CUSTOMER SEGMENTATION

FEATURE ENGINEERING

Feature engineering or feature extraction or feature discovery is the process of extracting features from raw data. Due to deep learning networks, such as convolutional neural networks, that are able to learn it by itself, domain-specific- based feature engineering has become obsolete for vision and manipulation — addition, deletion, combination, mutation — of your data set to improve machine learning model training, leading to better performance and greater accuracy.

```
data.describe()
```

	CustomerID	Age	AnnualIncome	SpendingScore
count	200.000000	200.000000	200.000000	200.000000
mean	100.500000	38.850000	60.560000	50.200000
std	57.879185	13.969007	26.264721	25.823522
min	1.000000	18.000000	15.000000	1.000000
25%	50.750000	28.750000	41.500000	34.750000
50%	100.500000	36.000000	61.500000	50.000000
75%	150.250000	49.000000	78.000000	73.000000
max	200.000000	70.000000	137.000000	99.000000

The data have been described over.

	CustomerID	Genre	Age	AnnualIncome	Spending Score
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40
...	...	...	...	...	...
195	196	Female	35	120	79
196	197	Female	45	126	28
197	198	Male	32	126	74
198	199	Male	32	137	18
199	200	Male	30	137	83

The data is trans formed using the Standards scaler and TSNE algorithm for reduce the dimension of the data set and find predictions.

```python
sc=StandardScaler()
x=data.iloc[:,2:4]
y=data.iloc[:,4:]
scaler=sc.fit_transform(x)
print(scaler)
import plotly.express as px
```

```
[[-1.42456879 -1.73899919]
 [-1.28103541 -1.73899919]
 [-1.3528021  -1.70082976]
 [-1.13750203 -1.70082976]
 [-0.56336851 -1.66266033]
 [-1.20926872 -1.66266033]
 [-0.27630176 -1.62449091]
 [-1.13750203 -1.62449091]
 [ 1.80493225 -1.58632148]
 [-0.6351352  -1.58632148]
 [ 2.02023231 -1.58632148]
 [-0.27630176 -1.58632148]
 [ 1.37433211 -1.54815205]
 [-1.06573534 -1.54815205]
 [-0.13276838 -1.54815205]
 [-1.20926872 -1.54815205]
 [-0.27630176 -1.50998262]
 [-1.3528021  -1.50998262]
 [ 0.94373197 -1.43364376]
```

```python
tsne=TSNE(learning_rate=200,n_components=2)
x_tsne=tsne.fit_transform(scaler)
y_tsne=y
print(x_tsne)
```

```
[[-10.197836     8.096215  ]
 [ -9.690498     8.600782  ]
 [ -9.981253     7.664772  ]
 [ -8.964528     8.807945  ]
 [ -6.844892     9.280128  ]
 [ -9.342688     8.01517   ]
 [ -5.9495425    9.676451  ]
 [ -8.853804     8.242036  ]
 [  8.907679     9.888261  ]
 [ -7.088064     8.811263  ]
 [  9.418162     9.788504  ]
 [ -5.5555515    9.635909  ]
 [  7.8716226    9.81476   ]
 [ -8.397869     8.092192  ]
 [ -4.826714     9.533949  ]
 [ -9.05111      7.475388  ]
 [ -5.7145967    9.097713  ]
 [ -9.663504     7.1045003 ]
 [  6.211253     9.682639  ]
```

KMEANS ALGORITHM

Let implement the data into Kmeans algorithm.

