yil6qjm6t

October 9, 2024

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
from sklearn.cluster import AgglomerativeClustering
from sklearn.preprocessing import StandardScaler, normalize
from sklearn.metrics import silhouette_score
import scipy.cluster.hierarchy as shc
```

[2]:
```
# Changing the working location to the location of the file
# (Note: Use this command in your terminal or Jupyter environment if needed)
# !cd C:\Users\Dev\Desktop\Kaggle\Credit_Card

# Load the dataset
X = pd.read_csv('CC_GENERAL.csv')

# Dropping the CUST_ID column from the data
X = X.drop('CUST_ID', axis=1)

# Handling the missing values using forward fill
X.ffill(inplace=True)
```

[3]:
```
# Scaling the data so that all the features become comparable
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Normalizing the data so that the data approximately follows a Gaussian␣
 ↪distribution
X_normalized = normalize(X_scaled)

# Converting the numpy array into a pandas
DataFrameX_normalized = pd.DataFrame(X_normalized)
```

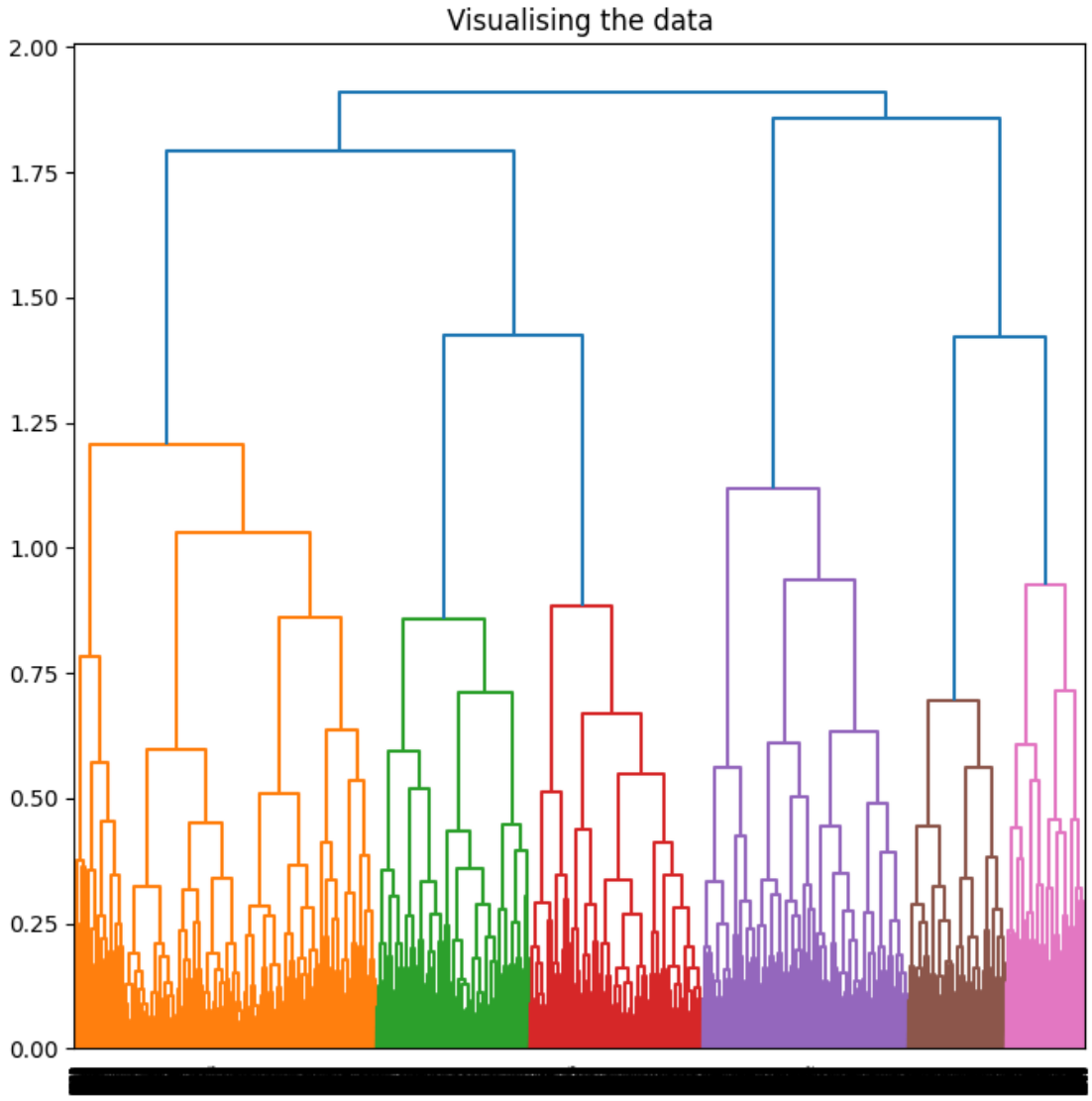
[4]:
```
# Reducing the dimensionality of the Data
pca = PCA(n_components = 2)
X_principal = pca.fit_transform(X_normalized)
X_principal = pd.DataFrame(X_principal)
```

```
X_principal.columns = ['P1', 'P2']
```

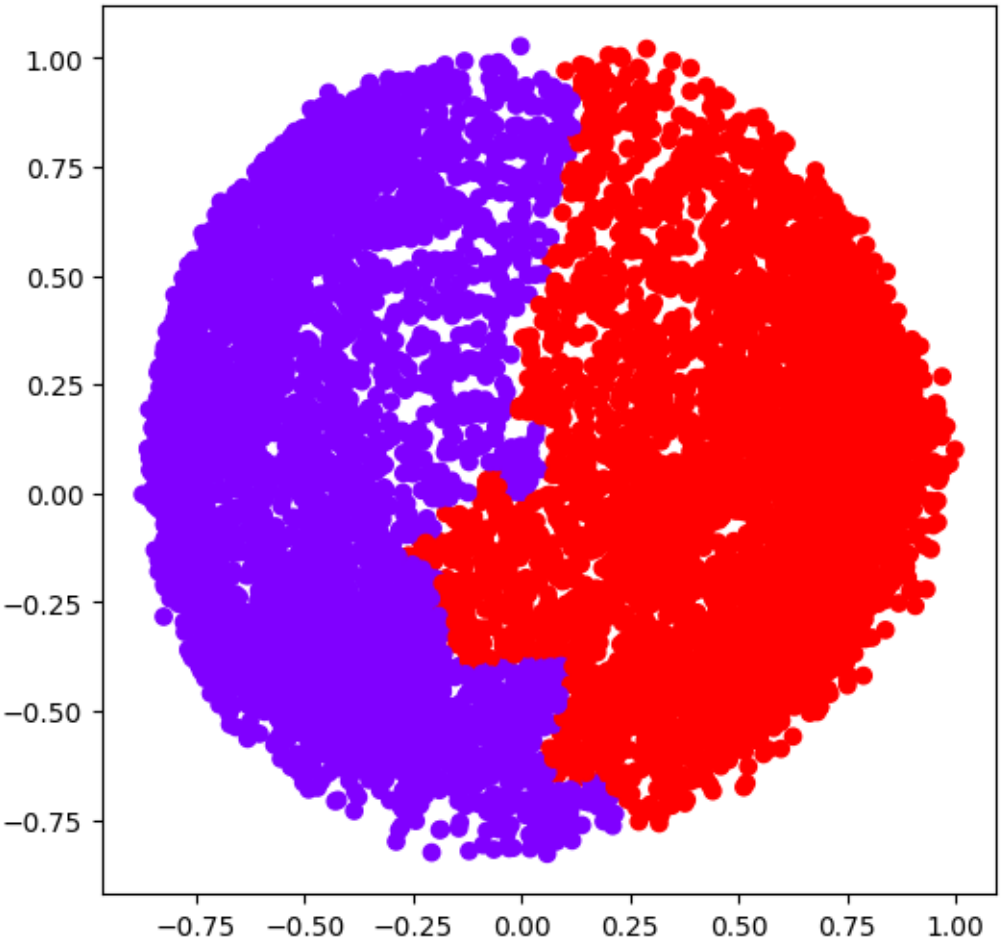
[5]:
```
print(X_principal.shape)
