K Means

مقدمه :

اگر چه امروز بیشتر کاربردهای یادگیری ماشین مبتنی بر یادگیری نظارت شده است ، اما اکثر داده های موجود در واقع بدون Label هستند. به عبارت دیگر ویژگی های ورودی x را در داده ها داریم اما برچسب y را در داده ها نداریم.

یکی از کارهایی که می توان به واسطه الگوریتم های یادگیری بدون ناظر انجام داد ، Label زدن برای داده های بدون Label است. به این روش اصطلاحا خوشه بندی یا Clustering می گویند.

یکی دیگر از اهدافی که الگوریتم های یادگیری بدون ناظر دارند ، تشخیص ناهنجاری است. در این روش نمونه های معیوب را از بین نمونه های سالم و عادی پیدا می کنند.

خوشه بندی (Clustering) :

خوشه بندی شناسایی نمونه های مشابه و اختصاص دادن نمونه های مشابه به خوشه های یکسان است.

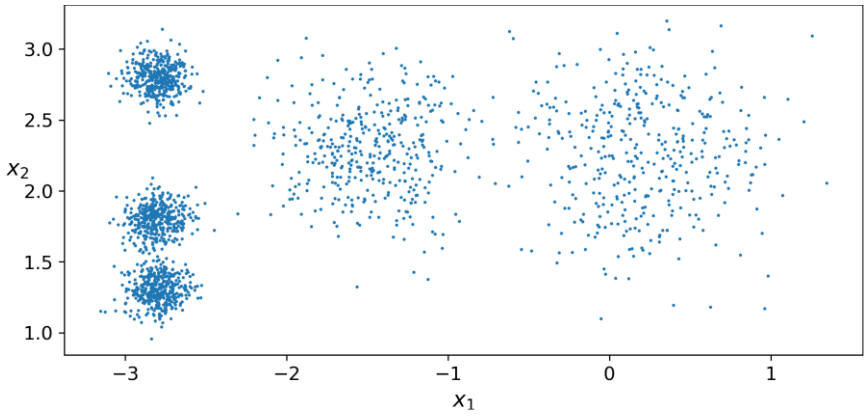
خوشه بندی درست مانند طبقه بندی است که هر نمونه به یک گروه اختصاص می یابد اما بدون ناظر.

یعنی در طبقه بندی داده های آموزش دارای Label بودند و مدل بر اساس آن ها فرآیند آموزش خود را انجام می داد اما در خوشه بندی داده ها دارای Label نیستند و هدف مدل آن است که داده ها را خوشه بندی کند تا برای هر خوشه یک Label قرار دهد. دلیل این کار آن است که در مدل های طبقه بندی حتما داده ها باید Label داشته باشند.

