



TECHNICAL REPORT



Klasifikasi Gambar Anemia Menggunakan Metode Decision Tree

Disusun oleh:

1. Yan Stephen Christian I.H (105222010)
2. Arshanda Geulis Nawajaputri (105222045)

1. Pendahuluan

Anemia merupakan salah satu kondisi medis yang paling umum di dunia, mempengaruhi miliaran orang dan sering kali tidak terdiagnosis secara tepat waktu, terutama di daerah dengan keterbatasan fasilitas kesehatan. Proses diagnosis konvensional melalui pemeriksaan laboratorium memerlukan waktu, biaya, serta keahlian medis yang khusus. Seiring dengan kemajuan teknologi dalam bidang computer vision, muncul peluang untuk mendeteksi anemia secara otomatis melalui analisis gambar sel darah merah dari mikroskop digital. Citra mikroskopik dapat mengungkap karakteristik visual yang menandai anemia, seperti tingkat kepuatan, bentuk yang tidak normal, dan variasi ukuran sel, sehingga pendekatan ini menjanjikan efisiensi dalam proses diagnosis dan potensi jangkauan yang lebih luas.

Laporan teknis ini bertujuan untuk membangun sistem klasifikasi biner menggunakan algoritma Decision Tree guna membedakan antara sel darah merah penderita anemia dan sel normal berdasarkan gambar mikroskopik. Model ini dikembangkan melalui proses ekstraksi fitur visual utama seperti warna, tekstur, dan morfologi sel. Decision Tree dipilih karena kesederhanaannya dalam pemodelan, kemampuannya menangani data berdimensi tinggi, serta interpretasi hasil yang mudah dipahami melalui representasi grafis berupa pohon keputusan. Pendekatan ini dinilai cocok untuk studi awal klasifikasi gambar medis karena proses pelatihannya yang efisien serta memberikan pemahaman awal terhadap pola visual yang membedakan sel darah normal dan anemia.

2. Deskripsi Dataset


- Dataset digunakan untuk klasifikasi gambar sel darah merah menjadi dua kategori: anemic dan nonanemic.
- Dataset berasal dari tugas yang diberikan oleh asisten dosen pada praktikum Machine Learning.
- Total terdapat 4.222 gambar berformat .png.dengan terbagi sebagai berikut:
 - anemic: 2.563 gambar sel darah merah penderita anemia.
 - nonanemic: 1714 gambar sel darah merah normal.
- Setiap gambar merupakan hasil citra mikroskopik dengan resolusi yang relatif beragam.

3. Exploratory Data Analysis (EDA)

Analisis eksploratif dilakukan untuk memahami struktur dan distribusi awal dari dataset gambar sebelum dilakukan preprocessing dan training. Langkah-langkah EDA yang dilakukan mencakup:

- Pemuatan Dataset Awal:

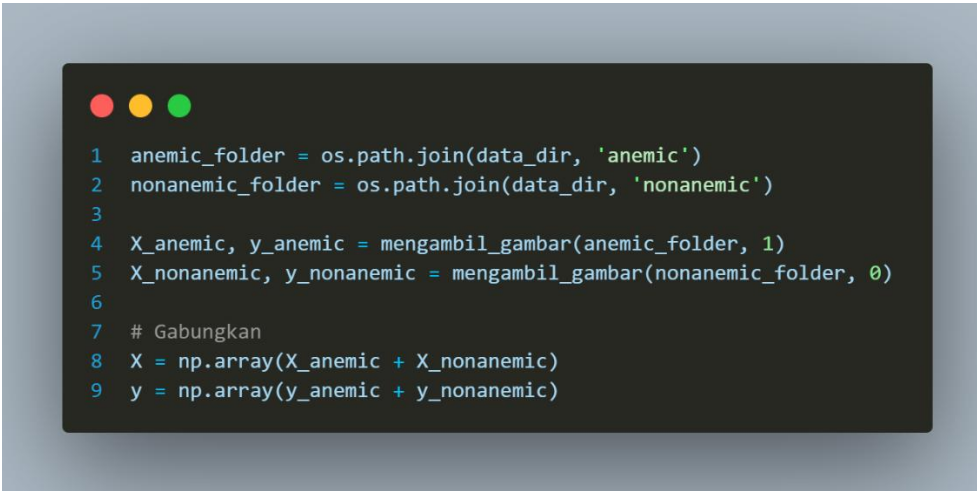
Dataset dimuat menggunakan fungsi `image_dataset_from_directory` dari TensorFlow. Seluruh gambar diresize menjadi ukuran 128x128 piksel untuk menyamakan dimensi input, dan dibagi ke dalam batch sebanyak 32 gambar. Proses ini juga secara otomatis mengenali dua kelas yang tersedia, yaitu anemic dan nonanemic.



```
1 train_dataset = image_dataset_from_directory(
2     data_dir,
3     image_size=(128, 128), # resize semua gambar ke 128x128
4     batch_size=32,
5     label_mode='binary'
6 )
```

