K Means Algorithm

September 30, 2020

[3]:
```
import pandas as pd
import numpy as np
from sklearn.cluster import KMeans
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import MinMaxScaler
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

[4]:
```
train_url = "http://s3.amazonaws.com/assets.datacamp.com/course/Kaggle/train.
 ↪csv"
train = pd.read_csv(train_url)
test_url = "http://s3.amazonaws.com/assets.datacamp.com/course/Kaggle/test.csv"
test = pd.read_csv(test_url)
```

[5]:
```
print("***** Train_Set *****")
print(train.head())
```

```
***** Train_Set *****
   PassengerId  Survived  Pclass  \
0            1         0       3
1            2         1       1
2            3         1       3
3            4         1       1
4            5         0       3

                                                Name     Sex   Age  SibSp  \
0                            Braund, Mr. Owen Harris    male  22.0      1
1  Cumings, Mrs. John Bradley (Florence Briggs Th…  female  38.0      1
2                             Heikkinen, Miss. Laina  female  26.0      0
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1
4                           Allen, Mr. William Henry    male  35.0      0

   Parch            Ticket     Fare Cabin Embarked
0      0         A/5 21171   7.2500   NaN        S
1      0          PC 17599  71.2833   C85        C
2      0  STON/O2. 3101282   7.9250   NaN        S
3      0            113803  53.1000  C123        S
```

```
4      0            373450   8.0500   NaN        S
```

[6]:
```
print("***Test set***")
print(test.head())
```

```
***Test set***
   PassengerId  Pclass                                          Name     Sex  \
0          892       3                              Kelly, Mr. James    male
1          893       3              Wilkes, Mrs. James (Ellen Needs)  female
2          894       2                     Myles, Mr. Thomas Francis    male
3          895       3                              Wirz, Mr. Albert    male
4          896       3  Hirvonen, Mrs. Alexander (Helga E Lindqvist)  female

    Age  SibSp  Parch   Ticket     Fare Cabin Embarked
0  34.5      0      0   330911   7.8292   NaN        Q
1  47.0      1      0   363272   7.0000   NaN        S
2  62.0      0      0   240276   9.6875   NaN        Q
3  27.0      0      0   315154   8.6625   NaN        S
4  22.0      1      1  3101298  12.2875   NaN        S
```

[7]:
```
print("***** Train_Set *****")
print(train.describe())
```

```
***** Train_Set *****
       PassengerId    Survived      Pclass         Age       SibSp  \
count   891.000000  891.000000  891.000000  714.000000  891.000000
mean    446.000000    0.383838    2.308642   29.699118    0.523008
std     257.353842    0.486592    0.836071   14.526497    1.102743
min       1.000000    0.000000    1.000000    0.420000    0.000000
25%     223.500000    0.000000    2.000000   20.125000    0.000000
50%     446.000000    0.000000    3.000000   28.000000    0.000000
75%     668.500000    1.000000    3.000000   38.000000    1.000000
max     891.000000    1.000000    3.000000   80.000000    8.000000

            Parch        Fare
count  891.000000  891.000000
mean     0.381594   32.204208
std      0.806057   49.693429
min      0.000000    0.000000
25%      0.000000    7.910400
50%      0.000000   14.454200
75%      0.000000   31.000000
max      6.000000  512.329200
```

[8]:
```
print("***test set***")
print(test.describe())
```

