Relatório do Teste Prático de Machine Learning Junior

1. Metodologia

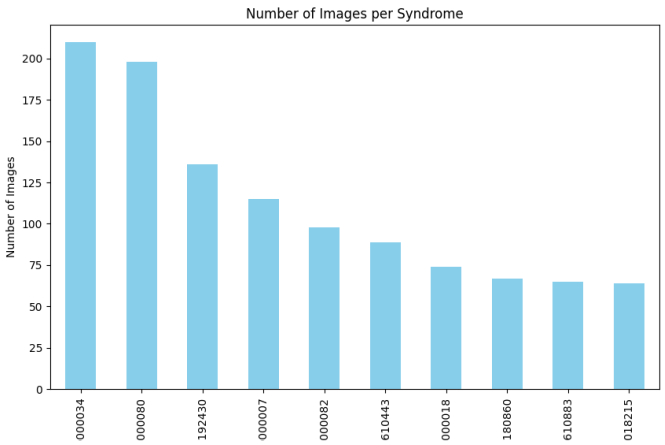
1.1 Processamento de Dados

O conjunto de dados fornecido era um arquivo pickle hierárquico contendo embeddings derivados de imagens. Para preparar esses dados para análise:

- **Carregamento** : Carreguei o arquivo pickle usando o módulo pickle do Python. Essa etapa foi essencial para garantir que os dados estivessem prontos para processamento.
- **Transformação** : A estrutura hierárquica (‘syndrome_id’ → ‘subject_id’ → ‘image_id’) foi convertida em um formato tabular, resultando em um DataFrame com as colunas: syndrome_id, subject_id, image_id e embedding. Essa abordagem facilitou a manipulação dos dados nas etapas subsequentes.
- **Validação** :
- Verifiquei dados ausentes ou inconsistentes, garantindo que os embeddings tivessem a dimensionalidade correta (320 dimensões).
- Removi entradas duplicadas para assegurar a integridade dos dados.
- **Saída** : Os dados processados foram salvos como um arquivo CSV para facilitar o uso futuro.

1.2 Análise Exploratória de Dados (EDA)

- **Estatísticas** :
  - O conjunto de dados continha embeddings correspondentes a **1.116 imagens** distribuídas entre vários syndrome_id.
  - Para entender melhor a distribuição das amostras, gerei um gráfico de barras que mostrou o número de imagens por síndrome.

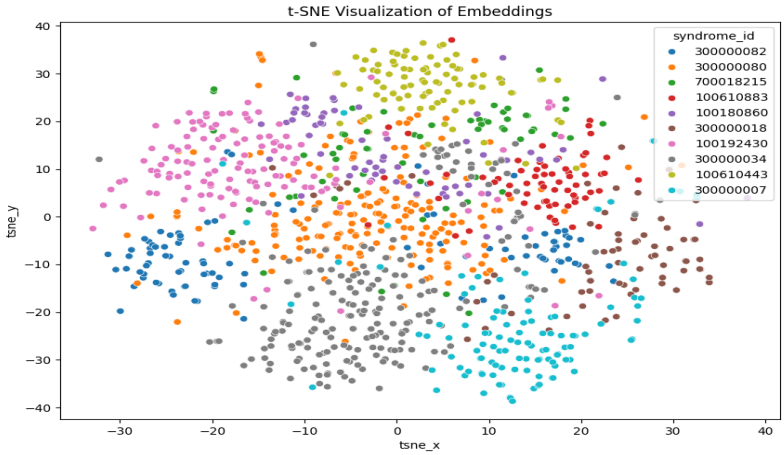


- Observei um volume de dados limitado no conjunto analisado, com algumas síndromes apresentando menos de 100 amostras. Esse cenário pode levar a desafios relacionados à generalização do modelo e ao possível desequilíbrio de classes, o que pode impactar negativamente os resultados da classificação. Um aumento no volume de dados para