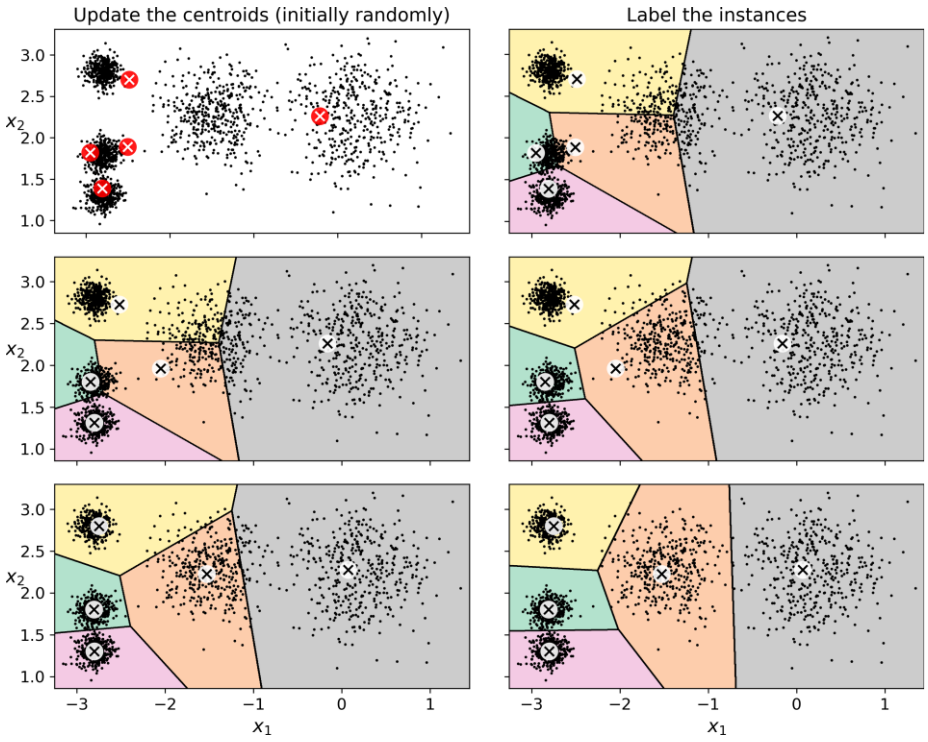
از خوشه بندی در برنامه های مختلفی اعم از : تقسیم بندی مشتری ها ، تجزیه و تحلیل داده ها ، یک روش برای کاهش بعد ، تشخیص ناهنجاری ، یادگیری نیمه نظارت شده ، موتورهای جستجو ، تقسیم بندی تصاویر و ... استفاده می شود.

❖ K Means :

الگوریتم K Means محبوب ترین و پرکاربردترین الگوریتم برای خوشه بندی داده ها است.
برای مثال تصور کنید پراکندگی داده های بدون Label به شکل زیر باشد :

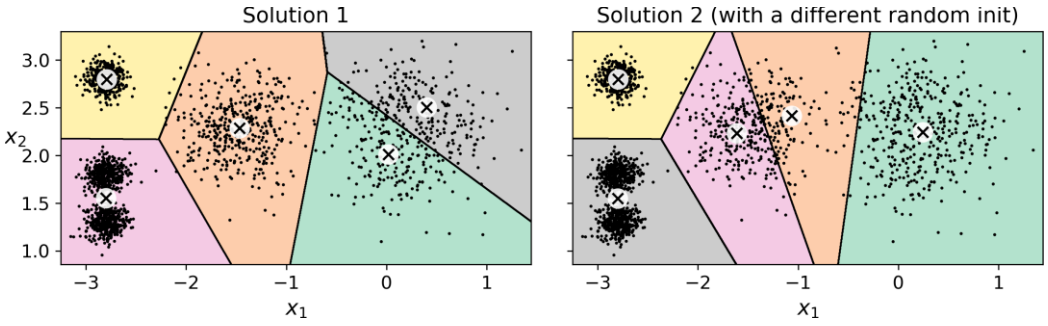


قصد داریم که داده ها را خوشه بندی کنیم. الگوریتم K Means به این صورت کار می کند که ابتدا به صورت تصادفی K تا مرکز خوشه را انتخاب می کند. تعداد مرکز خوشه یا همان K نیز به طور تصادفی انتخاب می شود که درباره آن جلوتر صحبت خواهیم کرد.
سپس بر اساس فاصله داده های دیگر از مراکز خوشه ها آن ها در خوشه های مختلف دسته بندی می کند. انتخاب مراکز خوشه ها به صورت تصادفی آن قدر ادامه می یابد تا الگوریتم همگرا شود.



اگر چه الگوریتم همگرا می شود ولی لزوما به راه حل درست همگرا نمی شود. درست همگرا شدن الگوریتم به مقادیر اولیه مراکز خوشه ها بستگی دارد.