- Ekstraksi Gambar dan Label:

Gambar dari masing-masing folder anemic dan nonanemic dibaca satu per satu. Gambar kemudian diubah menjadi array numerik dan dinormalisasi ke dalam rentang 0 hingga 1. Label juga diberikan dalam format biner, di mana 1 untuk kelas anemic dan 0 untuk nonanemic. Proses ini menghasilkan dua buah array: X untuk gambar dan y untuk label.



```
1 anemic_folder = os.path.join(data_dir, 'anemic')
2 nonanemic_folder = os.path.join(data_dir, 'nonanemic')
3
4 X_anemic, y_anemic = mengambil_gambar(anemic_folder, 1)
5 X_nonanemic, y_nonanemic = mengambil_gambar(nonanemic_folder, 0)
6
7 # Gabungkan
8 X = np.array(X_anemic + X_nonanemic)
9 y = np.array(y_anemic + y_nonanemic)
```

- Statistik Dataset:

Setelah proses ekstraksi, jumlah total gambar dalam dataset adalah 4.277 gambar, yang terdiri dari 2.563 gambar anemic dan 1.714 gambar nonanemic. Informasi ini penting untuk mengetahui distribusi awal label dan mengantisipasi potensi ketidakseimbangan kelas (class imbalance).

```
1 print(f"Jumlah gambar: {len(X)}")
2 print(f"Jumlah gambar anemia: {sum(y == 1)}")
3 print(f"Jumlah gambar nonanemia: {sum(y == 0)}")
4 print(f"Label: {np.unique(y, return_counts=True)}")
```

```
Jumlah gambar: 4277
Jumlah gambar anemia: 2563
Jumlah gambar nonanemia: 1714
Label: (array([0, 1]), array([1714, 2563], dtype=int64))
```

