابع چگالی دارد که دارای یک احتمال یا وزن در این ترکیب است.

- خوشه بندی فازی
- در این نوع خوشه بندی ، نقاط داده می توانند به بیش از یک دسته تعلق داشته باشند. هر مولفه موجود در خوشه، یک ضریب عضویت دارد که به میزان حضور در آن خوشه مرتبط است. همچنین روش خوشه بندی فازی به عنوان روش خوشه بندی نرم شناخته می شود.

روش k_mean

- الگوریتم K-Means، با فرض داشتن ورودی‌های x1، x2، x3، ...، xn به شکل زیر کار می‌کند.

- گام اول: انتخاب Kنقطه تصادفی به عنوان مرکز خوشه‌ها که به آن «مرکزوار» Centroid گفته می‌شود.
- گام دوم: هر xiبه نزدیک‌ترین خوشه با محاسبه فاصله آن از هر مرکزوار تخصیص داده می‌شود.
- گام سوم: پیدا کردن مرکز خوشه‌های جدید با محاسبه میانگین نقاط تخصیص داده شده به یک خوشه
- گام ۴: تکرار گام ۲ و ۳ تا هنگامی که هیچ یک از نقاط تخصیص داده شده به خوشه‌ها تغییر نکنند.

مثال الگوریتم kmean

- فرض کنیم سن مراجعه کنندگان یک فروشگاه در یک روز بصورت زیر باشد (تعداد نمونه n=19)

15	15	16	19
19	20	20	21
22	28	35	40
41	42	43	44
60	61	65	

خوشه اولیه (centroidیا میانگین تصادفی):

$$Distance\ 1 = |x_i - c_1|$$

$$Distance\ 2 = |x_i - c_2|$$

تکرار اول

x_i	c_1	c_2	Distance 1	Distance 2	Nearest Cluster	New Centroid
15	16	22	1	7	1	
15	16	22	1	7	1	**15.33**
16	16	22	0	6	1	
19	16	22	9	3	2	
19	16	22	9	3	2	
20	16	22	16	2	2	
20	16	22	16	2	2	
21	16	22	25	1	2	
22	16	22	36	0	2	
28	16	22	12	6	2	
35	16	22	19	13	2	
40	16	22	24	18	2	**36.25**
41	16	22	25	19	2	
42	16	22	26	20	2	
43	16	22	27	21	2	
44	16	22	28	22	2	
60	16	22	44	38	2	
61	16	22	45	39	2	
65	16	22	49	43	2	

تکرار دوم

x_i	c_1	c_2	Distance 1	Distance 2	Nearest Cluster	New Centroid
15	15.33	36.25	0.33	21.25	1	
15	15.33	36.25	0.33	21.25	1	
16	15.33	36.25	0.67	20.25	1	
19	15.33	36.25	3.67	17.25	1	
19	15.33	36.25	3.67	17.25	1	**18.56**
20	15.33	36.25	4.67	16.25	1	
20	15.33	36.25	4.67	16.25	1	
21	15.33	36.25	5.67	15.25	1	
22	15.33	36.25	6.67	14.25	1	
28	15.33	36.25	12.67	8.25	2	
35	15.33	36.25	19.67	1.25	2	
40	15.33	36.25	24.67	3.75	2	
41	15.33	36.25	25.67	4.75	2	
42	15.33	36.25	26.67	5.75	2	
43	15.33	36.25	27.67	6.75	2	**45.9**
44	15.33	36.25	28.67	7.75	2	
60	15.33	36.25	44.67	23.75	2	
61	15.33	36.25	45.67	24.75	2	
65	15.33	36.25	49.67	28.75	2	

