Nombre: Mauricio Josiel Villalobos Rodriguez

Materia: Seminario de IA 2

Profesor: CARLOS ALBERTO VILLASENOR PADILLA

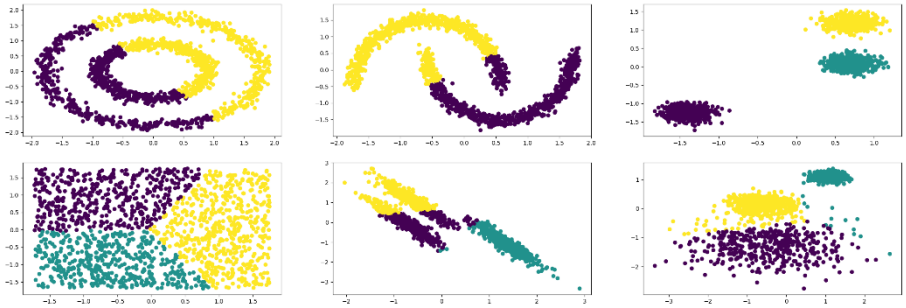
Fecha de entrega: 05/10/2022

Actividad: A10 Comparacion de algoritmos de agrupacion

Algoritmo Kmeans

```python
#kmeans funciona mejor particularmente en situaciones donde las muestras sean redondidas
for c, x in zip(n_cluster, X):
    model = cluster.KMeans(n_clusters=c, n_init=10)  # Establecer n_init en 10
    model.fit(x)
    if hasattr(model, "labels_"):
        y.append(model.labels_.astype(int))
    else:
        y.append(model.predict(x))
```

Kmeans



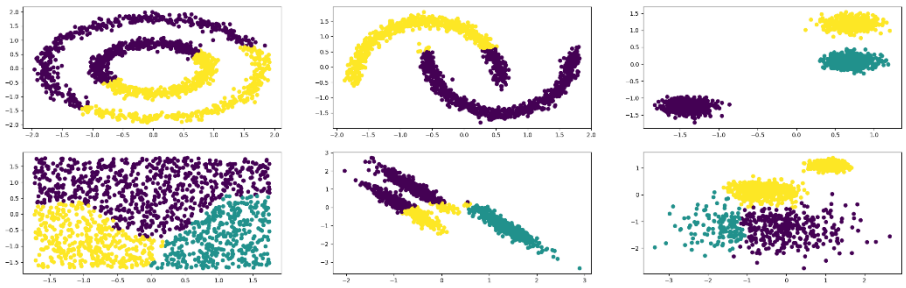
Este algoritmo en particular funciona mejor para situaciones donde los elementos se encuentran mas separadas y redondos

Birch

Al igual que el kmeans también nos da buenos resultados en el caso de la agrupación de las boltas

```python
#birch
for c, x in zip(n_cluster,X):
    model = cluster.Birch(n_clusters = c)
    model.fit(x)
    if hasattr(model, "labels_"):
        y.append(model.labels_.astype( int))
    else:
        y.append(model.predict(x))
```

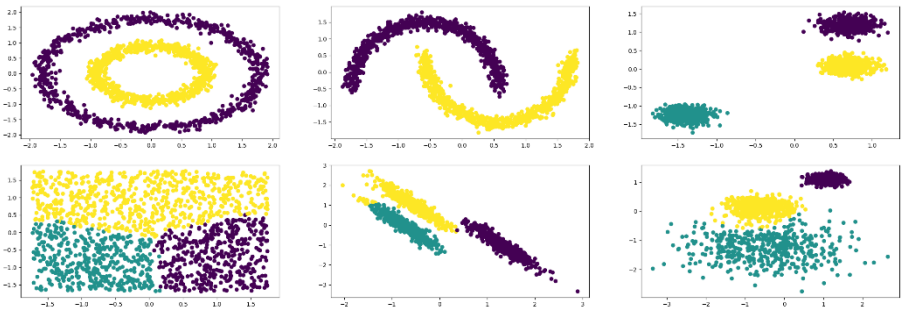
birch



Spectral Clustering

```
for c, x in zip(n_cluster,X):
    model = cluster.SpectralClustering(n_clusters = c, affinity = "nearest_neighbors")
    model.fit(x)
    if hasattr(model, "labels_"):
        y.append(model.labels_.astype(int))
    else:
        y.append(model.predict(x))
```

