K_means Algorithm

In [1]:

```
import pandas as pd
import matplotlib.pyplot as plt
data=pd.read_csv('kmeans_data.csv')
```

Visualize Data

In [2]:

```
data.head()
```

Out[2]:

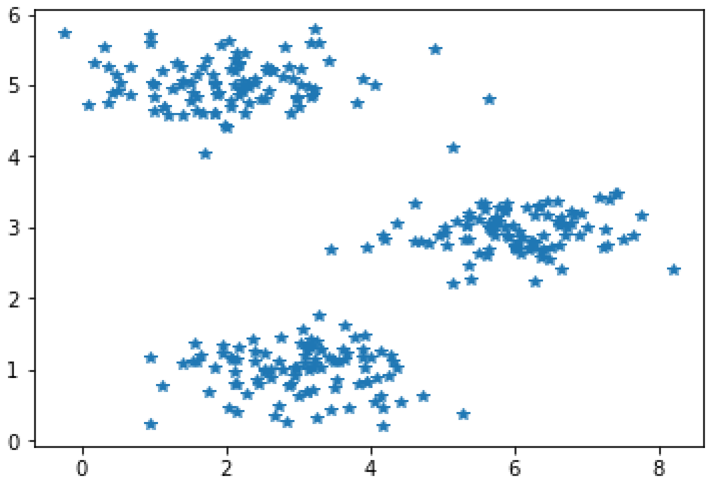
	x1	x2
0	1.842080	4.607572
1	5.658583	4.799964
2	6.352579	3.290854
3	2.904017	4.612204
4	3.231979	4.939894

In [3]:

```
plt.plot(data['x1'],data['x2'],'*')
```

Out[3]:

```
[<matplotlib.lines.Line2D at 0x1c6048c1978>]
```



In [4]:

```
%matplotlib inline
import numpy as np
from sklearn.cluster import KMeans
```

In [5]:

```
X=data.as_matrix()
```

```
C:\Users\Rajat_PC\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: Fut
ureWarning: Method .as_matrix will be removed in a future version. Use .va
lues instead.
  """Entry point for launching an IPython kernel.
```

In [6]:

```
X.shape
```

Out[6]:

```
(300, 2)
```

Apply Kmeans with k=2 Clusters

In [9]:

```
kmeans = KMeans(n_clusters=2)
kmeans.fit(X)
```

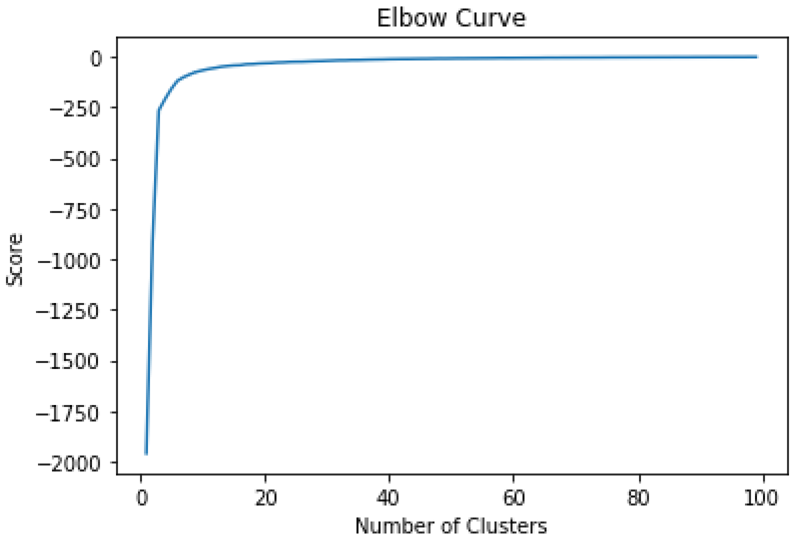
Out[9]:

```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
    n_clusters=2, n_init=10, n_jobs=None, precompute_distances='auto',
    random_state=None, tol=0.0001, verbose=0)
```

Apply Kmeans with k=1 to 100 and Plot Elbow Curve for max_iter=300

In [10]:

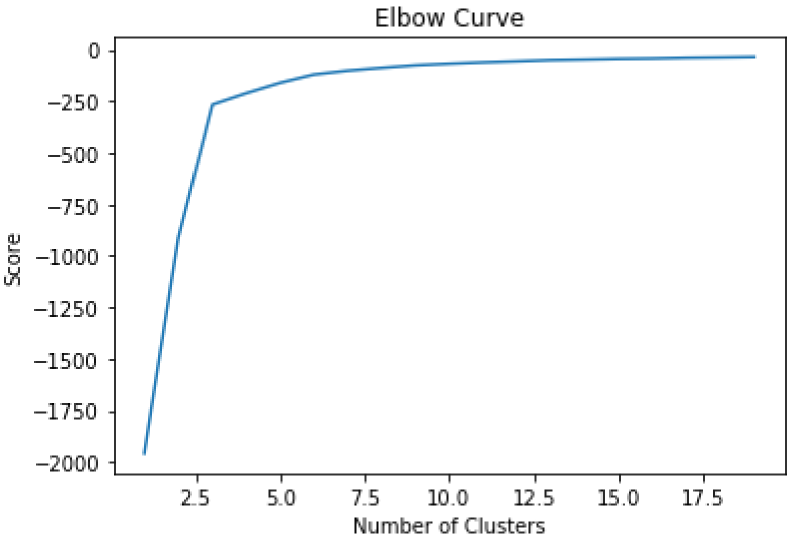
```
Nc = range(1, 100)

kmeans = [KMeans(n_clusters=i) for i in Nc]

kmeans

score = [kmeans[i].fit(X).score(X) for i in range(len(kmeans))]

score

plt.plot(Nc,score)

plt.xlabel('Number of Clusters')

plt.ylabel('Score')

plt.title('Elbow Curve')

plt.show()
```



Apply Kmeans with k=1 to 20 and Plot Elbow Curve for max_iter=1000

In [15]:

```
Nc = range(1, 20)

kmeans = [KMeans(n_clusters=i,max_iter=1000) for i in Nc]

kmeans

score = [kmeans[i].fit(X).score(X) for i in range(len(kmeans))]

score

plt.plot(Nc,score)

plt.xlabel('Number of Clusters')

plt.ylabel('Score')

plt.title('Elbow Curve')

plt.show()
```



As can be seen, K=3 suits best

In [16]:

```
# Best K=3 as can be seen
kmeans = KMeans(n_clusters=3)
kmeans.fit(X)
kmeans.get_params(deep=True)
```

Out[16]:

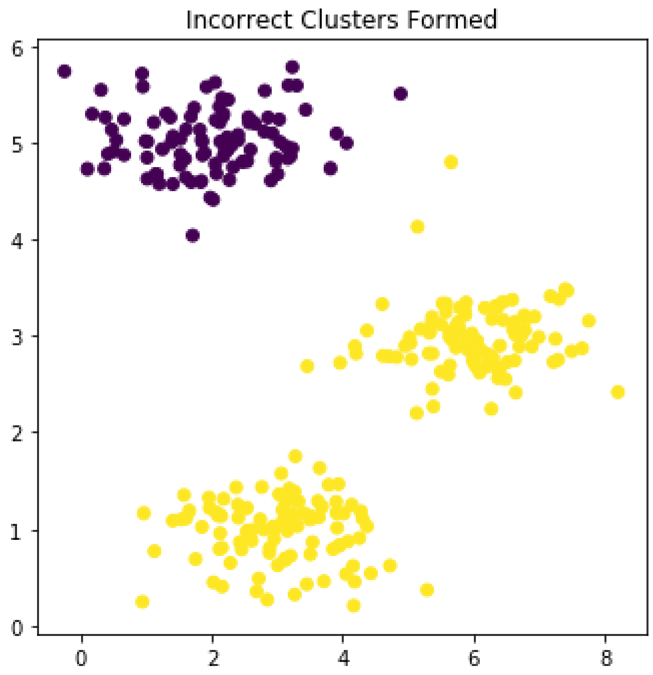
```
{'algorithm': 'auto',
 'copy_x': True,
 'init': 'k-means++',
 'max_iter': 300,
 'n_clusters': 3,
 'n_init': 10,
 'n_jobs': None,
 'precompute_distances': 'auto',
 'random_state': None,
 'tol': 0.0001,
 'verbose': 0}
```

In [22]:

```
n_samples = 1500
random_state = 170
y_pred = KMeans(n_clusters=2, random_state=random_state).fit_predict(X)
plt.figure(figsize=(12, 12))
plt.subplot(221)
plt.scatter(X[:, 0], X[:, 1], c=y_pred)
plt.title("Incorrect Clusters Formed")
```

Out[22]:

```
Text(0.5, 1.0, 'Incorrect Clusters Formed')
```

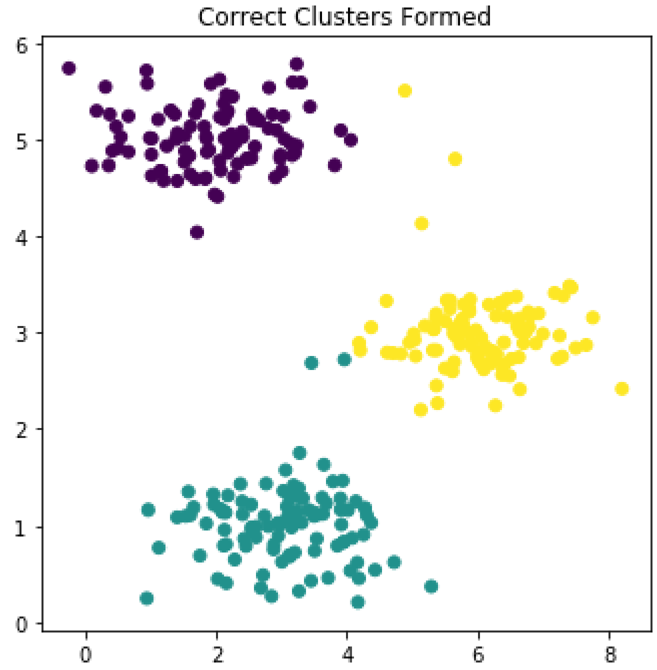


In [24]:

```
n_samples = 1500
random_state = 170
y_pred = KMeans(n_clusters=3, random_state=random_state).fit_predict(X)
plt.figure(figsize=(12, 12))
plt.subplot(221)
plt.scatter(X[:, 0], X[:, 1], c=y_pred)
plt.title("Correct Clusters Formed")
```

Out[24]:

```
Text(0.5, 1.0, 'Correct Clusters Formed')
```

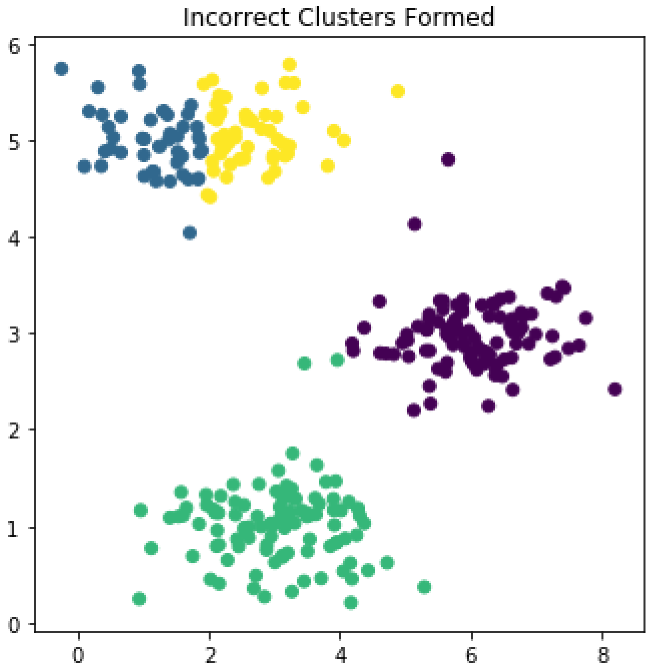


In [25]:

```
n_samples = 1500
random_state = 170
y_pred = KMeans(n_clusters=4, random_state=random_state).fit_predict(X)
plt.figure(figsize=(12, 12))
plt.subplot(221)
plt.scatter(X[:, 0], X[:, 1], c=y_pred)
plt.title("Incorrect Clusters Formed")
```

Out[25]:

```
Text(0.5, 1.0, 'Incorrect Clusters Formed')
```



In [ ]:

```
```