Deep Clustering with Autoencoders: K-Means, Agglomerative Clustering and Evaluation Metrics

1. Introduction

Clustering is one of the most widely used unsupervised learning methods, aiming to group data into meaningful structures without the use of labels. In this project, we evaluate two popular clustering algorithms—**K-Means** and **Agglomerative Hierarchical Clustering—**on a tabular dataset consisting of 20 features and a categorical target variable (price range).

The study is performed in two phases:

1. **Clustering on the original input space**
2. **Clustering on a lower-dimensional latent space**, learned using a fully connected **Autoencoder**

We analyze how dimensionality reduction affects clustering quality and compare results using two external evaluation metrics: **Purity** and **F-measure**.

The project demonstrates practical experience in:

- Unsupervised learning
- Deep learning for feature extraction
- Dimensionality reduction
- Design and execution of ML experiments
- Quantitative evaluation of clustering results

---

2. Dataset

The dataset contains:

- **20 numerical input features**
- **1 categorical label** (price_range), used only for evaluation — *not* for training

Since the provided test.csv does not include labels, all evaluations are performed only on the training dataset.

---

3. Methods

3.1 Clustering Algorithms

K-Means

K-Means partitions the dataset into *K* clusters by minimizing the within-cluster sum of squared distances. It alternates between:

1. **Assignment step** — assigning points to the nearest centroid

2. **Update step —** recomputing centroids as the mean of all assigned points

We use sklearn.KMeans with 10 different random initializations (n_init=10).

Agglomerative Hierarchical Clustering

This method builds a bottom-up hierarchy of clusters. Each data point starts as a separate cluster, and the closest clusters are merged iteratively until only *K* clusters remain.

We use sklearn.AgglomerativeClustering with default linkage (Ward's method).

3.2 Autoencoder for Dimensionality Reduction

To explore whether a compact representation improves clustering, we train a symmetric Autoencoder of architecture:

20 → 100 → M → 100 → 20

where **M ∈ {2, 10, 50}** represents the latent dimensionality.

Training settings:

- Loss: Mean Squared Error (MSE)
- Optimizer: Adam
- Epochs: 50
- Batch size: 256

After training, only the **encoder** part is used to generate latent features for clustering.

---

4. Evaluation Metrics

4.1 Purity

Purity measures the extent to which each cluster contains data points from a single class. For each cluster *i*:

- Count the class that appears most frequently inside the cluster
- Sum these counts across all clusters
- Divide by the total number of data points

Purity ∈ [0, 1], with 1 representing perfect clustering.

4.2 F-Measure

The F-measure combines **precision** and **recall** for each cluster relative to the true labels.

For a cluster *i*:

- **TP —** points in cluster *i* belonging to the cluster's majority class
- **FP —** points in cluster *i* belonging to another class
- **FN —** points of the majority class assigned to other clusters

Then:

precision = TP / (TP + FP)

recall = TP / (TP + FN)

F1 = 2 * (precision * recall) / (precision + recall)

The final F-measure is the average across all clusters.

---

5. Experimental Setup

We evaluate:

- **Five values of K**: {2, 4, 6, 8, 10}
- **Three latent dimensions**: M = {2, 10, 50}

All experiments are repeated **10 times**, and mean results are reported to reduce randomness.

---

6. Results

6.1 Clustering on Original Features

Method	K	Purity	F-measure
K-Means	2	0.50	0.67
K-Means	4	0.66	0.64
K-Means	6	0.66	0.54
K-Means	8	0.71	0.47
K-Means	10	0.74	0.42
Agglomerative	2	0.48	0.65
Agglomerative	4	0.62	0.60
Agglomerative	6	0.62	0.49
Agglomerative	8	0.65	0.43

Method	K	Purity	F-measure
Agglomerative	10	0.65	0.36

6.2 Clustering on Autoencoder Latent Space

M = 2

K	Purity	F-measure
2	0.50	0.58
4	0.745	0.55
6	0.717	0.42
8	0.742	0.36
10	0.745	0.30

The 2-D latent space is also visualized using scatter plots.

M = 10

K	Purity	F-measure
2	0.50	0.67
4	0.662	0.65
6	0.672	0.54
8	0.737	0.51
10	0.709	0.46

M = 50

K	Purity	F-measure
2	0.50	0.67
4	0.720	0.68
6	0.692	0.55
8	0.857	0.50
10	0.821	0.47

Agglomerative clustering at **M = 50, K = 4** achieves exceptionally high performance.

---

7. Discussion

The experiments demonstrate several important observations:

1. Higher K tends to increase purity but decrease F-measure.

This is expected: more clusters yield purer groups but often fragment class structure.

2. Autoencoder representations improve clustering quality.

For **M = 50**, both algorithms achieve higher Purity and F-measure compared to the original space.
The Autoencoder removes noise and captures more semantically meaningful features.

3. Agglomerative performs exceptionally well in latent space.

Particularly at **M = 50, K = 4**, suggesting that hierarchical clustering benefits from the compact, structured embedding learned by the neural network.

4. The 2-dimensional latent space is visually interpretable but less effective.

Although useful for visualization, M = 2 loses too much information compared to M = 10 or M = 50.

---

8. Conclusion

This study shows that:

- Non-linear dimensionality reduction through an Autoencoder **significantly enhances clustering performance**.
- K-Means performs well in the original feature space with low K but is surpassed by Agglomerative in the latent space.
- Higher latent dimensions (M = 50) deliver the strongest results, preserving more structure while filtering noise.
- The combination of deep learning and traditional clustering methods forms a powerful unsupervised learning pipeline.

The project highlights practical skills in machine learning, deep learning, clustering, evaluation metrics, and experimental methodology—making it a strong addition to an AI engineering portfolio.

---

9. Future Work

Potential extensions include:

- Testing different Autoencoder architectures (sparse, denoising, variational).

- Using alternative clustering algorithms (DBSCAN, Spectral Clustering).
- Evaluating intrinsic clustering metrics (Silhouette, Davies–Bouldin).
- Hyperparameter tuning of the Autoencoder to optimize latent embeddings.