In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
data = pd.read_csv('College.csv')
```

In [3]:
```
data.head()
```

Out[3]:

	Unnamed: 0	Private	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate	Room.Board	Books	Personal	PhD
0	Abilene Christian University	Yes	1660	1232	721	23	52	2885	537	7440	3300	450	2200	70
1	Adelphi University	Yes	2186	1924	512	16	29	2683	1227	12280	6450	750	1500	29
2	Adrian College	Yes	1428	1097	336	22	50	1036	99	11250	3750	400	1165	53
3	Agnes Scott College	Yes	417	349	137	60	89	510	63	12960	5450	450	875	92
4	Alaska Pacific University	Yes	193	146	55	16	44	249	869	7560	4120	800	1500	76

In [4]:
```
data.tail()
```

Out[4]:

	Unnamed: 0	Private	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate	Room.Board	Books	Personal	P
772	Worcester State College	No	2197	1515	543	4	26	3089	2029	6797	3900	500	1200	
773	Xavier University	Yes	1959	1805	695	24	47	2849	1107	11520	4960	600	1250	
774	Xavier University of Louisiana	Yes	2097	1915	695	34	61	2793	166	6900	4200	617	781	
775	Yale University	Yes	10705	2453	1317	95	99	5217	83	19840	6510	630	2115	
776	York College of Pennsylvania	Yes	2989	1855	691	28	63	2988	1726	4990	3560	500	1250	

In [5]:
```
data.shape
```

Out[5]:
```
(777, 19)
```

In [6]:
```
data.isnull().sum()
```

Out[6]:
```
Unnamed: 0    0
Private       0
Apps          0
Accept        0
Enroll        0
Top10perc     0
Top25perc     0
F.Undergrad   0
P.Undergrad   0
Outstate      0
Room.Board    0
Books         0
Personal      0
PhD           0
Terminal      0
S.F.Ratio     0
perc.alumni   0
Expend        0
Grad.Rate     0
dtype: int64
```

In [7]:
```
data.nunique()
```

Out[7]:
```
Unnamed: 0    777
Private         2
Apps          711
Accept        693
Enroll        581
Top10perc      82
Top25perc      89
F.Undergrad   714
P.Undergrad   566
Outstate      640
Room.Board    553
Books         122
Personal      294
PhD            78
Terminal       65
S.F.Ratio     173
perc.alumni    61
Expend        744
Grad.Rate      81
dtype: int64
```

In [8]:
```
data.duplicated().sum()
```

Out[8]:
```
0
```

In [9]:
```
data.describe()
```

Out[9]:

	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate	Room.Board	Books
count	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000
mean	3001.638353	2018.804376	779.972973	27.558559	55.796654	3699.907336	855.298584	10440.669241	4357.526384	549.380952
std	3870.201484	2451.113971	929.176190	17.640364	19.804778	4850.420531	1522.431887	4023.016484	1096.696416	165.105360
min	81.000000	72.000000	35.000000	1.000000	9.000000	139.000000	1.000000	2340.000000	1780.000000	96.000000
25%	776.000000	604.000000	242.000000	15.000000	41.000000	992.000000	95.000000	7320.000000	3597.000000	470.000000
50%	1558.000000	1110.000000	434.000000	23.000000	54.000000	1707.000000	353.000000	9990.000000	4200.000000	500.000000
75%	3624.000000	2424.000000	902.000000	35.000000	69.000000	4005.000000	967.000000	12925.000000	5050.000000	600.000000
max	48094.000000	26330.000000	6392.000000	96.000000	100.000000	31643.000000	21836.000000	21700.000000	8124.000000	2340.000000

In [10]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 777 entries, 0 to 776
Data columns (total 19 columns):
Unnamed: 0    777 non-null object
Private       777 non-null object
Apps          777 non-null int64
Accept        777 non-null int64
Enroll        777 non-null int64
Top10perc     777 non-null int64
Top25perc     777 non-null int64
F.Undergrad   777 non-null int64
P.Undergrad   777 non-null int64
Outstate      777 non-null int64
Room.Board    777 non-null int64
Books         777 non-null int64
Personal      777 non-null int64
PhD           777 non-null int64
Terminal      777 non-null int64
S.F.Ratio     777 non-null float64
perc.alumni   777 non-null int64
Expend        777 non-null int64
Grad.Rate     777 non-null int64
dtypes: float64(1), int64(16), object(2)
memory usage: 109.3+ KB
```

