

In []: First, we will **import** the necessary Python libraries **and** the dataset

```
In [1]: import pandas as pd
import numpy as np
from sklearn import cluster
```

```
In [2]: data = pd.read_csv("CC_GENERAL.csv")
print(data.head())
```

	CUST_ID	BALANCE	BALANCE_FREQUENCY	PURCHASES	ONEOFF_PURCHASES	\
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	

	INSTALLMENTS_PURCHASES	CASH_ADVANCE	PURCHASES_FREQUENCY	\
0	95.4	0.000000	0.166667	
1	0.0	6442.945483	0.000000	
2	0.0	0.000000	1.000000	
3	0.0	205.788017	0.083333	
4	0.0	0.000000	0.083333	

	ONEOFF_PURCHASES_FREQUENCY	PURCHASES_INSTALLMENTS_FREQUENCY	\
0	0.000000	0.083333	
1	0.000000	0.000000	
2	1.000000	0.000000	
3	0.083333	0.000000	
4	0.083333	0.000000	

	CASH_ADVANCE_FREQUENCY	CASH_ADVANCE_TRX	PURCHASES_TRX	CREDIT_LIMIT	\
0	0.000000	0	2	1000.0	
1	0.250000	4	0	7000.0	
2	0.000000	0	12	7500.0	
3	0.083333	1	1	7500.0	
4	0.000000	0	1	1200.0	

	PAYMENTS	MINIMUM_PAYMENTS	PRC_FULL_PAYMENT	TENURE
0	201.802084	139.509787	0.000000	12
1	4103.032597	1072.340217	0.222222	12
2	622.066742	627.284787	0.000000	12
3	0.000000	NaN	0.000000	12
4	678.334763	244.791237	0.000000	12

In []: Now, let's check whether this dataset contains any null values or not.

```
In [3]: data.isnull().sum()
```

```
Out[3]: CUST_ID          0
BALANCE          0
BALANCE_FREQUENCY  0
PURCHASES        0
ONEOFF_PURCHASES  0
INSTALLMENTS_PURCHASES  0
CASH_ADVANCE      0
PURCHASES_FREQUENCY  0
ONEOFF_PURCHASES_FREQUENCY  0
PURCHASES_INSTALLMENTS_FREQUENCY  0
CASH_ADVANCE_FREQUENCY  0
CASH_ADVANCE_TRX      0
PURCHASES_TRX        0
CREDIT_LIMIT        1
PAYMENTS            0
MINIMUM_PAYMENTS    313
PRC_FULL_PAYMENT     0
TENURE              0
dtype: int64
```

In []: The dataset has some null values **in** the minimum payments column. I will drop the rows **with** null values **and** move further

```
In [4]: data = data.dropna()
```

```
In [5]: clustering_data = data[["BALANCE", "PURCHASES", "CREDIT_LIMIT"]]
from sklearn.preprocessing import MinMaxScaler
for i in clustering_data.columns:
    MinMaxScaler(i)
```

```
In [6]: from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=5)
clusters = kmeans.fit_predict(clustering_data)
data["CREDIT_CARD_SEGMENTS"] = clusters
```

/Users/tavi/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.py:1412: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
super()._check_params_vs_input(X, default_n_init=10)

In []: I have added a new column **as** **"CREDIT_CARD_SEGMENTS"**. It contains labels about the group of credit card customers. The groups formed range **from** 0 to 4

```
In [7]: data["CREDIT_CARD_SEGMENTS"] = data["CREDIT_CARD_SEGMENTS"].map({0: "Cluster 1", 1:
"Cluster 2", 2: "Cluster 3", 3: "Cluster 4", 4: "Cluster 5"})
print(data["CREDIT_CARD_SEGMENTS"].head(10))
```