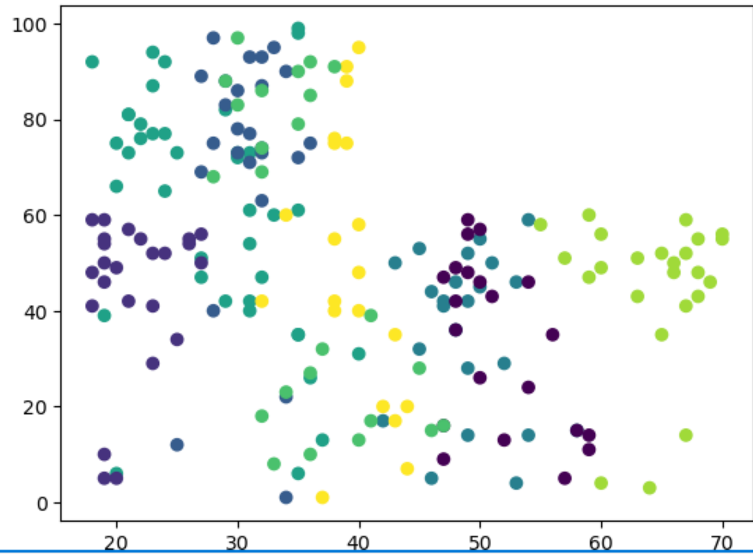
```
from sklearn.cluster import KMeans
kmeans=KMeans()
predict=kmeans.fit_predict(x_tsne)
data['kmeans']=kmeans.labels_
print(data)
```

	CustomerID	Genre	Age	AnnualIncome	SpendingScore	kmeans
0	1	Male	19	15	39	4
1	2	Male	21	15	81	4
2	3	Female	20	16	6	4
3	4	Female	23	16	77	4
4	5	Female	31	17	40	4
..	...	...	...	...	...	...
195	196	Female	35	120	79	5
196	197	Female	45	126	28	5
197	198	Male	32	126	74	5
198	199	Male	32	137	18	5
199	200	Male	30	137	83	5

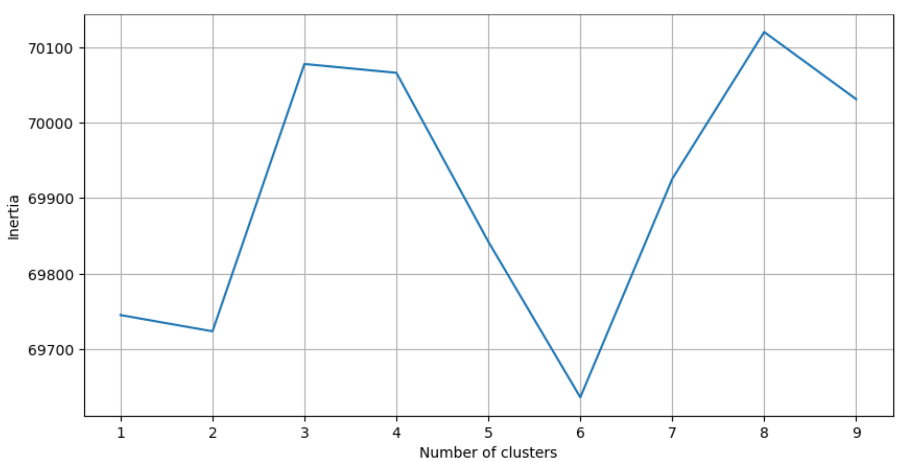
Let find out the prediction value and add it into data set for visualisation.

VISUALIZATION

```python
plt.scatter(x=data['Age'],y=data['SpendingScore'],c=data['kmeans'])
plt.show()
```




```python
value=data.drop(columns='Genre')
means=[]
inertias=[]
for k in range(1,10):
    kmeans=KMeans(n_clusters=10)
    kmeans.fit(value)
    means.append(k)
    inertias.append(kmeans.inertia_)
fig=plt.subplots(figsize=(10,5))
plt.plot(means,inertias)
plt.xlabel('Number of clusters')
plt.ylabel('Inertia')
plt.grid(True)
plt.show()
```



Interpreting the Meaning of the Clusters

The above visual part defines the data predicton of kmeans Algorithm.

The graph represents the number of clustures and the value of inertia with the data value of data set.