```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
df=pd.read_csv('/content/CC GENERAL.csv')
df
```

	CUST_ID	BALANCE	BALANCE_FREQUENCY	PURCHASES	ONEOFF_PURCHASES	INSTALLMENTS
0	C10001	40.900749	0.818182	95.40	0.00	
1	C10002	3202.467416	0.909091	0.00	0.00	
2	C10003	2495.148862	1.000000	773.17	773.17	
3	C10004	1666.670542	0.636364	1499.00	1499.00	
4	C10005	817.714335	1.000000	16.00	16.00	
...	...	...	...	...	...	
8945	C19186	28.493517	1.000000	291.12	0.00	
8946	C19187	19.183215	1.000000	300.00	0.00	
8947	C19188	23.398673	0.833333	144.40	0.00	
8948	C19189	13.457564	0.833333	0.00	0.00	
8949	C19190	372.708075	0.666667	1093.25	1093.25	

8950 rows × 18 columns

Next steps: Generate code with df View recommended plots

```
df.head()
```

```
df.isna().sum()
```

```
df.dtypes
```

```
df['CREDIT_LIMIT']=df['CREDIT_LIMIT'].fillna(df['CREDIT_LIMIT'].mean())
df['MINIMUM_PAYMENTS']=df['MINIMUM_PAYMENTS'].fillna(df['MINIMUM_PAYMENTS'].mean())

df.isna().sum()
```

```
df.drop(['CUST_ID'],axis=1,inplace=True)
df
```

```
df.dtypes
```

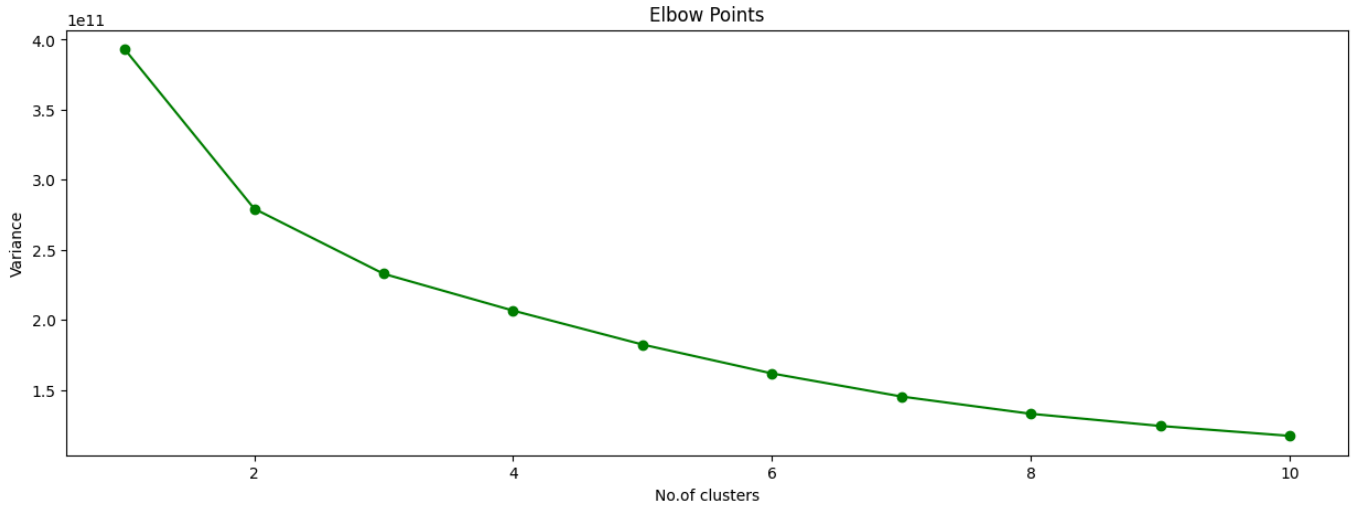
```
x=df
x
```

```
from sklearn.cluster import KMeans
lst=[]
for i in range(1,11):
  model=KMeans(n_clusters=i,init='k-means++',random_state=42)
  model.fit(x)
  lst.append(model.inertia_)
```

```
lst
```

```
plt.figure(figsize=(15,5))
plt.plot(range(1,11),lst,marker='o',color='g')
plt.xlabel('No.of clusters')
plt.ylabel('Variance')
plt.title('Elbow Points')
```

```
Text(0.5, 1.0, 'Elbow Points')
```




```
#Here contant value starts from 4..so elbow point=7
model1=KMeans(n_clusters=4,init='k-means++',random_state=42)
model1.fit(x)
y=model1.predict(x)
y
```

```
df['Clusters']=y
df
```

```
df.dtypes
```

```
df.isna().sum()
```

```
x=df.iloc[:,:-1].values
x
```

```
y=df.iloc[:,-1].values
y
```

```
array([1, 0, 0, ..., 1, 1, 1], dtype=int32)
```

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.30,random_state=42)
x_train
```

```
from sklearn.preprocessing import StandardScaler
norm=StandardScaler()
norm.fit(x_train)
x_train=norm.transform(x_train)
x_test=norm.transform(x_test)
```

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
model2=RandomForestClassifier(n_estimators=10,criterion='entropy',random_state=42)
model3=DecisionTreeClassifier(criterion='entropy')
lst1=[model2,model3]
```

```
for i in lst1:
  print("model is", i)
  i.fit(x_train,y_train)
  y_pred=i.predict(x_test)
  print("score is",accuracy_score(y_test,y_pred))
  print("\n")
```

```
model is RandomForestClassifier(criterion='entropy', n_estimators=10, random_state=42)
score is 0.9761638733705773


model is DecisionTreeClassifier(criterion='entropy')
score is 0.9772811918063314
```

```
from sklearn.metrics import silhouette_score
print(silhouette_score(x,y))
```

```
0.46558299560999516
```