Genetic Syndrome Classification Report - Apollo Solutions

Machine Learning Developer Test

João Pedro Pereira Santiago

Date: March 20, 2025

1. Introduction

This report summarizes the analysis of genetic syndromes using embeddings derived from pre-processed image data. The primary goal is to classify the `syndrome_id` associated with each image based on its corresponding 320-dimensional embedding vector. The project encompasses data preprocessing, exploratory data analysis (EDA), dimensionality reduction and visualization using t-SNE, classification using the K-Nearest Neighbors (KNN) algorithm, and a thorough evaluation of the results using various metrics.

2. Methodology

The following steps were implemented in this analysis:

1. **Data Loading and Preprocessing:**
    - The dataset was loaded from the provided pickle file (`mini_gm_public_v0.1.p`).
    - The hierarchical data structure was flattened into a tabular format (Pandas DataFrame) for easier manipulation and analysis. This created columns for `syndrome_id`, `subject_id`, `image_id`, and the `embedding` (a list of 320 floats).
    - The string-based `syndrome_id` values were mapped to numerical integer labels (0 to 9) to be compatible with scikit-learn's algorithms. An inverse mapping was also created to convert numerical predictions back to original syndrome names.
    - A check for `NaN` (Not a Number) values within the embeddings was performed. The presence of `NaN`s would indicate data quality issues and would need to be addressed (e.g., through imputation or removal of affected samples).
    - To avoid redundant preprocessing, the processed data (embeddings, numerical labels, and the syndrome mapping) were saved to a separate pickle file (`processed_data.pkl`). Subsequent runs load this processed data if it exists.

    - **Crucially, even when loading preprocessed data, the *raw* data is still loaded. This is necessary for the dataset statistics calculations.**
    - All embeddings were standardized using `sklearn.preprocessing.StandardScaler`. This ensures that each feature (dimension of the embedding) has a mean of 0 and a standard deviation of 1. Standardization is essential for distance-based algorithms like KNN, preventing features with larger magnitudes from dominating the distance calculations. The scaler is fit *only* on the training data and then used to transform both training and test sets (during cross-validation) to prevent data leakage.
2. **Exploratory Data Analysis (EDA):**
    - Dataset statistics were calculated, including:
        - Total number of syndromes.
        - Total number of subjects.
        - Total number of images.
        - Average number of images per syndrome.
        - Average number of images per subject.
        - Average number of subjects per syndrome.
    - Data imbalance was analyzed by:
        - Calculating the ratio between the number of images in the most frequent and least frequent syndromes.
        - Identifying syndromes with significantly lower representation (below half the average images per syndrome).
3. **Data Visualization (t-SNE):**
    - t-distributed Stochastic Neighbor Embedding (t-SNE) was employed to reduce the dimensionality of the 320-dimensional embeddings to 2 dimensions for visualization.
    - t-SNE plots were generated and saved, with data points colored according to their `syndrome_id`. This allows for visual inspection of cluster formation, providing insights into the separability of the syndromes.
    - t-SNE was run with multiple perplexity values (5, 30, 50, and 100) to assess the impact of this parameter on the visualization.
    - A basic cluster analysis was performed on the 2D t-SNE output, calculating:

* The number of points in each cluster (syndrome).
* The centroid (center) of each cluster.
* The average distance of points within a cluster to their centroid.
* An overlap score between each pair of clusters. This score is based on the distance between cluster centers and the average distances within the clusters, providing a measure of how well-separated the syndromes are.

4. **Classification (KNN):**

- The K-Nearest Neighbors (KNN) algorithm was implemented as a custom Python class (`KNNClassifier`). This class supports both Euclidean and Cosine distance metrics.
- The classifier includes methods for:
  - `fit(X, y)`: Stores the training data (embeddings and labels).
  - `_calculate_distances(X)`: Computes distances between test points and training points using the specified metric (efficiently implemented using NumPy).
  - `predict(X)`: Predicts the class label for each test point by majority vote among its k-nearest neighbors.
  - `predict_proba(X)`: Estimates class probabilities for each test point, based on the proportion of neighbors belonging to each class. This is crucial for ROC AUC calculation.
- 10-fold cross-validation was used to evaluate the performance of the KNN classifier rigorously. This provides a more reliable estimate of the model's generalization ability than a single train/test split.
- The optimal value of `k` (number of neighbors) was determined through cross-validation, testing values from 1 to 15. The `k` value yielding the highest average accuracy across the folds was selected.
- Both Euclidean and Cosine distances were evaluated to compare their effectiveness for this specific dataset.

