Unsupervised Learning

- It is divided into 2 types:
  - Clustering
  - Association

Clustering:

K means Clustering Model

- K Means is a simple and easy way to classify the given dataset through a certain number of clusters
- K means algorithm takes dataset with constant K value and returns K centroids
  - centroid defines the cluster of data in the datset which are similar to one another

In [1]:
```
# sno x y
#----------------
#1   1  1
#2  1.5 2
#3  3   4
#4  5   7
#5 3.5  5
#6 4.5  5
```

Working on K Means

- step1 : randomly take K number of clusters
- step2 : calculate the distance
- step3 : calculate centroids
- step4 : Divide into number of groups
- step5 : same process will repeats

In [2]:
```
#k =2
# distance between (1,1),(1,1) and (5,7),(1,1)
#                   (x1,y1),(x2,y2)and (x1,y1),(x2,y2)
```

In [3]:
```
import math

math.sqrt((1-1)**2+((1-1)**2))
```

Out[3]: 0.0

In [4]:
```
math.sqrt((1-5)**2+((1-7)**2))
```

Out[4]: 7.211102550927978

In [5]:
```
# distance between (1.5,2),(1,1) and (1.5,2),(5,7)
```

In [6]:
```
import pandas as pd
```

In [69]:
```
dataset = pd.read_csv("https://raw.githubusercontent.com/AP-State-Skill-Deve
```

In [70]:
```
dataset.head()
```

Out[70]:

	CustomerID	Genre	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

In [72]:
```
#dataset["CustomerID"].value_counts()
```

In [10]:
```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 5 columns):
CustomerID                200 non-null int64
Genre                     200 non-null object
Age                       200 non-null int64
Annual Income (k$)        200 non-null int64
Spending Score (1-100)    200 non-null int64
dtypes: int64(4), object(1)
memory usage: 7.9+ KB
```