برای مثال :



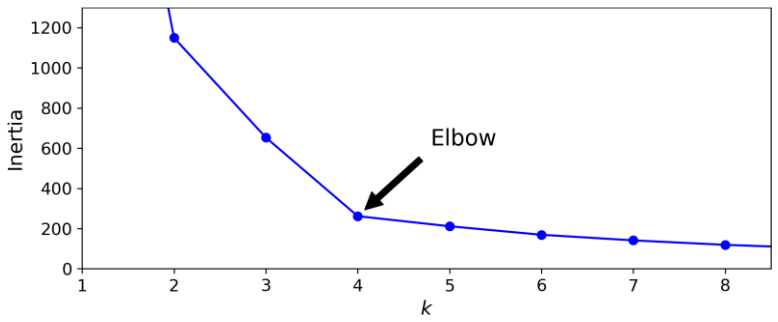
در انتخاب مراکز خوشه ها دو حالت کلی وجود دارد.

۱- مراکز خوشه ها را از قبل می دانیم. در این حالت آرایه ای از مختصات مراکز خوشه ها تنظیم می کنیم و در هنگام پیاده سازی مدل K Means آن را به مدل می دهیم.

۲- مراکز خوشه ها را نمی دانیم. در این حالت کاملا به صورت تصادفی مراکز را انتخاب می کنیم و می توانیم مشخص کنیم که مراکز خوشه ها چندین بار به صورت تصادفی انتخاب

شوند. سپس مدل هر کدام از حالاتی که بهترین معیار عملکرد را داشته باشد به عنوان مراکز خوشه نگه می دارد.
معیار عملکردی که برای تعیین بهترین مراکز خوشه ها استفاده می شود را اینرسی مدل می گویند. اینرسی مدل ، میانگین فاصله مربع بین هر نمونه و نزدیکترین مرکز آن است. اینرسی مدل اگر کمتر باشد آن مرکز خوشه ها به نسبت مابقی مراکز خوشه مناسب تر هستند.

موضوع مهم بعدی یافتن تعداد بهینه خوشه ها است. به طور کلی یافتن تعداد بهینه خوشه ها یا همان مقدار K ، کار ساده ای نیست و اگر این مقدار به درستی پیدا نشود اثر بسیار بدی بر روی مدل خواهد داشت.
اگر بر اساس اینرسی مدل بخواهیم تعداد K بهینه را پیدا کنیم ، برای مثال :



هر چه اینرسی کمتر باشد بهتر است پس اگر تعداد خوشه ها را ۸ قرار دهیم اینرسی کمتری خواهیم داشت اما می دانیم بر اساس پراکندگی داده ها تعداد خوشه ها اگر برابر ۸ قرار گیرد برخی از خوشه های خوب بی دلیل به دو خوشه تقسیم می شوند و حالت بهینه مدل از بین می رود. به همین دلیل استفاده از اینرسی مدل برای پیدا کردن بهینه ترین تعداد خوشه ها روشی نسبتا ناهنجار است.
روش دیگری وجود دارد که دقیق تر است اما از نظر هزینه محاسباتی گران تر است. در این روش به جای استفاده از اینرسی مدل از امتیاز Silhouette استفاده می شود.
امتیاز Silhouette که میانگین ضریب Silhouette برای تمام نمونه ها است برابر است با :

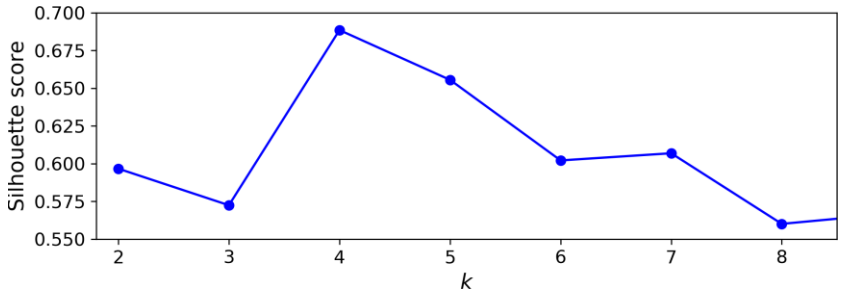
$$(b-a) \Big/ \max(a,b)$$

که a میانگین فاصله نسبت به نمونه های دیگر در همان خوشه است (میانگین فاصله درون خوشه).

و b میانگین نزدیک ترین فاصله خوشه است (میانگین فاصله تا نمونه های نزدیک ترین خوشه بعدی).

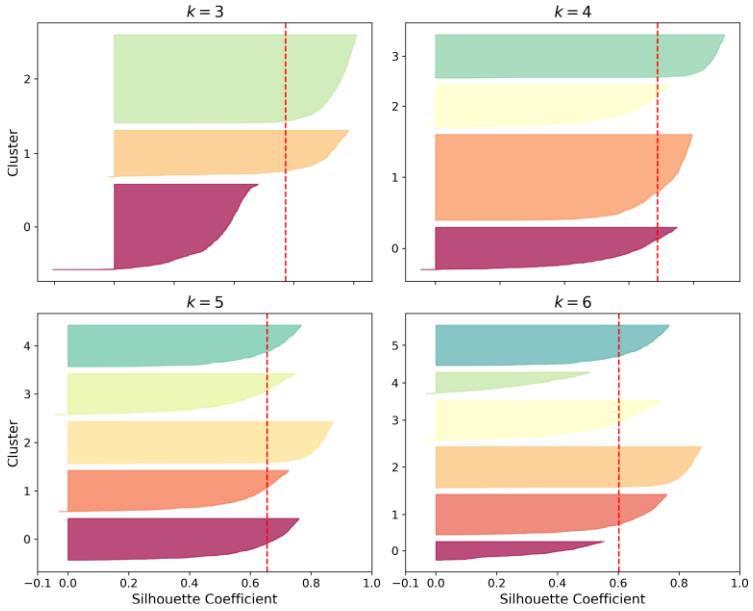
Silhouette مقداری بین بازه 1- تا 1 می گیرد. هر چه به 1 نزدیک تر باشد به این معنی است که نمونه کاملا در خوشه خودش است. هر چه به 1- نزدیک تر باشد به این معنی است که نمونه به خوشه اشتباه تعلق گرفته است. هر چه به 0 نزدیک تر باشد به این معنی است که به مرکز خوشه نزدیک است.

برای مثال :



نمودار بالا اثر تعداد خوشه بر امتیاز Silhouette را نشان می دهد.

همچنین نمودار زیر اثر تعداد خوشه بر ضریب Silhouette را نشان می دهد.



Written by Hossein Kariznoei

الگوریتم K Means با وجود شایستگی های فراوان از جمله سریع بودن ، بی عیب و نقص نیست. همانطور که قبل تر به آن پرداختیم برای رسیدن به راه حل بهینه لازم است چندین بار الگوریتم را اجرا کنیم و تعداد خوشه ها را مشخص کنیم که خیلی ساده نیست. علاوه بر این ها K Means برای داده هایی که خوشه های آن ها دارای اندازه ها و تراکم های مختلف و یا اشکال غیر کروی هستند ، خیلی خوب کار نمی کند.

پروژه :

می خواهیم با استفاده از الگوریتم K Means برای داده ها Label قرار دهیم.

داریم :

```
import matplotlib.pyplot as plt
from sklearn.datasets._samples_generator import make_blobs
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
```

```
X, y = make_blobs(n_samples=500, n_features=2, centers=5, cluster_std=0.75, random_state=1)
plt.figure(figsize=(8, 5))
plt.scatter(X[:, 0], X[:, 1], s=30)
```

```
...
```

```
k_means = KMeans(n_clusters=5)
k_means.fit(X)
```

```
...
```

```
y_pred = k_means.predict(X)
y_pred
```

```
...
```

```
plt.figure(figsize=(8, 5))
plt.scatter(X[:, 0], X[:, 1], c=y, cmap="viridis")
centers = k_means.cluster_centers_
plt.scatter(centers[:, 0], centers[:, 1], c="red", s=100)
```

```
...
```

```
k_means.inertia_
```

```
...
```

```
silhouette_score(X, k_means.labels_)
```