# Clustering

December 10, 2017

```
In [34]: import pandas as pd
         import numpy as np
         import matplotlib.pyplot as plt
         import seaborn as sns
         %matplotlib inline

In [35]: teens = pd.read_csv('snsdata.csv')

In [36]: teens.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30000 entries, 0 to 29999
Data columns (total 40 columns):
gradyear        30000 non-null int64
gender          27276 non-null object
age             24914 non-null float64
friends         30000 non-null int64
basketball      30000 non-null int64
football        30000 non-null int64
soccer          30000 non-null int64
softball        30000 non-null int64
volleyball      30000 non-null int64
swimming        30000 non-null int64
cheerleading    30000 non-null int64
baseball        30000 non-null int64
tennis          30000 non-null int64
sports          30000 non-null int64
cute            30000 non-null int64
sex             30000 non-null int64
sexy            30000 non-null int64
hot             30000 non-null int64
kissed          30000 non-null int64
dance           30000 non-null int64
band            30000 non-null int64
marching        30000 non-null int64
music           30000 non-null int64
rock            30000 non-null int64
god             30000 non-null int64
```

```
church          30000 non-null int64
jesus           30000 non-null int64
bible           30000 non-null int64
hair            30000 non-null int64
dress           30000 non-null int64
blonde          30000 non-null int64
mall            30000 non-null int64
shopping        30000 non-null int64
clothes         30000 non-null int64
hollister       30000 non-null int64
abercrombie     30000 non-null int64
die             30000 non-null int64
death           30000 non-null int64
drunk           30000 non-null int64
drugs           30000 non-null int64
dtypes: float64(1), int64(38), object(1)
memory usage: 9.2+ MB
```

```
In [37]: teens.head()
```

```
Out[37]:    gradyear gender     age  friends  basketball  football  soccer  softball  \
         0      2006      M  18.982        7           0         0       0         0
         1      2006      F  18.801        0           0         1       0         0
         2      2006      M  18.335       69           0         1       0         0
         3      2006      F  18.875        0           0         0       0         0
         4      2006    NaN  18.995       10           0         0       0         0

            volleyball  swimming  ...    blonde  mall  shopping  clothes  hollister  \
         0           0         0  ...         0     0         0        0          0
         1           0         0  ...         0     1         0        0          0
         2           0         0  ...         0     0         0        0          0
         3           0         0  ...         0     0         0        0          0
         4           0         0  ...         0     0         2        0          0

            abercrombie  die  death  drunk  drugs
         0            0    0      0      0      0
         1            0    0      0      0      0
         2            0    0      1      0      0
         3            0    0      0      0      0
         4            0    0      0      1      1

         [5 rows x 40 columns]
```

```
In [38]: teens.describe()
```

```
Out[38]:            gradyear           age       friends    basketball      football  \
         count  30000.000000  24914.000000  30000.000000  30000.000000  30000.000000
         mean    2007.500000     17.993950     30.179467      0.267333      0.252300
```

```
std        1.118053     7.858054    36.530877     0.804708     0.705357
min     2006.000000     3.086000     0.000000     0.000000     0.000000
25%     2006.750000    16.312000     3.000000     0.000000     0.000000
50%     2007.500000    17.287000    20.000000     0.000000     0.000000
75%     2008.250000    18.259000    44.000000     0.000000     0.000000
max     2009.000000   106.927000   830.000000    24.000000    15.000000

             soccer     softball   volleyball     swimming  cheerleading  \
count  30000.000000 30000.000000 30000.000000  30000.00000  30000.000000
mean       0.222767     0.161200     0.143133      0.13440      0.106633
std        0.917226     0.739707     0.639943      0.51699      0.514333
min        0.000000     0.000000     0.000000      0.00000      0.000000
25%        0.000000     0.000000     0.000000      0.00000      0.000000
50%        0.000000     0.000000     0.000000      0.00000      0.000000
75%        0.000000     0.000000     0.000000      0.00000      0.000000
max       27.000000    17.000000    14.000000     31.00000      9.000000

           ...           blonde         mall     shopping      clothes  \
count      ...     30000.000000 30000.000000 30000.000000  30000.00000
mean       ...         0.098933     0.257367     0.353000      0.14850
std        ...         1.942319     0.695758     0.724391      0.47264
min        ...         0.000000     0.000000     0.000000      0.00000
25%        ...         0.000000     0.000000     0.000000      0.00000
50%        ...         0.000000     0.000000     0.000000      0.00000
75%        ...         0.000000     0.000000     1.000000      0.00000
max        ...       327.000000    12.000000    11.000000      8.00000

          hollister  abercrombie          die        death         drunk  \
count  30000.000000 30000.000000 30000.000000 30000.000000  30000.000000
mean       0.069867     0.051167     0.184100     0.114233      0.087967
std        0.346779     0.279555     0.624516     0.436796      0.399125
min        0.000000     0.000000     0.000000     0.000000      0.000000
25%        0.000000     0.000000     0.000000     0.000000      0.000000
