- Dataset can be found at Mall Customer Segmentation Data (https://www.kaggle.com/vjchoudhary7/customer-segmentation-tutorial-in-python)
- More about K-Means clustering at Hierarchical Clustering (https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html#sklearn.cluster.AgglomerativeClustering)

In [1]:
```
import numpy as np
import pandas as pd

from sklearn.cluster import AgglomerativeClustering
from matplotlib import pyplot as plt
```

In [2]:
```
df = pd.read_csv("Mall_Customers.csv")
df.head()
```

Out[2]:

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

What does the dataset contain ?

The data is about customers like Customer ID, age, gender, annual income and spending score. Spending Score is assigned to the customer based on a defined parameters like customer behavior and purchasing data.

In [3]:
```
df.drop(columns=["CustomerID"], inplace=True)
df['Gender'] = df['Gender'].apply(lambda x: 1 if x == "Male" else 0)
df.head()
```

Out[3]:

	Gender	Age	Annual Income (k$)	Spending Score (1-100)
0	1	19	15	39
1	1	21	15	81
2	0	20	16	6
3	0	23	16	77
4	0	31	17	40

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 4 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Gender                  200 non-null    int64
 1   Age                     200 non-null    int64
 2   Annual Income (k$)      200 non-null    int64
 3   Spending Score (1-100)  200 non-null    int64
dtypes: int64(4)
memory usage: 6.4 KB
```

What is the algorithm

Hierarchical clustering is an unsupervised clustering method that you can use to group your data. This algorithm is unsupervised because it uses a random, unlabelled dataset. The resulting clusters are displayed in a hierarchical tree called a dendrogram. This is helpful because the algorithm produces a clear graphical depiction of your clusters that you can understand and interpret easily. With this algorithm, you can create decision trees as well as category hierarchies for your business.

How does it work

Hierarchical clustering is a general family of clustering algorithms that build nested clusters by merging or splitting them successively. This hierarchy of clusters is represented as a tree (or dendrogram). The root of the tree is the unique cluster that gathers all the samples, the leaves being the clusters with only one sample.
The AgglomerativeClustering object performs a hierarchical clustering using a bottom up approach: each observation starts in its own cluster, and clusters are successively merged together.

Advantages and Disadvantges of the algorithm

Advantages:
* Hierarchical Clustering preserves the structure of the dataset by adding connectivity constraints.
* Output of the algorithm can be used to understand the big picture as well as the groups in your data.

Disadvantages:
* It does not always provide the best solution: When you cluster multi-dimensional retail data that cannot always be visualised on a plot, poor solutions may be tricky to spot and resolve.
* The algorithm cannot run if there is missing data: You will need to remove these lines or estimate values to ensure the algorithm is able to run.
* The algorithm cannot run with different data types: When you use many different data types, it becomes difficult to compute a distance matrix. There is no simple formula that can work with both qualitative and numerical data at the same time.
* The dendrogram can be misinterpreted: The descriptors and composition of clusters may be difficult to interpret for all your business stakeholders involved with clustering.

How is it performed on the dataset

In [5]:
```
X = df.values
```

In [6]:
```
X.shape
```

Out[6]: (200, 4)

In [7]:
```
algo = AgglomerativeClustering(n_clusters=5)
algo.fit(X)
labels = algo.labels_
df['Clusters'] = labels
```

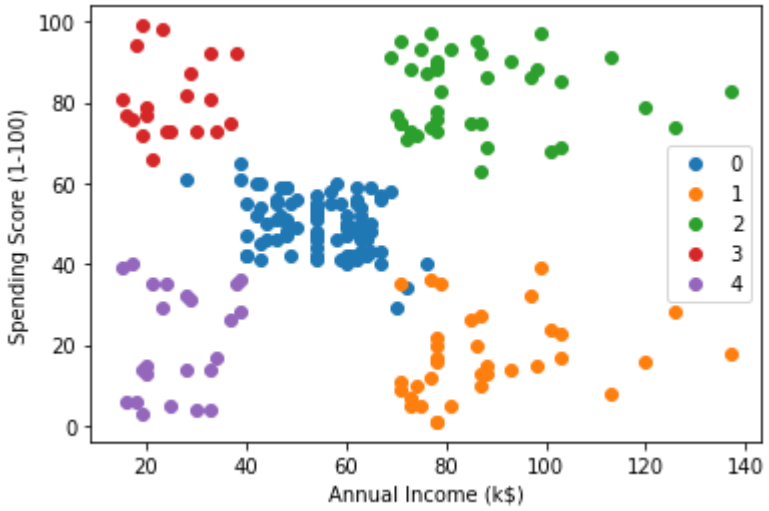
In [8]:
```
df.head()
```

Out[8]:

	Gender	Age	Annual Income (k$)	Spending Score (1-100)	Clusters
0	1	19	15	39	4
1	1	21	15	81	3
2	0	20	16	6	4
3	0	23	16	77	3
4	0	31	17	40	4

In [9]:

```
for i in np.unique(labels):
    label = df['Clusters']
    dx = df[label == i]
    plt.scatter(dx.iloc[:, 2], dx.iloc[:, 3], label=i)
plt.xlabel("Annual Income (k$)")
plt.ylabel("Spending Score (1-100)")
plt.legend()
plt.show();
```



Summary

- Hierarchical clustering with 5 clusters is found to be optimal
- The results are very similar to K-Means
- Annual Income and Spending Score are found to be the important features
- Fitting the model with just these 2 features might result in better clusters