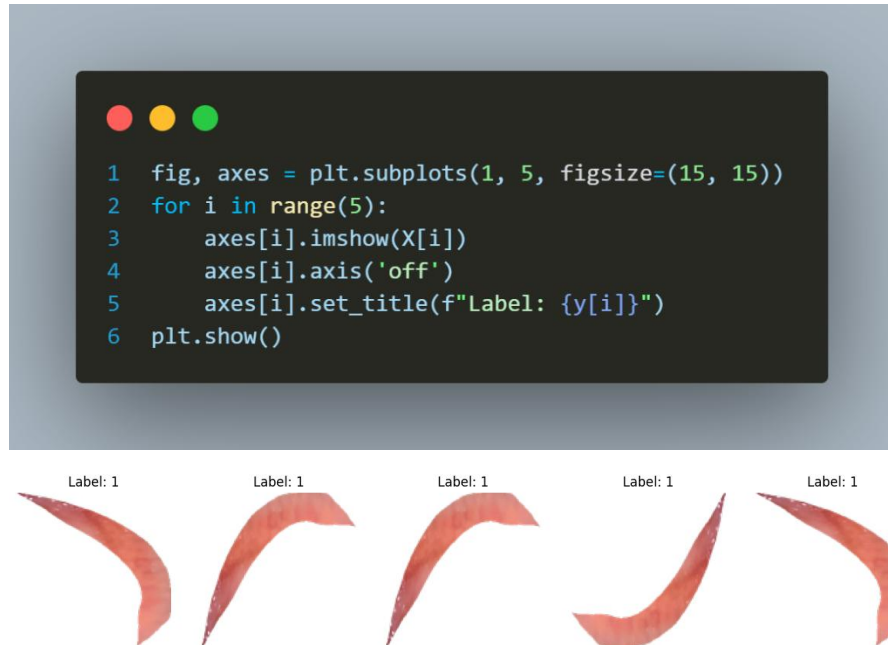
- Melakukan pengecekan gambar rusak

Pengecekan gambar rusak dilakukan untuk memastikan bahwa semua file citra yang digunakan dalam pelatihan model dapat dibaca dan diproses tanpa error. Proses ini mencakup upaya membuka setiap file gambar dalam folder dataset dan melakukan verifikasi menggunakan fungsi `img.verify()` dari library PIL. Jika ditemukan gambar yang tidak dapat dibuka atau rusak formatnya, nama file tersebut akan dicatat agar dapat diperbaiki atau dihapus dari dataset.

```
1 gambar_rusak = []
2 for folder in [anemic_folder, nonanemic_folder]:
3     for filename in os.listdir(folder):
4         if filename.endswith('.png'):
5             try:
6                 img_path = os.path.join(folder, filename)
7                 img = Image.open(img_path) # Cek apakah gambar dapat dibuka
8                 img.verify() # Verifikasi gambar
9             except (IOError, SyntaxError):
10                gambar_rusak.append(filename)
11
12 print(f"Gambar rusak: {gambar_rusak}")
```

- Visualisasi Sample Gambar:

Lima gambar pertama dalam dataset divisualisasikan untuk memberikan gambaran awal bentuk dan warna sel darah. Setiap gambar diberi keterangan label untuk memperkuat pemahaman terhadap perbedaan karakteristik visual antar kelas.



- Distribusi Label:

Visualisasi menggunakan `seaborn.countplot` memperlihatkan perbandingan jumlah antara gambar anemic dan nonanemic. Terlihat bahwa jumlah gambar anemic lebih banyak dibandingkan nonanemic, yang menunjukkan ketidakseimbangan data yang ringan.



- Insight dari EDA:

Dari hasil analisis eksploratif, terlihat bahwa karakteristik visual seperti tingkat kecerahan (brightness), bentuk, dan ukuran sel dapat menjadi indikator kuat dalam membedakan kondisi anemic dan nonanemic. Selain itu, ketidakseimbangan data yang teridentifikasi perlu menjadi perhatian dalam proses training agar model tidak bias terhadap kelas mayoritas.

4. Preprocessing Data

Pada tahap Preprocessing Data, langkah yang dilakukan sebagai berikut.

- Transformasi Gambar dengan flattening dan Normalisasi Gambar

Pada tahap ini, Gambar yang sudah dibaca dan dinormalisasi memiliki dimensi (jumlah gambar x 128 x 128 x 3), sehingga dilakukan perubahan dimensi gambar menjadi vektor 1D menggunakan metode reshape. Selanjutnya, data gambar yang telah diratakan dilakukan penskalaan (scaling) menggunakan StandardScaler untuk menormalkan fitur, memastikan bahwa setiap fitur memiliki rata-rata 0 dan standar deviasi 1, yang sangat penting untuk meningkatkan kinerja model pembelajaran mesin.