50%        0.000000     0.000000     0.000000     0.000000      0.000000
75%        0.000000     0.000000     0.000000     0.000000      0.000000
max        9.000000     8.000000    22.000000    14.000000      8.000000

              drugs
count  30000.000000
mean       0.060433
std        0.345522
min        0.000000
25%        0.000000
50%        0.000000
75%        0.000000
max       16.000000

[8 rows x 39 columns]
```

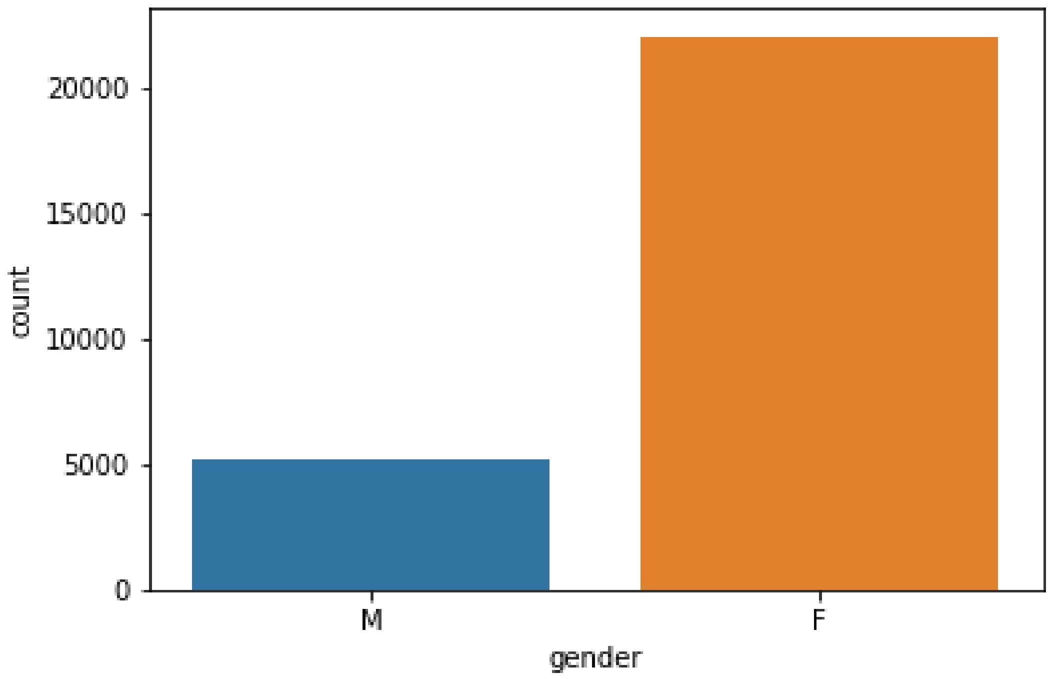
```
In [39]: teens['age'].describe()

Out[39]: count    24914.000000
         mean        17.993950
         std          7.858054
         min          3.086000
         25%         16.312000
         50%         17.287000
         75%         18.259000
         max        106.927000
         Name: age, dtype: float64
```

```
In [40]: # Managing Outliers

         def impute_age(cols):
             age = cols[0]
             if age >= 20:
                 age = None
             else:
                 if age < 13:
                     age = None
                 else:
                     return age;
```

```
In [41]: teens['age'] = teens[['age']].apply(impute_age, axis = 1)
         teens['age'].describe()

Out[41]: count    24477.000000
         mean        17.252429
         std          1.157465
         min         13.027000
         25%         16.304000
         50%         17.265000
         75%         18.220000
         max         19.995000
         Name: age, dtype: float64
```

```
In [42]: teens['gender'].value_counts(dropna = False)

Out[42]: F      22054
         M       5222
         NaN     2724
         Name: gender, dtype: int64
```

```
In [43]: sns.countplot(x= 'gender', data=teens)

Out[43]: <matplotlib.axes._subplots.AxesSubplot at 0x7f55842ec790>
```



```
In [44]: dummies= pd.get_dummies(data = teens['gender'])
         dummies.head()

Out[44]:    F  M
         0  0  1
         1  1  0
         2  0  1
         3  1  0
         4  0  0

In [45]: teens= pd.concat([teens, dummies], axis= 1)
         teens.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30000 entries, 0 to 29999
Data columns (total 42 columns):
gradyear        30000 non-null int64
gender          27276 non-null object
age             24477 non-null float64
friends         30000 non-null int64
basketball      30000 non-null int64
football        30000 non-null int64
soccer          30000 non-null int64
softball        30000 non-null int64
volleyball      30000 non-null int64
```

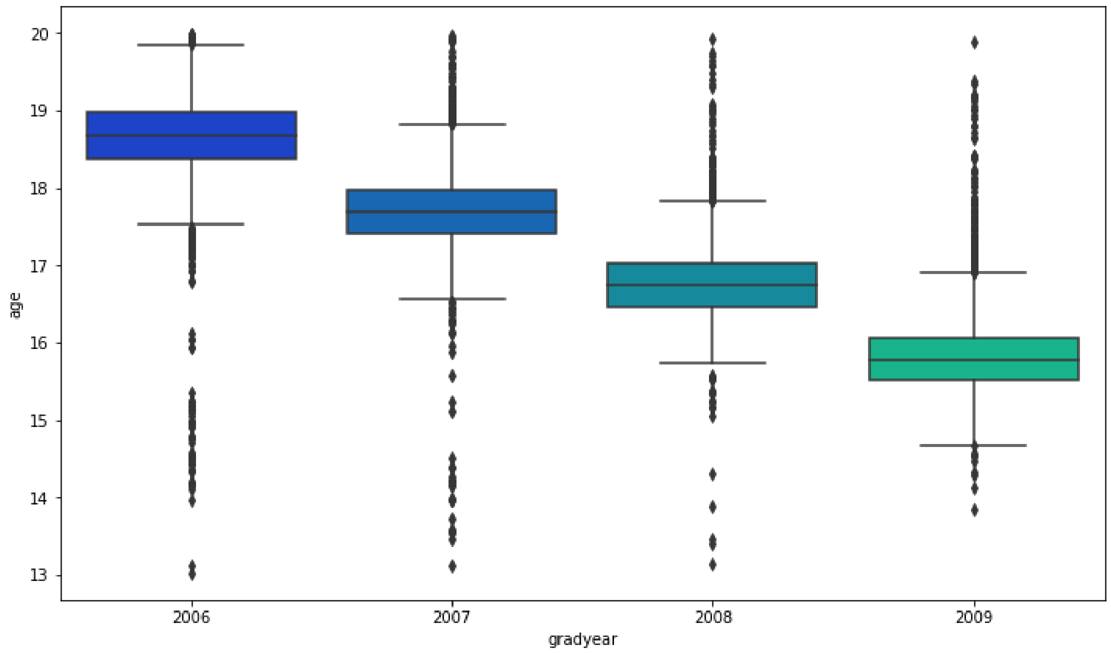
```
swimming        30000 non-null int64
cheerleading    30000 non-null int64
baseball        30000 non-null int64
tennis          30000 non-null int64
sports          30000 non-null int64
cute            30000 non-null int64
sex             30000 non-null int64
sexy            30000 non-null int64
hot             30000 non-null int64
kissed          30000 non-null int64
dance           30000 non-null int64
band            30000 non-null int64
marching        30000 non-null int64
music           30000 non-null int64
rock            30000 non-null int64
god             30000 non-null int64
church          30000 non-null int64
jesus           30000 non-null int64
bible           30000 non-null int64
hair            30000 non-null int64
dress           30000 non-null int64
blonde          30000 non-null int64
mall            30000 non-null int64
shopping        30000 non-null int64
clothes         30000 non-null int64
hollister       30000 non-null int64
abercrombie     30000 non-null int64
die             30000 non-null int64
death           30000 non-null int64
drunk           30000 non-null int64
drugs           30000 non-null int64
F               30000 non-null uint8
M               30000 non-null uint8
dtypes: float64(1), int64(38), object(1), uint8(2)
memory usage: 9.2+ MB
```

```
In [46]: teens[['gender', 'F', 'M']].head()
```

```
Out[46]:   gender  F  M
         0      M  0  1
         1      F  1  0
         2      M  0  1
         3      F  1  0
         4    NaN  0  0
```

```
In [47]: plt.figure(figsize=(12, 7))
         sns.boxplot(x='gradyear',y='age',data=teens,palette='winter')
```

```
Out[47]: <matplotlib.axes._subplots.AxesSubplot at 0x7f5584227190>
```



```
In [48]: GradyearMeansByAge = teens['age'].groupby(teens['gradyear']).mean()

In [49]: print(GradyearMeansByAge)

gradyear
2006    18.655858
2007    17.706172
2008    16.767701
2009    15.819573
Name: age, dtype: float64


In [50]: def impute_age(cols):
             Age = cols[0]
             Gradyear = cols[1]

             if pd.isnull(Age):

                 if Gradyear == 2006:
                     return 18.655858

                 elif Gradyear == 2007:
                     return 17.706172
```

```
                elif Gradyear == 2008:
                    return 16.767701

                else:
                    return 15.819573

            else:
                return Age
```

```
In [51]: teens['age'] = teens[['age','gradyear']].apply(impute_age,axis=1)
```

```
In [52]: teens['age'].describe()
```

```
Out[52]: count    30000.000000
         mean        17.237326
         std          1.141821
         min         13.027000
         25%         16.282000
         50%         17.238000
         75%         18.212000
         max         19.995000
         Name: age, dtype: float64
```

```
In [53]: interests= teens.iloc[:,4:40]
         interests.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30000 entries, 0 to 29999
Data columns (total 36 columns):
basketball      30000 non-null int64
football        30000 non-null int64
soccer          30000 non-null int64
softball        30000 non-null int64
volleyball      30000 non-null int64
swimming        30000 non-null int64
cheerleading    30000 non-null int64
baseball        30000 non-null int64
tennis          30000 non-null int64
sports          30000 non-null int64
cute            30000 non-null int64
sex             30000 non-null int64
sexy            30000 non-null int64
hot             30000 non-null int64
kissed          30000 non-null int64
dance           30000 non-null int64
band            30000 non-null int64
marching        30000 non-null int64
music           30000 non-null int64
rock            30000 non-null int64
```

```
god             30000 non-null int64
church          30000 non-null int64
jesus           30000 non-null int64
bible           30000 non-null int64
hair            30000 non-null int64
dress           30000 non-null int64
blonde          30000 non-null int64
mall            30000 non-null int64
shopping        30000 non-null int64
clothes         30000 non-null int64
hollister       30000 non-null int64
abercrombie     30000 non-null int64
die             30000 non-null int64
death           30000 non-null int64
drunk           30000 non-null int64
drugs           30000 non-null int64
dtypes: int64(36)
memory usage: 8.2 MB
```

```
In [54]: from sklearn.preprocessing import StandardScaler
         scaler= StandardScaler()

In [55]: scaler.fit(interests)

Out[55]: StandardScaler(copy=True, with_mean=True, with_std=True)

In [56]: scaled_features= scaler.transform(interests)
         interests_z = pd.DataFrame(scaled_features, columns= interests.columns)
         interests_z.describe()

Out[56]:          basketball      football        soccer      softball    volleyball  \
         count  3.000000e+04  3.000000e+04  3.000000e+04  3.000000e+04  3.000000e+04
         mean   5.494864e-17 -4.547474e-17  1.515825e-17  3.031649e-17  3.789561e-18
         std    1.000017e+00  1.000017e+00  1.000017e+00  1.000017e+00  1.000017e+00
         min   -3.322173e-01 -3.576974e-01 -2.428741e-01 -2.179278e-01 -2.236696e-01
         25%   -3.322173e-01 -3.576974e-01 -2.428741e-01 -2.179278e-01 -2.236696e-01
         50%   -3.322173e-01 -3.576974e-01 -2.428741e-01 -2.179278e-01 -2.236696e-01
         75%   -3.322173e-01 -3.576974e-01 -2.428741e-01 -2.179278e-01 -2.236696e-01
         max    2.949277e+01  2.090850e+01  2.919421e+01  2.276453e+01  2.165366e+01

                    swimming  cheerleading      baseball        tennis        sports  \
         count  3.000000e+04  3.000000e+04  3.000000e+04  3.000000e+04  3.000000e+04
         mean   1.894781e-17 -3.789561e-17  3.789561e-18 -2.889540e-17 -2.842171e-17
         std    1.000017e+00  1.000017e+00  1.000017e+00  1.000017e+00  1.000017e+00
         min   -2.599706e-01 -2.073271e-01 -2.011306e-01 -1.689389e-01 -2.971234e-01
         25%   -2.599706e-01 -2.073271e-01 -2.011306e-01 -1.689389e-01 -2.971234e-01
         50%   -2.599706e-01 -2.073271e-01 -2.011306e-01 -1.689389e-01 -2.971234e-01
         75%   -2.599706e-01 -2.073271e-01 -2.011306e-01 -1.689389e-01 -2.971234e-01
