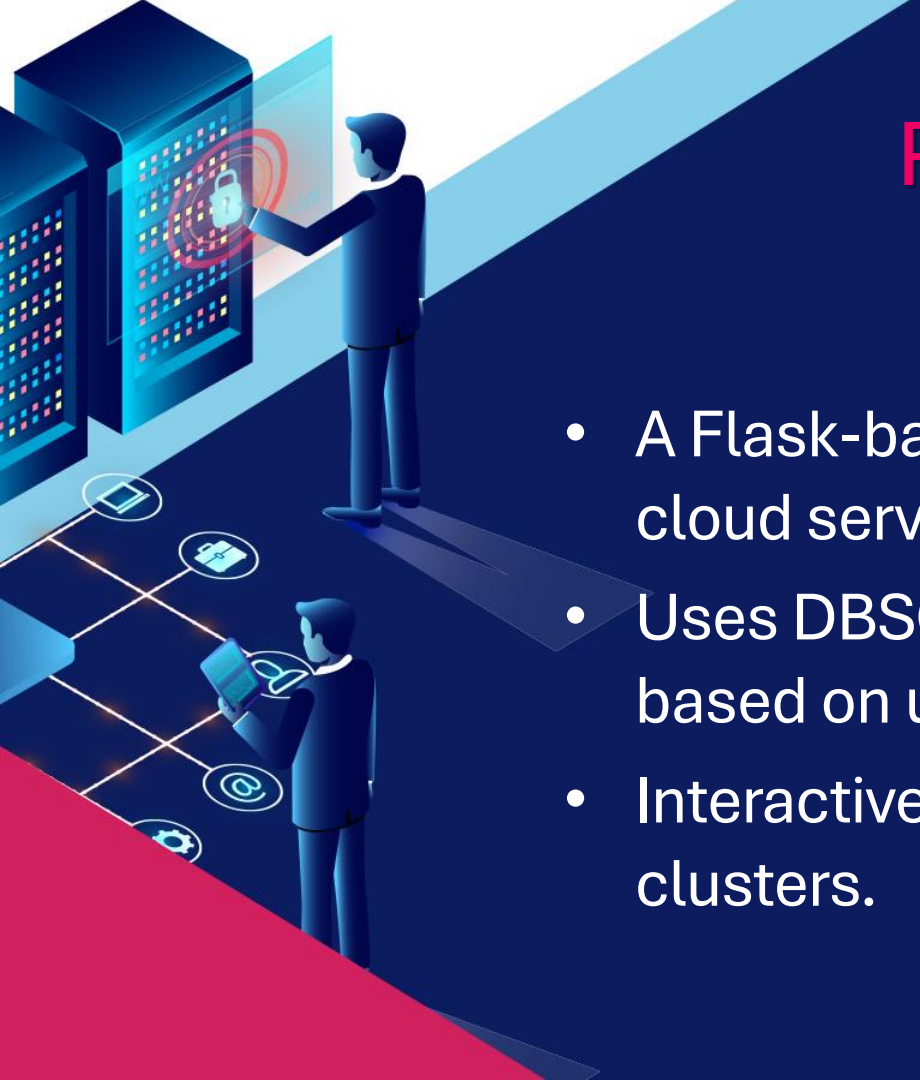
MULTI-CLOUD SERVICE RECOMMENDATION SYSTEM

Using DBSCAN Clustering Algorithm

Group Members :
249341G - Wanasinghe.W.P.N.M
249311P - Kaumadi .I. A. S
249296V - Devindi. P.H

PROBLEM STATEMENT

- Users face difficulty choosing the best cloud service due to many options.
- No intelligent support for selecting services based on specific requirements.
- Manual comparison is time-consuming and error-prone.



PROPOSED SOLUTION

- A Flask-based web app that recommends cloud services using ML.
- Uses DBSCAN to group similar services based on user needs.
- Interactive filtering and visualization of clusters.

OBJECTIVES

- Help users choose optimal cloud services based on needs.
- Use unsupervised ML to cluster services.
- Display results with interactive charts and filters.

DATASET USED

File: multi_cloud_service_composition.csv

- Contains data from AWS, Azure, GCP, IBM, etc.
- Attributes:

Provider, Type, CPU, Memory, Bandwidth, etc.

- Preprocessed and normalized for clustering.

TECHNOLOGIES USED

- Frontend: HTML, CSS, Bootstrap
- Backend: Flask (Python)
- ML Libraries: Scikit-learn (DBSCAN), Pandas, NumPy
- Visualization: Plotly

ML ALGORITHM: DBSCAN

- DBSCAN = Density-Based Spatial Clustering of Applications with Noise
- No need to predefine number of clusters
- Detects noise/outliers
- Parameters: eps (radius), min_samples (density threshold)

WORKFLOW

User selects provider, type, and resource needs (CPU, memory, bandwidth).