In [11]:
```
x = dataset.iloc[:,[3,4]].values
```

In [12]:
```
1 x
```

Out[12]:
```
array([[ 15,  39],
       [ 15,  81],
       [ 16,   6],
       [ 16,  77],
       [ 17,  40],
       [ 17,  76],
       [ 18,   6],
       [ 18,  94],
       [ 19,   3],
       [ 19,  72],
       [ 19,  14],
       [ 19,  99],
       [ 20,  15],
       [ 20,  77],
       [ 20,  13],
       [ 20,  79],
       [ 21,  35],
       [ 21,  66],
       [ 23,  29],
       [ 23,  98],
       [ 24,  35],
       [ 24,  73],
       [ 25,   5],
       [ 25,  73],
       [ 28,  14],
       [ 28,  82],
       [ 28,  32],
       [ 28,  61],
       [ 29,  31],
       [ 29,  87],
       [ 30,   4],
       [ 30,  73],
       [ 33,   4],
       [ 33,  92],
       [ 33,  14],
       [ 33,  81],
       [ 34,  17],
       [ 34,  73],
       [ 37,  26],
       [ 37,  75],
       [ 38,  35],
       [ 38,  92],
       [ 39,  36],
       [ 39,  61],
       [ 39,  28],
       [ 39,  65],
       [ 40,  55],
       [ 40,  47],
       [ 40,  42],
       [ 40,  42],
       [ 42,  52],
       [ 42,  60],
       [ 43,  54],
       [ 43,  60],
```

```
       [ 43,  45],
       [ 43,  41],
       [ 44,  50],
       [ 44,  46],
       [ 46,  51],
       [ 46,  46],
       [ 46,  56],
       [ 46,  55],
       [ 47,  52],
       [ 47,  59],
       [ 48,  51],
       [ 48,  59],
       [ 48,  50],
       [ 48,  48],
       [ 48,  59],
       [ 48,  47],
       [ 49,  55],
       [ 49,  42],
       [ 50,  49],
       [ 50,  56],
       [ 54,  47],
       [ 54,  54],
       [ 54,  53],
       [ 54,  48],
       [ 54,  52],
       [ 54,  42],
       [ 54,  51],
       [ 54,  55],
       [ 54,  41],
       [ 54,  44],
       [ 54,  57],
       [ 54,  46],
       [ 57,  58],
       [ 57,  55],
       [ 58,  60],
       [ 58,  46],
       [ 59,  55],
       [ 59,  41],
       [ 60,  49],
       [ 60,  40],
       [ 60,  42],
       [ 60,  52],
       [ 60,  47],
       [ 60,  50],
       [ 61,  42],
       [ 61,  49],
       [ 62,  41],
       [ 62,  48],
       [ 62,  59],
       [ 62,  55],
       [ 62,  56],
       [ 62,  42],
       [ 63,  50],
       [ 63,  46],
       [ 63,  43],
       [ 63,  48],
       [ 63,  52],
```

```
       [ 63,  54],
       [ 64,  42],
       [ 64,  46],
       [ 65,  48],
       [ 65,  50],
       [ 65,  43],
       [ 65,  59],
       [ 67,  43],
       [ 67,  57],
       [ 67,  56],
       [ 67,  40],
       [ 69,  58],
       [ 69,  91],
       [ 70,  29],
       [ 70,  77],
       [ 71,  35],
       [ 71,  95],
       [ 71,  11],
       [ 71,  75],
       [ 71,   9],
       [ 71,  75],
       [ 72,  34],
       [ 72,  71],
       [ 73,   5],
       [ 73,  88],
       [ 73,   7],
       [ 73,  73],
       [ 74,  10],
       [ 74,  72],
       [ 75,   5],
       [ 75,  93],
       [ 76,  40],
       [ 76,  87],
       [ 77,  12],
       [ 77,  97],
       [ 77,  36],
       [ 77,  74],
       [ 78,  22],
       [ 78,  90],
       [ 78,  17],
       [ 78,  88],
       [ 78,  20],
       [ 78,  76],
       [ 78,  16],
       [ 78,  89],
       [ 78,   1],
       [ 78,  78],
       [ 78,   1],
       [ 78,  73],
       [ 79,  35],
       [ 79,  83],
       [ 81,   5],
       [ 81,  93],
       [ 85,  26],
       [ 85,  75],
       [ 86,  20],
       [ 86,  95],
```

```
       [ 87,  27],
       [ 87,  63],
       [ 87,  13],
       [ 87,  75],
       [ 87,  10],
       [ 87,  92],
       [ 88,  13],
       [ 88,  86],
       [ 88,  15],
       [ 88,  69],
       [ 93,  14],
       [ 93,  90],
       [ 97,  32],
       [ 97,  86],
       [ 98,  15],
       [ 98,  88],
       [ 99,  39],
       [ 99,  97],
       [101,  24],
       [101,  68],
       [103,  17],
       [103,  85],
       [103,  23],
       [103,  69],
       [113,   8],
       [113,  91],
       [120,  16],
       [120,  79],
       [126,  28],
       [126,  74],
       [137,  18],
       [137,  83]], dtype=int64)
```

In [14]:
```
1 dataset.isnull().sum()
```

Out[14]:
```
CustomerID                0
Genre                     0
Age                       0
Annual Income (k$)        0
Spending Score (1-100)    0
dtype: int64
```

In [35]:
```
1 from sklearn.cluster import KMeans
```

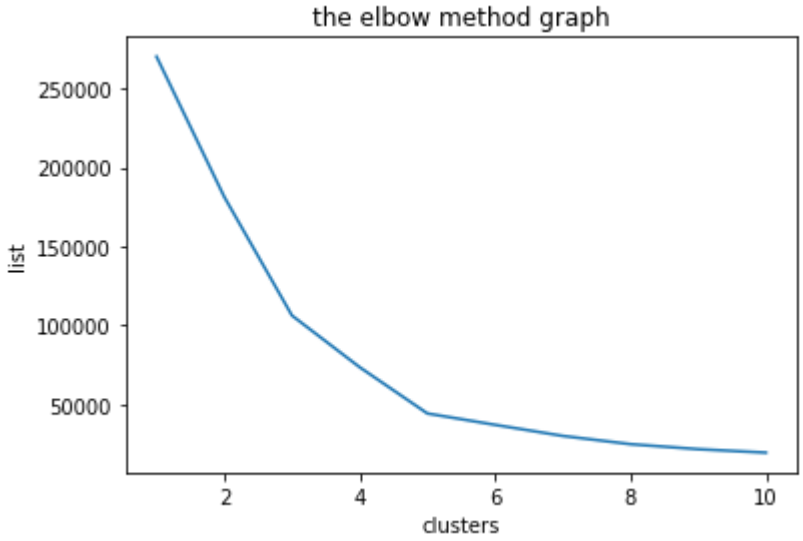
In [64]:
```
1
2 print(dir((KMeans)))
```

```
['__class__', '__delattr__', '__dict__', '__dir__', '__doc__', '__eq__', '__for
mat__', '__ge__', '__getattribute__', '__getstate__', '__gt__', '__hash__', '__
init__', '__init_subclass__', '__le__', '__lt__', '__module__', '__ne__', '__ne
w__', '__reduce__', '__reduce_ex__', '__repr__', '__setattr__', '__setstate__',
'__sizeof__', '__str__', '__subclasshook__', '__weakref__', '_check_test_data',
'_estimator_type', '_get_param_names', '_transform', 'fit', 'fit_predict', 'fit
_transform', 'get_params', 'predict', 'score', 'set_params', 'transform']
```

In [47]:
```
import matplotlib.pyplot as plt
l = []
for i in range(1,11):
    kmeans = KMeans(n_clusters = i,init = "k-means++",random_state = 40)

    kmeans.fit(x)
    l.append(kmeans.inertia_)


plt.plot(range(1,11),l)
plt.title("the elbow method graph")
plt.xlabel("clusters")
plt.ylabel("list")
plt.show()
```

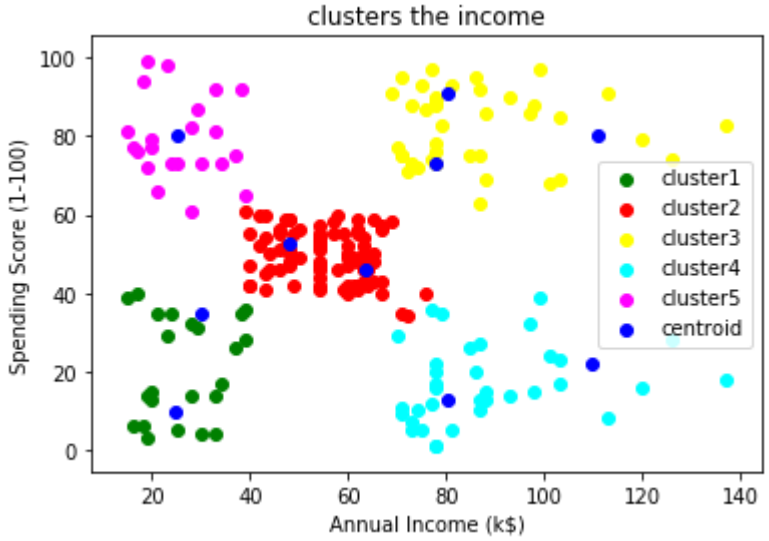


In [49]:
```
kmeans1 = KMeans(n_clusters=5,init="k-means++",random_state=40)
y_predict = kmeans1.fit_predict(x)
```

In [50]:
```
y_predict
```

Out[50]:
```
array([0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4,
       0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 4, 0, 1,
       0, 4, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 3, 2, 1, 2, 3, 2, 3, 2,
       1, 2, 3, 2, 3, 2, 3, 2, 3, 2, 1, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2,
       3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2,
       3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2, 3, 2,
       3, 2])
```

In [67]:
```
# visuvalize
plt.scatter(x[y_predict==0,0],x[y_predict==0,1],c= "green",label = "cluster1
plt.scatter(x[y_predict==1,0],x[y_predict==1,1],c= "red",label = "cluster2")
plt.scatter(x[y_predict==2,0],x[y_predict==2,1],c= "yellow",label = "cluster
plt.scatter(x[y_predict==3,0],x[y_predict==3,1],c= "cyan",label = "cluster4"
plt.scatter(x[y_predict==4,0],x[y_predict==4,1],c= "magenta",label = "cluste
plt.scatter(kmeans.cluster_centers_[:,0],kmeans.cluster_centers_[:,1],c="blu
plt.legend()
plt.title("clusters the income")
plt.xlabel("Annual Income (k$)")
plt.ylabel("Spending Score (1-100)")
plt.show()
```



In [53]:
```
x[y_predict==0,0],x[y_predict==0,1]
```

Out[53]:
```
(array([15, 16, 17, 18, 19, 19, 20, 20, 21, 23, 24, 25, 28, 28, 29, 30, 33,
        33, 34, 37, 38, 39, 39], dtype=int64),
 array([39,  6, 40,  6,  3, 14, 15, 13, 35, 29, 35,  5, 14, 32, 31,  4,  4,
        14, 17, 26, 35, 36, 28], dtype=int64))
```

In [ ]:
```

```