```

```
max    5.970348e+01  1.729137e+01  3.046682e+01  2.884728e+01  2.517666e+01

            ...             blonde          mall      shopping       clothes  \
count       ...       3.000000e+04  3.000000e+04  3.000000e+04  3.000000e+04
mean        ...      -8.289665e-18 -1.136868e-17  5.873820e-17  1.515825e-17
std         ...       1.000017e+00  1.000017e+00  1.000017e+00  1.000017e+00
min         ...      -5.093652e-02 -3.699147e-01 -4.873142e-01 -3.141979e-01
25%         ...      -5.093652e-02 -3.699147e-01 -4.873142e-01 -3.141979e-01
50%         ...      -5.093652e-02 -3.699147e-01 -4.873142e-01 -3.141979e-01
75%         ...      -5.093652e-02 -3.699147e-01  8.931794e-01 -3.141979e-01
max         ...       1.683073e+02  1.687776e+01  1.469812e+01  1.661229e+01

          hollister   abercrombie           die         death         drunk  \
count  3.000000e+04  3.000000e+04  3.000000e+04  3.000000e+04  3.000000e+04
mean   5.494864e-17  1.136868e-17 -9.687066e-17 -1.610564e-17 -1.515825e-17
std    1.000017e+00  1.000017e+00  1.000017e+00  1.000017e+00  1.000017e+00
min   -2.014763e-01 -1.830317e-01 -2.947932e-01 -2.615302e-01 -2.204026e-01
25%   -2.014763e-01 -1.830317e-01 -2.947932e-01 -2.615302e-01 -2.204026e-01
50%   -2.014763e-01 -1.830317e-01 -2.947932e-01 -2.615302e-01 -2.204026e-01
75%   -2.014763e-01 -1.830317e-01 -2.947932e-01 -2.615302e-01 -2.204026e-01
max    2.575205e+01  2.843431e+01  3.493308e+01  3.179061e+01  1.982379e+01

              drugs
count  3.000000e+04
mean   1.752672e-17
std    1.000017e+00
min   -1.749076e-01
25%   -1.749076e-01
50%   -1.749076e-01
75%   -1.749076e-01
max    4.613268e+01

[8 rows x 36 columns]
```

```
In [57]: from sklearn.cluster import KMeans

In [58]: kmeans = KMeans(n_clusters=5)

In [59]: kmeans.fit(interests_z)

Out[59]: KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
             n_clusters=5, n_init=10, n_jobs=1, precompute_distances='auto',
             random_state=None, tol=0.0001, verbose=0)

In [60]: kmeans.labels_

Out[60]: array([0, 2, 0, ..., 0, 0, 0], dtype=int32)

In [61]: labels = pd.DataFrame(kmeans.labels_)
In [62]: teens_labels = pd.concat([teens,labels], axis=1)
```

```
In [63]: teens_labels.rename(columns={0: 'labels'}, inplace=True)
```

```
In [64]: teens_labels.head()
```

```
Out[64]:    gradyear gender     age  friends  basketball  football  soccer  softball  \
         0      2006      M  18.982        7           0         0       0         0
         1      2006      F  18.801        0           0         1       0         0
         2      2006      M  18.335       69           0         1       0         0
         3      2006      F  18.875        0           0         0       0         0
         4      2006    NaN  18.995       10           0         0       0         0

            volleyball  swimming   ...    clothes  hollister  abercrombie  die  death  \
