Chapter 7: Clustering

7.2 How clustering works? ¶

```
import warnings
warnings.filterwarnings('ignore')
```

```
import pandas as pd
customers_df = pd.read_csv( "customers.csv" )
```

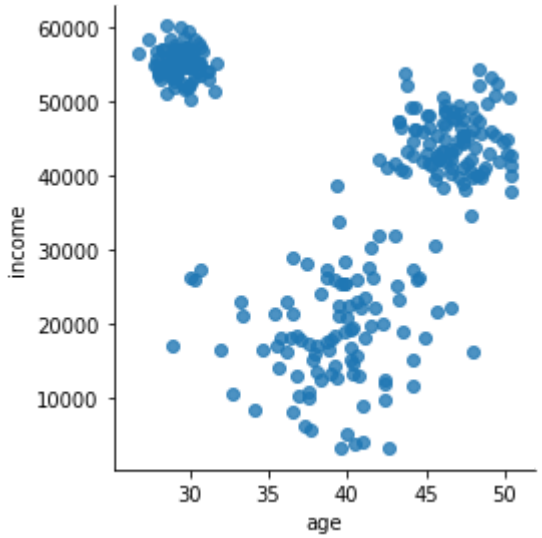
```
customers_df.head( 5 )
```

	income	age
0	41100.0	48.75
1	54100.0	28.10
2	47800.0	46.75
3	19100.0	40.25
4	18200.0	35.80

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sn
%matplotlib inline
```

```
sn.lmplot( "age", "income", data=customers_df, fit_reg = False, size = 4 );
#plt.title( "Fig 1: Customer Segments Based on Income and Age");
```

```
/Users/manaranjan/anaconda/lib/python3.5/site-packages/seaborn/regre
ssion.py:546: UserWarning: The `size` paramter has been renamed to `
height`; please update your code.
  warnings.warn(msg, UserWarning)
```



7.3 K-means Clustering

```
from sklearn.cluster import KMeans

clusters = KMeans( 3 )
clusters.fit( customers_df )
```

```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=30
0,
    n_clusters=3, n_init=10, n_jobs=None, precompute_distances='aut
o',
    random_state=None, tol=0.0001, verbose=0)
```

```
customers_df["clusterid"] = clusters.labels_
```

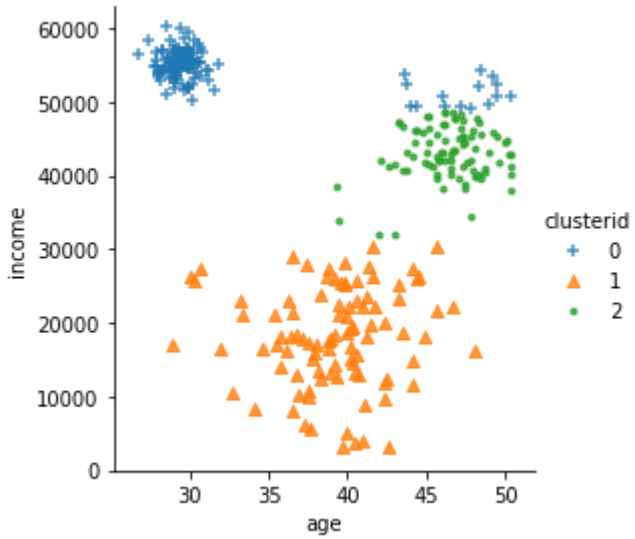
```
customers_df[0:5]
```

	income	age	clusterid
0	41100.0	48.75	2
1	54100.0	28.10	0
2	47800.0	46.75	2
3	19100.0	40.25	1
4	18200.0	35.80	1

7.3.1 Plotting the customers with their segments

```
markers = ['+','^','.']

sn.lmplot( "age", "income",
           data=customers_df,
           hue = "clusterid",
           fit_reg=False,
           markers = markers,
           size = 4 );
```

```
/Users/manaranjan/anaconda/lib/python3.5/site-packages/seaborn/regre
ssion.py:546: UserWarning: The `size` paramter has been renamed to `
height`; please update your code.
  warnings.warn(msg, UserWarning)
```



7.3.2 Normalizing the features

```
from sklearn.preprocessing import StandardScaler
```

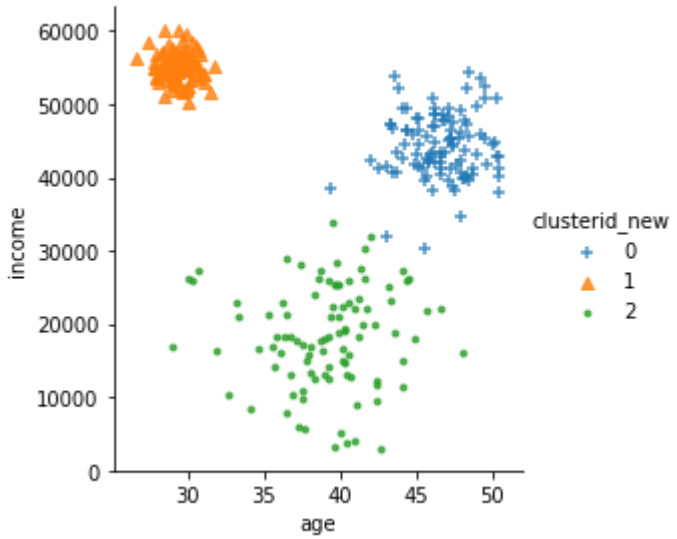
```
scaler = StandardScaler()
scaled_customers_df = scaler.fit_transform(
    customers_df[["age", "income"]] )

scaled_customers_df[0:5]
```

```
array([[ 1.3701637 ,  0.09718548],
       [-1.3791283 ,  0.90602749],
       [ 1.10388844,  0.51405021],
       [ 0.23849387, -1.27162408],
       [-0.35396857, -1.32762083]])
```

```
from sklearn.cluster import KMeans

clusters_new = KMeans( 3, random_state=42 )
clusters_new.fit( scaled_customers_df )
customers_df["clusterid_new"] = clusters_new.labels_
```

```
markers = ['+','^','.']

sn.lmplot( "age", "income",
           data=customers_df,
           hue = "clusterid_new",
           fit_reg=False,
           markers = markers,
           size = 4 );
#plt.title( "Fig 3: Customer segments created after normalization");
```

```
/Users/manaranjan/anaconda/lib/python3.5/site-packages/seaborn/regre
ssion.py:546: UserWarning: The `size` paramter has been renamed to `
height`; please update your code.
  warnings.warn(msg, UserWarning)
```




```
clusters.cluster_centers_
```

```
array([[5.46756522e+04, 3.17004348e+01],
       [1.81447917e+04, 3.91744792e+01],
       [4.30539326e+04, 4.64191011e+01]])
```

7.3.3 Cluster Centers and Interpreting the Clusters

```
customers_df.groupby( 'clusterid' )['age',
                              'income'].agg( ["mean",
                                             'std'] ).reset_index()
```

	clusterid	age		income	
		mean	std	mean	std
0	0	31.700435	6.122122	54675.652174	2362.224320
1	1	39.174479	3.626068	18144.791667	6745.241906
2	2	46.419101	2.289620	43053.932584	3613.769632

7.4 Creating Product Segments

7.4.1 Beer Dataset

```
beer_df = pd.read_csv( 'beer.csv' )
```

```
beer_df
```

	name	calories	sodium	alcohol	cost
0	Budweiser	144	15	4.7	0.43
1	Schlitz	151	19	4.9	0.43
2	Lowenbrau	157	15	0.9	0.48
3	Kronenbourg	170	7	5.2	0.73
4	Heineken	152	11	5.0	0.77
5	Old_Milwaukee	145	23	4.6	0.28
6	Augsberger	175	24	5.5	0.40
7	Srohs_Bohemian_Style	149	27	4.7	0.42
8	Miller_Lite	99	10	4.3	0.43
9	Budweiser_Light	113	8	3.7	0.40
10	Coors	140	18	4.6	0.44
11	Coors_Light	102	15	4.1	0.46
12	Michelob_Light	135	11	4.2	0.50
13	Becks	150	19	4.7	0.76
14	Kirin	149	6	5.0	0.79
15	Pabst_Extra_Light	68	15	2.3	0.38
16	Hamms	139	19	4.4	0.43
17	Heilemans_Old_Style	144	24	4.9	0.43
18	Olympia_Goled_Light	72	6	2.9	0.46
19	Schlitz_Light	97	7	4.2	0.47

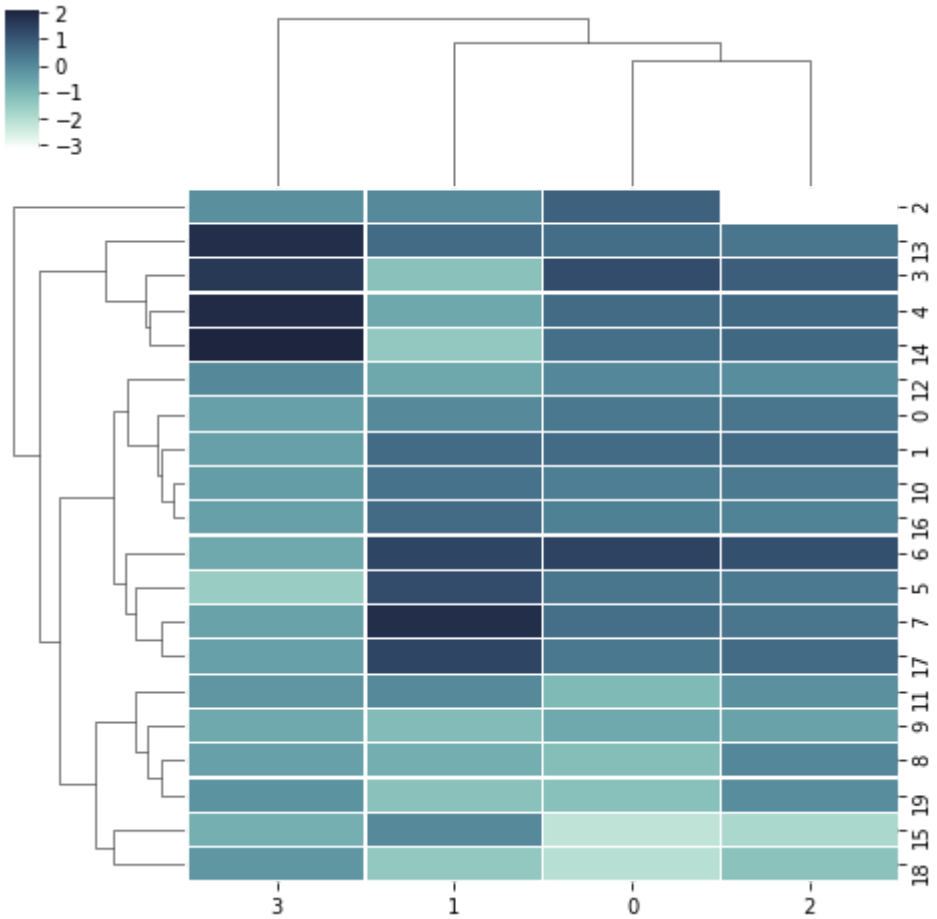
```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
scaled_beer_df = scaler.fit_transform( beer_df[['calories',
                                                'sodium',
                                                'alcohol',
                                                'cost']] )
```

```
/Users/manaranjan/anaconda/lib/python3.5/site-packages/sklearn/prepr
ocessing/data.py:617: DataConversionWarning: Data with input dtype i
nt64, float64 were all converted to float64 by StandardScaler.
  return self.partial_fit(X, y)
/Users/manaranjan/anaconda/lib/python3.5/site-packages/sklearn/base.
py:462: DataConversionWarning: Data with input dtype int64, float64
were all converted to float64 by StandardScaler.
  return self.fit(X, **fit_params).transform(X)
```

7.4.2 How many clusters exist?

7.4.2.1 Using Dendrogram

```
cmap = sn.cubehelix_palette(as_cmap=True, rot=-.3, light=1)
sn.clustermap(scaled_beer_df, cmap=cmap, linewidths=.2,
              figsize = (8,8) );
#plt.title( "Fig 4: Dendrogram of Beer Dataset");
```



```
beer_df.ix[[10, 16]]
```

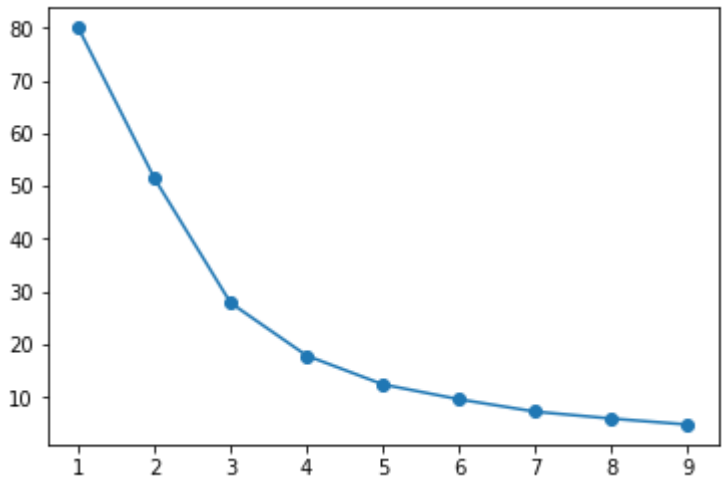
	name	calories	sodium	alcohol	cost
10	Coors	140	18	4.6	0.44
16	Hamms	139	19	4.4	0.43

```
beer_df.ix[[2,18]]
```

	name	calories	sodium	alcohol	cost
2	Lowenbrau	157	15	0.9	0.48
18	Olympia_Goled_Light	72	6	2.9	0.46

7.4.2.2 Finding Optimal Number of Clusters using Elbow Method