تکرار سوم

x_i	c_1	c_2	Distance 1	Distance 2	Nearest Cluster	New Centroid
15	18.56	45.9	3.56	30.9	1	
15	18.56	45.9	3.56	30.9	1	
16	18.56	45.9	2.56	29.9	1	
19	18.56	45.9	0.44	26.9	1	
19	18.56	45.9	0.44	26.9	1	
20	18.56	45.9	1.44	25.9	1	**19.50**
20	18.56	45.9	1.44	25.9	1	
21	18.56	45.9	2.44	24.9	1	
22	18.56	45.9	3.44	23.9	1	
28	18.56	45.9	9.44	17.9	1	
35	18.56	45.9	16.44	10.9	2	
40	18.56	45.9	21.44	5.9	2	
41	18.56	45.9	22.44	4.9	2	
42	18.56	45.9	23.44	3.9	2	
43	18.56	45.9	24.44	2.9	2	**47.89**
44	18.56	45.9	25.44	1.9	2	
60	18.56	45.9	41.44	14.1	2	
61	18.56	45.9	42.44	15.1	2	
65	18.56	45.9	46.44	19.1	2	

تکرار چهارم

x_i	c_1	c_2	Distance 1	Distance 2	Nearest Cluster	New Centroid
15	19.5	47.89	4.50	32.89	1	
15	19.5	47.89	4.50	32.89	1	
16	19.5	47.89	3.50	31.89	1	
19	19.5	47.89	0.50	28.89	1	
19	19.5	47.89	0.50	28.89	1	
20	19.5	47.89	0.50	27.89	1	**19.50**
20	19.5	47.89	0.50	27.89	1	
21	19.5	47.89	1.50	26.89	1	
22	19.5	47.89	2.50	25.89	1	
28	19.5	47.89	8.50	19.89	1	
35	19.5	47.89	15.50	12.89	2	
40	19.5	47.89	20.50	7.89	2	
41	19.5	47.89	21.50	6.89	2	
42	19.5	47.89	22.50	5.89	2	
43	19.5	47.89	23.50	4.89	2	**47.89**
44	19.5	47.89	24.50	3.89	2	
60	19.5	47.89	40.50	12.11	2	
61	19.5	47.89	41.50	13.11	2	
65	19.5	47.89	45.50	17.11	2	

- بین تکرارهای ۳ و ۴ تغییری نکرده است. با استفاده از خوشه بندی، دو گروه ۱۵ تا ۲۸ و ۳۵-۶۵ شناسایی شدند. انتخاب اولیه centroidsمی تواند بر خوشه های خروجی تأثیر بگذارد، بنابراین الگوریتم اغلب چند بار با شرایط مختلف شروع می شود تا دیدگاه مناسبی از خوشه ها داشته باشد.

مثال

- ما دو دسته ماشین (مثلا پراید و اتوبوس) داریم. ما فقط اطلاعات ماشین ها (طول و ارتفاع) را داریم اما نوع ماشین را نمی دانیم (برای اطلاعات برچسب نداریم. حال با روش k- میانگین می خواهیم نوع ماشین را مشخص کنیم.

ارتفاع ماشین	طول ماشین	
۳	۷	۱
۴	۷	۲
۳	۸	۳
۳	۹	۴
۲	۵	۵
۳	۴	۶
۴	۳	۷

- داده ها را به دو خوشه دسته بندی می کنیم.
- دو نقطه رندم تولید میکنیم و فاصله تا داده ها تا این نقطه را محاسبه میکنیم و داده ها را در دو دسته تقسیم میکنیم
- در مرحله بعد دوباره دو نقطه رندم تولید میکنیم و فاصله تا داده ها تا این نقطه را محاسبه میکنیم و داده ها را در دو دسته تقسیم میکنیم
- در مرحله بعد مرحله فوق مجدد تکرار میشه
- شروط پایان الگوریتم:
- ۱- الگوریتم وقتی به این حالت رسید که در چند دورِ متوالی تغییری در خوشه‌ی نمونه‌ها (در این‌جا ماشین‌ها) به وجود نیامد، به ایم معنی است که الگوریتم دیگر نمی‌تواند زیاد تغییر کند و این حالتِ پایانی برای خوشه‌هاست.
- ۲- الگوریتم‌های مبتنی بر تکرار iterative algorithmsمی‌توان تعدادِ تکرارها را محدود کرد تا الگوریتم بی‌نهایت تکرار نداشته باشد.

```
from sklearn.cluster import KMeans
import numpy as np
X = np.array([[7, 3], [7, 4], [8, 3],[9, 3], [5, 2], [4, 3],[3,3]])
model = KMeans(n_clusters=2, max_iter=300).fit(X)
print(model.labels_)
print(model.predict([[8,4]]))
print(model.cluster_centers_)
```

بصورت پیش فرض max_iter=300 است.