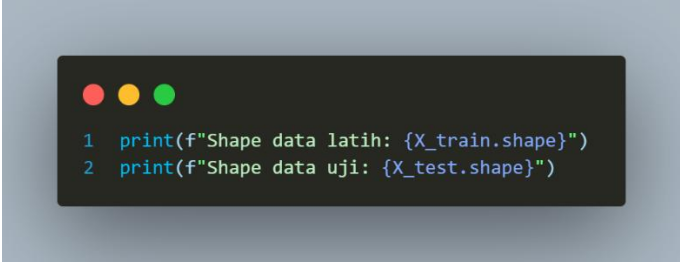
```
1 X_flattened = X.reshape(X.shape[0], -1)
2 X_flattened
```

```
1 scaler = StandardScaler()
2 X_scaled = scaler.fit_transform(X_flattened)
3 X_scaled
```

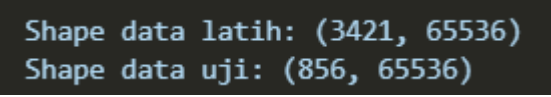
- Split Data

Dataset dibagi menjadi dua bagian: data latih dan data uji. Pembagian ini dilakukan menggunakan fungsi `train_test_split` dengan 80% data untuk pelatihan dan 20% sisanya untuk pengujian. Pembagian ini memastikan bahwa model dapat dilatih menggunakan data yang cukup dan dievaluasi dengan data yang terpisah untuk menghindari overfitting.

```
1 X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)
```



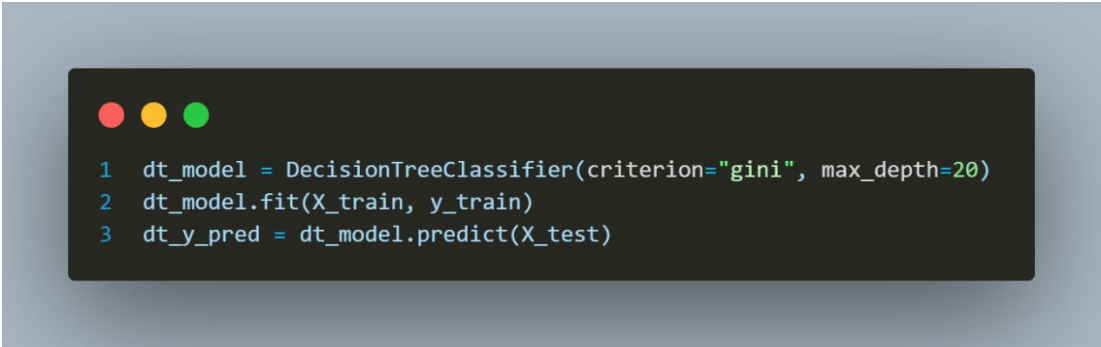
```
1 print(f"Shape data latih: {X_train.shape}")
2 print(f"Shape data uji: {X_test.shape}")
```



```
Shape data latih: (3421, 65536)
Shape data uji: (856, 65536)
```

5. Pemilihan Model dan Training

Pemilihan model dilakukan dengan mempertimbangkan karakteristik dari data serta kebutuhan interpretabilitas dalam konteks medis. Oleh karena itu, algoritma Decision Tree Classifier dipilih sebagai model awal dalam klasifikasi gambar antara kondisi anemic dan nonanemic. Decision Tree menawarkan keuntungan berupa transparansi dalam pengambilan keputusan melalui visualisasi pohon keputusan, serta efisiensi dalam proses pelatihan untuk dataset berukuran sedang.



```
1 dt_model = DecisionTreeClassifier(criterion="gini", max_depth=20)
2 dt_model.fit(X_train, y_train)
3 dt_y_pred = dt_model.predict(X_test)
```

Model dikonstruksi menggunakan parameter kriteria pemisahan "gini" dan kedalaman maksimum (*max_depth*) sebesar 20 untuk mengontrol kompleksitas pohon serta mengurangi risiko overfitting. Model dilatih menggunakan data pelatihan yang telah diproses sebelumnya melalui tahap flattening dan scaling. Setelah pelatihan selesai, prediksi dilakukan terhadap data uji untuk mendapatkan hasil klasifikasi. Hasil ini akan digunakan dalam tahap evaluasi model, termasuk perhitungan akurasi, visualisasi pohon keputusan, confusion matrix, serta kurva ROC.

6. Evaluasi Model

Setelah model dilatih, selanjutnya dilakukan evaluasi model menggunakan beberapa metrik. Berikut tahapan evaluasi model yang dilakukan.

- Akurasi dan classification report

Evaluasi pertama menggunakan akurasi dan laporan klasifikasi, yang memberikan gambaran umum tentang kinerja model dalam mengklasifikasikan kedua kelas (anemic dan nonanemic).

```
1 print(f"Decision Tree - Akurasi: {accuracy_score(y_test, dt_y_pred):.2f}")
2 print("Decision Tree - Classification Report:")
3 print(classification_report(y_test, dt_y_pred))
```

```
Decision Tree - Akurasi: 0.89
Decision Tree - Classification Report:
              precision    recall  f1-score   support

     0       0.84         0.85         0.85         321
     1       0.91         0.90         0.91         535

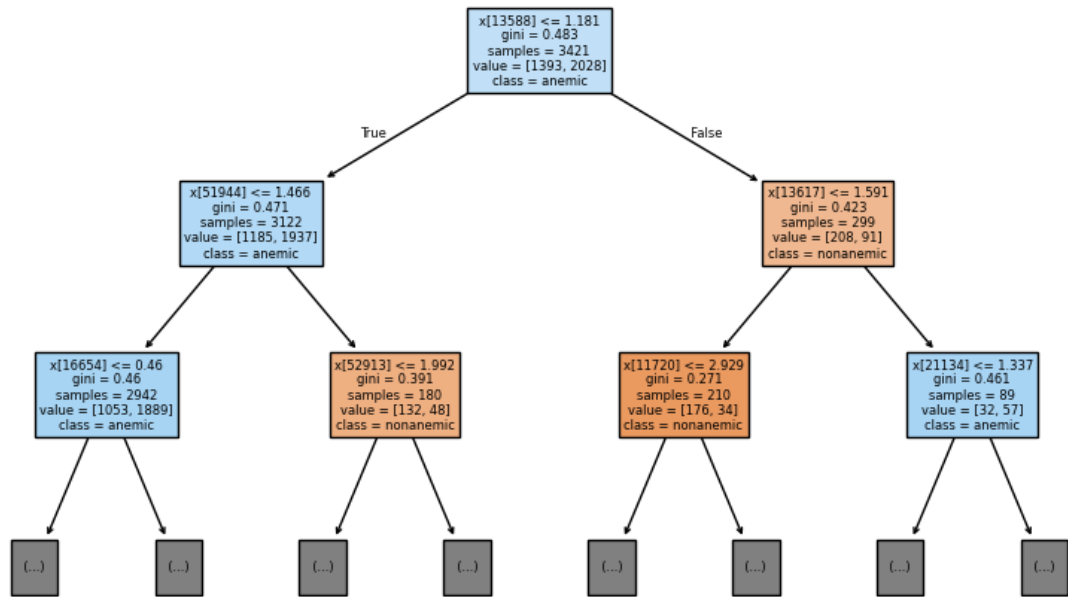
 accuracy          0.89
 macro avg         0.88         0.88         0.88         856
weighted avg         0.89         0.89         0.89         856
```

- Visualisasi pohon Keputusan

Visualisasi pohon keputusan dilakukan untuk memahami lebih lanjut bagaimana model membuat keputusan berdasarkan fitur-fitur input. Pohon keputusan yang divisualisasikan menggambarkan pembagian data pada setiap level dan kriteria keputusan yang digunakan pada setiap node.

```
1 plt.figure(figsize=(10, 6))
2 plot_tree(dt_model, filled=True, max_depth=2, class_names=['nonanemic', 'anemic'])
3 plt.title("Pohon Keputusan (Sebagian)")
4 plt.show()
5
```

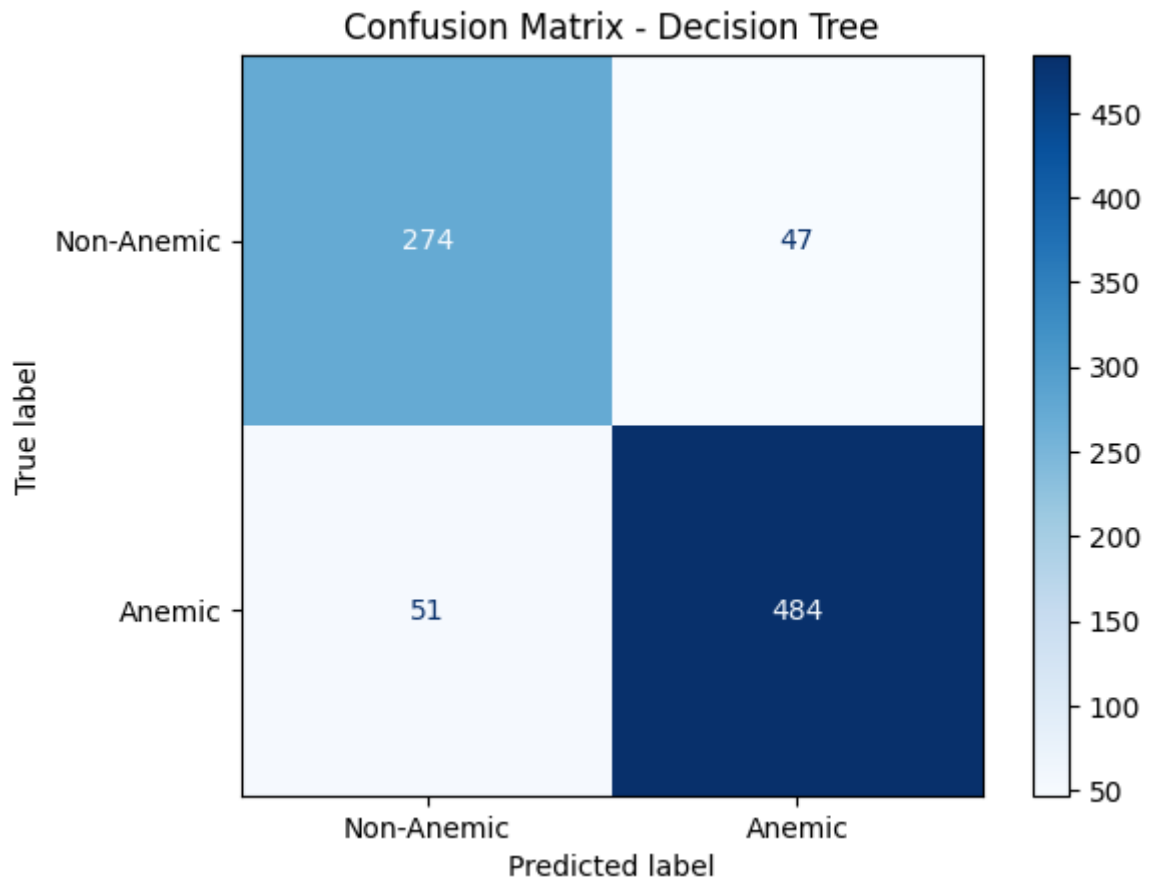

Pohon Keputusan (Sebagian)



- Confusion Matrix

Selain itu, evaluasi juga dilakukan dengan menggunakan Confusion Matrix, yang menunjukkan perbandingan antara prediksi model dan label yang sebenarnya. Hal ini memungkinkan kita untuk mengidentifikasi jenis kesalahan yang terjadi (misalnya, false positives atau false negatives).

```
1 cm = confusion_matrix(y_test, dt_y_pred)
2 disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=['Non-Anemic', 'Anemic'])
3 disp.plot(cmap='Blues')
4 plt.title("Confusion Matrix - Decision Tree")
5 plt.show()
6
```