Spectral Clustering

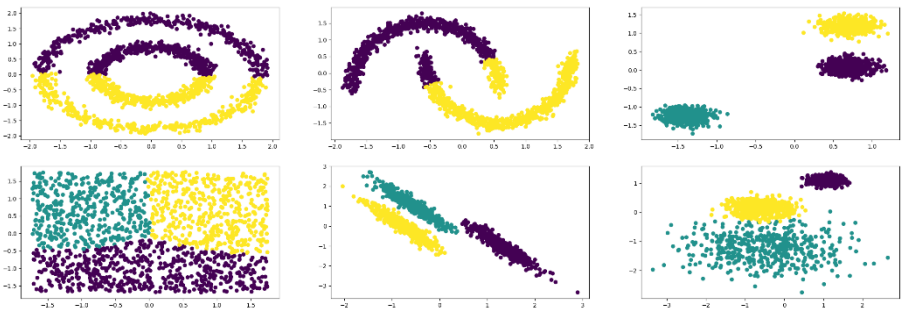


Aquí es donde empieza las situaciones interesantes, porque como podemos ver nos da muy buenos resultados en casos donde los algoritmos pasados no, como en el de los espirales y las líneas, aun que su potencial es mayor en los círculos y el espiral

Gaussian mixture

```
#Gaussian Mixture
for c, x in zip(n_cluster,X):
    model = mixture.GaussianMixture(n_components = c, covariance_type= "full")
    model.fit(x)
    if hasattr(model, "labels_"):
        y.append(model.labels_.astype(int))
    else:
        y.append(model.predict(x))
```

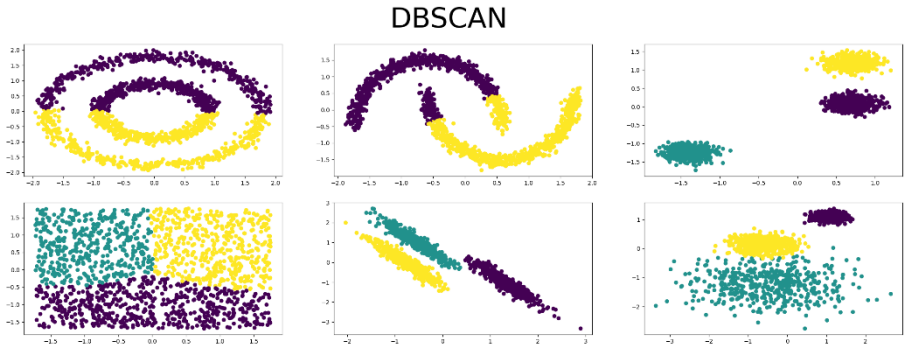
Gaussian Mixture



El Gaussian mixtrure es bueno para las líneas, pero no nos da resultados diferentes en las demás

DBSCAN

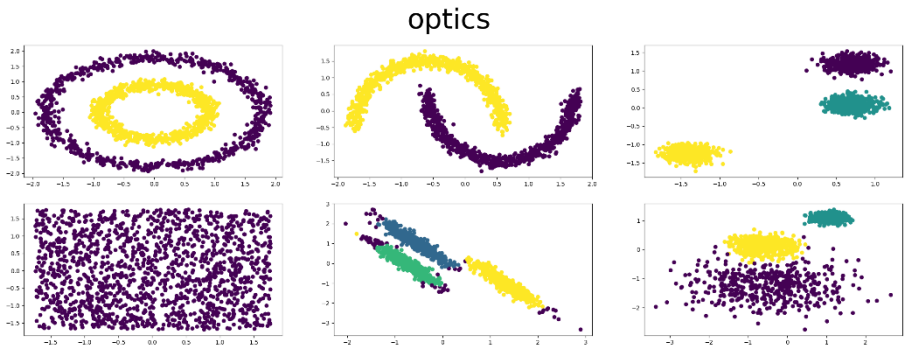
```
eps = [0.3,0.3, 0.3,0.3,0.15, 0.18]
for e, x in zip(eps,X):
    model = cluster.DBSCAN(eps = e)
    model.fit(x)
    if hasattr(model, "labels_"):
        y.append(model.labels_.astype( int))
    else:
        y.append(model.predict(x))
```



El DBSCAN si da resultados buenos pero suele ayudarnos a detectar puntos muy lejanos a los cluster

Optics

```
for c, x in zip(n_cluster,X):
    model = cluster.OPTICS(min_samples=20, xi=0.05, min_cluster_size=0.1)
    model.fit(x)
    if hasattr(model, "labels_"):
        y.append(model.labels_.astype( int))
    else:
        y.append(model.predict(x))
```



Optics es un algoritmo muy versátil y eficiente cuando se trabaja con conjuntos de formas irregulares