         0           0         0   ...          0          0            0    0      0
         1           0         0   ...          0          0            0    0      0
         2           0         0   ...          0          0            0    0      1
         3           0         0   ...          0          0            0    0      0
         4           0         0   ...          0          0            0    0      0

            drunk  drugs  F  M  labels
         0      0      0  0  1       0
         1      0      0  1  0       2
         2      0      0  0  1       0
         3      0      0  1  0       0
         4      1      1  0  0       1

         [5 rows x 43 columns]
```

```
In [65]: teens_labels['labels'].value_counts()
```

```
Out[65]: 0    21530
         2     4201
         3     2635
         1     1035
         4      599
         Name: labels, dtype: int64
```

```
In [66]: AgeMeansByLabels = teens_labels['age'].groupby(teens_labels['labels']).mean()
         print(AgeMeansByLabels)
```

```
labels
0    17.300643
1    17.098047
2    17.050091
3    17.040383
4    17.381677
Name: age, dtype: float64
```

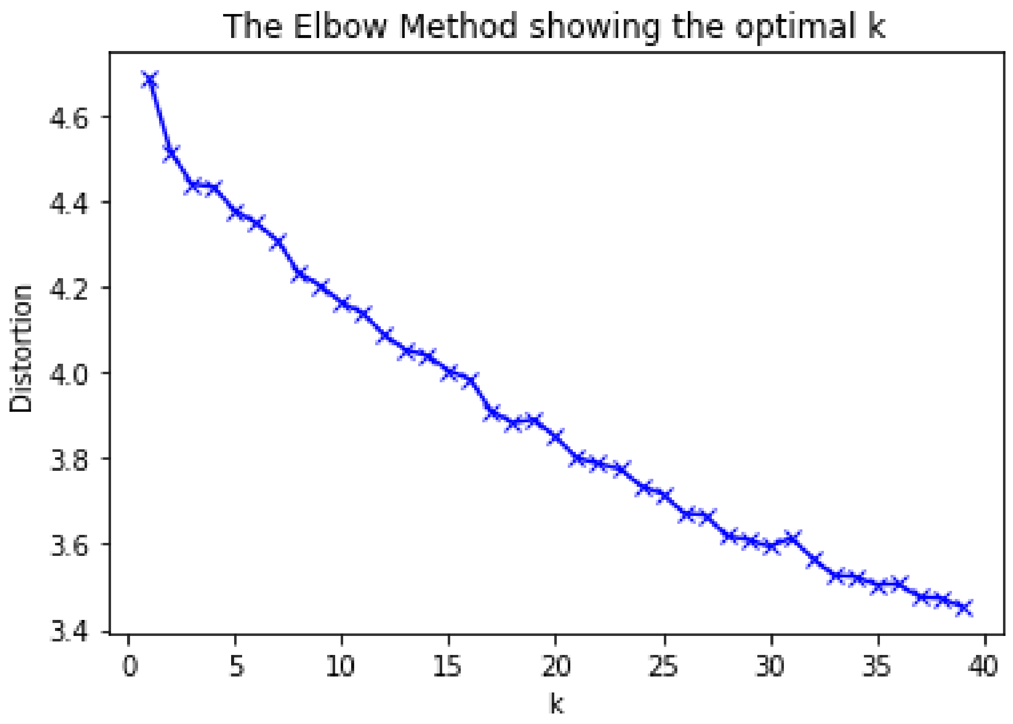
```
In [67]: FemaleMeansByLabels = teens_labels['F'].groupby(teens_labels['labels']).mean()
         print(FemaleMeansByLabels)

labels
0    0.706781
1    0.801932
2    0.887170
3    0.700949
4    0.722871
Name: F, dtype: float64
```

```
In [68]: FriendsMeanByLabels = teens_labels['friends'].groupby(teens_labels['labels']).mean()
         print(FriendsMeanByLabels)

labels
0    27.772782
1    30.727536
2    38.387527
3    35.909298
4    32.964942
Name: friends, dtype: float64
```

```
In [69]: teens.columns

Out[69]: Index([u'gradyear', u'gender', u'age', u'friends', u'basketball', u'football',
               u'soccer', u'softball', u'volleyball', u'swimming', u'cheerleading',
               u'baseball', u'tennis', u'sports', u'cute', u'sex', u'sexy', u'hot',
               u'kissed', u'dance', u'band', u'marching', u'music', u'rock', u'god',
               u'church', u'jesus', u'bible', u'hair', u'dress', u'blonde', u'mall',
               u'shopping', u'clothes', u'hollister', u'abercrombie', u'die', u'death',
               u'drunk', u'drugs', u'F', u'M'],
              dtype='object')
```

```
In [77]: from scipy.spatial.distance import cdist
         distortions = []
         for i in range(1,40):
             kmeans = KMeans(n_clusters=i)
             kmeans.fit(interests_z)
             distortions.append(sum(np.min(cdist(interests_z, kmeans.cluster_centers_, 'euclidea
```

```
In [80]: plt.plot(range(1,40), distortions, 'bx-')
         plt.xlabel('k')
         plt.ylabel('Distortion')
         plt.title('The Elbow Method showing the optimal k')
         plt.show()
```


The Elbow Method showing the optimal k
Distortion
4.6
4.4
4.2
4.0
3.8
3.6
3.4
0
5
10
15
20
25
30
35
40
k