In [12]:
```
data['Private'].value_counts()
```

Out[12]:
```
Yes    565
No     212
Name: Private, dtype: int64
```

In [13]:
```
s = {'Yes':0,'No':1}
data['Private']=data['Private'].map(s)
```

In [15]:
```
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import silhouette_score
```

In [16]:
```
x = data[['F.Undergrad','F.Undergrad','PhD']]
```

In [18]:
```
data.head()
```

Out[18]:

	Unnamed: 0	Private	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate	Room.Board	Books	Personal	PhD
0	Abilene Christian University	0	1660	1232	721	23	52	2885	537	7440	3300	450	2200	70
1	Adelphi University	0	2186	1924	512	16	29	2683	1227	12280	6450	750	1500	29
2	Adrian College	0	1428	1097	336	22	50	1036	99	11250	3750	400	1165	53
3	Agnes Scott College	0	417	349	137	60	89	510	63	12960	5450	450	875	92
4	Alaska Pacific University	0	193	146	55	16	44	249	869	7560	4120	800	1500	76

In [105]:
```
scaler = StandardScaler()
x = scaler.fit_transform(x)
```

In [106]:
```
k_means = KMeans(n_clusters=5,init='k-means++',random_state=42)
k_means.fit(x)
k_clusters = k_means.fit_predict(x)
k_clusters
```

Out[106]:
```
array([1, 4, 4, 1, 1, 4, 1, 1, 1, 4, 1, 1, 1, 4, 1, 1, 1, 4, 4, 4, 1, 3,
       4, 2, 1, 4, 1, 3, 4, 1, 4, 4, 4, 4, 4, 4, 4, 1, 1, 1, 3, 1, 1, 1, 4,
       1, 4, 4, 4, 1, 1, 4, 4, 4, 4, 1, 1, 4, 4, 1, 1, 3, 4, 1, 1, 1, 1,
       1, 4, 1, 1, 1, 1, 1, 4, 4, 4, 4, 4, 1, 1, 3, 1, 3, 4, 1, 1, 1, 1,
       4, 4, 4, 4, 4, 1, 1, 1, 3, 4, 1, 1, 1, 3, 1, 1, 1, 1, 4, 1, 4, 4,
       4, 4, 4, 4, 1, 1, 4, 4, 1, 1, 1, 1, 4, 4, 4, 4, 1, 1, 1, 1, 1, 1,
       1, 3, 4, 1, 1, 1, 4, 1, 1, 1, 4, 4, 1, 1, 4, 1, 3, 4, 4, 4, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
       1, 1, 1, 4, 1, 4, 4, 4, 4, 4, 1, 1, 1, 1, 1, 4, 3, 4, 4, 1, 1, 1,
       1, 1, 4, 4, 1, 4, 4, 4, 1, 1, 3, 4, 1, 1, 3, 4, 1, 1, 4, 1, 4, 1,
       4, 4, 1, 4, 1, 4, 4, 1, 4, 1, 4, 4, 1, 4, 1, 4, 4, 4, 4, 4, 1, 4,
       1, 3, 4, 4, 1, 1, 1, 3, 1, 4, 3, 1, 1, 4, 4, 1, 4, 4, 4, 4, 4, 1,
       1, 4, 1, 4, 3, 1, 4, 1, 4, 1, 1, 1, 4, 4, 4, 4, 4, 3, 1, 4, 4, 4,
       2, 3, 3, 4, 4, 4, 1, 4, 3, 1, 1, 1, 3, 1, 4, 4, 3, 4, 4, 4, 4, 4,
       3, 1, 4, 1, 1, 3, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 1, 1, 1, 1,
       1, 1, 4, 3, 1, 1, 1, 1, 1, 1, 1, 1, 4, 1, 4, 4, 4, 1, 1, 3, 0,
       1, 1, 1, 3, 4, 4, 1, 1, 4, 4, 1, 1, 1, 1, 4, 4, 1, 1, 1, 4, 4, 1,
       4, 4, 1, 1, 3, 1, 1, 4, 4, 1, 1, 1, 1, 1, 4, 1, 4, 1, 4, 1, 4, 1,
       1, 3, 4, 4, 3, 1, 4, 1, 4, 4, 4, 4, 1, 4, 4, 4, 4, 1, 1, 4, 1, 4,
       1, 1, 1, 4, 4, 4, 1, 1, 1, 3, 3, 0, 3, 3, 3, 3, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 3, 4, 4, 4, 4, 4, 1, 4, 4, 1, 1, 4, 4, 1, 1, 1, 4, 1,
       4, 4, 1, 1, 1, 1, 4, 1, 1, 3, 0, 3, 2, 4, 1, 3, 3, 1, 1, 1, 0, 1,
       1, 1, 4, 0, 4, 4, 3, 0, 4, 4, 2, 4, 4, 4, 4, 4, 0, 0, 3, 3, 0, 0,
       3, 1, 2, 3, 3, 1, 3, 4, 1, 0, 1, 1, 3, 1, 3, 3, 3, 3, 3, 0, 3, 3,
       1, 3, 3, 0, 3, 4, 1, 1, 1, 1, 4, 4, 4, 2, 3, 4, 4, 4, 3, 1,
       1, 2, 0, 3, 4, 1, 1, 1, 2, 3, 3, 0, 3, 1, 1, 1, 3, 1, 0, 2, 3, 4,
       1, 4, 4, 1, 1, 1, 3, 3, 0, 4, 1, 4, 4, 1, 4, 1, 4, 4, 4, 1, 4, 1,
       1, 3, 1, 4, 4, 4, 4, 1, 1, 4, 1, 3, 4, 4, 4, 1, 0, 1, 1, 3, 1,
       4, 4, 4, 2, 1, 4, 4, 4, 4, 1, 1, 4, 4, 1, 4, 4, 1, 4, 1, 4, 4, 1,
       1, 1, 4, 4, 1]1]
```