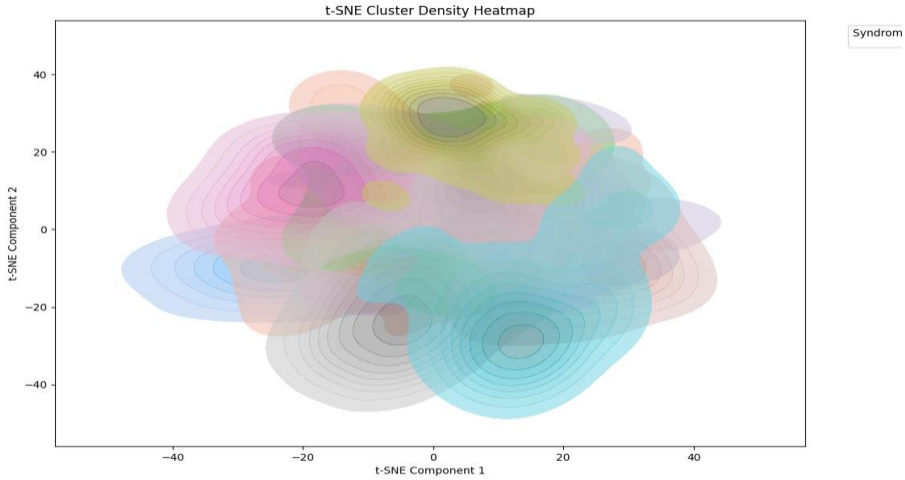
síndromes com menos amostras seria uma estratégia importante para melhorar a qualidade do modelo.

1.3 Visualização de Dados

- **Visualização com t-SNE** :
  - Os embeddings foram reduzidos a duas dimensões usando t-SNE. Essa técnica ajudou a revelar padrões e clusters nos dados.
- Eu criei:
  - Um gráfico de dispersão com os pontos coloridos por syndrome_id.



  - Um mapa de densidade para destacar regiões de alta concentração de pontos.



- **Observações** :
  - Algumas síndromes formaram clusters distintos, indicando que os embeddings podem ter separabilidade suficiente para classificação.
  - Entretanto, também identifiquei clusters sobrepostos, sugerindo que alguns casos poderiam ser desafiadores para os modelos.

1.4 Tarefa de Classificação

- **K-Nearest Neighbors (KNN)** :
  - O uso do KNN foi especificado nos requisitos do teste, e ele foi implementado com métricas de distância cosseno e euclidiana. Apesar disso, em um cenário prático, eu também consideraria testar outros modelos, como redes neurais ou SVM, para avaliar se esses poderiam oferecer resultados melhores.
  - Para determinar o valor ideal de k no modelo KNN, utilizei a métrica de validação cruzada por meio da função cross_val_score do Scikit-learn. Essa abordagem permitiu avaliar o desempenho do modelo em diferentes valores de k (variando de 1 a 15), utilizando validação cruzada com 10 folds. O critério para a seleção do melhor k foi a **precisão média obtida** ao longo dos folds. Cada valor de k foi testado com duas métricas de distância distintas (euclidiana e cosseno), e o k que apresentou o maior desempenho médio foi escolhido como o ideal para cada métrica de distância.
  - Essa análise ajudou a garantir que o modelo fosse configurado para maximizar sua capacidade de generalização e evitar tanto underfitting quanto overfitting. Os valores selecionados para k foram documentados nas tabelas de resultados e mostraram um impacto significativo no desempenho final do modelo.
- **Métricas** :
- Para os melhores modelos, calculei as seguintes métricas:
  - Precisão
  - F1-Score
  - Top-k Accuracy
  - AUC (Area Under the ROC Curve)
- As matrizes de confusão e os relatórios de classificação foram gerados para ambas as métricas.

1.5 Avaliação de Métricas

- **Curvas ROC AUC** :
  - Tracei as curvas ROC AUC para ambas as métricas, calculando a média dos resultados ao longo dos folds de validação cruzada.
  - Sobre as curvas ROC, observei valores muito altos tanto para o AUC da distância euclidiana quanto para o AUC da distância cosseno (0,92 e 0,94, respectivamente). Em um ambiente prático, eu investigaria mais profundamente a razão desses valores elevados, buscando entender se há algum fator externo ou característica específica do conjunto de dados que os explique. No entanto, como as outras métricas também apresentaram resultados consistentes e bem avaliados, considerei que uma investigação detalhada não seria necessária para este teste.
  - As curvas foram exibidas juntas em um único gráfico para facilitar a comparação.
- **Tabelas Resumo** :
  - Gerei um arquivo CSV resumindo as métricas de desempenho para ambas as métricas de distância. Essa tabela incluía Precisão, F1-Score, Top-k Accuracy e AUC.