5. **Metrics and Evaluation:**

   - The following metrics were calculated for each fold and averaged across folds:
     - **Accuracy:** The proportion of correctly classified samples.
     - **Macro-averaged F1-score:** A balanced measure of precision and recall, particularly useful for imbalanced datasets.
     - **Top-k Accuracy (k=3 and k=5):** The proportion of samples where the true label is among the top-k predicted labels.
     - **Area Under the Receiver Operating Characteristic Curve (ROC AUC):** A measure of the classifier's ability to distinguish between classes, considering various thresholds. Macro-averaged ROC AUC was calculated to handle the multi-class nature of the problem.
     - Custom functions (in `metrics.py`) were implemented for calculating these metrics, including a robust multi-class ROC AUC implementation that handles edge cases (e.g., missing classes in a fold) correctly.
   - ROC curves were generated and plotted, showing the trade-off between true positive rate and false positive rate for both distance metrics. Results were averaged across the cross-validation folds.
   - A confusion matrix was generated and plotted after training the best-performing KNN model (optimal `k` and metric) on the *entire* dataset. This visualizes the classification performance across all syndromes, highlighting potential misclassifications.

- ○ A summary table was produced using pandas to display the different metrics for various k values, in both metrics.

3. Results

3.1 Dataset Statistics

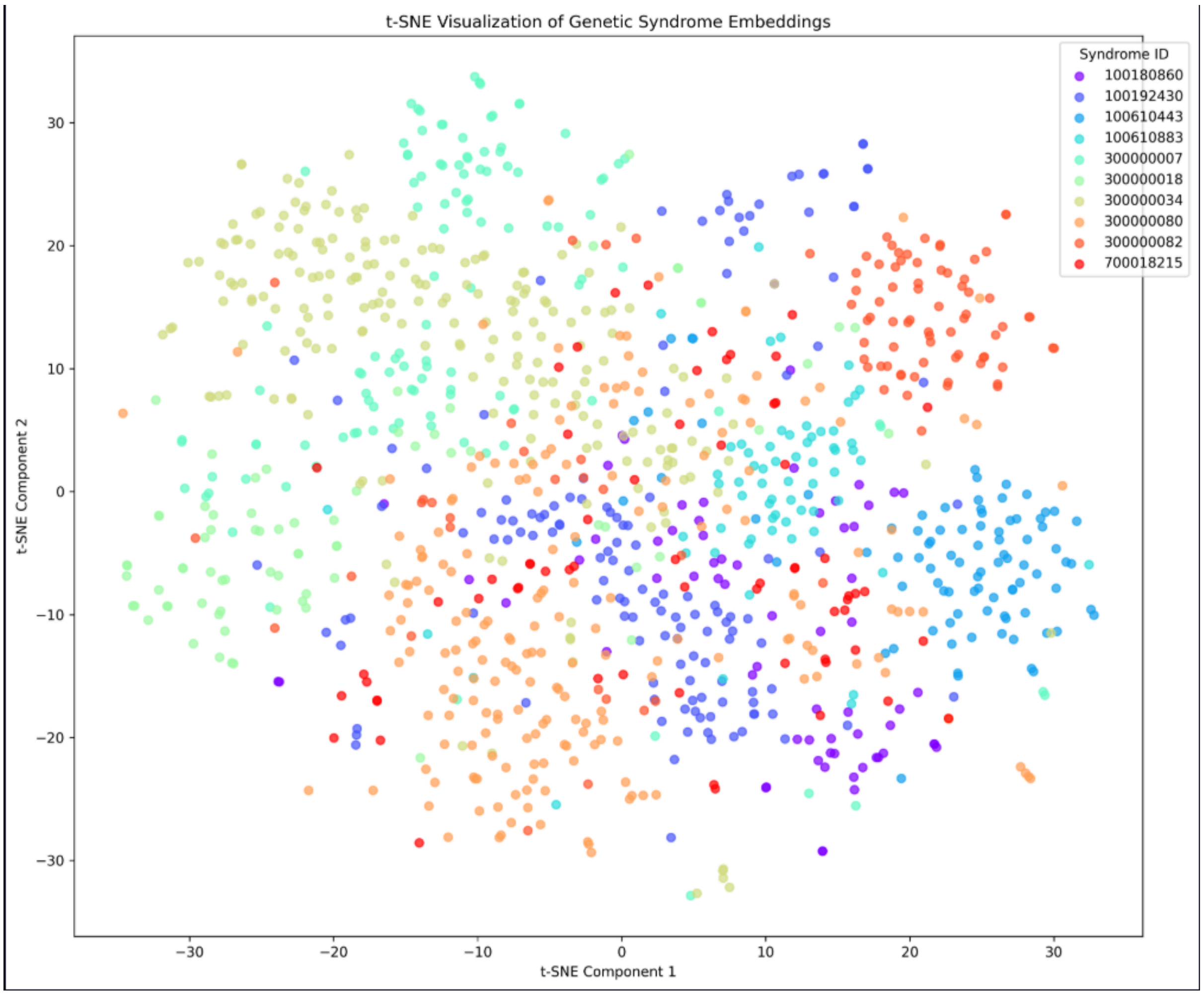
Statistic	Value
Number of Syndromes	10
Total Subjects	941
Total Images	1116
Average Images per Syndrome	111.60
Imbalance Ratio	3.28
Low Representation Syndromes	None

The dataset contains 1116 images representing 941 subjects and 10 different genetic syndromes. The average number of images per syndrome is approximately 112. The imbalance ratio of 3.28 (calculated as the maximum number of images for a syndrome divided by the minimum) indicates a moderate level of class imbalance. While no syndromes fell below the threshold defined as "low representation" (half the average images per syndrome), the imbalance should still be considered during model evaluation and potential future improvements.

3.2 t-SNE Visualization

(See attached image: `tsne_plot.png`)

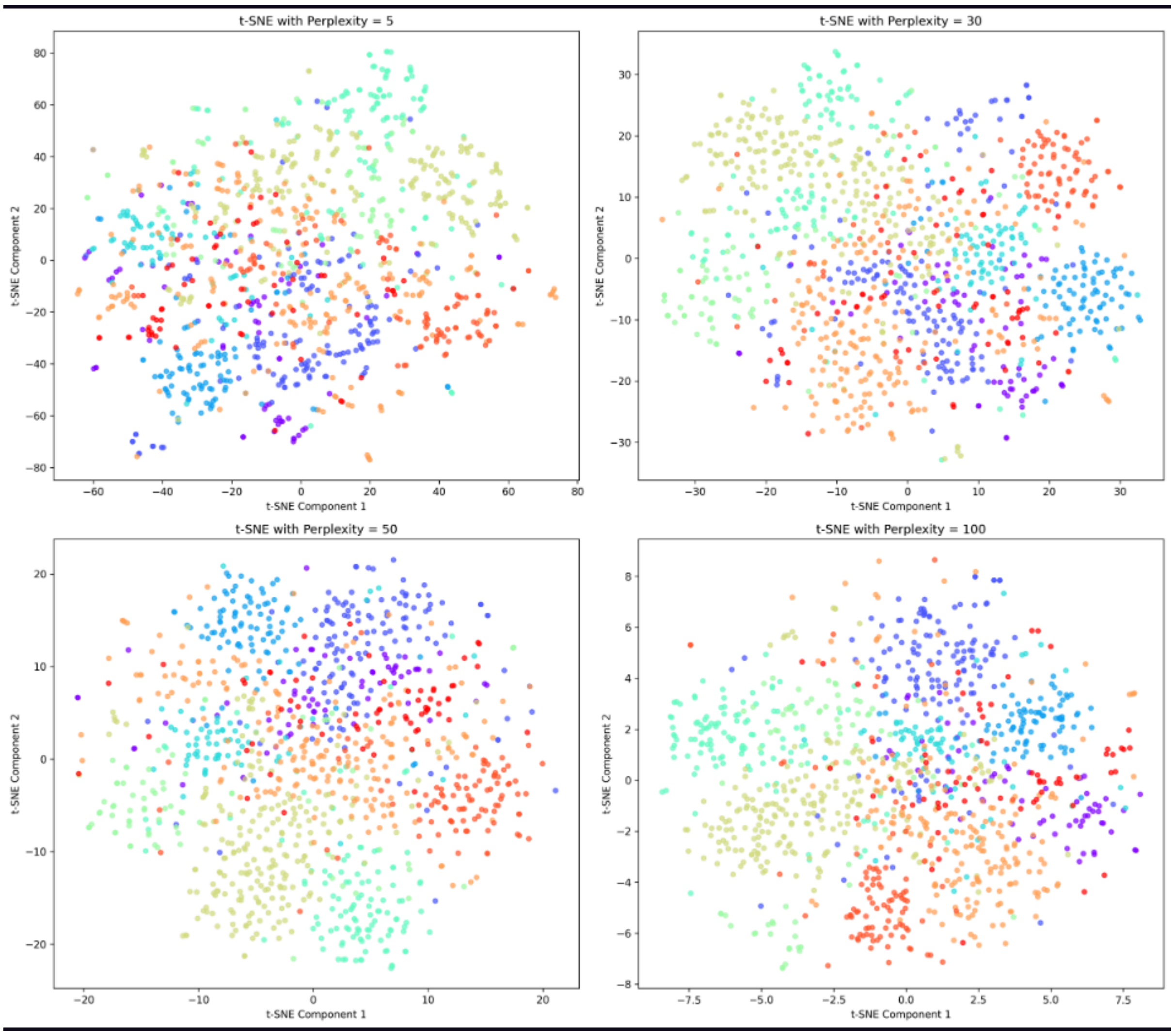
The t-SNE plot with perplexity 30 provides a 2D visualization of the 320-dimensional embeddings. Visual inspection reveals some degree of clustering, suggesting that the embeddings contain information that is relevant for distinguishing between different syndromes. However, the clusters are not perfectly separated, indicating that the classification task is not trivial. Some syndromes appear more tightly clustered than others, which might correlate with classification performance.



3.3 Multiple t-SNE Visualizations

(See attached image: `multiple_tsne_plot.png`)

This plot shows t-SNE visualizations with different perplexity values (5, 30, 50, and 100). Varying perplexity can reveal different aspects of the data structure. Lower perplexity values tend to emphasize local structure, while higher values emphasize global structure. Comparing these plots can provide additional insights into the relationships between syndromes.



3.4 Cluster Analysis (Based on t-SNE with perplexity=30)

The following table summarizes the cluster analysis performed on the 2D t-SNE embeddings:

Syndrome	# Points	Avg. Distance to Center	Overlap Scores (Selected Examples)
100180860	67	12.60	100192430: 0.39, 300000034: 1.16, 700018215: 0.31
100192430	136	15.08	100180860: 0.39, 300000080: 0.25, 700018215: 0.10
100610443	89	7.84	100180860: 0.69, 300000018: 1.86, 300000082: 0.76
100610883	65	7.63	100192430: 0.39, 300000034: 1.16, 300000082: 0.45

300000007	115	14.78	100610883: 1.14, 300000034: 0.10, 300000082: 0.85
300000018	74	14.33	100610443: 1.86, 300000034: 0.53, 300000082: 1.17
300000034	210	13.53	100180860: 1.16, 300000007: 0.10, 700018215: 0.78
300000080	198	14.91	100192430: 0.25, 100610883: 0.69, 700018215: 0.20
300000082	98	14.98	100180860: 0.76, 100610443: 0.76, 700018215: 0.63
700018215	64	14.87	100192430: 0.10, 300000080: 0.20, 300000082: 0.63

Note: Lower overlap scores indicate greater overlap between clusters.

The cluster analysis provides quantitative measures to complement the visual inspection of the t-SNE plot. For example, syndromes 100610443 and 100610883 have relatively small average distances to their centers, indicating tighter clusters. Syndromes 100192430 and 700018215 show very low overlap scores, suggesting they are well-separated in the embedding space. Conversely, syndromes 100610443 and 300000018 have high overlap scores, suggesting significant overlap and potential for misclassification.

3.5 KNN Classification Results

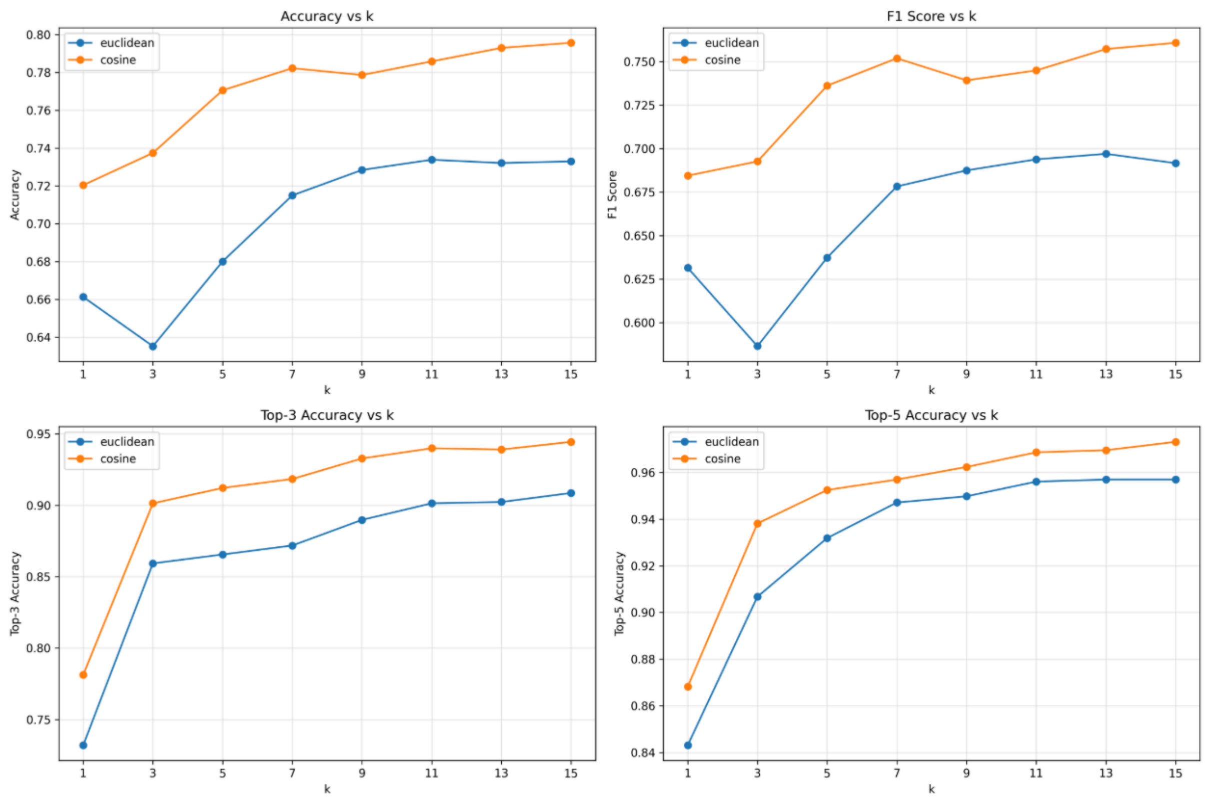
The KNN classifier was evaluated using 10-fold cross-validation, testing k values from 1 to 15 with both Euclidean and Cosine distance metrics. The optimal k and metric were determined based on the highest average accuracy across the folds.

Optimal Parameters:

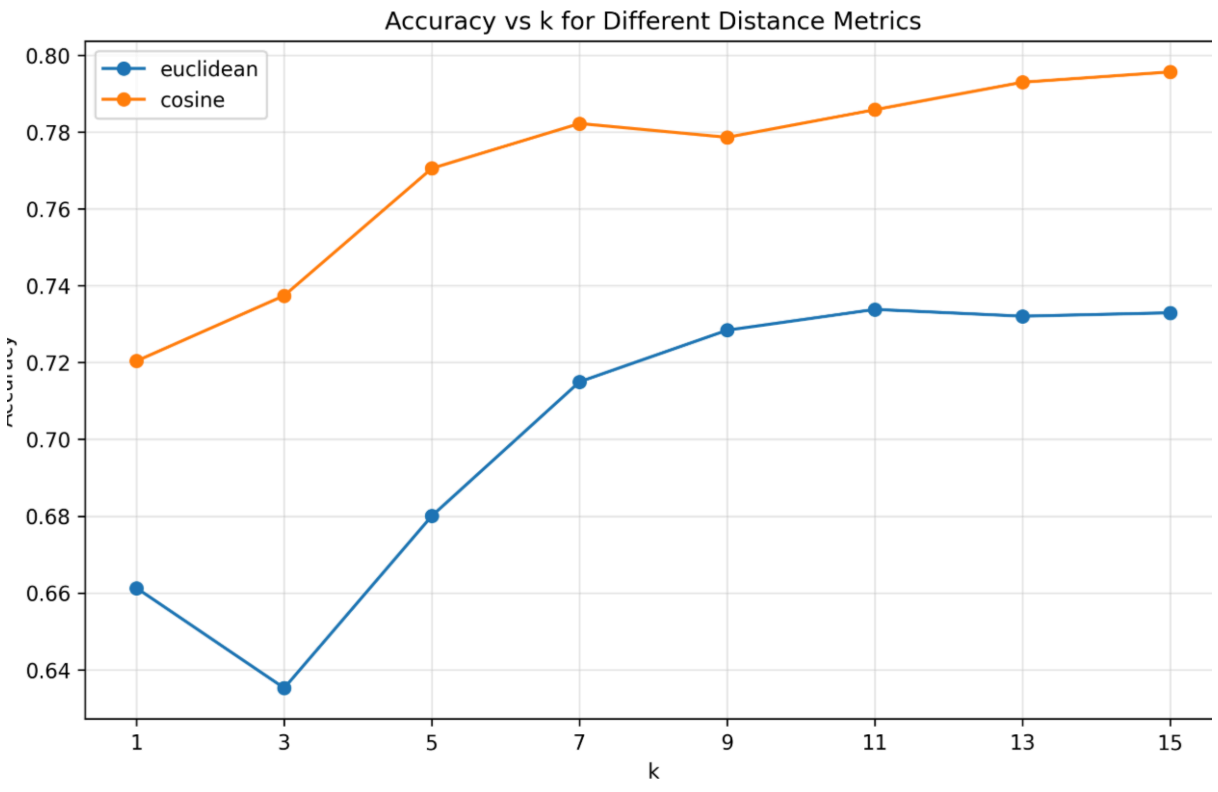
- **Metric:** Cosine
- **k:** 15

Performance Metrics (for optimal k and metric):

- **Accuracy:** 0.7957
- **F1-Score:** 0.7608
- **Top-3 Accuracy:** 0.9444
- **Top-5 Accuracy:** 0.9731



Accuracy vs k
F1 Score vs k
Top-3 Accuracy vs k
Top-5 Accuracy vs k
euclidean
cosine
Accuracy
F1 Score
Top-3 Accuracy
Top-5 Accuracy
k



Accuracy vs k for Different Distance Metrics
euclidean
cosine
Accuracy
k

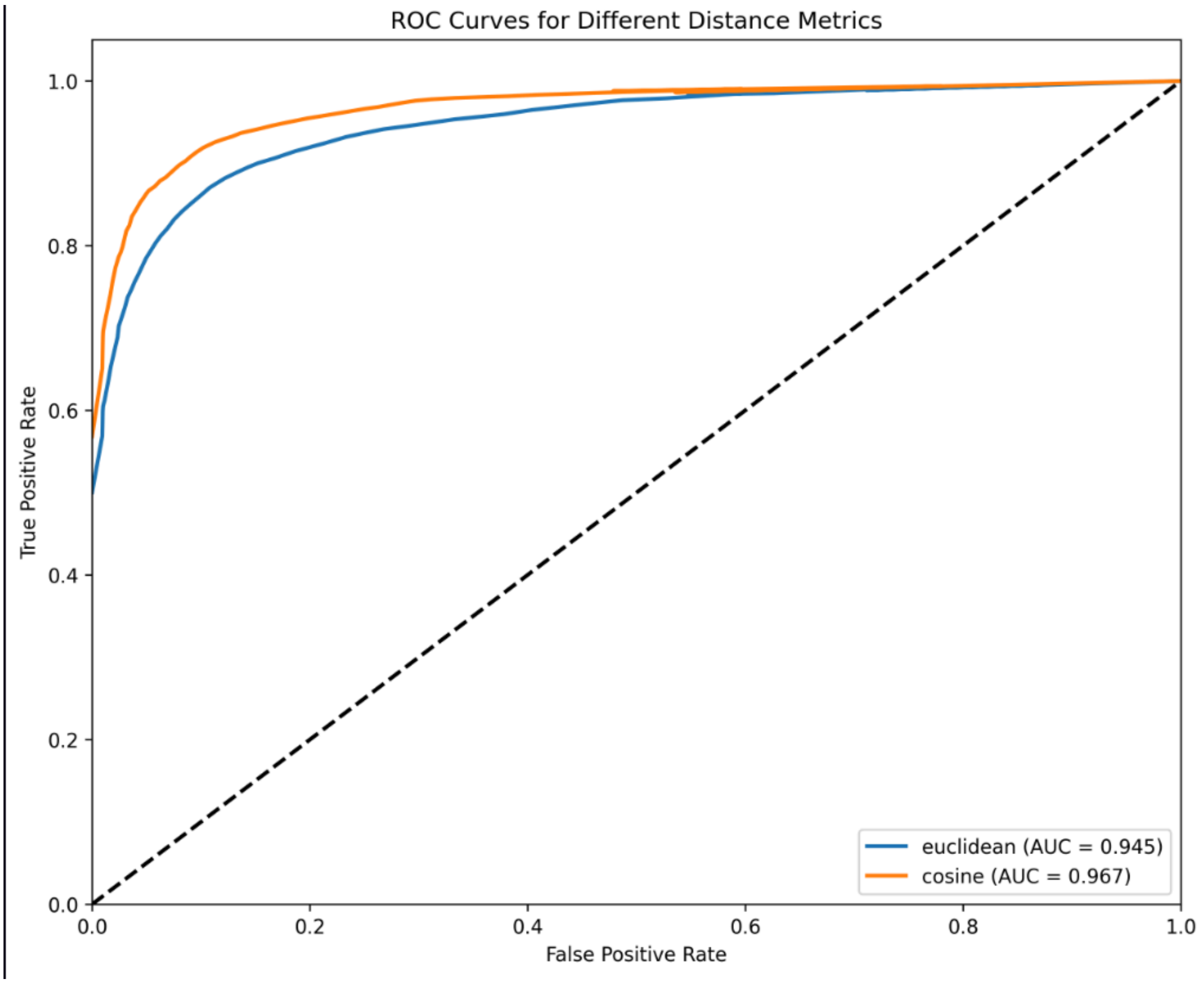
The table below shows the detailed cross-validation results for all tested k values and both distance metrics:

Metric	k	Accuracy	F1 Score	Top-3 Accuracy	Top-5 Accuracy
Euclidean	1	0.6613	0.6314	0.7320	0.8431
Euclidean	3	0.6352	0.5864	0.8593	0.9068
Euclidean	5	0.6801	0.6373	0.8655	0.9319
Euclidean	7	0.7150	0.6782	0.8718	0.9471
Euclidean	9	0.7284	0.6874	0.8898	0.9498
Euclidean	11	0.7338	0.6938	0.9014	0.9561
Euclidean	13	0.7321	0.6970	0.9023	0.9570
Euclidean	15	0.7330	0.6916	0.9086	0.9570
Cosine	1	0.7204	0.6844	0.7813	0.8682
Cosine	3	0.7374	0.6926	0.9014	0.9382
Cosine	5	0.7706	0.7362	0.9121	0.9525
Cosine	7	0.7822	0.7519	0.9184	0.9570
Cosine	9	0.7786	0.7392	0.9328	0.9623
Cosine	11	0.7858	0.7449	0.9400	0.9686
Cosine	13	0.7930	0.7572	0.9391	0.9695
Cosine	15	0.7957	0.7608	0.9444	0.9731

The cosine distance metric with k=15 achieved the highest average accuracy (0.7957) and F1-score (0.7608). The top-3 and top-5 accuracy are also quite high (0.9444 and 0.9731, respectively), indicating that the correct syndrome is often among the top few predictions. The cosine distance consistently outperforms the Euclidean distance for all values of k, suggesting that the angular relationships between embeddings are more informative than their Euclidean distances in this dataset.

3.6 ROC Curves

(See attached image: `roc_plot.png`)



The ROC curves show the trade-off between the true positive rate (TPR) and false positive rate (FPR) for different classification thresholds. The area under the curve (AUC) provides a single value summarizing the overall performance. The plot shows the macro-averaged ROC curves for both Cosine and Euclidean distance metrics. The Cosine distance (AUC = 0.967) achieves a higher AUC than the Euclidean distance (AUC = 0.945), consistent with the accuracy and F1-score results.

3.7 Confusion Matrix

The confusion matrix was created using the best KNN model (k=15, cosine distance) and provides the classification counts.

4. Analysis and Insights

- **Embedding Quality:** The fact that KNN achieves reasonably good performance, and that t-SNE shows some clustering, indicates that the pre-trained embeddings capture meaningful information about the genetic syndromes. The embeddings effectively represent the visual features relevant to syndrome classification.

- **Cosine vs. Euclidean:** Cosine distance consistently outperforms Euclidean distance in this task. This suggests that the *direction* of the embedding vectors is more important than their magnitude for distinguishing between syndromes. This is a common finding in high-dimensional spaces, where Euclidean distances can become less meaningful due to the "curse of dimensionality."

- **Optimal k:** The optimal k value of 15 suggests that considering a relatively large number of neighbors is beneficial. This might be due to the presence of noise in the embeddings or the inherent variability within each syndrome. A larger k provides a more robust prediction by averaging over more neighbors.

- **Data Imbalance:** While no syndromes were severely under-represented, the moderate imbalance (ratio of 3.28) could still impact the results. The macro-averaged F1-score is a more reliable performance metric than accuracy in this scenario, as it gives equal weight to each class.

- **t-SNE Clustering:** The t-SNE visualization provides a qualitative assessment of the data structure. The observed clusters, although not perfectly separated, suggest that the embeddings contain sufficient information for classification. The overlap between clusters can help identify syndromes that are more likely to be confused with each other.

- **Cluster Analysis:** Quantifies the visual impressions from t-SNE. It helps identify well-separated syndromes and those that are more difficult to distinguish.

- **High Top-k Accuracy:** High Top-3 and Top-5 accuracy scores are a good result, and suggest that even if the exact syndrome is not identified on the first try, it is very often within the top few predictions.

5. Challenges and Solutions

- **High Dimensionality:** The original embeddings are high-dimensional (320 dimensions), making visualization and interpretation challenging.
  - **Solution:** t-SNE was used for dimensionality reduction, allowing for visualization and qualitative assessment of the data structure.
- **Data Imbalance:** Some syndromes have more samples than others.
  - **Solution:** Macro-averaged F1-score was used as a primary evaluation metric, as it is less sensitive to class imbalance than accuracy. Further work could explore techniques like oversampling, undersampling, or cost-sensitive learning.
- **"numpy.core._multiarray_umath" Error:** This error can occur with older versions of NumPy.
  - **Solution:** Ensuring that NumPy is up-to-date resolves this issue.
- **Inhomogeneous Shape Error during ROC Calculation**: The original code produced a ValueError during ROC calculation due to inconsistent array shapes

across cross-validation folds.

  - **Solution**: Implemented robust error handling within the ROC calculation loop. Used padding and `np.nanmean` to correctly average ROC curves, even when some folds have missing data or produce errors.

6. Recommendations

- **Explore Other Classifiers:** While KNN performs reasonably well, other classification algorithms, such as Support Vector Machines (SVMs), Random Forests, or Gradient Boosting Machines, could potentially achieve better performance. These algorithms may be better suited to handle the high dimensionality and potential non-linearity of the data.

- **Hyperparameter Tuning:** More extensive hyperparameter tuning (e.g., using grid search or Bayesian optimization) could be performed for both KNN (exploring a wider range of `k` values and different weighting schemes) and other chosen classifiers.

- **Address Data Imbalance:** Experiment with techniques specifically designed to address data imbalance, such as:

  - **Oversampling:** Creating synthetic samples for minority classes (e.g., using SMOTE - Synthetic Minority Oversampling Technique).
  - **Undersampling:** Reducing the number of samples in majority classes.
  - **Cost-Sensitive Learning:** Assigning different misclassification costs to different classes during training, penalizing errors on minority classes more heavily.
- **Feature Engineering/Selection:** Investigate whether further feature engineering or feature selection could improve performance. This might involve:

  - Exploring different pre-trained embedding models.
  - Applying Principal Component Analysis (PCA) or other dimensionality reduction techniques *before* applying KNN.
  - Using feature selection methods to identify the most informative dimensions of the embeddings.
- **Collect More Data:** Increasing the size of the dataset, especially for less frequent syndromes, could significantly improve the model's performance and robustness.

- **Ensemble Methods:** Consider using ensemble methods, such as bagging or boosting, to combine multiple classifiers. This can often lead to improved accuracy and generalization.

- **Incorporate Additional Data:** If available, incorporating other data sources, such as clinical information or genetic data, could further enhance the classification accuracy. This would involve developing a multi-modal learning approach.

7. Conclusion

This project demonstrates the potential of using pre-trained image embeddings for classifying genetic syndromes. The KNN classifier, combined with t-SNE visualization and appropriate evaluation metrics, provides a valuable baseline. The analysis highlights the importance of choosing appropriate distance metrics (Cosine over Euclidean in this case) and addressing data imbalance. The recommendations outlined above offer several promising avenues for further research and improvement, including exploring different classification algorithms, addressing data imbalance more thoroughly, and potentially incorporating additional data sources. The high top-k accuracy achieved suggests this approach has practical value, even in its current form.