

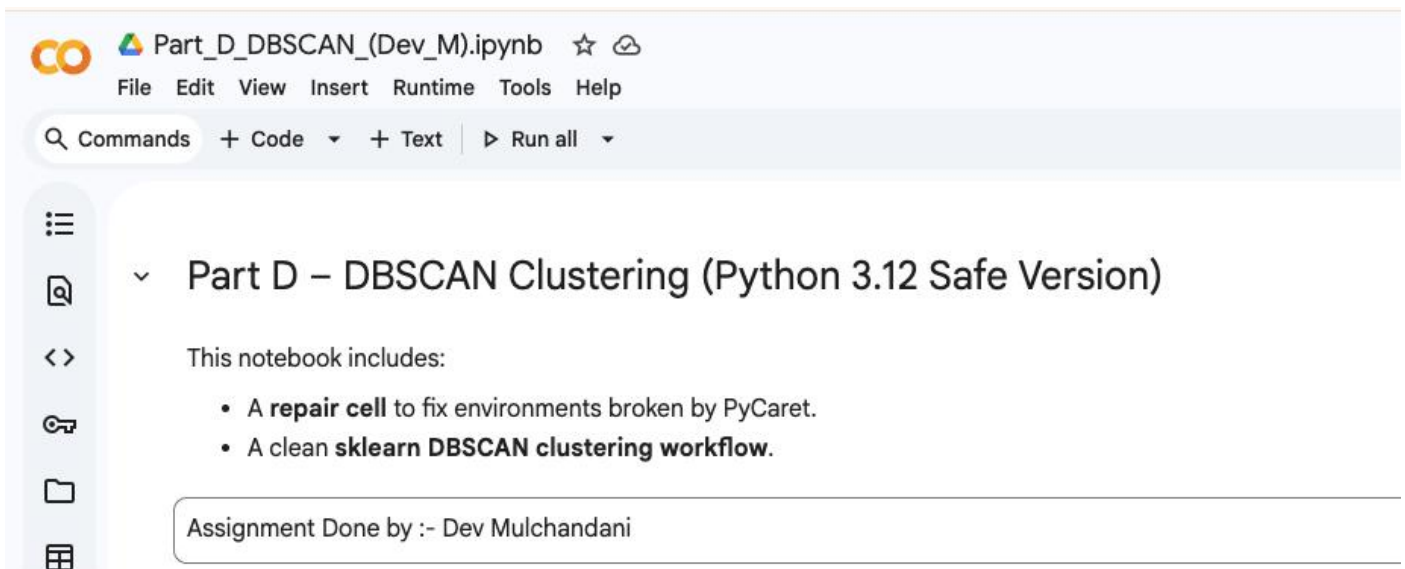
Clustering Assignments

❖ Assignment done by :- Dev Mulchandani

❖ Part-D :- DB Scan clustering

In Part D, I performed density-based clustering using the DBSCAN algorithm implemented in sklearn, which is fully compatible with Python 3.12. After uploading and preprocessing the dataset by selecting numerical features and standardizing them with StandardScaler, I applied DBSCAN to detect natural density groups and isolate noise points. Unlike algorithms that require a preset number of clusters, DBSCAN automatically identified clusters based on regions of high point density while marking sparse points as outliers. I evaluated the clustering structure using silhouette score when applicable and visualized the results through PCA, allowing me to observe how the density-based clusters formed in two-dimensional space. This alternative approach provided a robust and flexible way to cluster data without relying on PyCaret.

❖ Screenshots:-



[1]
✓ 8s

```
# 📁 Upload dataset AFTER restarting runtime
import pandas as pd
from google.colab import files

uploaded = files.upload()
fname = list(uploaded.keys())[0]
df = pd.read_csv(fname)
df.head()
```

▼

... credit_default.csv

credit_default.csv(text/csv) - 3421 bytes, last modified: 02/12/2025 - 100% done
Saving credit_default.csv to credit_default.csv

	limit_bal	sex	education	marriage	age	default_payment_next_month
0	28638	1	1	1	47	0
1	33166	1	3	3	68	0
2	29727	1	3	1	67	1
3	37059	1	1	2	60	0
4	31620	2	1	1	21	0

Next steps: [Generate code with df](#) [New interactive sheet](#)

[2]
✓ 7s

```
# 🌟 DBSCAN clustering using sklearn (PyCaret alternative)
from sklearn.cluster import DBSCAN
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import silhouette_score
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt

X = df.select_dtypes(include=['float64', 'int64']).values

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

db = DBSCAN(eps=0.5, min_samples=5).fit(X_scaled)
labels_ = db.labels_

print("Cluster labels:", set(labels_))
print("Clusters:", len(set(labels_)) - (1 if -1 in labels_ else 0))
print("Noise points:", list(labels_).count(-1))

if len(set(labels_)) > 1 and -1 not in set(labels_):
    print("Silhouette:", silhouette_score(X_scaled, labels_))

pca = PCA(n_components=2)
X_2d = pca.fit_transform(X_scaled)

plt.figure(figsize=(7,6))
plt.scatter(X_2d[:,0], X_2d[:,1], c=labels_, cmap='tab10')
plt.title("DBSCAN Clusters (sklearn)")
plt.xlabel("PC1")
plt.ylabel("PC2")
plt.show()
```

Cluster labels: {-1}
Clusters: 0
Noise points: 200