```
***test set***
```

```
       PassengerId      Pclass         Age       SibSp       Parch        Fare
count   418.000000  418.000000  332.000000  418.000000  418.000000  417.000000
mean   1100.500000    2.265550   30.272590    0.447368    0.392344   35.627188
std     120.810458    0.841838   14.181209    0.896760    0.981429   55.907576
min     892.000000    1.000000    0.170000    0.000000    0.000000    0.000000
25%     996.250000    1.000000   21.000000    0.000000    0.000000    7.895800
50%    1100.500000    3.000000   27.000000    0.000000    0.000000   14.454200
75%    1204.750000    3.000000   39.000000    1.000000    0.000000   31.500000
max    1309.000000    3.000000   76.000000    8.000000    9.000000  512.329200
```

[9]:
```
train.isna().head()
```

[9]:
```
   PassengerId  Survived  Pclass   Name    Sex    Age  SibSp  Parch  Ticket  \
0        False     False   False  False  False  False  False  False   False
1        False     False   False  False  False  False  False  False   False
2        False     False   False  False  False  False  False  False   False
3        False     False   False  False  False  False  False  False   False
4        False     False   False  False  False  False  False  False   False

    Fare  Cabin  Embarked
0  False   True     False
1  False  False     False
2  False   True     False
3  False  False     False
4  False   True     False
```

[10]:
```
test.isna().head()
```

[10]:
```
   PassengerId  Pclass   Name    Sex    Age  SibSp  Parch  Ticket   Fare  \
0        False   False  False  False  False  False  False   False  False
1        False   False  False  False  False  False  False   False  False
2        False   False  False  False  False  False  False   False  False
3        False   False  False  False  False  False  False   False  False
4        False   False  False  False  False  False  False   False  False

   Cabin  Embarked
0   True     False
1   True     False
2   True     False
3   True     False
4   True     False
```

[11]:
```
print("*****In the train set*****")
print(train.isna().sum())
```

```
*****In the train set*****
PassengerId      0
Survived         0
```

```
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

[12]:
```
train.fillna(train.mean(), inplace=True)
```

[13]:
```
test.fillna(train.mean(), inplace=True)
```

[14]:
```
print(train.isna().sum())
```

```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age              0
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

[15]:
```
print(test.isna().sum())
```

```
PassengerId      0
Pclass           0
Name             0
Sex              0
Age              0
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          327
Embarked         0
dtype: int64
```

[16]:
```
train['Ticket'].head()
```

[16]:
```
0           A/5 21171
1            PC 17599
2    STON/O2. 3101282
3              113803
4              373450
Name: Ticket, dtype: object
```

[17]:
```
train['Cabin'].head()
```

[17]:
```
0     NaN
1     C85
2     NaN
3    C123
4     NaN
Name: Cabin, dtype: object
```

[18]:
```
train[['Pclass', 'Survived']].groupby(['Pclass'], as_index=False).mean().
↪sort_values(by='Survived', ascending=False)
```

[18]:

	Pclass	Survived
0	1	0.629630
1	2	0.472826
2	3	0.242363

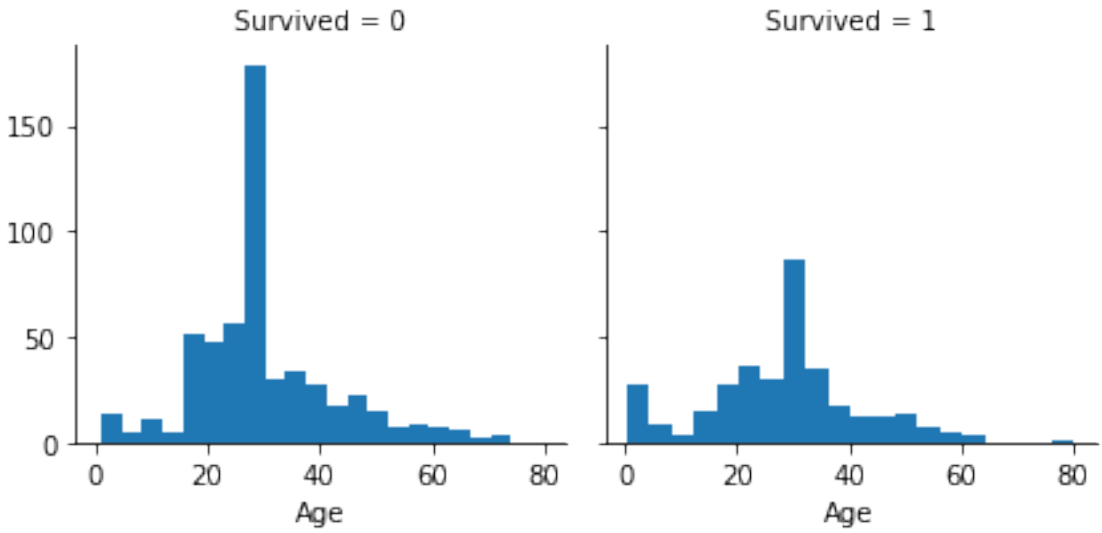
[19]:
```
train[["Sex", "Survived"]].groupby(['Sex'], as_index=False).mean().
↪sort_values(by='Survived', ascending=False)
```

[19]:

	Sex	Survived
0	female	0.742038
1	male	0.188908

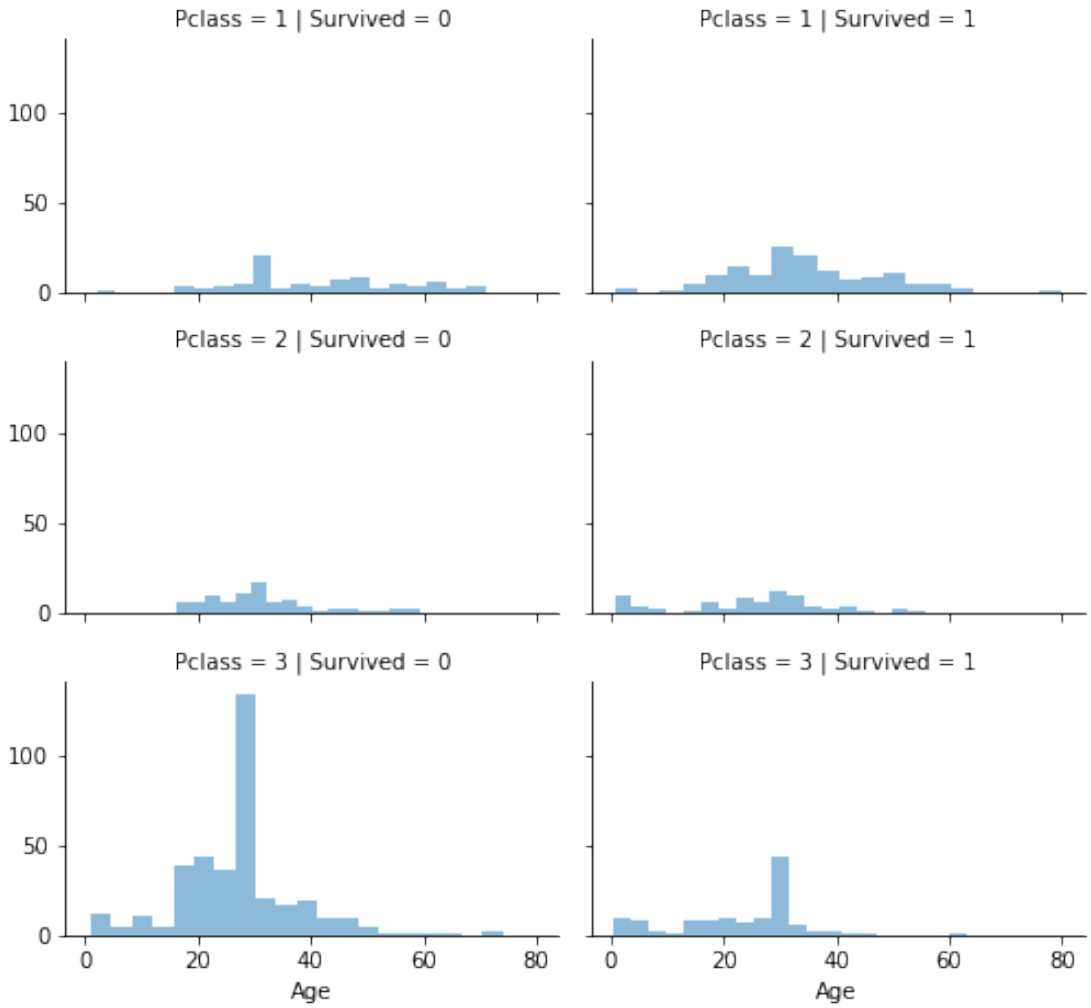
[20]:
```
g = sns.FacetGrid(train, col='Survived')
g.map(plt.hist, 'Age', bins=20)
```

[20]:
```
<seaborn.axisgrid.FacetGrid at 0x152bec9ee48>
```



[21]:
```
grid = sns.FacetGrid(train, col='Survived', row='Pclass', size=2.2, aspect=1.6)
grid.map(plt.hist, 'Age', alpha=.5, bins=20)
grid.add_legend();
```