```
cluster_range = range( 1, 10 )
cluster_errors = []

for num_clusters in cluster_range:
  clusters = KMeans( num_clusters )
  clusters.fit( scaled_beer_df )
  cluster_errors.append( clusters.inertia_ )

plt.figure(figsize=(6,4))
plt.plot( cluster_range, cluster_errors, marker = "o" );
#plt.title( "Fig 5: Elbow Diagram");
```



7.4.2.3 Normalizing Features

Rescaling the dataset

```
scaler = StandardScaler()
scaled_beer_df = scaler.fit_transform( beer_df[['calories',
                                                'sodium',
                                                'alcohol',
                                                'cost']] )
```

```
/Users/manaranjan/anaconda/lib/python3.5/site-packages/sklearn/prepr
ocessing/data.py:617: DataConversionWarning: Data with input dtype i
nt64, float64 were all converted to float64 by StandardScaler.
  return self.partial_fit(X, y)
/Users/manaranjan/anaconda/lib/python3.5/site-packages/sklearn/base.
py:462: DataConversionWarning: Data with input dtype int64, float64
were all converted to float64 by StandardScaler.
  return self.fit(X, **fit_params).transform(X)
```

7.4.3 Creating Clusters

```
k = 3

clusters = KMeans( k, random_state = 42 )
clusters.fit( scaled_beer_df )
beer_df["clusterid"] = clusters.labels_
```

7.4.4 Interpreting the Clusters

Cluster 0

```
beer_df[beer_df.clusterid == 0]
```

	name	calories	sodium	alcohol	cost	clusterid
0	Budweiser	144	15	4.7	0.43	0
1	Schlitz	151	19	4.9	0.43	0
5	Old_Milwaukee	145	23	4.6	0.28	0
6	Augsberger	175	24	5.5	0.40	0
7	Srohs_Bohemian_Style	149	27	4.7	0.42	0
10	Coors	140	18	4.6	0.44	0
16	Hamms	139	19	4.4	0.43	0
17	Heilemans_Old_Style	144	24	4.9	0.43	0

Cluster 1

```
beer_df[beer_df.clusterid == 1]
```

	name	calories	sodium	alcohol	cost	clusterid
2	Lowenbrau	157	15	0.9	0.48	1
8	Miller_Lite	99	10	4.3	0.43	1
9	Budweiser_Light	113	8	3.7	0.40	1
11	Coors_Light	102	15	4.1	0.46	1
12	Michelob_Light	135	11	4.2	0.50	1
15	Pabst_Extra_Light	68	15	2.3	0.38	1
18	Olympia_Goled_Light	72	6	2.9	0.46	1
19	Schlitz_Light	97	7	4.2	0.47	1

Cluster 2

```
beer_df[beer_df.clusterid == 2]
```

	name	calories	sodium	alcohol	cost	clusterid
3	Kronenbourg	170	7	5.2	0.73	2
4	Heineken	152	11	5.0	0.77	2
13	Becks	150	19	4.7	0.76	2
14	Kirin	149	6	5.0	0.79	2

7.5 Hierarchical clustering

```
from sklearn.cluster import AgglomerativeClustering
```

```
h_clusters = AgglomerativeClustering( 3 )
h_clusters.fit( scaled_beer_df )
beer_df["h_clusterid"] = h_clusters.labels_
```

```
beer_df[beer_df.h_clusterid == 0]
```

	name	calories	sodium	alcohol	cost	clusterid	h_clusterid
2	Lowenbrau	157	15	0.9	0.48	1	0
8	Miller_Lite	99	10	4.3	0.43	1	0
9	Budweiser_Light	113	8	3.7	0.40	1	0
11	Coors_Light	102	15	4.1	0.46	1	0
12	Michelob_Light	135	11	4.2	0.50	1	0
15	Pabst_Extra_Light	68	15	2.3	0.38	1	0
18	Olympia_Goled_Light	72	6	2.9	0.46	1	0
19	Schlitz_Light	97	7	4.2	0.47	1	0

```
beer_df[beer_df.h_clusterid == 1]
```

	name	calories	sodium	alcohol	cost	clusterid	h_clusterid
0	Budweiser	144	15	4.7	0.43	0	1
1	Schlitz	151	19	4.9	0.43	0	1
5	Old_Milwaukee	145	23	4.6	0.28	0	1
6	Augsberger	175	24	5.5	0.40	0	1
7	Srohs_Bohemian_Style	149	27	4.7	0.42	0	1
10	Coors	140	18	4.6	0.44	0	1
16	Hamms	139	19	4.4	0.43	0	1
17	Heilemans_Old_Style	144	24	4.9	0.43	0	1

```
beer_df[beer_df.h_clusterid == 2]
```

	name	calories	sodium	alcohol	cost	clusterid	h_clusterid
3	Kronenbourg	170	7	5.2	0.73	2	2
4	Heineken	152	11	5.0	0.77	2	2
13	Becks	150	19	4.7	0.76	2	2
14	Kirin	149	6	5.0	0.79	2	2