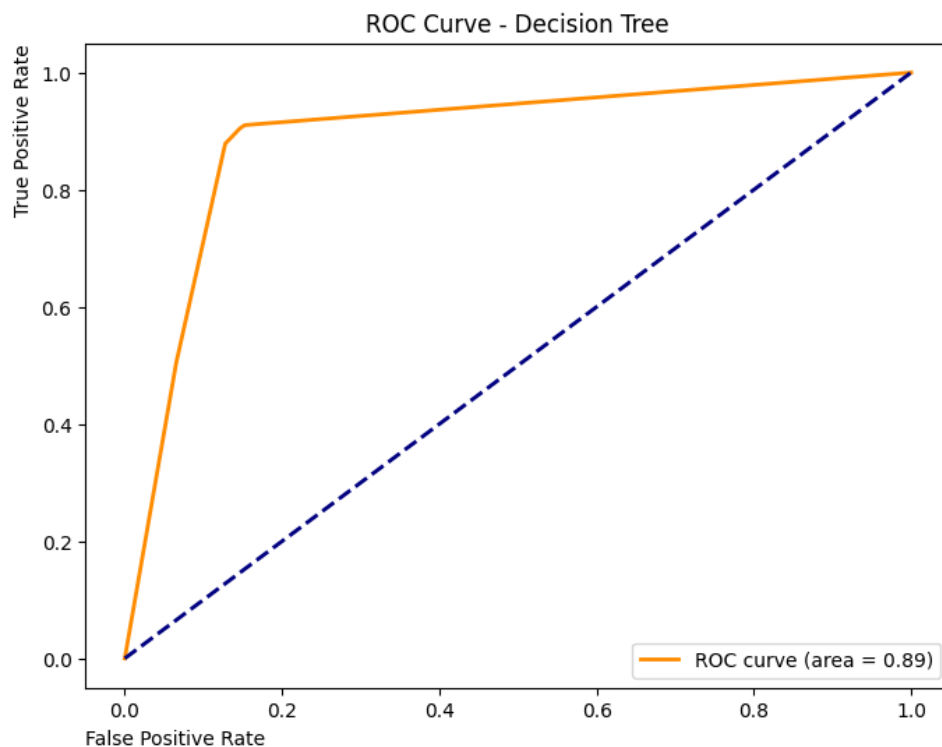


Berdasarkan confusion matrix tersebut, model Decision Tree menunjukkan performa yang baik dengan akurasi tinggi, di mana dari total prediksi ($274+47+51+484=856$), model berhasil mengklasifikasikan 274 non-anemia (true negative) dan 484 anemia (true positive) dengan benar, namun masih terdapat 47 false positive (non-anemia diprediksi sebagai anemia) dan 51 false negative (anemia diprediksi sebagai non-anemia). Nilai presisi dan recall untuk kelas anemia cenderung tinggi karena jumlah true positive (484) jauh lebih besar dibanding false positive (47), sementara untuk kelas non-anemia, false negative (51) relatif lebih tinggi dibanding false positive (47), menunjukkan bahwa model sedikit lebih baik dalam mendeteksi anemia daripada non-anemia. Secara keseluruhan, model ini cukup andal tetapi dapat dioptimalkan lebih lanjut untuk mengurangi kesalahan klasifikasi, terutama pada false negative yang berpotensi lebih kritis dalam konteks medis.

- ROC Curve

Kurva ROC (Receiver Operating Characteristic) juga digunakan untuk mengevaluasi performa model dalam membedakan antara kedua kelas. Area under the curve (AUC) dihitung untuk mengetahui seberapa baik model dapat membedakan antara kelas *anemic* dan *nonanemic*.

```
1 fpr, tpr, thresholds = roc_curve(y_test, dt_model.predict_proba(X_test)[: , 1])
2 roc_auc = auc(fpr, tpr)
3
4 plt.figure(figsize=(8, 6))
5 plt.plot(fpr, tpr, color='darkorange', lw=2, label=f'ROC curve (area = {roc_auc:.2f})')
6 plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
7 plt.xlabel('False Positive Rate', loc='left')
8 plt.ylabel('True Positive Rate', loc='top')
9 plt.title('ROC Curve - Decision Tree')
10 plt.legend(loc='lower right')
11 plt.grid(False)
12 plt.show()
13
```



Pada gambar tersebut, analisis yang didapat adalah model Decision Tree memiliki performa klasifikasi yang sangat baik dengan nilai AUC sebesar 0.89, menunjukkan kemampuan yang kuat dalam membedakan kelas positif dan negatif. Kurva ROC yang cembung ke kiri atas dan berada jauh di atas garis diagonal (AUC =

0.5) mengindikasikan bahwa model mencapai True Positive Rate (TPR) yang relatif tinggi (misalnya 0.6 pada FPR 0.2) sambil mempertahankan False Positive Rate (FPR) yang rendah, sehingga menyeimbangkan antara sensitivitas dan spesifisitas dengan efektif. Hasil ini menyarankan bahwa model sudah optimal, namun masih dapat ditingkatkan melalui teknik seperti tuning hyperparameter atau ensemble methods jika diperlukan.

7. Kesimpulan

Adapun kesimpulan dari laporan teknis ini sebagai berikut:

1. Model Decision Tree menunjukkan kinerja yang baik dalam mengklasifikasikan sel darah merah anemia dan non-anemia dengan akurasi tinggi (AUC 0.89). Hal ini ditunjukkan melalui confusion matrix yang memiliki 484 true positive dan 274 true negative, meskipun masih terdapat 47 false positive dan 51 false negative.
2. Kelebihan utama model ini terletak pada kemudahan interpretasi melalui visualisasi pohon keputusan serta efisiensi komputasi yang cocok untuk dataset berukuran sedang. Selain itu, ROC curve (AUC 0.89) membuktikan kemampuan model dalam menyeimbangkan sensitivitas (TPR) dan spesifisitas (FPR).
3. Terdapat beberapa keterbatasan, antara lain:
 - Ketidakseimbangan kelas (lebih banyak data anemia) yang berpotensi menyebabkan bias.
 - False negative (51 kasus) yang dapat berisiko dalam diagnosis medis.
 - Dependensi pada fitur manual (warna, tekstur, morfologi) yang mungkin kurang optimal dibanding pendekatan deep learning.
4. Rekomendasi untuk pengembangan lebih lanjut meliputi:
 - Optimasi hyperparameter (seperti max_depth dan min_samples_split) untuk mengurangi overfitting.
 - Penerapan ensemble methods (misalnya Random Forest atau XGBoost) guna meningkatkan akurasi.
 - Eksplorasi arsitektur CNN untuk ekstraksi fitur otomatis dari gambar mikroskopik.
 - Penambahan data non-anemia atau teknik class weighting/SMOTE untuk menangani ketidakseimbangan kelas.

Secara keseluruhan, model ini telah membuktikan potensinya sebagai alat bantu

diagnosis anemia berbasis citra mikroskopik. Namun, uji validasi klinis dan pengembangan lebih lanjut diperlukan sebelum implementasi skala besar.

8. Referensi

- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). *Scikit-learn: Machine learning in Python*. Journal of Machine Learning Research, 12, 2825–2830.
Tersedia di: <https://scikit-learn.org/stable/>
- TensorFlow. (2024). *image_dataset_from_directory API Documentation*. TensorFlow Official Documentation.
Tersedia di: https://www.tensorflow.org/api_docs/python/tf/keras/utils/image_dataset_from_directory
- Google AI Research. (n.d.). *Technical Report Style Examples*.
Tersedia di: <https://research.google.com/pubs/>
- Overleaf. (n.d.). *Template for a Technical Report*.
Tersedia di: <https://www.overleaf.com/latex/templates/template-for-a-technical-report/>
- GitHub Docs. (n.d.). *About READMEs*.
Tersedia di: <https://docs.github.com/en/repositories/managing-your-repositorys-settings-and-features/customizing-your-repository/about-readmes>