Filters applied to dataset.

DBSCAN clusters filtered data into service groups.

Results visualized with cluster plots and recommendation table.

USER INTERFACE

- **Input form for selecting filters**

Cloud Service Recommender

Find the perfect cloud service for your needs with AI-powered recommendations

Service Configuration

Cloud Provider

AWS

Service Type

Compute

Performance Requirements

Minimum CPU Utilization (%)

4.0

Minimum Memory Usage (MB)

16.0

Minimum Network Bandwidth (Mbps)

500.0

Advanced Options

- [x] Run DBSCAN Clustering
- [x] Apply LDA Classification
- [ ] Optimize by QoS Score

DBSCAN eps

0.5

DBSCAN min_samples

3

- **Recommendation table with top services**

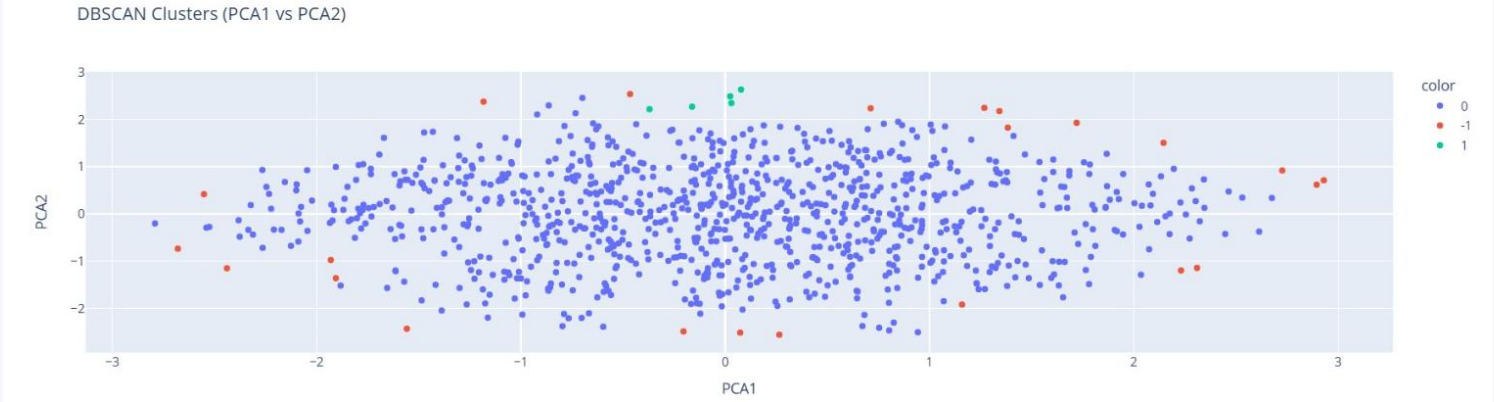
GET RECOMMENDATIONS

Top Recommended Services

SERVICE ID	TYPE	PROVIDER	CPU (%)	MEMORY (MB)	BANDWIDTH (MBPS)	LATENCY (MS)	QOS SCORE	CLUSTER	LDA
S0046	Compute	AWS	52.4057662208147	4131	505.472542708051	244.6362231728832	**0.5102558015116947**	-1	-1.90
S0082	Compute	AWS	58.127041018721	3164	569.765767931665	132.10319123776986	**0.6832284603765957**	-1	-0.49
S0105	Compute	AWS	61.56698855990125	3520	859.3755400059066	180.9480166145697	**0.8191708752905449**	-1	-0.85
S0156	Compute	AWS	36.545896453340845	3187	804.8226969055905	40.1364280836313	**0.8297228293578192**	-1	1.03
S0178	Compute	AWS	58.85590746791116	7959	509.5728343616186	213.69075117277004	**0.7602852721495867**	-1	-1.06
S0248	Compute	AWS	28.60766539019131	5103	637.0776228397058	187.07977698336515	**0.6754610474828397**	-1	-0.81
S0257	Compute	AWS	32.433090512245684	4820	841.4221486814535	157.5767811695912	**0.5209108118932326**	-1	-0.29
S0266	Compute	AWS	58.43769865777488	7593	994.4711148832848	38.42338329748897	**0.9401239972377818**	-1	1.66

- **Interactive cluster visualization using plotly**

Cluster Visualization (PCA1 vs PCA2)



All services have been loaded successfully!

Demonstration

CONCLUSION & FUTURE WORK

Conclusion

- Smart recommendation using clustering achieved.
- Improves cloud service decision-making.

Possible Future Implementations

- Add more algorithms (e.g., K-Means, LDA).
- Integrate live data from cloud providers.

THANK YOU !