0	Cluster 3
1	Cluster 1
2	Cluster 1
4	Cluster 3
5	Cluster 3
6	Cluster 2
7	Cluster 3
8	Cluster 1
9	Cluster 2
10	Cluster 3

Name: CREDIT_CARD_SEGMENTS, dtype: object

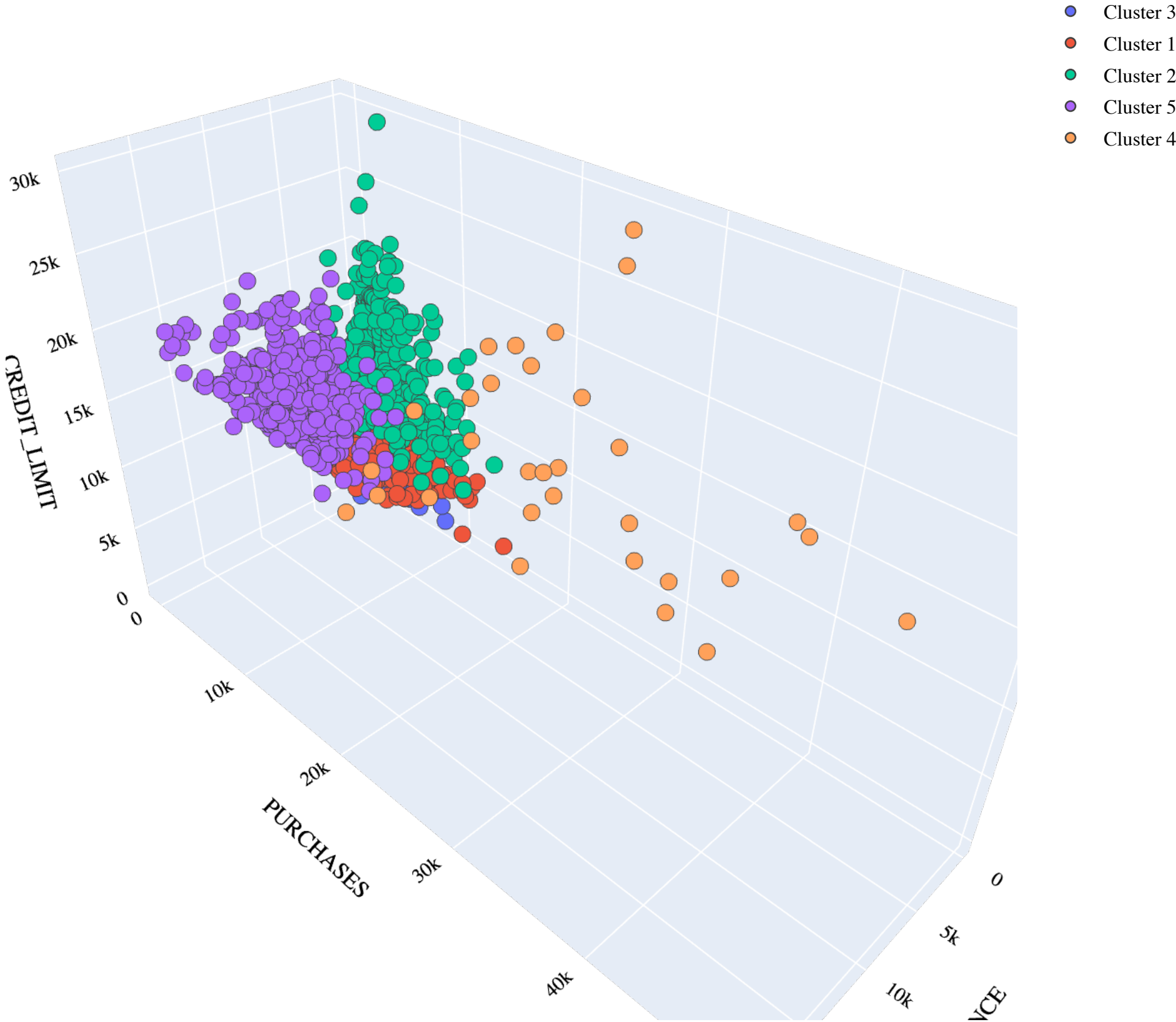
In []: Next we will visualize the credit card clusters we found **from** our cluster analysis

```
In [8]: import plotly.graph_objects as go
PLOT = go.Figure()
for i in list(data["CREDIT_CARD_SEGMENTS"].unique()):

    PLOT.add_trace(go.Scatter3d(x = data[data["CREDIT_CARD_SEGMENTS"]== i]['BALANCE'],
                                y = data[data["CREDIT_CARD_SEGMENTS"] == i]['PURCHASES'],
                                z = data[data["CREDIT_CARD_SEGMENTS"] == i]['CREDIT_LIMIT'],
                                mode = 'markers',marker_size = 6, marker_line_width = 1,
                                name = str(i)))
PLOT.update_traces(hovertemplate='BALANCE: %{x} <br>PURCHASES %{y} <br>DCREDIT_LIMIT: %{z}')
```



```
PLOT.update_layout(width = 800, height = 800, autosize = True, showlegend = True,
                    scene = dict(xaxis=dict(title = 'BALANCE', titlefont_color = 'black'),
                                yaxi=dict(title = 'PURCHASES', titlefont_color = 'black'),
                                zaxis=dict(title = 'CREDIT_LIMIT', titlefont_color = 'black')),
                    font = dict(family = 'Gilroy', color = 'black', size = 12))
```



In []: This **is** how you can perform credit card segmentation **with** Machine Learning using Python.