In [107]:
```
k_means.cluster_centers_
```

Out[107]:
```
array([[ 1.45084133,  3.34439759,  0.83326435],
       [-0.32969619, -0.33050804,  0.61725594],
       [ 5.85032298,  1.89177584,  0.59465721],
       [ 0.70653799,  1.05803017,  0.43136133],
       [-0.25776543, -0.47496849, -0.10961961]])
```

In [108]:
```
print('Silhouette Score :-',silhouette_score(x,k_clusters))
```

```
silhouette Score :- 0.42469460468088921
```

In [109]:
```
SSE =[]
K=range(1,50)

for i in K:
    k_means = KMeans(n_clusters=i,init='k-means++',random_state=42)
    k_means.fit(x)
    SSE.append(k_means.inertia_)
```

In [110]:
```
SSE
```

Out[110]:
```
[2330.999999999995,
 1426.81669387028,
 947.8408504037858,
 744.5615461757828,
 621.3924300210142,
 512.6105557748377,
 436.5048084739908,
 384.1573669240294,
 335.7992764565858,
 297.5130935281439,
 272.9016064200234,
 250.7937593058892,
 232.9098768620597,
 222.03307841046,
 208.9714785897292,
 193.4207342689105,
 185.5265157266649,
 178.3946350882626,
 171.5983593334441,
 164.4275000000001,
 158.4270364667386,
 153.2584913762804,
 146.2024603894932,
 139.9299476685478,
 133.3612661067108,
 129.3638049027893,
 124.6490045972059,
 120.4697241276524,
 115.7842067287107,
 112.4612768913945,
 109.5456373378267,
 105.6122499126923,
 101.5540926770448829,
 99.2933279102639,
 97.3141335069752,
 94.4961840079959,
 92.7435771821784,
 90.7614657862787,
 89.1033593398744,
 87.4443296061327,
 85.6865285254559,
 84.395602664357,
 83.1613697535715,
 80.9433078168419,
 79.0813457028195,
 77.6395514619937,
 76.0228994391963,
 73.0775136680902,
 70.8917399802148]
```

In [111]:
```
plt.plot(K,SSE)
plt.ylabel('Sum Of Squred Error')
plt.grid()
plt.xlabel('Number Of K values')
plt.show()
```

In [117]:
```
k_means = KMeans(n_clusters=5,init='k-means++',random_state=42)
k_means.fit(x)
```

Out[117]:
```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
       n_clusters=5, n_init=10, n_jobs=None, precompute_distances='auto',
       random_state=42, tol=0.0001, verbose=0)
```

In [118]:
```
y_label=k_means.fit_predict(x)
```

In [119]:
```
k_means.cluster_centers_
```

Out[119]:
```
array([[ 1.45084133,  3.34439759,  0.83326435],
       [-0.32969619, -0.33050804,  0.61725594],
       [ 5.85032298,  1.89177584,  0.59465721],
       [ 0.70653799,  1.05803017,  0.43136133],
       [-0.25776543, -0.47496849, -0.10961961]])
```

In [120]:
```
from sklearn.metrics import silhouette_score
```

In [121]:
```
silhouette_score(x,y_label)
```

Out[121]:
```
0.42469460468088921
```

In [ ]: