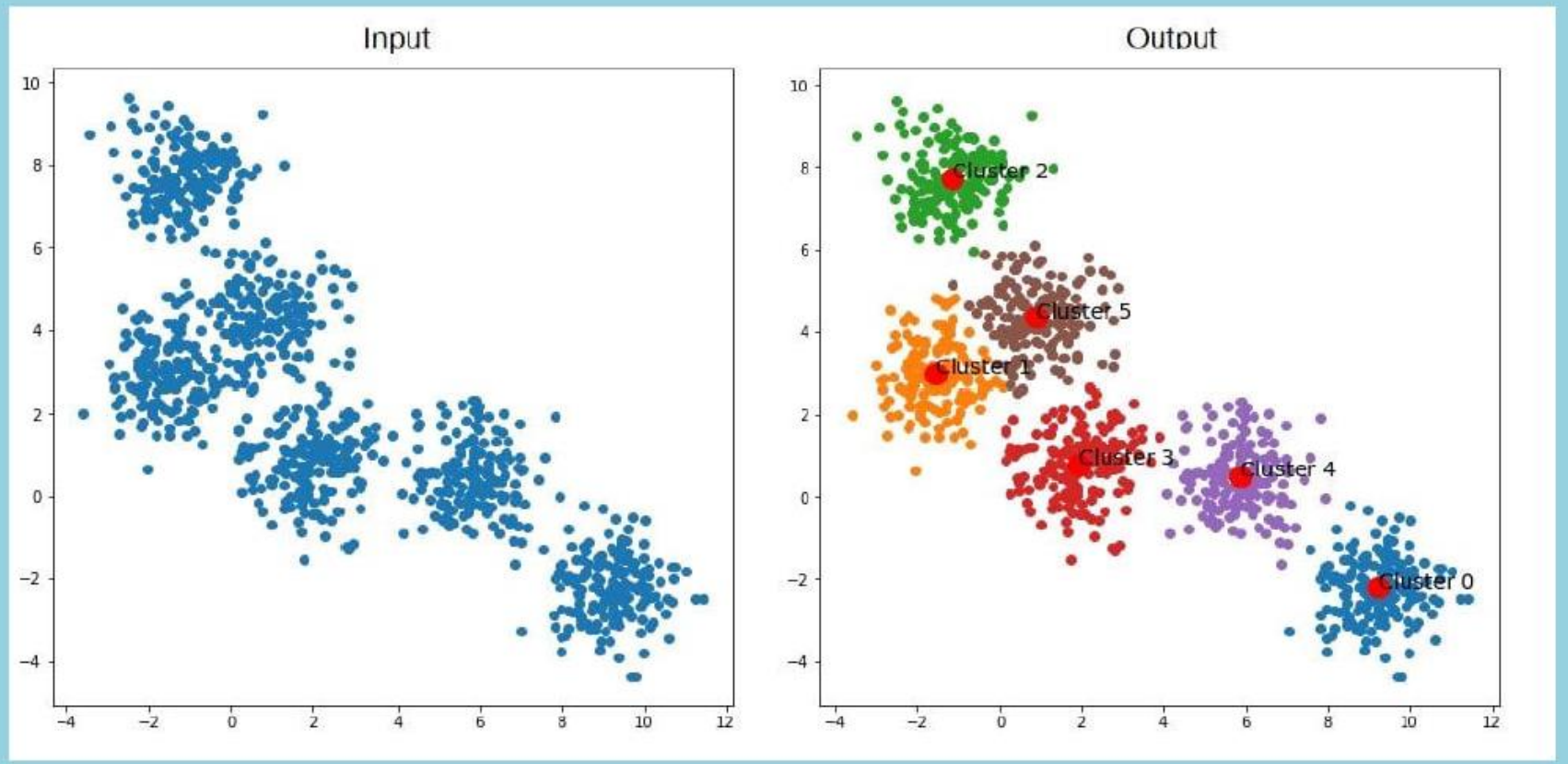
Apuntes Ayudantía
Algoritmos de Segmentación

By Natalie Julian

En un problema de agrupación, ¿cuál es el output?