```

```
(8950, 2)
```

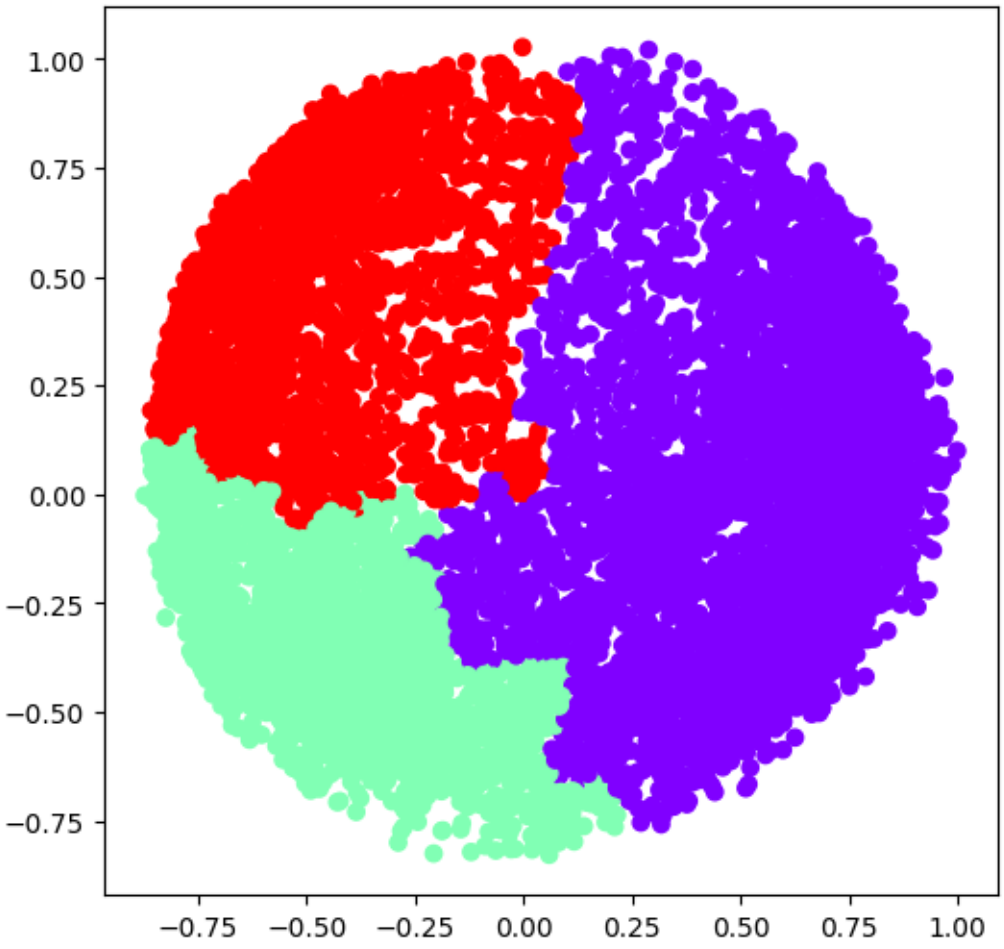
[6]:
```
# Visualizing theworking of the Dendrograms
# Dendrograms are used to divide a given clusterinto many different clusters
plt.figure(figsize =(8, 8))
plt.title('Visualising the data')
Dendrogram = shc.dendrogram((shc.linkage(X_principal, method ='complete')))
```



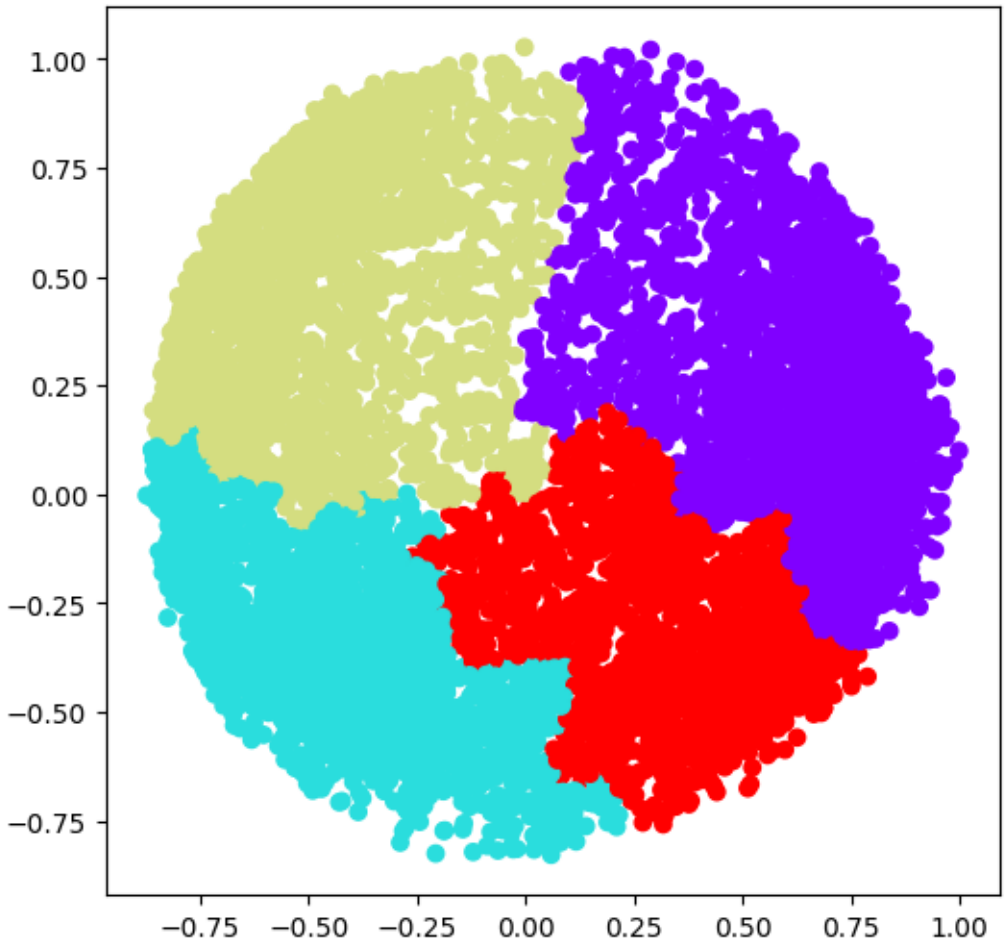
[7]:
```
# Building and Visualizing the different clustering models for different values␣
↪of k
# k = 2
ac2 = AgglomerativeClustering(n_clusters = 2)
plt.figure(figsize =(6, 6))
plt.scatter(X_principal['P1'], X_principal['P2'],
c = ac2.fit_predict(X_principal), cmap ='rainbow')
plt.show()
```



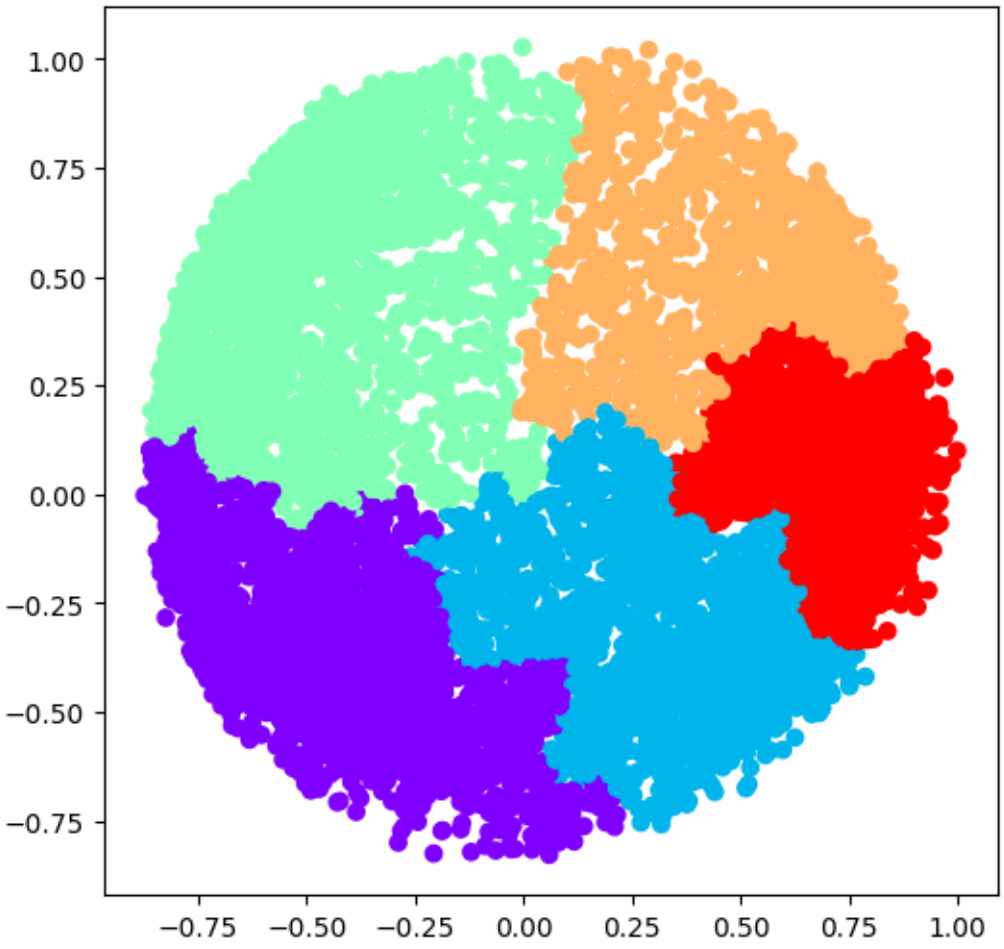
[8]:
```
# k = 3
ac3 = AgglomerativeClustering(n_clusters = 3)
plt.figure(figsize =(6, 6))
plt.scatter(X_principal['P1'],
X_principal['P2'],
c = ac3.fit_predict(X_principal), cmap ='rainbow')
plt.show()
```



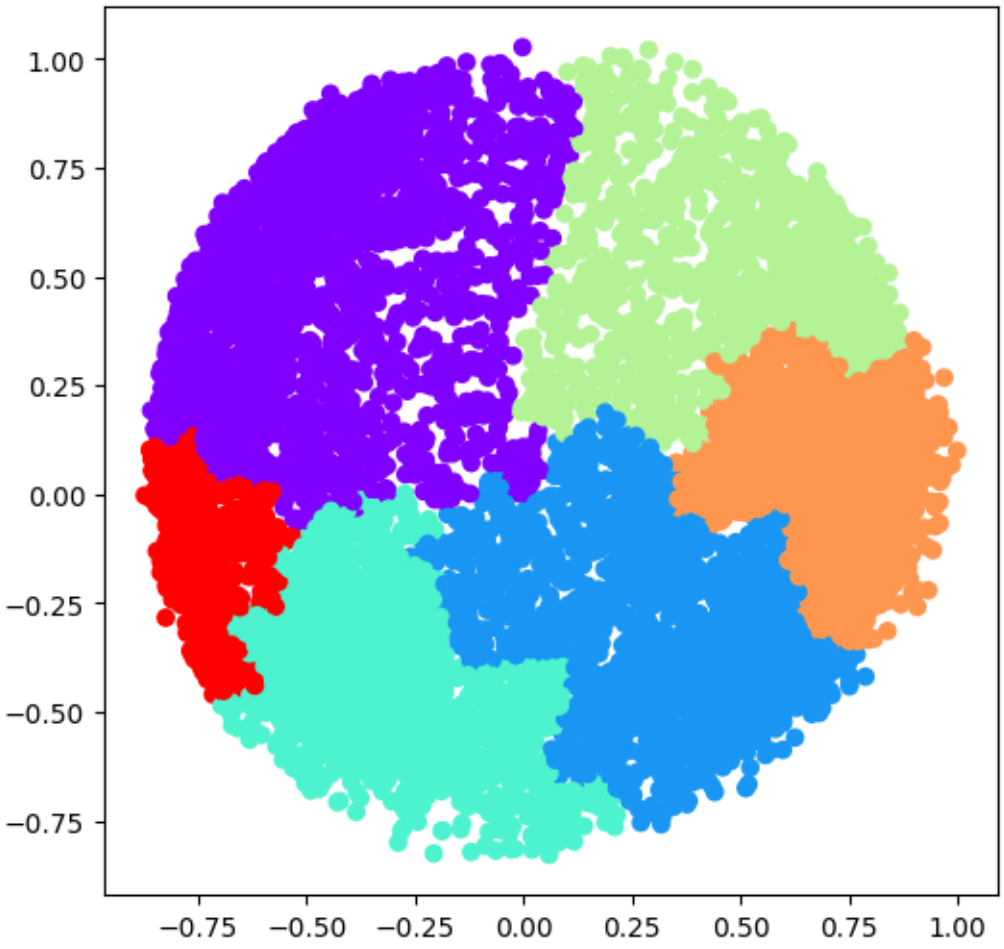
[9]:
```
# k = 4
ac4 = AgglomerativeClustering(n_clusters = 4)
plt.figure(figsize =(6, 6))
plt.scatter(X_principal['P1'],
X_principal['P2'],
c = ac4.fit_predict(X_principal), cmap ='rainbow')
plt.show()
```



[10]:
```
# k = 5
ac5 = AgglomerativeClustering(n_clusters = 5)
plt.figure(figsize =(6, 6))
plt.scatter(X_principal['P1'],
X_principal['P2'],
c = ac5.fit_predict(X_principal), cmap ='rainbow')
plt.show()
```



[11]:
```
# k = 6
ac6 = AgglomerativeClustering(n_clusters = 6)
plt.figure(figsize =(6, 6))
plt.scatter(X_principal['P1'],
X_principal['P2'],
c = ac6.fit_predict(X_principal), cmap ='rainbow')
plt.show()
```

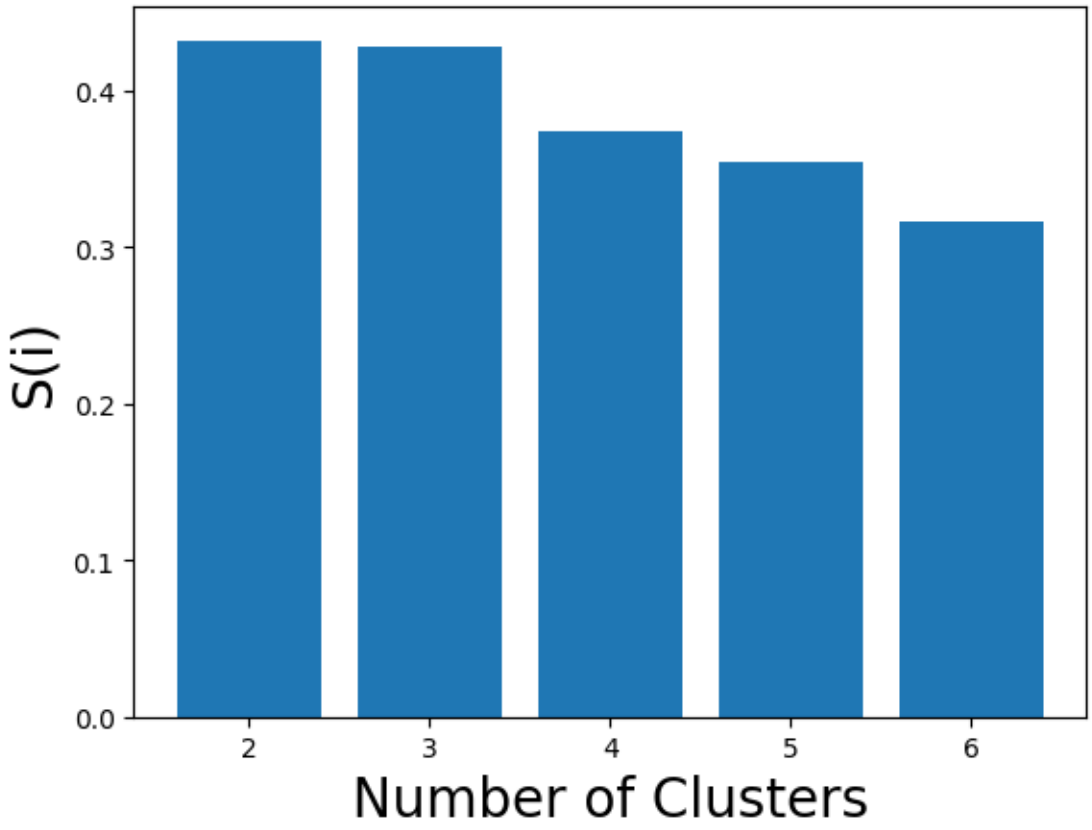


[12]:
```
# Evaluating the different models and Visualizing the results.
k = [2, 3, 4, 5, 6]
```

[13]:
```
# Appending the silhouette scores of the different models to the list
silhouette_scores = []
silhouette_scores.append(silhouette_score(X_principal, ac2.
 ↪fit_predict(X_principal)))
silhouette_scores.append(silhouette_score(X_principal, ac3.
 ↪fit_predict(X_principal)))
silhouette_scores.append(silhouette_score(X_principal, ac4.
 ↪fit_predict(X_principal)))
silhouette_scores.append(silhouette_score(X_principal, ac5.
 ↪fit_predict(X_principal)))
silhouette_scores.append(silhouette_score(X_principal, ac6.
 ↪fit_predict(X_principal)))
print(silhouette_scores)
```

```
[np.float64(0.43236768272675374), np.float64(0.4278447856411723),
np.float64(0.3746584392060403), np.float64(0.35512470426139253),
np.float64(0.31685879889867724)]
```

[14]:
```
# Plotting a bar graph to compare the results
plt.bar(k,silhouette_scores)
plt.xlabel('Number of Clusters', fontsize = 20)
plt.ylabel('S(i)',fontsize = 20)
plt.show()
```



[ ]: