K Means

In [ ]:
```
import pandas as pd
from sklearn.cluster import KMeans
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
```

In [ ]:
```
df=pd.read_csv('Live.csv',sep=",")
```

In [ ]:
```
df
```

Out[ ]:

	status_id	status_type	status_published	num_reactions	num_comments	num_shares	num_likes	num_loves	num_wows	num_hahas	num
0	246675545449582_1649696485147474	video	4/22/2018 6:00	529	512	262	432	92	3	1	
1	246675545449582_1649426988507757	photo	4/21/2018 22:45	150	0	0	150	0	0	0	
2	246675545449582_1648730588577397	video	4/21/2018 6:17	227	236	57	204	21	1	1	
3	246675545449582_1648576705259452	photo	4/21/2018 2:29	111	0	0	111	0	0	0	
4	246675545449582_1645700502213739	photo	4/18/2018 3:22	213	0	0	204	9	0	0	
...	...	...	...	...	...	...	...	...	...	...	
7045	1050855161656896_1061863470556065	photo	9/24/2016 2:58	89	0	0	89	0	0	0	
7046	1050855161656896_1061334757275603	photo	9/23/2016 11:19	16	0	0	14	1	0	1	
7047	1050855161656896_1060126464063099	photo	9/21/2016 23:03	2	0	0	1	1	0	0	
7048	1050855161656896_1058663487542730	photo	9/20/2016 0:43	351	12	22	349	2	0	0	
7049	1050855161656896_1050858841656528	photo	9/10/2016 10:30	17	0	0	17	0	0	0	

7050 rows × 16 columns

In [ ]:
```
df.describe()
```

Out[ ]:

	num_reactions	num_comments	num_shares	num_likes	num_loves	num_wows	num_hahas	num_sads	num_angrys	Column1	Column2	Column3	Column4
count	7050.000000	7050.000000	7050.000000	7050.000000	7050.000000	7050.000000	7050.000000	7050.000000	7050.000000	0.0	0.0	0.0	0.0
mean	230.117163	224.356028	40.022553	215.043121	12.728652	1.289362	0.696454	0.243688	0.113191	NaN	NaN	NaN	NaN
std	462.625309	889.636820	131.599965	449.472357	39.972930	8.719650	3.957183	1.597156	0.726812	NaN	NaN	NaN	NaN
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	NaN	NaN	NaN	NaN
25%	17.000000	0.000000	0.000000	17.000000	0.000000	0.000000	0.000000	0.000000	0.000000	NaN	NaN	NaN	NaN
50%	59.500000	4.000000	0.000000	58.000000	0.000000	0.000000	0.000000	0.000000	0.000000	NaN	NaN	NaN	NaN
75%	219.000000	23.000000	4.000000	184.750000	3.000000	0.000000	0.000000	0.000000	0.000000	NaN	NaN	NaN	NaN
max	4710.000000	20990.000000	3424.000000	4710.000000	657.000000	278.000000	157.000000	51.000000	31.000000	NaN	NaN	NaN	NaN

In [ ]:
```
df=df.drop(["Column1","Column2","Column3","Column4"],axis=1)
```

In [ ]:
```
df["status_id"].unique()
```

Out[ ]:
```
array(['246675545449582_1649696485147474',
       '246675545449582_1649426988507757',
       '246675545449582_1648730588577397', ...,
       '1050855161656896_1060126464063099',
       '1050855161656896_1058663487542730',
       '1050855161656896_1050858841656528'], dtype=object)
```

In [ ]:
```
df["status_published"].unique()
```

Out[ ]:
```
array(['4/22/2018 6:00', '4/21/2018 22:45', '4/21/2018 6:17', ...,
       '9/21/2016 23:03', '9/20/2016 0:43', '9/10/2016 10:30'],
      dtype=object)
```

In [ ]:
```
df=df.drop(["status_published","status_id"],axis=1)
```

In [ ]:
```
df.isna().sum()
```

Out[ ]:
```
status_type      0
num_reactions    0
num_comments     0
num_shares       0
num_likes        0
num_loves        0
num_wows         0
num_hahas        0
num_sads         0
num_angrys       0
dtype: int64
```

In [ ]:
```
le = LabelEncoder()
df['status_type'] = le.fit_transform(df['status_type'])
```

In [ ]:
```
df
```

Out[ ]:

	status_type	num_reactions	num_comments	num_shares	num_likes	num_loves	num_wows	num_hahas	num_sads	num_angrys
0	3	529	512	262	432	92	3	1	1	0
1	1	150	0	0	150	0	0	0	0	0
2	3	227	236	57	204	21	1	1	0	0
3	1	111	0	0	111	0	0	0	0	0
4	1	213	0	0	204	9	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...
7045	1	89	0	0	89	0	0	0	0	0
7046	1	16	0	0	14	1	0	1	0	0
7047	1	2	0	0	1	1	0	0	0	0
7048	1	351	12	22	349	2	0	0	0	0
7049	1	17	0	0	17	0	0	0	0	0

7050 rows × 10 columns

In [ ]:
```
X=df.drop(["status_type"],axis=1)
y=df["status_type"]
```

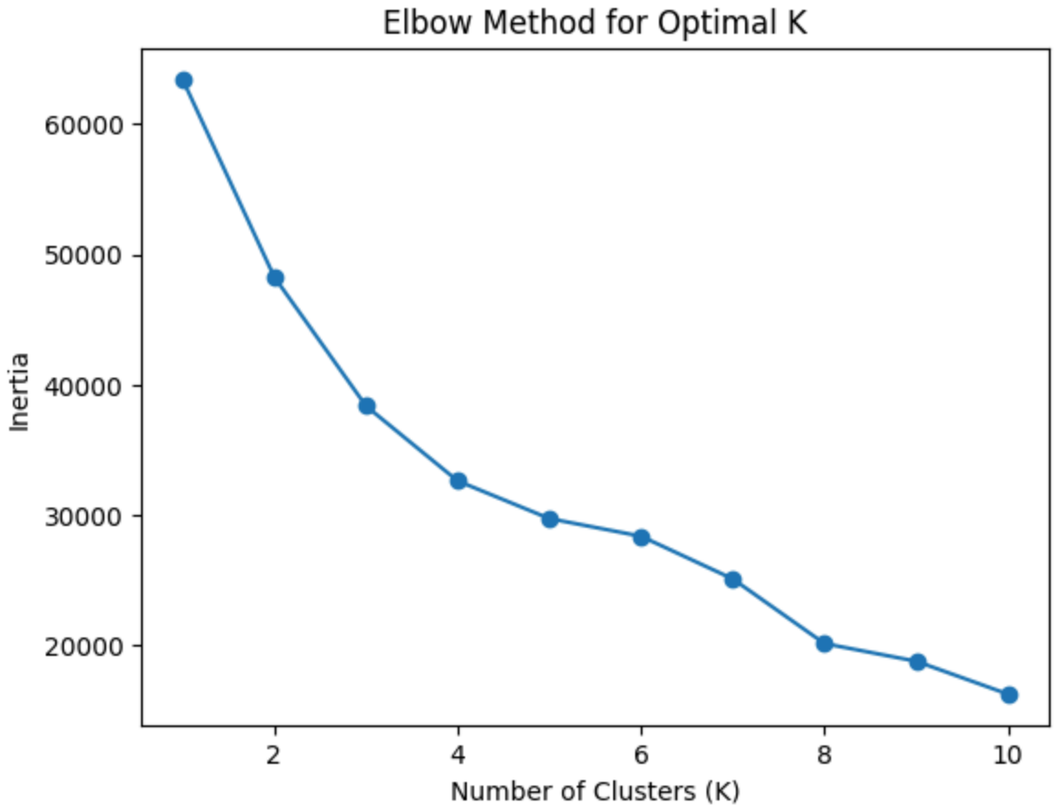
In [ ]:
```
sc = StandardScaler()
X = sc.fit_transform(X)
```

In [ ]:
```
inertia = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, random_state=42)
    kmeans.fit(X)
    inertia.append(kmeans.inertia_)
```

In [ ]:
```
inertia
```

Out[ ]:
```
[63449.999999999876,
 48278.098242205444,
 38372.997723869616,
 32616.927771149505,
 29726.631702081617,
 28368.06407358091,
 25093.341908878676,
 20129.409640055328,
 18773.60665082319,
 16249.783728898929]
```

In [ ]:
```
plt.plot(range(1, 11), inertia, marker='o')
plt.title('Elbow Method for Optimal K')
plt.xlabel('Number of Clusters (K)')
plt.ylabel('Inertia')
plt.show()
```



In [ ]:
```
kmeans = KMeans(n_clusters=3)
kmeans.fit(X)
```

Out[ ]:
```
KMeans
KMeans(n_clusters=3)
```

In [ ]:
```
plt.figure(figsize=(8, 6))
plt.scatter(X[:, 0], X[:, 1], c=df['status_type'], cmap='viridis', alpha=0.5, edgecolors='k')
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], c='red', marker='x', label='Centroids')
plt.title('K-Means Clustering Results')
plt.xlabel('Feature 1 (Standardized)')
plt.ylabel('Feature 2 (Standardized)')
```

```
plt.colorbar(label='Cluster')
plt.legend()
plt.show()
```