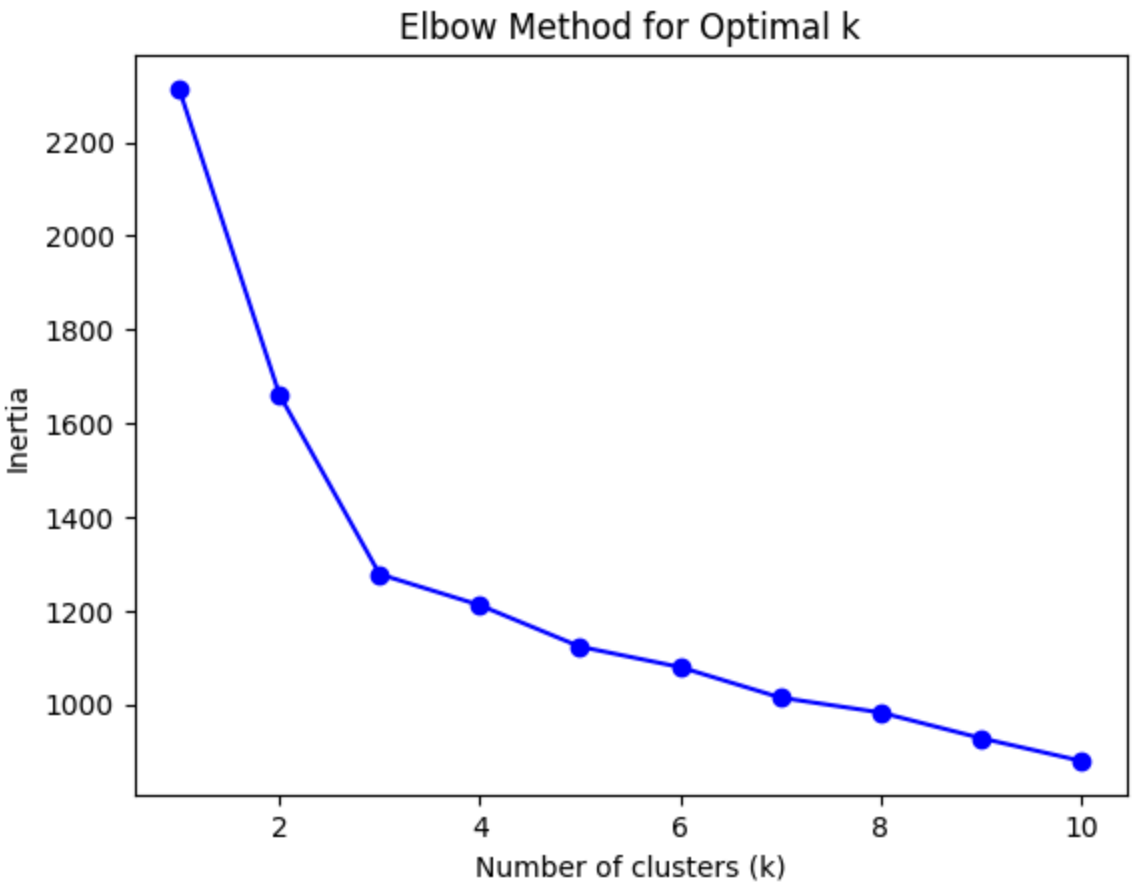
In [1]:
```
from sklearn.datasets import load_wine
import pandas as pd
wine = load_wine()
X = pd.DataFrame(wine.data, columns=wine.feature_names)
X.head()
```

Out[1]:

	alcohol	malic_acid	ash	alcalinity_of_ash	magnesium	total_phenols	flavan
0	14.23	1.71	2.43	15.6	127.0	2.80	
1	13.20	1.78	2.14	11.2	100.0	2.65	
2	13.16	2.36	2.67	18.6	101.0	2.80	
3	14.37	1.95	2.50	16.8	113.0	3.85	
4	13.24	2.59	2.87	21.0	118.0	2.80	

In [2]:
```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

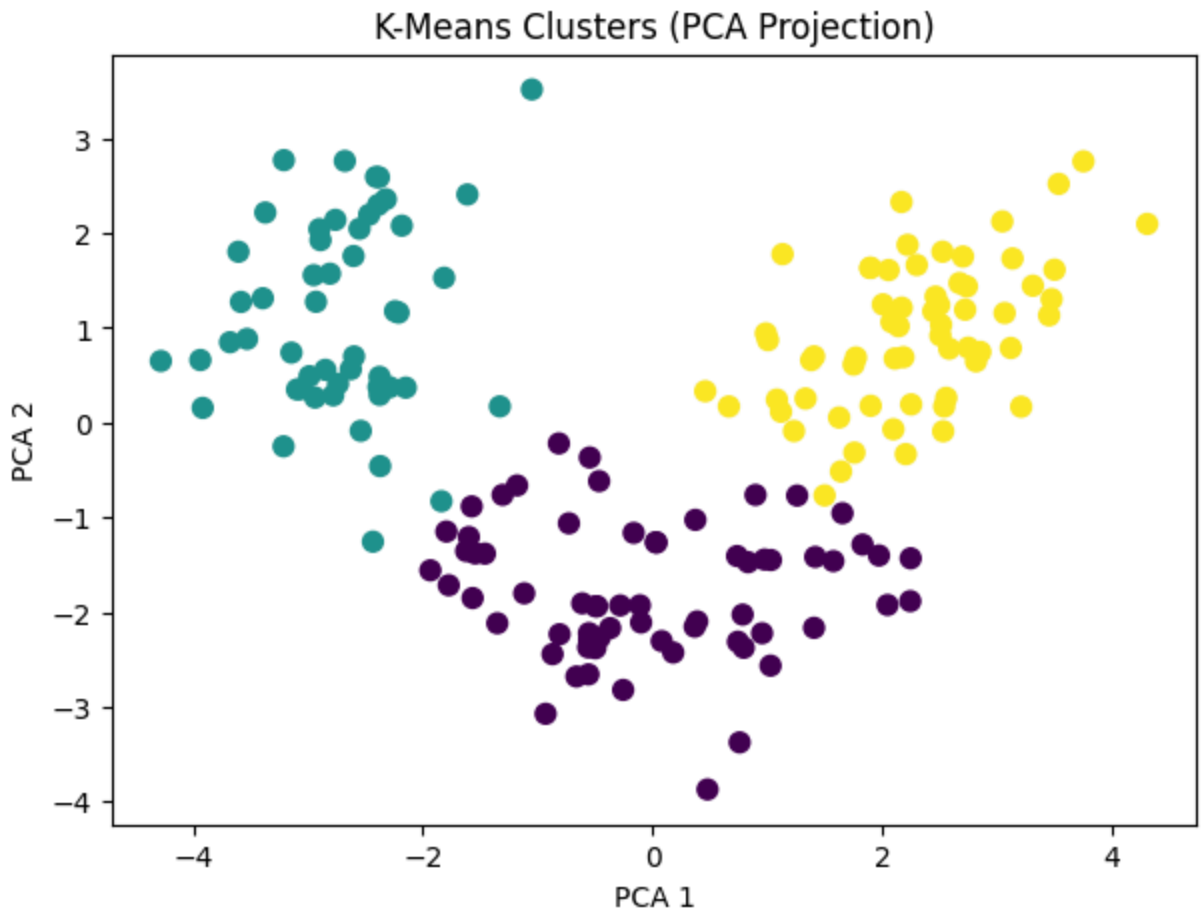
In [3]:
```
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
inertias = []
Kmax = 10
for k in range(1, Kmax + 1):
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(X_scaled)
    inertias.append(kmeans.inertia_)
plt.plot(range(1, Kmax + 1), inertias, 'bo-')
plt.xlabel('Number of clusters (k)')
plt.ylabel('Inertia')
plt.title('Elbow Method for Optimal k')
plt.show()
```



In [4]:
```python
from sklearn.metrics import silhouette_score
k_opt = 3  # use the elbow value
kmeans = KMeans(n_clusters=k_opt, random_state=42)
labels_kmeans = kmeans.fit_predict(X_scaled)
silhouette_kmeans = silhouette_score(X_scaled, labels_kmeans)
print("KMeans Silhouette Score:", silhouette_kmeans)
```

```
KMeans Silhouette Score: 0.2848589191898987
```

In [5]:
```python
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)
plt.figure(figsize=(7,5))
plt.scatter(X_pca[:,0], X_pca[:,1], c=labels_kmeans, cmap='viridis', s=50)
plt.title('K-Means Clusters (PCA Projection)')
plt.xlabel('PCA 1')
plt.ylabel('PCA 2')
plt.show()
```



In [6]:
```
import numpy as np
n_base = 10
base_labels = []
for i in range(n_base):
    k = np.random.choice([2, 3, 4, 5])  # varying k
    kmeans = KMeans(n_clusters=k, random_state=i)
    base_labels.append(kmeans.fit_predict(X_scaled))
base_labels = np.array(base_labels)
print("Base clusterings shape:", base_labels.shape)
```

```
Base clusterings shape: (10, 178)
```

In [7]:
```
n_samples = X_scaled.shape[0]
C = np.zeros((n_samples, n_samples))
for labels in base_labels:
    for i in range(n_samples):
        for j in range(n_samples):
            if labels[i] == labels[j]:
                C[i, j] += 1
C /= n_base
```

In [8]:
```
from sklearn.cluster import SpectralClustering
spectral = SpectralClustering(n_clusters=k_opt, affinity='precomputed', random
labels_ensemble = spectral.fit_predict(C)
silhouette_ensemble = silhouette_score(X_scaled, labels_ensemble)
```

```
print("Ensemble (Spectral) Silhouette Score:", silhouette_ensemble)
```

```
Ensemble (Spectral) Silhouette Score: 0.2848589191898987
```

In [9]:
```
fig, ax = plt.subplots(1, 2, figsize=(12,5))
ax[0].scatter(X_pca[:,0], X_pca[:,1], c=labels_kmeans, cmap='viridis', s=50)
ax[0].set_title('K-Means Clusters')
ax[1].scatter(X_pca[:,0], X_pca[:,1], c=labels_ensemble, cmap='plasma', s=50)
ax[1].set_title('Ensemble (Spectral) Clusters')
plt.show()
```