# Importing Libraries

In [1]:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scikitplot.cluster import plot_elbow_curve
from scikitplot.decomposition import plot_pca_component_variance
from scikitplot.metrics import plot_silhouette
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
import plotly.express as px
from plotly.offline import init_notebook_mode,iplot
init_notebook_mode(connected=True)
```

# Parsing Data and Getting Metrics of Features

In [2]:

```
data = pd.read_csv("data.csv",index_col='id')
data
```

Out[2]:

| id | f_00 | f_01 | f_02 | f_03 | f_04 | f_05 | f_06 | f_07 | f_08 | f_09 | ... | f_19 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | -0.389420 | -0.912791 | 0.648951 | 0.589045 | -0.830817 | 0.733624 | 2.258560 | 2 | 13 | 14 | ... | -0.478412 |
| **1** | -0.689249 | -0.453954 | 0.654175 | 0.995248 | -1.653020 | 0.863810 | -0.090651 | 2 | 3 | 6 | ... | -0.428791 |
| **2** | 0.809079 | 0.324568 | -1.170602 | -0.624491 | 0.105448 | 0.783948 | 1.988301 | 5 | 11 | 5 | ... | -0.413534 |
| **3** | -0.500923 | 0.229049 | 0.264109 | 0.231520 | 0.415012 | -1.221269 | 0.138850 | 6 | 2 | 13 | ... | 0.619283 |
| **4** | -0.671268 | -1.039533 | -0.270155 | -1.830264 | -0.290108 | -1.852809 | 0.781898 | 8 | 7 | 5 | ... | -1.628830 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **97995** | 0.237591 | 1.657034 | -0.689282 | 0.313710 | -0.299039 | 0.329139 | 1.607378 | 5 | 7 | 8 | ... | -0.290116 |
| **97996** | 0.322696 | 0.710411 | 0.562625 | -1.321713 | -0.357708 | 0.182024 | 0.178558 | 3 | 9 | 2 | ... | 0.117687 |
| **97997** | -0.249364 | -0.459545 | 1.886122 | -1.340310 | 0.195029 | -0.559520 | -0.379767 | 8 | 9 | 10 | ... | -0.850223 |
| **97998** | 0.311408 | 2.185237 | 0.761367 | 0.436723 | 0.464967 | 0.062321 | -0.334025 | 1 | 8 | 11 | ... | -0.010839 |
| **97999** | 0.755170 | 0.567483 | 1.456767 | -0.579071 | -0.048474 | -1.206240 | 0.784305 | 0 | 11 | 3 | ... | 1.180805 |

98000 rows × 29 columns

In [3]:

```
data.describe()
```

Out[3]:

| | f_00 | f_01 | f_02 | f_03 | f_04 | f_05 | f_06 | |
|---|---|---|---|---|---|---|---|---|
| **count** | 98000.000000 | 98000.000000 | 98000.000000 | 98000.000000 | 98000.000000 | 98000.000000 | 98000.000000 | 98000.( |
| **mean** | 0.001220 | 0.005580 | -0.001042 | -0.000700 | -0.003522 | -0.001612 | -0.003042 | 5.! |
| **std** | 1.002801 | 1.000742 | 1.001373 | 1.000422 | 1.003061 | 1.000532 | 0.997434 | 3.( |
| **min** | -4.732235 | -4.202795 | -4.377021 | -4.010826 | -4.535903 | -4.300767 | -4.894525 | 0.( |
| **25%** | -0.675226 | -0.670985 | -0.672779 | -0.672540 | -0.682510 | -0.675066 | -0.680421 | 3.( |

| | f_00 | f_01 | f_02 | f_03 | f_04 | f_05 | f_06 | |
|---|---|---|---|---|---|---|---|---|
| **50%** | 0.002022 | 0.006650 | -0.000324 | -0.003185 | -0.003307 | 0.001024 | -0.002053 | 5.( |
| **75%** | 0.677271 | 0.677746 | 0.677086 | 0.672097 | 0.677589 | 0.673344 | 0.668112 | 8.( |
| **max** | 4.490521 | 4.324974 | 4.560247 | 4.399373 | 4.050549 | 4.710316 | 3.998595 | 32.( |

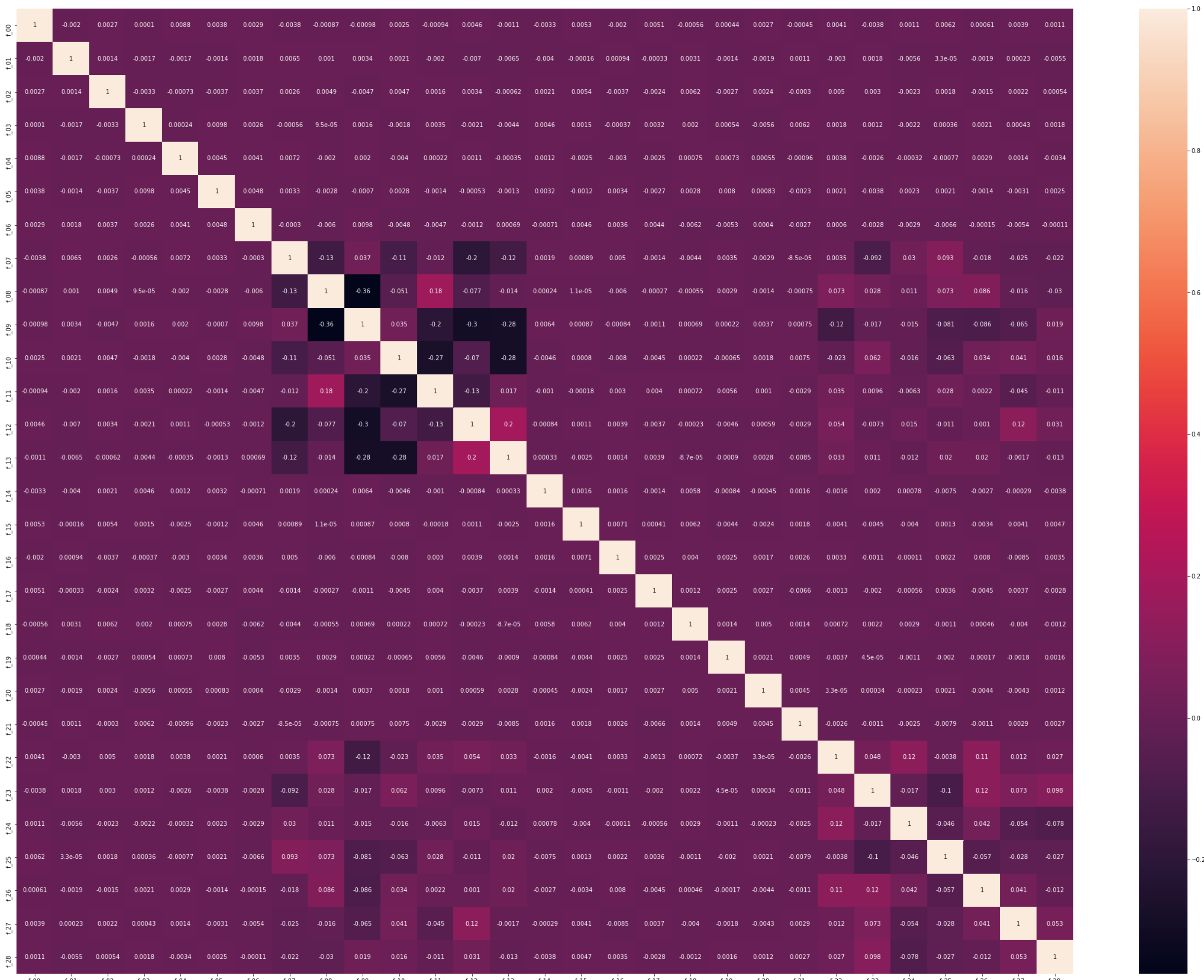
8 rows × 29 columns

# Heatmap to show correlations between features

In [4]:

```
plt.figure(figsize=(40,30))
X=data[['f_00','f_01','f_02','f_03','f_04','f_05','f_06','f_07','f_08','f_09','f_10','f_1
sns.heatmap(X.corr(),annot=True)
```

Out[4]: <AxesSubplot:>



# Values are standardized to simplify clustering process

In [5]:

```
scaler = StandardScaler()
X_std = scaler.fit_transform(X)
X_std
```

Out[5]:
```
array([[-0.3895505 , -0.91769495,  0.649105  , ...,  0.96048158,
         1.04529612,  0.68332274],
       [-0.6885438 , -0.45919476,  0.65432122, ..., -0.55294213,
         0.3554352 , -1.60267076],
       [ 0.80560694,  0.31875345, -1.16796258, ...,  0.97917134,
        -0.92625017, -2.22432704],
       ...,
       [-0.24988501, -0.46478181,  1.88458578, ...,  1.54422962,
         1.1855134 ,  0.57075203],
       [ 0.30932332,  2.17805293,  0.7613671 , ..., -1.08428749,
        -0.5474945 ,  0.10775421],
       [ 0.75184799,  0.56149   ,  1.45581708, ..., -0.63608329,
         1.00085474, -0.31416284]])
```

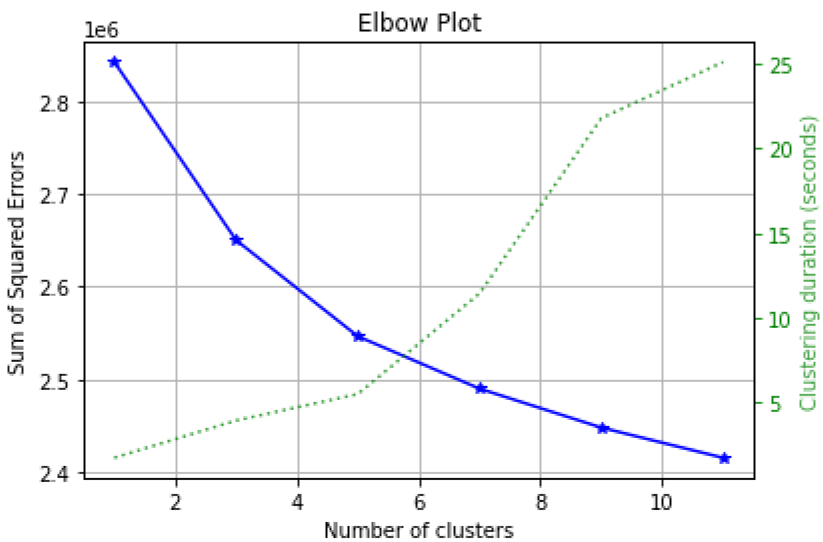
# Initial KMeans Model

In [6]:
```python
model = KMeans(n_clusters=5)
```

# Scikitplot module is used to plot SSE values and clustering durations from 1 to 11 clusters

In [7]:
```python
plot_elbow_curve(model, X_std)
```

Out[7]:
```
<AxesSubplot:title={'center':'Elbow Plot'}, xlabel='Number of clusters', ylabel='Sum of Sq
uared Errors'>
```




# SSE values are shown based on number of clusters in dataframe

In [8]:
```python
sse = []
for n in range(1, 12):
    kmeans = KMeans(n_clusters=n)
    kmeans.fit(X_std)
    sse.append(kmeans.inertia_)

sse_data = pd.DataFrame()
sse_data['Number of Clusters'] = range(1,12)
sse_data['SSE'] = sse
sse_data
```

Out[8]:

| | Number of Clusters | SSE |
|---|---|---|
| **0** | 1 | 2.842000e+06 |
| **1** | 2 | 2.727188e+06 |
| **2** | 3 | 2.650155e+06 |
| **3** | 4 | 2.591869e+06 |
| **4** | 5 | 2.546540e+06 |
| **5** | 6 | 2.515479e+06 |
| **6** | 7 | 2.490155e+06 |
| **7** | 8 | 2.467124e+06 |
| **8** | 9 | 2.448185e+06 |
| **9** | 10 | 2.431116e+06 |
| **10** | 11 | 2.415682e+06 |

# PCA is used with six components to create silhouette plot through cluster labels

In [9]:

```
pca = PCA(n_components=6)
pca.fit(X)
X_pca=pca.transform(X)
X_pca
```

Out[9]:

```
array([[ 1.55615749, -5.55761901, -3.27033444, -8.10451241,  0.07160189,
        -1.38545715],
       [-4.56847689,  0.73947323, 10.15778924, -2.02184815,  2.6929648 ,
        -0.60118678],
       [-4.35162356, -1.55351235, -3.40598442, -0.57164416, -5.90081356,
         1.27071841],
       ...,
       [ 3.22977612, -0.46293048, -2.8466608 ,  1.53698353, -1.34452581,
        -4.53622615],
       [ 3.00420965,  5.38436588,  1.90883229, -4.45553122, -1.48407363,
        -2.2999682 ],
       [-8.76284074,  0.40598569, -1.99838805, -7.08410411,  6.23535882,
         1.76794389]])
```

In [10]:

```
cluster_labels = model.fit_predict(X_std)
cluster_labels
```

Out[10]:

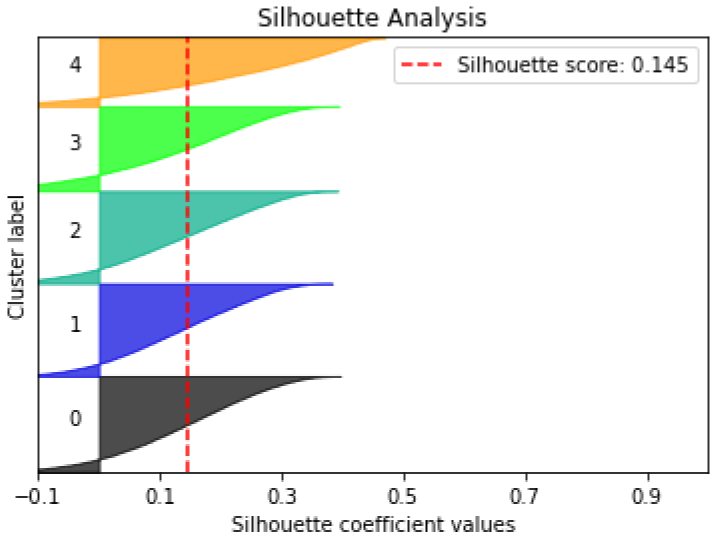
```
array([2, 1, 2, ..., 3, 4, 2])
```

In [11]:

```
plot_silhouette(X_pca, cluster_labels)
```

Out[11]:

```
<AxesSubplot:title={'center':'Silhouette Analysis'}, xlabel='Silhouette coefficient value
s', ylabel='Cluster label'>
```

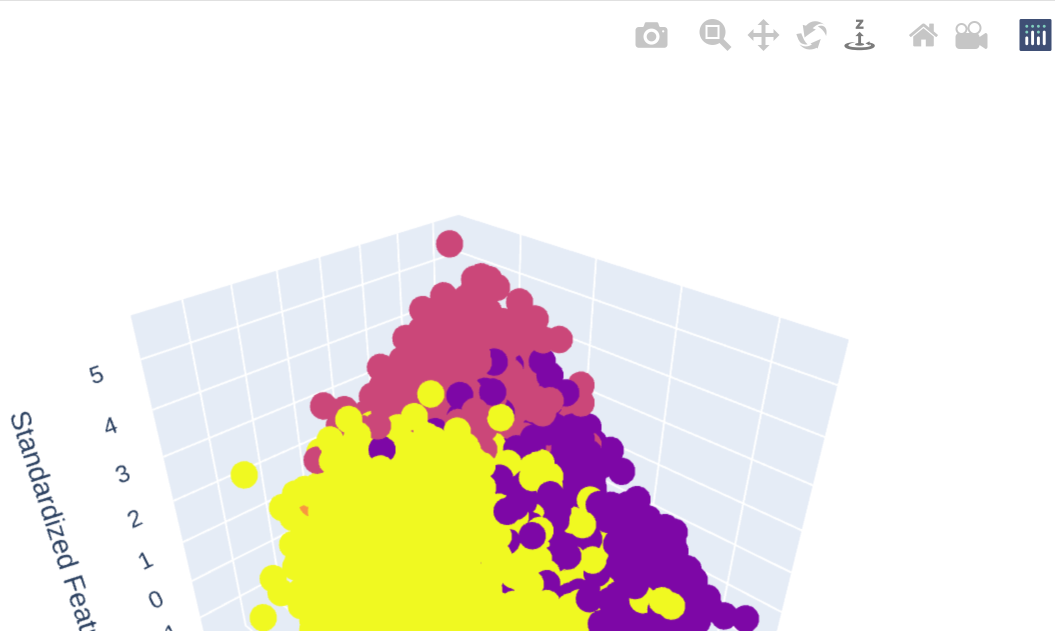



## Features 13 and 12 have the greatest correlation before standardization, and correlation between feature 8 and 11 had the second greatest correlation before standardization

## Testing is done through 3D scatter plot showing all three features catagorized by number of clusters in fitted and predicted model

In [12]:

```
model.fit(X_std)
ykmeans = model.predict(X_std)
cluster_num=[]
for i in range(len(ykmeans)):
    cluster_num.append(ykmeans[i]+1)
fig = px.scatter_3d(x=X_std[:,13], y=X_std[:,12], z=X_std[:,8],
                    color=cluster_num,
                    labels={
                     "x": "Standardized Feature 13",
                     "y": "Standardized Feature 12",
                     "z": "Standardized Feature 8",
                     "color": "Number of Clusters"
                 })
fig.show()
```




In [13]:

```
cluster_num=[]
for i in range(len(ykmeans)):
    cluster_num.append(ykmeans[i]+1)
fig = px.scatter_3d(x=X_std[:,13], y=X_std[:,12], z=X_std[:,11],
                    color=cluster_num,
                    labels={
                     "x": "Standardized Feature 13",
                     "y": "Standardized Feature 12",
                     "z": "Standardized Feature 11",
                     "color": "Number of Clusters"
                 })
fig.show()
```