مثال: اگر یک محصول با وزن ۱۲۵ ، اندازه ۴۵ و رنگ آبی دیده شود مربوط به کدام دسته A یا B خواهد بود.

برحسب	رنگ	اندازه (سانتی متر)	وزن (کیلوگرم)	نمونه
A	1	50	120	1
B	2	20	60	2
A	1	65	145	3
A	3	45	130	4
B	2	15	50	5

روش یادگیری بدون نظارت خوشه بندی k_mean

```
import numpy as np
dataset = np.array([[120, 50, 1, 0],[60, 20, 2, 1],
                    [145, 65, 1, 0],[130, 45, 3, 0],
                    [50, 15, 2, 1]])
features = dataset[:, :3]

from sklearn.cluster import KMeans
model = KMeans(n_clusters=2)
model.fit(features)
print (model.labels_)
```

تعداد بهینه خوشه ها در الگوریتم های خوشه بندی

- روش های مستقیم تعیین تعداد بهینه خوشه ها در الگوریتم های خوشه بندی:
- این روش ها به دنبال بهینه سازی یک معیار به‌خصوص، مانند مجموع مربعات فواصل درون خوشه ای Within-cluster Sum of Squareیا WSS یا سیلوئت میانگین Average Silhouette هستند. از جمله این متدها می‌توان به متد elbow و روش های مبتنی بر معیار silhouette اشاره کرد.

- روش های مبتنی بر آزمون های آماری در تعیین تعداد بهینه خوشه ها در الگوریتم های خوشه بندی: این متدها به دنبال تطبیق مشاهدات با فرض صفر یک آزمون آماری هستند. از جمله این روش ها می‌توان به Gap Statisticsاشاره کرد.

- علاوه بر متدهای elbow، silhouetteو gap statistics، بیش از ۳۰ مورد روش و شاخص مختلف برای تعیین تعداد بهینه خوشه ها در مقالات مختلف ارائه شده است.

متد elbow

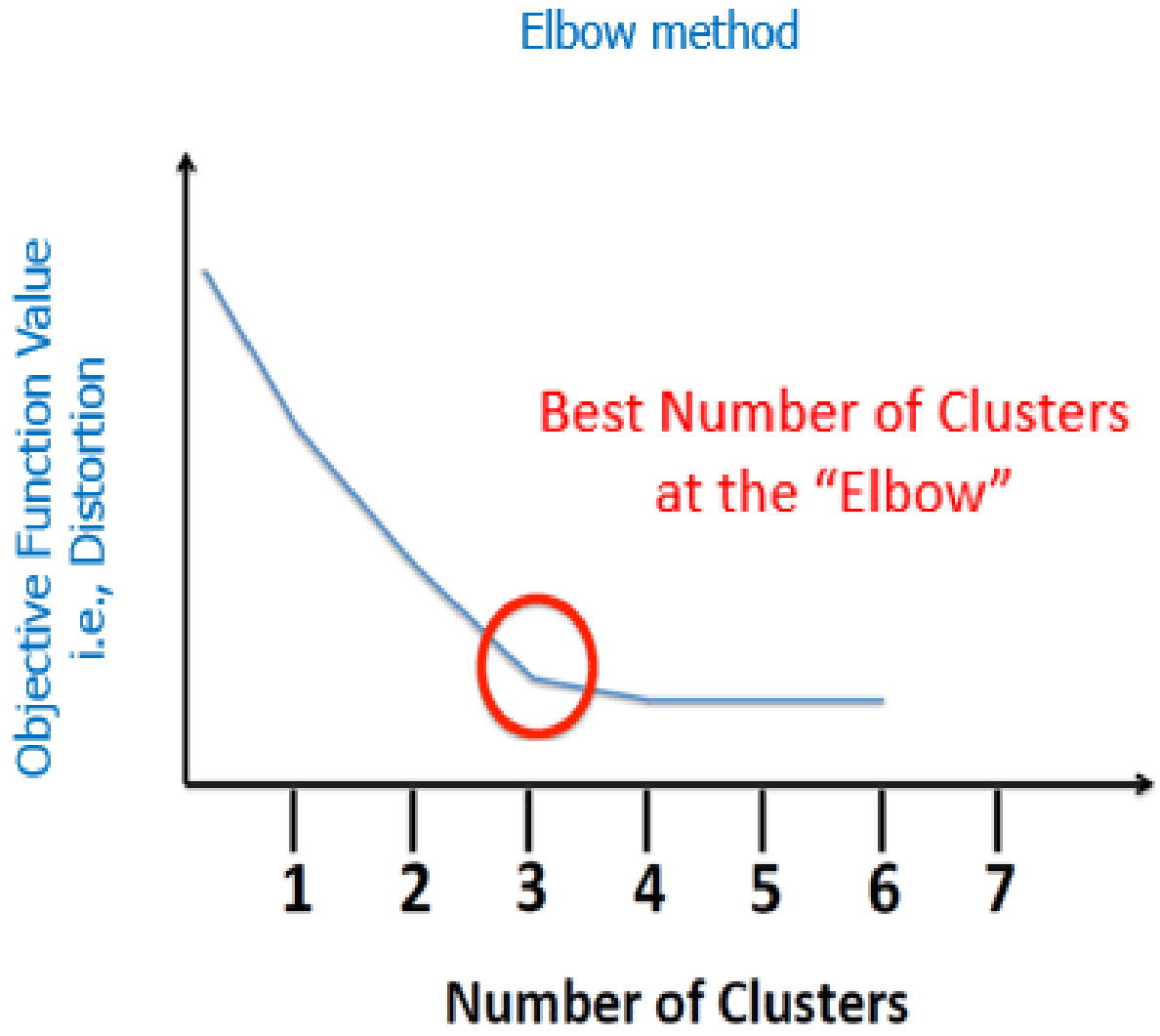
- همانطور که می‌دانیم ایده اصلی روش های خوشه بندی مبتنی بر تقسیم مانند k-means به دست آوردن تعداد خوشه ها به نحوی است که مجموع فواصل درون خوشه ای داده ها (یا مجموع مربعات فواصل درون خوشه ای) حداقل شود. مجموع فواصل درون خوشه ای داده ها، میزان فشردگی خوشه بندی انجام شده را نشان می‌دهد و هدف، حداقل سازی آن تا جای ممکن است.
- روش elbow، مجموع فواصل درون خوشه ای داده ها را به عنوان تابعی از تعداد خوشه ها در نظر می‌گیرد. به این ترتیب تعداد خوشه ها به نحوی انتخاب می‌شوند که افزودن یک خوشه دیگر، بهبودی در حداقل‌سازی WSS ایجاد نکند.
- تعداد بهینه خوشه ها طبق الگوریتم زیر به‌دست می‌آید:

۱- اجرای الگوریتم خوشه بندی مانند k-means برای مقادیر متفاوت) kبه‌طور مثال با در نظر گرفتن مقدار k در بازه ۱ تا ۱۰)

۲- محاسبه مقدار WSS برای هر مقدار k

۳-رسم مقدار WSS بر حسب مقادیر مختلف k

۴-نقطه زانوئی نمودار رسم شده، تعداد بهینه خوشه ها را نشان می‌دهد.


Elbow method
Objective Function Value
i.e., Distortion
Best Number of Clusters
at the "Elbow"
1
2
3
4
5
6
7
Number of Clusters

متد silhouetteمیانگین

- تمرکز معیار سیلوئت بر کیفیت خوشه بندی انجام شده متکی است. در واقع این معیار مشخص می‌کند که پراکندگی داده ها در خوشه ها به چه صورت است. هر چه مقدار سیلوئت بالاتر باشد، کیفیت خوشه بندی نیز بالاتر است.
- در متد سیلوئت میانگین ، الگوریتم خوشه بندی به ازای مقادیر مختلف kاجرا شده و به‌ازای هر اجرا، معیار سیلوئت برای هر یک از اعضای خوشه ها محاسبه می‌شود. سپس از سیلوئت های به‌دست آمده معدل گرفته می‌شود. مقدار بهینه kمقداری است که به ازای آن، سیلوئت میانگین ماکزیمم شود.
- الگوریتم این متد مشابه روش elbowبوده و می‌توان آن را به صورت زیر نوشت:

۱- اجرای الگوریتم خوشه بندی مانند k-meansبرای مقادیر متفاوت) kبه طور مثال با در نظر گرفتن مقدار k در بازه ۱ تا ۱۰)

۲- محاسبه مقدار سیلوئت هر یک از مشاهدات برای هر مقدار kو محاسبه میانگین آن‌ها

۳- رسم مقدار میانگین سیلوئت بر حسب مقادیر مختلف k

۴- نقطه ماکزیمم نمودار رسم شده، تعداد بهینه خوشه ها را نشان می‌دهد.

متد Gap Statistics

- این روش می‌تواند بر انواع مختلف الگوریتم های خوشه بندی اعمال شود. متد Gap Statisticsبه‌ازای هر یک از مقادیر در نظر گرفته شده برای k، مجموع تفاضلات درون خوشه ای داده ها را با مقادیر مورد انتظار آن‌ها (توزیع داده ها با فرض درست بودن فرض صفر) مقایسه می‌کند. مقدار بهینه خوشه ها، مقداری است که آماره gapرا ماکزیمم کند. این به آن معناست که ساختار خوشه بندی از توزیع یکنواخت تصادفی داده ها دور است.

- الگوریتم این روش را می‌توان به فرم زیر نوشت:

۱- خوشه بندی داده ها با در نظر گرفتن مقدار kدر بازه ۱ تا . kmaxمحاسبه مقدار تفاضلات درون خوشه ای برای هر یک از اجراها و قرار دادن آن در متغیر Wk

۲- -تولید تعداد Bمجموعه داده از مجموعه داده های اصلی، به نحوی که دارای توزیع یکنواخت تصادفی باشند. خوشه بندی هر یک از این Bمجموعه داده به‌ازای مقادیر مختلف kدر بازه ۱ تا .kmax محاسبه مقدار تفاضلات درون خوشه ای برای هر یک از اجراها و قرار دادن آن در متغیر Wkb

۳-محاسبه آماره gapبه صورت تفاضل مقادیر مشاهده شده Wkاز مقادیر مورد انتظار آن‌ها تحت فرض صفر Wkbو نیز محاسبه انحراف معیار آماره به‌دست آمده:

$$Gap(k) = \frac{1}{B}\sum_{b=1}^{B} log(W_{kb}^{*}) - log(W_k)$$

۴- انتخاب تعداد بهینه خوشه ها به صورت کم‌ترین مقدار kبه‌طوری‌که $Gap(k) \geq Gap(k+1) - s_{k+1}$

elbow method

```
from sklearn.cluster import KMeans
import numpy as np
import matplotlib.pyplot as plt
X = np.array([[7, 3], [7, 4], [8, 3],[9, 3], [5, 2], [4, 3],[3,3]])
#from sklearn.preprocessing import StandardScaler
#scaler = StandardScaler()
#scaled_features = scaler.fit_transform(X)
sse = []
for k in range(1, 7):
  model = KMeans(n_clusters=k)
  model.fit(X)
  sse.append(model.inertia_)
plt.plot(range(1, 7), sse)
plt.xticks(range(1, 7))
plt.xlabel("Number of Clusters")
plt.ylabel("SSE")
plt.show()
```

silhouette coefficient

```
from sklearn.cluster import KMeans
import numpy as np
import matplotlib.pyplot as plt
X = np.array([[7, 3], [7, 4], [8, 3],[9, 3], [5, 2], [4, 3],[3,3]])
#from sklearn.preprocessing import StandardScaler
#scaler = StandardScaler()
#scaled_features = scaler.fit_transform(X)
from sklearn.metrics import silhouette_score
silhouette_coefficients = []
for k in range(2, 7):
  model = KMeans(n_clusters=k)
  model.fit(X)
  score = silhouette_score(X, model.labels_)
  silhouette_coefficients.append(score)
plt.plot(range(2, 7), silhouette_coefficients)
plt.xticks(range(2, 7))
plt.xlabel("Number of Clusters")
plt.ylabel("Silhouette Coefficient")
plt.show()
```