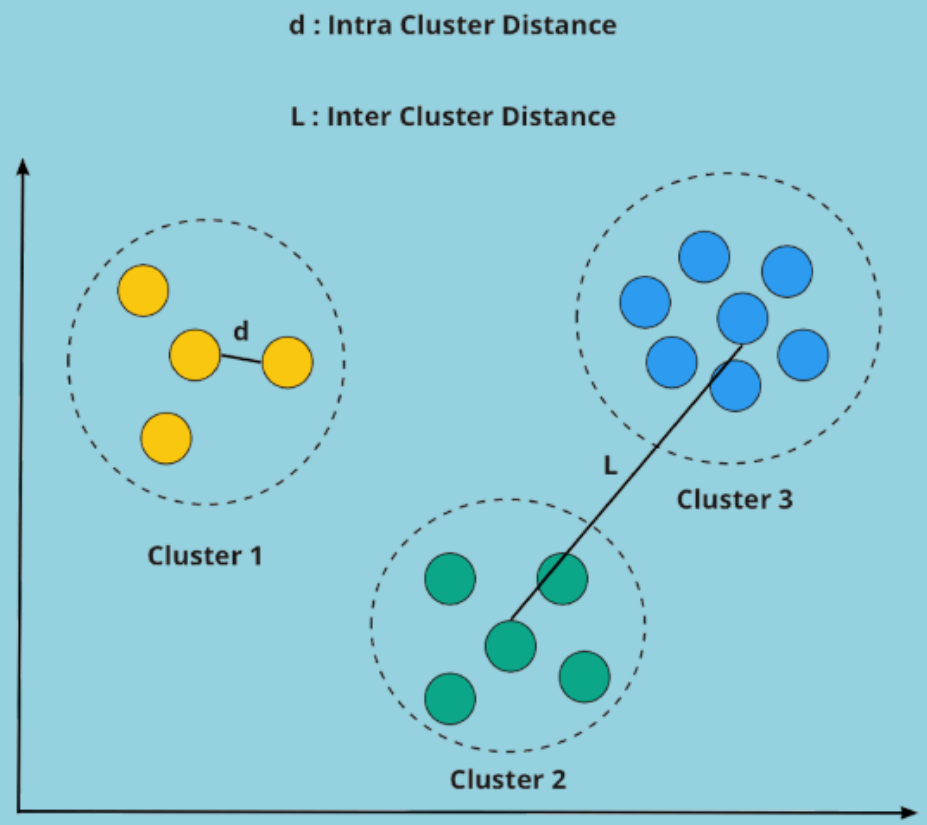
El output de los algoritmos de agrupación son precisamente, los grupos c_i y la caracterización de cada grupo (las coordenadas del centro) de modo que cada observación pertenezca a un grupo.



Fuente Imagen: www.linkedin.com/company/data-science-dojo

Sumas cuadráticas en K-Means

Dos conceptos muy importantes al realizar clustering es la varianza intra clúster e inter clúster:



Distancia intra clústers (Within Cluster Sums of Squares - WSS)

Indica qué tan separados están los elementos del mismo grupo y qué tan lejos están de su centroide. Los grupos más densos con elementos más cercanos tienen mayor similitud y son más probables para compartir realmente la misma calificación.

Se calcula:

$$WSS = \sum_{i=1}^{N_c} \sum_{x \in C_i} d(x, \bar{x}_{C_i})^2$$

Donde N_c es el número de clústers, C_i corresponde al clúster, x la observación y $\bar{x}_{C_i}$ el centroide del clúster.

Ejemplo en Python - Datos Mezclados

Utilizaremos las tres variables categóricas que creamos y además dos variables numéricas, Cantidad de Ceniza y Matiz del vino.

```
vinos_num=vinos[['Ceniza', 'Matiz del vino']] #Variables numéricas

from sklearn.preprocessing import StandardScaler #Estandarizamos

sc = StandardScaler()
vinosnum_stand = sc.fit_transform(vinos_num) #Cuando se estandariza se pierde toda la información de nombre de columnas y otros

vinosnum_stand = pd.DataFrame(vinosnum_stand, columns = vinos[['Ceniza', 'Matiz del vino']].columns) #Añadimos nombres de las variables

vinos_proto=pd.concat([vinosnum_stand, vinos_cat1], axis=1).drop('clusters', axis=1) #Unimos las variables categóricas y las numéricas estandarizadas

print(vinos_proto)
```

```
       Ceniza  Matiz del vino  alcohol_cat  flavonoides_cat  intensidad_cat
0    0.229683        0.354487            0                0               1
1   -0.827683        0.398599            0                1               1
2    1.104744        0.310376            0                0               1
3    0.484909       -0.439520            0                0               2
4    1.833962        0.354487            0                1               1
..        ...             ...          ...              ...             ...
172  0.302605       -1.409974            0                2               2
173  0.411987       -1.145305            0                2               2
174 -0.390152       -1.630531            0                2               2
175  0.010918       -1.586420            0                2               2
176  1.359971       -1.542308            0                2               2

[177 rows x 5 columns]
```

```
from kmodes.kprototypes import KPrototypes

cost = [] #Guardamos el costo para cada valor de K
for num_clusters in list(range(1,15)):
    kproto = KPrototypes(n_clusters=num_clusters)
    kproto.fit_predict(vinos_proto, categorical=[2,3,4])
    cost.append(kproto.cost_)
```

Ejemplo en Python - Gráfico de Elbow

```
import seaborn as sns #Gráfico con seaborn
sns.set_theme(style="whitegrid", palette="bright", font_scale=1.2)

plt.figure(figsize=(15, 7))
ax = sns.lineplot(x=range(1,15), y=cost, marker="o", dashes=False)
ax.set_title('Gráfico de Elbow para la elección del k óptimo', fontsize=18)
ax.set_xlabel('K', fontsize=14)
ax.set_ylabel('Costo', fontsize=14)
ax.set(xlim=(1-0.1, 15+0.1))
plt.plot()
```