```
C:\Users\blr0a\Anaconda3\lib\site-packages\seaborn\axisgrid.py:243: UserWarning:
The `size` parameter has been renamed to `height`; please update your code.
  warnings.warn(msg, UserWarning)
```



[22]:
```
train = train.drop(['Name','Ticket', 'Cabin','Embarked'], axis=1)
test = test.drop(['Name','Ticket', 'Cabin','Embarked'], axis=1)
```

[23]:
```
labelEncoder = LabelEncoder()
labelEncoder.fit(train['Sex'])
labelEncoder.fit(test['Sex'])
train['Sex'] = labelEncoder.transform(train['Sex'])
test['Sex'] = labelEncoder.transform(test['Sex'])
```

[24]:
```
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 8 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
```

```
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Sex          891 non-null    int32
 4   Age          891 non-null    float64
 5   SibSp        891 non-null    int64
 6   Parch        891 non-null    int64
 7   Fare         891 non-null    float64
dtypes: float64(2), int32(1), int64(5)
memory usage: 52.3 KB
```

[25]:
```
test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  418 non-null    int64
 1   Pclass       418 non-null    int64
 2   Sex          418 non-null    int32
 3   Age          418 non-null    float64
 4   SibSp        418 non-null    int64
 5   Parch        418 non-null    int64
 6   Fare         418 non-null    float64
dtypes: float64(2), int32(1), int64(4)
memory usage: 21.4 KB
```

[26]:
```
X = np.array(train.drop(['Survived'], 1).astype(float))
```

[27]:
```
y = np.array(train['Survived'])
```

[28]:
```
 train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 8 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Sex          891 non-null    int32
 4   Age          891 non-null    float64
 5   SibSp        891 non-null    int64
 6   Parch        891 non-null    int64
 7   Fare         891 non-null    float64
dtypes: float64(2), int32(1), int64(5)
```

```
memory usage: 52.3 KB
```

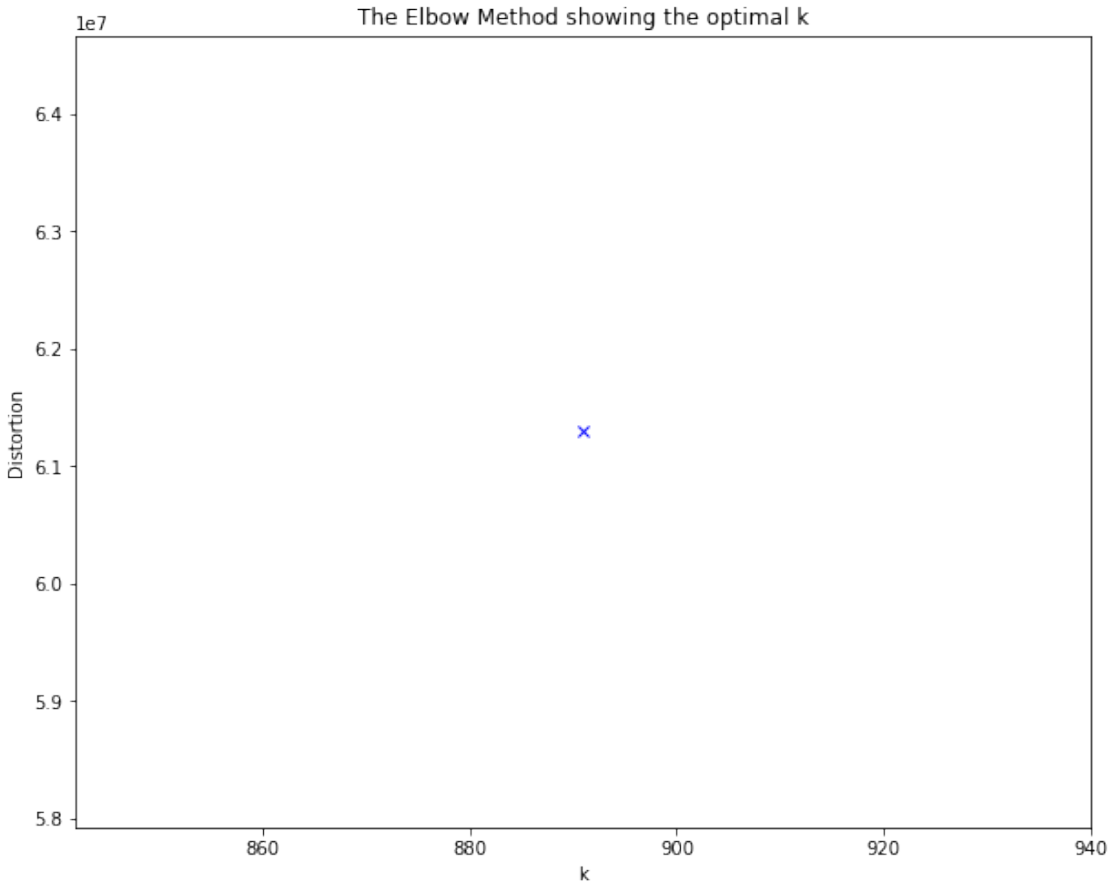
[29]:
```
kmeans = KMeans(n_clusters=2)
kmeans.fit(X)
```

[29]:
```
KMeans(n_clusters=2)
```

[38]:
```
correct = 0
distortions = []
for i in range(len(X)):
    distortions.append(kmeans.inertia_)
    predict_me = np.array(X[i].astype(float))
    predict_me = predict_me.reshape(-1, len(predict_me))
    prediction = kmeans.predict(predict_me)
    if prediction[0] == y[i]:
        correct += 1