2. Resultados

2.1 Resultados do Processamento de Dados

- Total de amostras após o processamento: **1.116** .
- Número de síndromes únicas: **10** .
- Distribuição de imagens por síndrome salva como syndrome_distribution.png.

2.2 Resultados da Visualização

- Gráfico de Dispersão t-SNE: Salvo como tsne_visualization.png.
- Mapa de Densidade t-SNE: Salvo como tsne_cluster_density_heatmap.png.

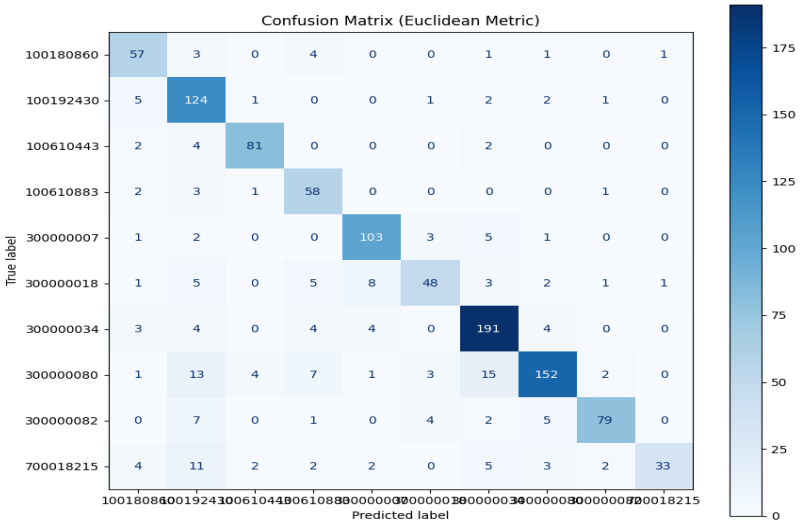
2.3 Resultados da Classificação

Resumo de Métricas de Desempenho

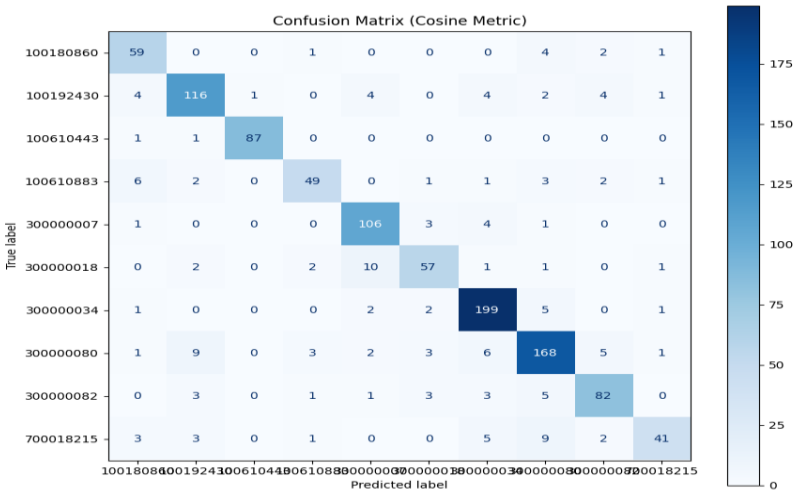
Métrica	Distância Euclidiana	Distância Cosseno
Precisão	0.829	0.863
Top-k	15	12
F1-Score	0.827	0.862
AUC	0.985	0.990

Matrizes de Confusão

- Salvas como:
  - confusion_matrix_euclidean.png



- confusion_matrix_cosine.png



- As matrizes de confusão apresentaram resultados satisfatórios, indicando que o modelo não enfrenta grandes dificuldades para identificar corretamente as classes e minimizar falsos positivos ou falsos negativos. Para uma avaliação mais detalhada e robusta, seria ideal incluir novas entradas de dados para validar ainda mais o desempenho do modelo em cenários variados.

Relatórios de Classificação

Distância Euclidiana:

	precision	recall	f1-score	support
100180860	0.75	0.85	0.80	67
100192430	0.70	0.91	0.79	136
100610443	0.91	0.91	0.91	89
100610883	0.72	0.89	0.79	65
300000007	0.87	0.90	0.88	115
300000018	0.81	0.65	0.72	74
300000034	0.85	0.91	0.88	210
300000080	0.89	0.77	0.83	198
300000082	0.92	0.81	0.86	98
700018215	0.94	0.52	0.67	64
accuracy			0.83	1116
macro avg	0.84	0.81	0.81	1116
weighted avg	0.84	0.83	0.83	1116

Distância Cosseno:

	precision	recall	f1-score	support
100180860	0.78	0.88	0.83	67
100192430	0.85	0.85	0.85	136
100610443	0.99	0.98	0.98	89
100610883	0.86	0.75	0.80	65
300000007	0.85	0.92	0.88	115
300000018	0.83	0.77	0.80	74
300000034	0.89	0.95	0.92	210
300000080	0.85	0.85	0.85	198
300000082	0.85	0.84	0.84	98
700018215	0.87	0.64	0.74	64
accuracy			0.86	1116
macro avg	0.86	0.84	0.85	1116
weighted avg	0.86	0.86	0.86	1116

---

3. Análise

3.1 Observações do t-SNE

- Clusters distintos indicam separabilidade potencial de algumas síndromes.
- Clusters sobrepostos sugerem possíveis desafios na classificação de síndromes com embeddings semelhantes.

3.2 Comparação de Métricas de Distância

- A **Distância Cosseno** superou ligeiramente a **Distância Euclidiana** em todas as métricas (Precisão, F1-Score, AUC, Top-k Accuracy).
- A diferença de desempenho pode ser atribuída a:

  - A capacidade da distância cosseno de lidar com dados de alta dimensionalidade onde a magnitude dos vetores pode importar menos que a direção.

3.3 Desafios e Soluções

- **Desafio**: Desequilíbrio de classes.
  - **Solução**: Aumentar os dados de treinamento para classes sub-representadas ou aplicar pesos nas classes em iterações futuras.
- **Desafio**: Clusters sobrepostos na visualização t-SNE.
  - **Solução**: Eu teria utilizado modelos mais avançados (e.g., SVM, redes neurais) em experimentos complementares para lidar com fronteiras de decisão complexas.
- **Desafio**: Ajuste de hiperparâmetros em KNN e outros modelos.
  - **Solução**: Implementaria uma busca sistemática de hiperparâmetros (e.g., Grid Search ou Random Search) para encontrar os melhores valores de k e parâmetros relacionados às métricas de distância, maximizando o desempenho do modelo.
- Manipulação de arquivos: Garantiu-se uma estrutura de diretórios apropriada para evitar erros de salvamento.
- Inconsistências de métricas: Verificaram-se os cálculos de métricas e as configurações de validação cruzada.

---

4. Recomendações

- Experimentar com outros classificadores (e.g., Random Forest, SVM).
- Abordar o desequilíbrio de classes com técnicas de aumento de dados ou oversampling.
- Realizar otimização de hiperparâmetros para KNN e outros modelos.
- Investigar os embeddings mais profundamente (e.g., PCA, clustering).

---

5. Conclusão

- Eu observei que a distância cosseno foi ligeiramente mais eficaz para esta tarefa, especialmente em métricas como AUC e F1-Score. Isso sugere que modelos que priorizem a direção dos embeddings em vez de sua magnitude podem ser mais adequados para dados de alta dimensionalidade, como os utilizados aqui.
- Durante o processo, identifiquei desafios relacionados ao desequilíbrio de classes e à presença de clusters sobrepostos nos dados. Esses fatores podem influenciar diretamente a capacidade do modelo de generalizar para novos dados e exigem maior atenção em projetos futuros.

- Apesar desses desafios, os resultados gerais indicam que o pipeline implementado serve como uma base promissora para classificações relacionadas a síndromes genéticas, oferecendo insights valiosos sobre os dados e o desempenho dos modelos.
- Em uma possível aplicação em um ambiente real, eu exploraria mais a fundo a distribuição dos dados e utilizaria técnicas para mitigar o impacto de classes sub-representadas, como o oversampling ou o uso de pesos adaptativos nas classes.
- Também considero que seria importante avaliar modelos mais complexos, como redes neurais profundas, que poderiam capturar padrões mais sofisticados nos embeddings e potencialmente melhorar a acurácia e a generalização dos resultados.
- A distância cosseno é ligeiramente mais adequada para esta tarefa.

---

6. Referências

- Documentação do Scikit-learn.
- Guias do Python Matplotlib e Seaborn para visualização.