[REDACTED] ML Developer Test Report

This document will detail the coding and analytical choices for the task of classifying genetic syndromes based on the 320-dimensional embeddings derived from images associated hierarchically to that syndrome as per described in the practical test itself.

The full pipeline of the data, from the embeddings to the finished plots is as follows:

- Data Loading
  "raw" data is received in the form of a pickle file (mini_gm_public_v0.1.p) hierarchically arranged in this configuration:

```
{
  'syndrome_id': {
    'subject_id': {
      'image_id': [320-dimensional embedding]
    }
  }
}
```

  The Dataset is then loaded into a Pandas Data Frame for processing.

- Data Preprocessing
  The hierarchies are flattened and the full dataset is checked for missing information and for any embeddings that are not in the expected format.
  the Data Frame has the following structure in the end:

```
image_id|subject_id|syndrome_id|dimension 1|dimension 2|[...]|dimension 320
```

- Data Processing
  The preprocessed data is then reduced in dimensionality from 320D to 2D using a t-SNE algorithm with the following parameters*:

```
PERPLEXITY = 60.0
LEARNING_RATE = 8.0
EARLY_EXAGGERATION = 90.0
```

  And thus the dataset's structure is altered to match the new dimensionality

```
image_id|subject_id|syndrome_id|dimension_1|dimension_2
```

- Data Classification
  A KNN algorithm is then used to classify the reduced embeddings into their respective syndromes using Euclidean and Cosine distance metrics and the results are validated using a 10-fold cross-validation optimizing for accuracy.

- Data Visualization
  Plots are generated for visualization of the data spread after the t-SNE, colored by syndrome, and also for the ROC AUC curves for both Euclidean and Cosine metrics, separated by syndrome. A table is also generated comparing the best-k, accuracy, top-k-accuracy, precision, recall, f1 scores and roc-auc scores between Euclidean and Cosine results. The Accuracy pertains to the best-k, but the other metrics are the average between folds.

* t-SNE was the reduction method used, as asked, however, other options might be better suited for the task at hand. Although the global structure of the data is not explicitly expected to influence on the final outcome, a reduction method like UMAP could allow new insights by preserving some of the global relationships. Initial parameters for the t-SNE were chosen based on the visual spread of the plot generated by the data visualization out of 120 options. However, after the full completion of the task, a testing mode allowed for the optimization of parameters aiming for better accuracy.

```
TSNE_PERPLEXITY = [5, 20, 30, 50, 60]
TSNE_LEARNING_RATE = [10, 200, 500, 800, 1000]
TSNE_EARLY_EXAGGERATION = [10, 15, 20, 35, 50]
```

Original parameters were perplexity=50.0, learning rate=10.0 early and exaggeration=20.0 chosen out of the combination of these options (estimated using scikit-learn documentation).

The following data was extracted from the dataset:

Metrics table

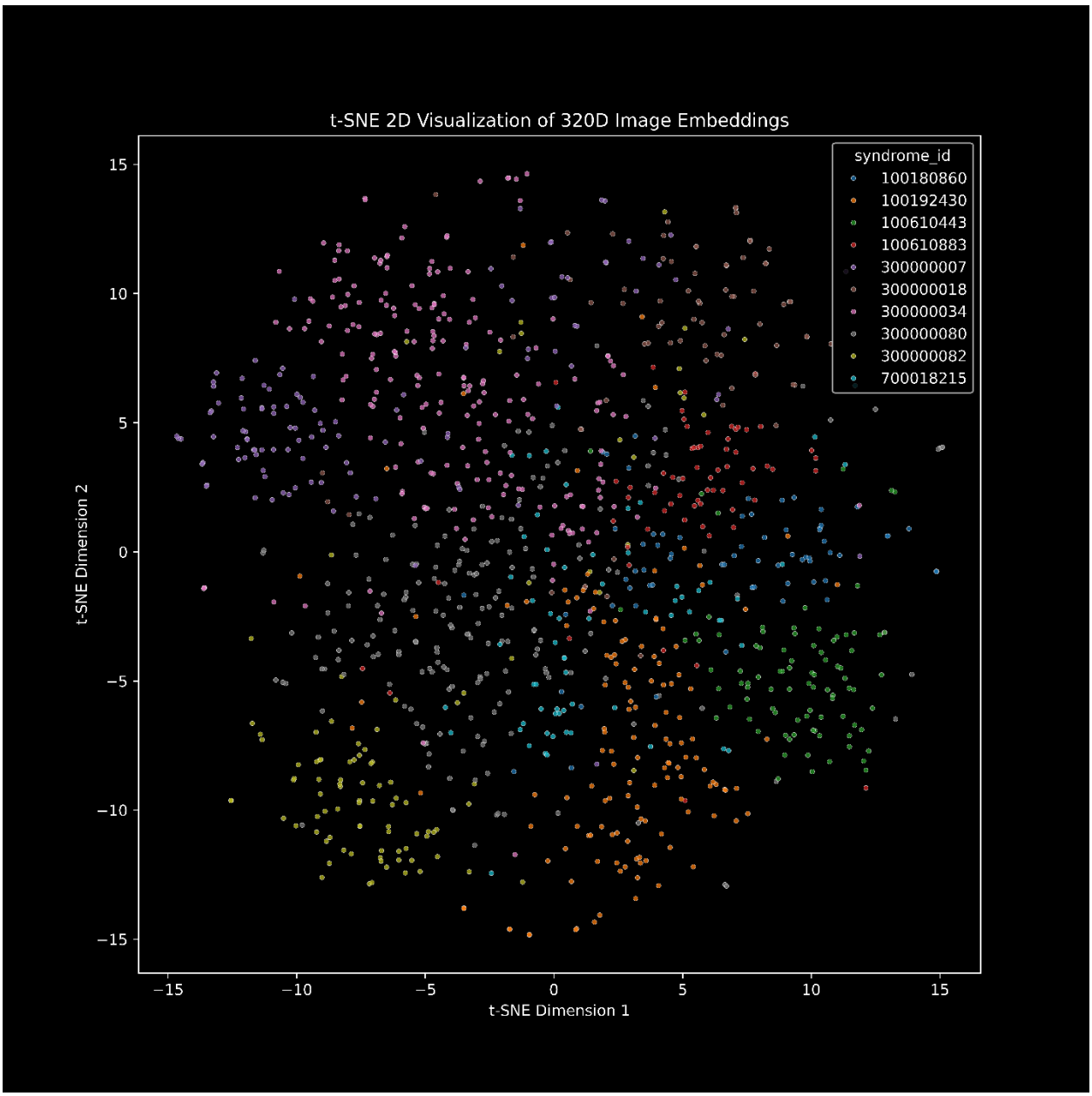
metric	euclidean_result	cosine_result
best_k	9	10
accuracy	0.754424067	0.628032497
top_3_accuracy	0.906581048	0.842679365
precision	0.767181508	0.636807798
recall	0.765135773	0.645284846
f1_score	0.760954194	0.634747995
roc_auc_score	0.930390328	0.865252112

The K in top_k_accuracy was arbitrarily chosen as 3 and future tests can be made to assess and visualize how close the model gets to the answer when it misses.

While the accuracy pertains to the best-k, the other metrics are averages weighed between folds. Metrics suggest a fairly effective classification model, especially top_k_accuracy, that suggest that the model can get the right answer on it's first 3 predictions most of the time.

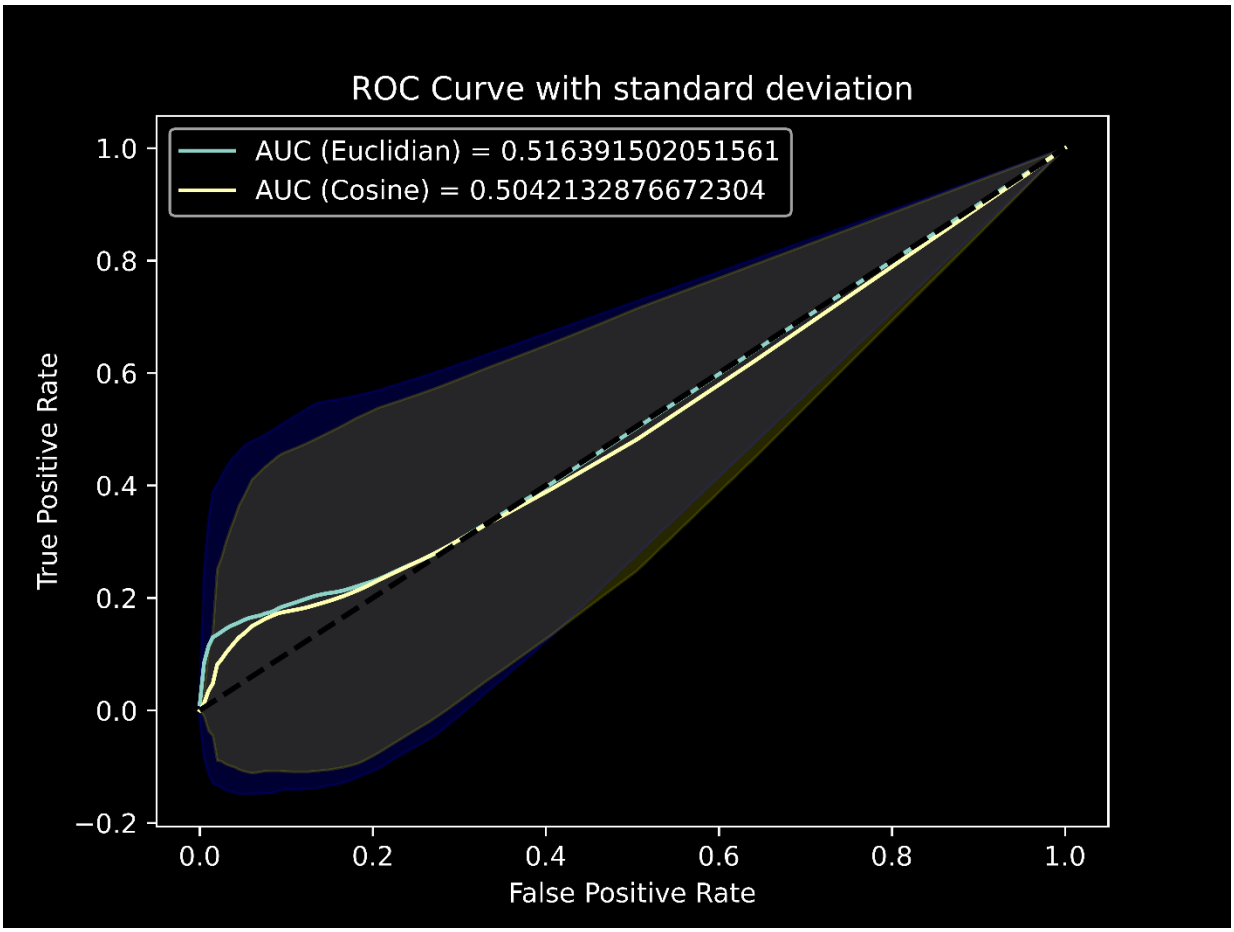
However, roc_auc_score is inconsistent with the other metrics, suggesting a model with excellent performance. While it's accuracy, precision and recall suggest some errors.

Data spread after t-SNE 320D reduction to 2D



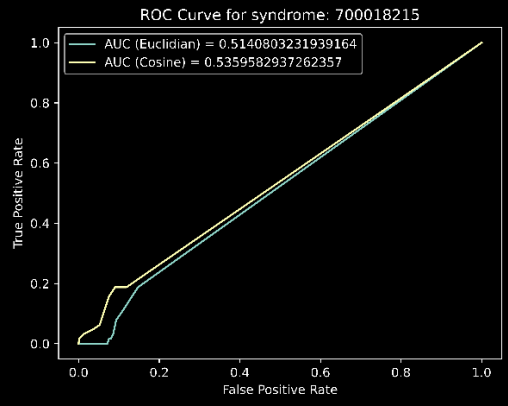
The plot shows some clear color aggregates suggesting a good match for clusterization.

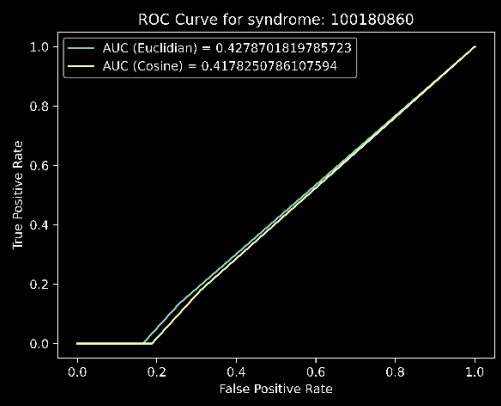
ROC AUC curve with standard deviation

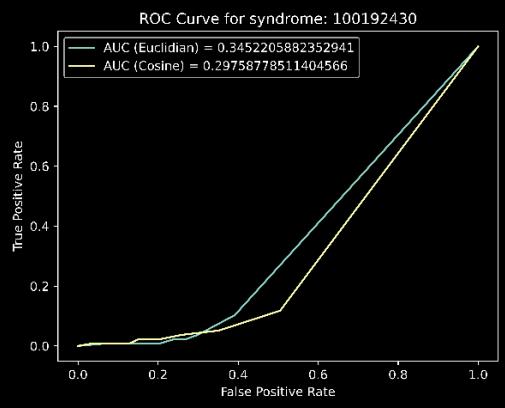


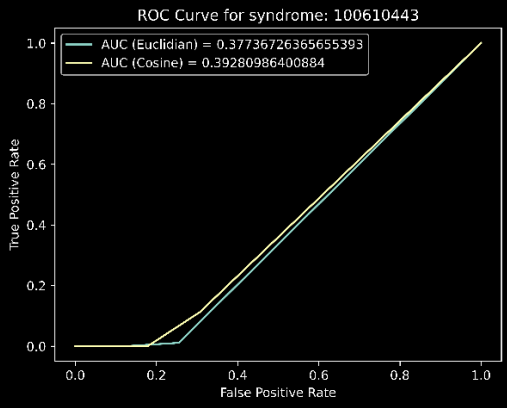
The ROC AUC curve by itself shows an abysmal performance on most of the cases, suggesting that the model is performing as well as a guess would. However, the high standard deviation colored around the line, obtained while standardizing the syndrome results, shows that the model's results is highly inconsistent across syndromes, as such, Individual plots were made for each syndrome.

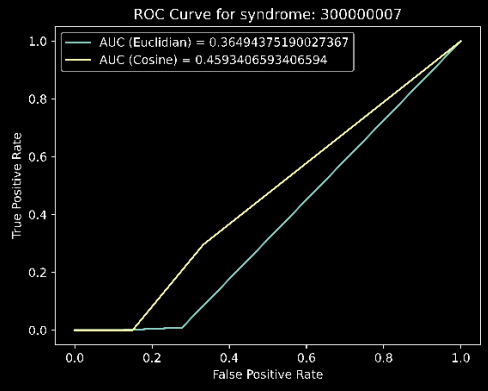
The individual plots for each syndrome (shown below) confirm that the model is inconsistent across syndromes. Only on syndromes 300000018 and 100610883 the model achieved satisfactory results, while on all other cases, the model's performance's effectiveness is close to guessing.

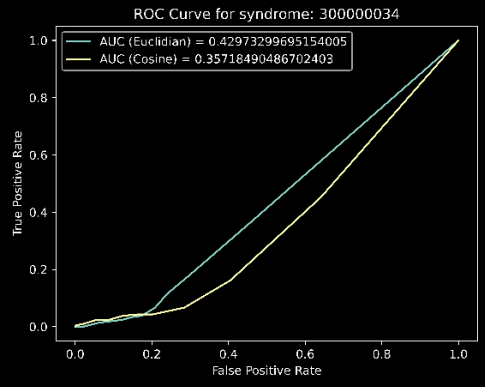

ROC Curve for syndrome: 700018215
AUC (Euclidian) = 0.5140803231939164
AUC (Cosine) = 0.5359582937262357
True Positive Rate
False Positive Rate

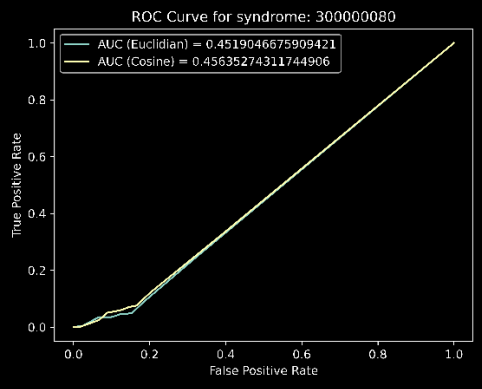

ROC Curve for syndrome: 100180860
AUC (Euclidian) = 0.4278701819785723
AUC (Cosine) = 0.4178250786107594
True Positive Rate
False Positive Rate

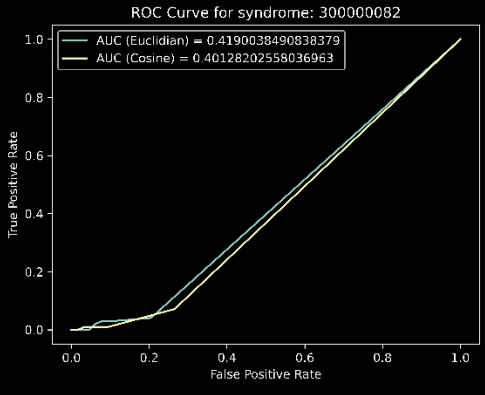

ROC Curve for syndrome: 100192430
AUC (Euclidian) = 0.3452205882352941
AUC (Cosine) = 0.29758778511404566
True Positive Rate
False Positive Rate

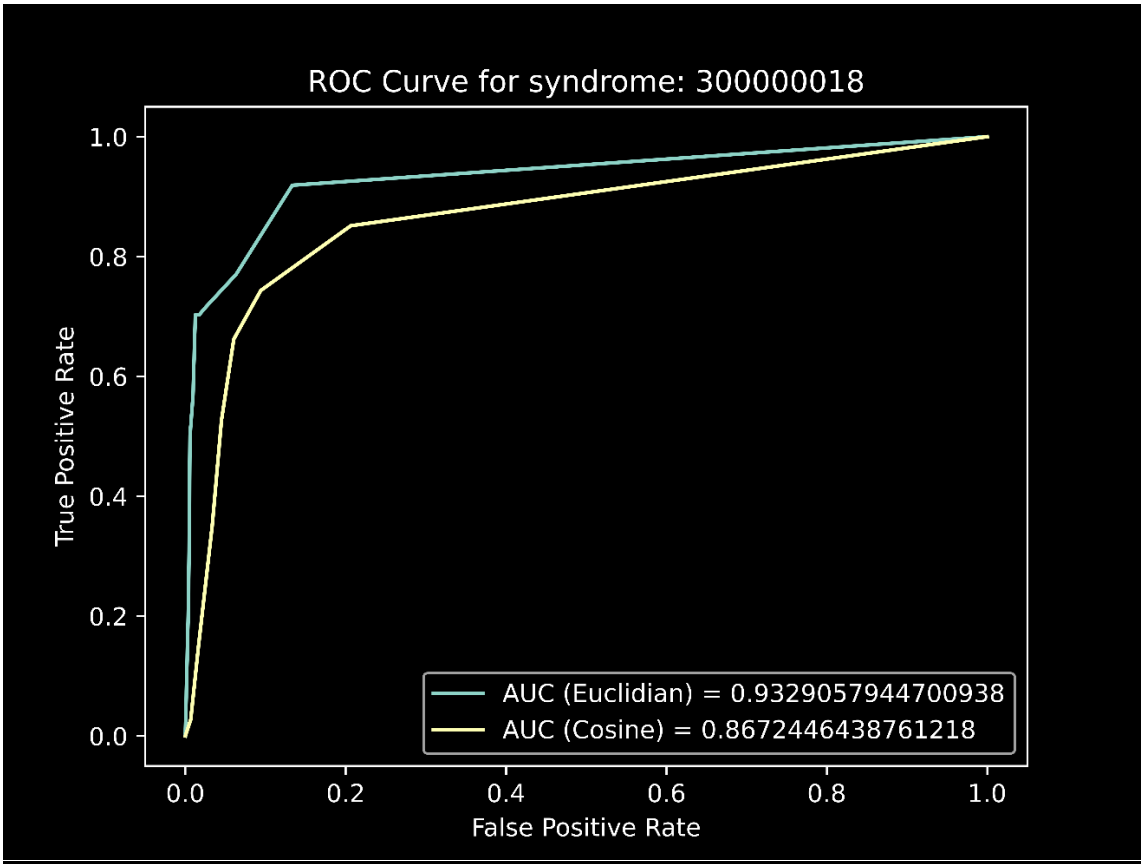

ROC Curve for syndrome: 100610443
AUC (Euclidian) = 0.37736726365655393
AUC (Cosine) = 0.39280986400884
True Positive Rate
False Positive Rate

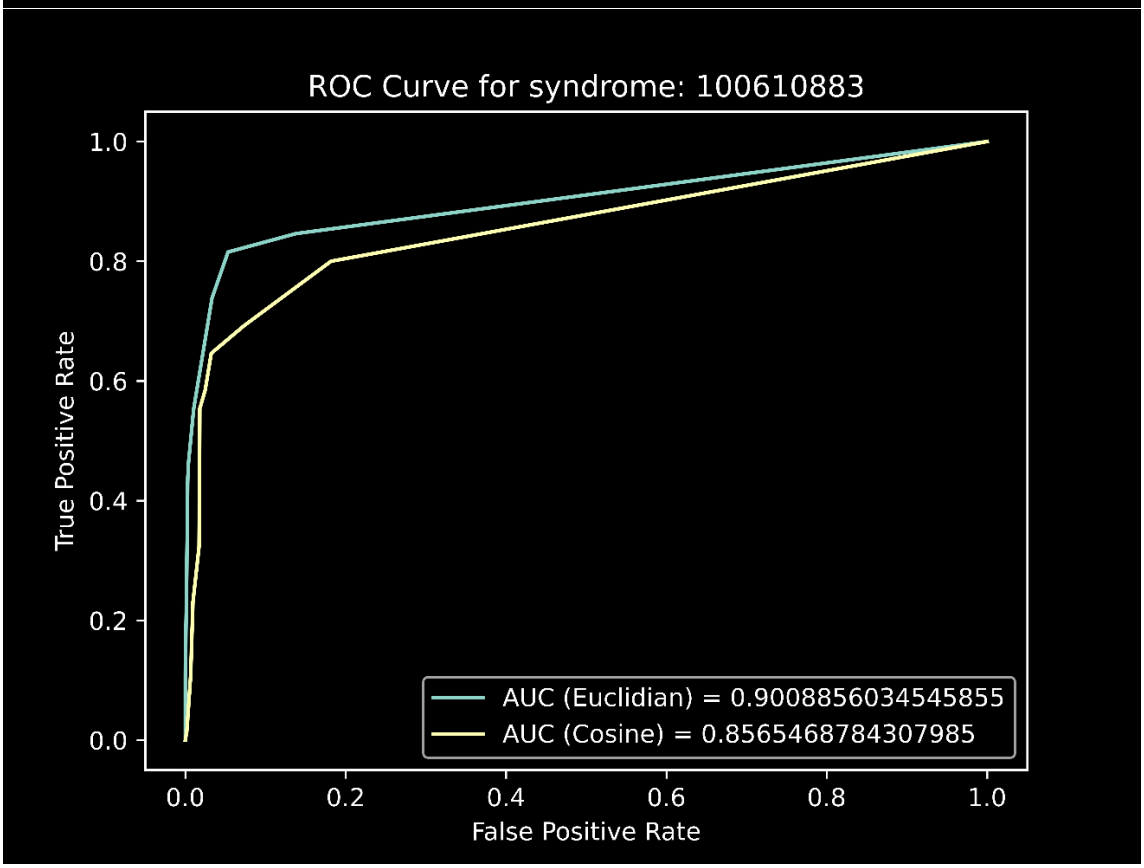

ROC Curve for syndrome: 300000007
AUC (Euclidian) = 0.36494375190027367
AUC (Cosine) = 0.4593406593406594
True Positive Rate
False Positive Rate


ROC Curve for syndrome: 300000034
AUC (Euclidian) = 0.42973299695154005
AUC (Cosine) = 0.3571849048670240 3
True Positive Rate
False Positive Rate


ROC Curve for syndrome: 300000080
AUC (Euclidian) = 0.4519046675909421
AUC (Cosine) = 0.456352743111744906
True Positive Rate
False Positive Rate


ROC Curve for syndrome: 300000082
AUC (Euclidian) = 0.4190038490838379
AUC (Cosine) = 0.4012820255803696 3
True Positive Rate
False Positive Rate


ROC Curve for syndrome: 300000018
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
AUC (Euclidian) = 0.9329057944700938
AUC (Cosine) = 0.8672446438761218


ROC Curve for syndrome: 100610883
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
AUC (Euclidian) = 0.9008856034545855
AUC (Cosine) = 0.8565468784307985

Comparing Cosine and Euclidean metrics indicates that Euclidean is the better metric for clustering this dataset, which is expected since the t-SNE reduced the dimensionality to 2, drastically reducing the possible angular complexities between the distances, the Euclidean distances are also more sensible to the scalar and normalized nature of the dataset.

The cosine model would likely perform better if all the 320 dimensions were used in the training, but at a great cost of computational power. That being said I would suggest testing with a 3D reduction instead of 2D and comparing again. The Cosine metric find relations in the angles between the nodes, leading to a better performance.

I would also suggest testing of the model removing the two syndromes that performed well, allowing for a more specific view of the performance on those cases, especially top_k_accuracy that might be used to at least identify the most likely syndromes in those cases.

Summarized Core Insights:

- In the case of t-SNE 2D reduction the model performs best with Euclidean metrics, as expected.
- The model is ineffective on the task of correctly identifying the syndrome for most syndromes.
- The model could still be used as is to correctly identify cases of syndromes 300000018 and 100610883 with ≈90% accuracy.
- The model can also be used to identify the 3 most likely syndromes with the same ≈90% accuracy.

Possible Future testing:

- Consider using a different reduction method, such as UMAP
- Reducing dimensionality to 3D instead of 2D and testing both Euclidean and Cosine metrics
- Plotting the top_k_accuracy for different K
- Testing the model on the cases that it's weak to assess its effectiveness on the worst case

References:

https://stackoverflow.com
https://docs.scipy.org
https://scikit-learn.org
https://www.w3schools.com
https://en.wikipedia.org
https://chatgpt.com
https://pandas.pydata.org
https://www.freecodecamp.org
https://www.kdnuggets.com

Now, leaving aside the fictitious biotech company.

In the making of this project, I had to learn a LOT about the workings of data pipelines like the one I was asked to create. I also had to learn on the fly about many methods and metrics that I only saw in passing glance while on my academic journey. So, the first thing I ought to do is to thank you for the opportunity and for the experience you already gave me.

The major difficulty I encountered was my own lack of experience with the processes, which led me to make some parts of the task in redundant ways and forced me to go back and fix them before sending. I hope I didn't let any of those into the final product.

For the sake of honesty, I have to say that I used a fair amount of ChatGPT, especially to help me understand the concepts I didn't know enough about and to help me fix the errors in my code. Still, this document and the code I'm sending are my own work, not only as requested by the assignment itself but also as the fruit of my professional work ethic.