print(correct/len(X))
```

```
0.37373737373737376
```

[42]:
```
distortions = []
K = range(1,2)
for k in K:
    kmeanModel = KMeans(n_clusters=k)
    kmeanModel.fit(X)
    distortions.append(kmeanModel.inertia_)
import matplotlib.pyplot as plt
plt.figure(figsize=(10,8))
plt.plot(len(X), distortions, 'bx-')
plt.xlabel('k')
plt.ylabel('Distortion')
plt.title('The Elbow Method showing the optimal k')
plt.show()
```



[32]:
```
scaler = MinMaxScaler()
X_scaled = scaler.fit_transform(X)
```

[33]:
```
kmeans.fit(X_scaled)
```

[33]:
```
KMeans(n_clusters=2)
```

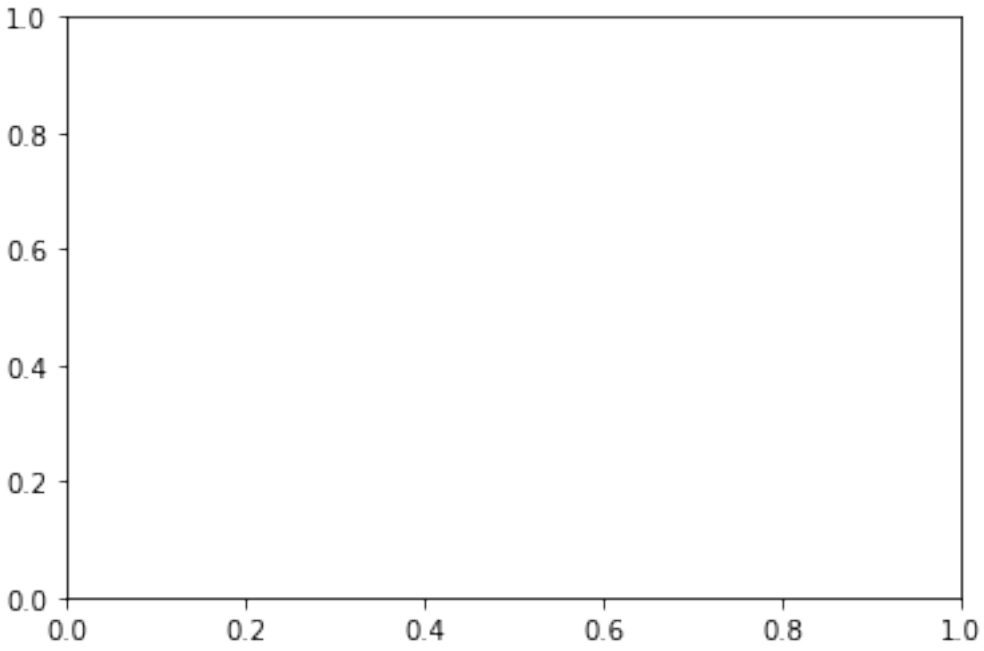
[34]:
```
correct = 0
for i in range(len(X)):
    predict_me = np.array(X[i].astype(float))
    predict_me = predict_me.reshape(-1, len(predict_me))
    prediction = kmeans.predict(predict_me)
    if prediction[0] == y[i]:
        correct += 1

print(correct/len(X))
```

```
0.37373737373737376
```

[35]:
```
K=range(len(X))
plt.plot(K,prediction)
plt.title('The Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('predicted')
plt.show()
```

```
␣
↪---------------------------------------------------------------------------

ValueError                                Traceback (most recent call␣
↪last)

<ipython-input-35-10e6ce5f1fc1> in <module>
  1 K=range(len(X))
----> 2 plt.plot(K,prediction)
  3 plt.title('The Elbow Method')
  4 plt.xlabel('Number of clusters')
  5 plt.ylabel('predicted')


~\Anaconda3\lib\site-packages\matplotlib\pyplot.py in plot(scalex,␣
↪scaley, data, *args, **kwargs)
 2761     return gca().plot(
 2762         *args, scalex=scalex, scaley=scaley, **({"data": data} if␣
↪data
-> 2763         is not None else {}), **kwargs)
 2764
 2765


~\Anaconda3\lib\site-packages\matplotlib\axes\_axes.py in plot(self,␣
↪scalex, scaley, data, *args, **kwargs)
 1644         """
 1645         kwargs = cbook.normalize_kwargs(kwargs, mlines.Line2D)
-> 1646         lines = [*self._get_lines(*args, data=data, **kwargs)]
 1647         for line in lines:
 1648             self.add_line(line)


~\Anaconda3\lib\site-packages\matplotlib\axes\_base.py in __call__(self,␣
↪*args, **kwargs)
 214                 this += args[0],
 215                 args = args[1:]
--> 216             yield from self._plot_args(this, kwargs)
 217
```

```
    218     def get_next_color(self):


~\Anaconda3\lib\site-packages\matplotlib\axes\_base.py in _plot_args(self, tup, kwargs)
    340
    341         if x.shape[0] != y.shape[0]:
--> 342             raise ValueError(f"x and y must have same first dimension, but "
    343                              f"have shapes {x.shape} and {y.shape}")
    344         if x.ndim > 2 or y.ndim > 2:


ValueError: x and y must have same first dimension, but have shapes (891,) and (1,)
```



[ ]: