#### **Context**

You are being hired by a fictional biotech company specializing in genetic research. The task involves analyzing **embeddings** derived from images to classify genetic syndromes. These embeddings are outputs from a pre-trained classification model. The company wants to improve its understanding of the data distribution and enhance the classification accuracy of genetic syndromes based on these embeddings.

Your objective is to implement a comprehensive pipeline that includes data preprocessing, visualization, classification, manual implementation of key metrics, and insightful analysis.

### **Dataset Description**

You are provided with a **pickle file** (mini\_gm\_public\_v0.1.p) that contains all the necessary data. The data structure is as follows:

* **Embeddings**: 320-dimensional vectors representing images.

**Hierarchy**:  
 {

'syndrome\_id': {

'subject\_id': {

'image\_id': [320-dimensional embedding]

}

}

}

* **Goal**: Use these embeddings to classify the **syndrome\_id** associated with each image.

**Note**: If you encounter the "numpy.core.\_multiarray\_umath" error when loading the pickle file, please upgrade your numpy package.

### 

### **Task Description**

### Development

1. **Data Processing**:
   * **Load and preprocess** the data from the pickle file.
     + Flatten the hierarchical data structure into a suitable format for analysis.
     + Ensure data integrity and handle any missing or inconsistent data.
   * **Exploratory Data Analysis**:
     + Provide statistics about the dataset (e.g., number of syndromes, images per syndrome).
     + Discuss any data imbalances or patterns observed.
2. **Data Visualization**:
   * Use **t-SNE** to reduce the dimensionality of the embeddings to 2D.
     + Generate a plot that visualizes the embeddings colored by their **syndrome\_id**.
     + Identify and interpret clusters or patterns in the visualization.
     + Discuss how these patterns might relate to the classification task.
3. **Classification Task**:
   * Implement the **K-Nearest Neighbors (KNN)** algorithm to classify the embeddings into their respective **syndrome\_id**.
     + Use both **Cosine** and **Euclidean** distance metrics.
     + Perform **10-fold cross-validation** to evaluate the model performance.
     + **Determine the optimal value of kk** (from 1 to 15) for KNN using cross-validation.
     + **Implement**:
       - Calculation of **AUC (Area Under the ROC Curve)**.
       - **F1-Score**.
       - **Top-k Accuracy**.
     + Compare the classification results between the two distance metrics.
     + Discuss any differences in performance and possible reasons.
4. **Metrics and Evaluation**:
   * **Generate ROC AUC curves** for both Cosine and Euclidean distance metrics.
     + Average the results across the cross-validation folds.
     + Plot both curves on the same graph for comparison.
   * **Create tables** summarizing the performance metrics for both algorithms.
     + Include metrics such as Top-k Accuracy, AUC, F1-Score, etc.
     + Ensure that the tables are clear and can be automatically generated (e.g., from code).