Technical Report: Klasifikasi Gambar Menggunakan Support Vector Machine

Praktikum Machine Learning

Nama Anggota 1 : Amanda Putri Aprilliani (105222001)

Nama Anggota 2 : Anom Wajawening (105222035)

Program Studi : Ilmu Komputer

---

1. Pendahuluan

1.1 Latar Belakang

Anemia adalah kondisi medis yang ditandai dengan kurangnya sel darah merah yang sehat untuk membawa oksigen ke jaringan tubuh. Deteksi dini anemia sangat penting untuk penanganan yang tepat dan mencegah komplikasi. Metode konvensional memerlukan pemeriksaan laboratorium yang membutuhkan waktu dan tenaga ahli. Dengan kemajuan teknologi *machine learning*, terdapat adanya peluang untuk mengembangkan sistem otomatis yang dapat membantu proses skrining awal anemia melalui analisis gambar sel darah.

Proyek ini mengembangkan sistem klasifikasi gambar untuk membedakan gambar sel darah *anemic* dan *non-anemic* menggunakan algoritma Support Vector Machine (SVM). SVM dipilih karena kemampuannya dalam menangani data berdimensi tinggi dan efisiensi komputasionalnya, yang menjadikannya pilihan yang tepat untuk klasifikasi gambar medis dengan dataset berukuran menengah.

1.2 Tujuan

Tujuan dari proyek ini adalah:

1. Mengembangkan model klasifikasi gambar yang dapat membedakan gambar sel darah *anemic* dan *non-anemic* dengan akurasi tinggi.
2. Melakukan eksplorasi dan analisis karakteristik visual yang membedakan gambar *anemic* dan *non-anemic*.
3. Mengevaluasi performa model dengan berbagai metrik untuk menilai keefektifannya sebagai alat skrining anemia.
4. Mengidentifikasi faktor-faktor yang berkontribusi terhadap akurasi klasifikasi untuk pengembangan model selanjutnya.

1.3 Deskripsi Dataset

Dataset terdiri dari gambar *inner eyelid* dikategorikan menjadi dua kelas:

- Jumlah gambar *anemic*: 2563

- Jumlah gambar *non-anemic*: 1714
- Total gambar: 4277
- Rasio kelas: 60% *anemic* dan 40% *non-anemic*

Gambar memiliki format PNG dengan dimensi yang bervariasi, sehingga diperlukan langkah standardisasi ukuran pada tahap preprocessing.

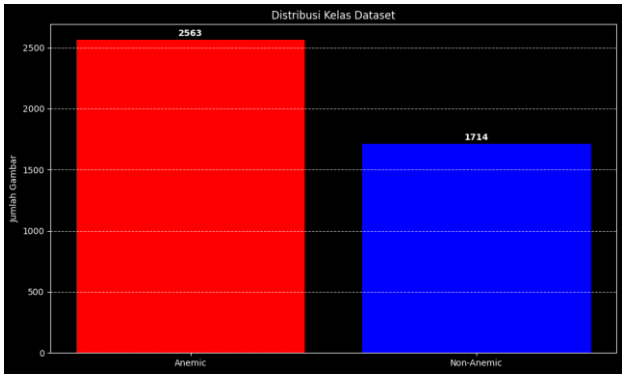
2. Exploratory Data Analysis (EDA)

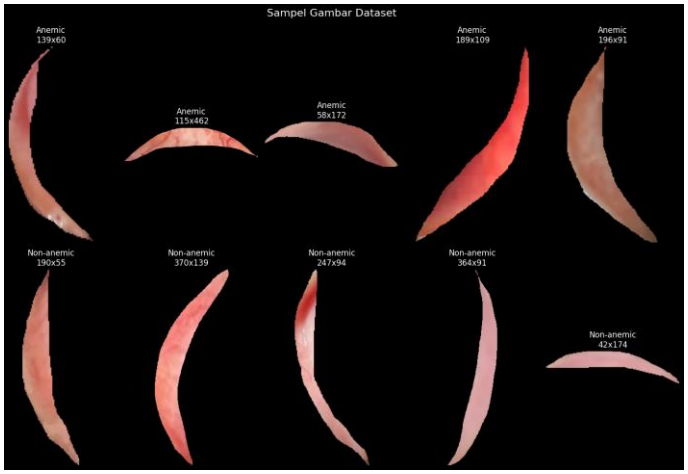
```
print ("===== DESKRIPSI DATASET =====")
num_anemic = len(os.listdir(anemic_folder))
num_nonanemic = len(os.listdir(
nonanemic_folder))
total_images = num_anemic + num_nonanemic

print ("Jumlah gambar anemic: ", num_anemic)
print ("Jumlah gambar non-anemic: ",
num_nonanemic)
print ("Total gambar: ", total_images)
print(f"Rasio anemic:nonanemic = {num_anemic/
total_images:.2f}:{num_nonanemic/total_images
```

2.1 Distribusi Kelas

Distribusi kelas dalam dataset menunjukkan ketidakseimbangan ringan, dengan 60% gambar *anemic* dan 40% *non-anemic*. Visualisasi distribusi ini penting untuk memahami struktur dataset dan potensi bias.





2.2 Visualisasi Sampel

Beberapa sampel dari masing-masing kelas divisualisasikan untuk membantu identifikasi pola visual yang membedakan gambar *anemic* (cenderung pucat dan homogen) dengan *non-anemic* (lebih bervariasi).

2.3 Analisis Dimensi

```python
# Analisis dimensi dan format gambar diperlukan untuk memahami variasi ukuran dan format gambar dalam dataset.

print("\n===== ANALISIS DIMENSI DAN FORMAT =====")

# Fungsi untuk menganalisis dimensi dan format gambar
def analyze_image_dimensions(folder_path, limit=100):
    widths, heights = [], []
    formats = {}
    sizes = []

    files = os.listdir(folder_path)[:limit]
    for filename in tqdm(files, desc=f"Menganalisis gambar di {folder_path}"):
        file_path = os.path.join(folder_path, filename)
        try:
            # Mendapatkan format gambar
            format_type = filename.split('.')[-1].lower()
            formats[format_type] = formats.get(format_type, 0) + 1

            # Mendapatkan dimensi
            img = Image.open(file_path)
            width, height = img.size
            widths.append(width)
            heights.append(height)

            # Mendapatkan ukuran file
            size_kb = os.path.getsize(file_path) / 1024  # KB
            sizes.append(size_kb)

        except Exception as e:
            print(f"Error dengan file {filename}: {e}")

    return {
        'widths': widths,
        'heights': heights,
        'formats': formats,
        'sizes': sizes
    }

# Menganalisis subset dari kedua folder
anemic_stats = analyze_image_dimensions(anemic_folder, limit=100)
nonanemic_stats = analyze_image_dimensions(nonanemic_folder, limit=100)

# Menampilkan statistik dimensi
print("\nStatistik Dimensi Gambar Anemic:")
print(f"Width - Min: {min(anemic_stats['widths'])}, Max: {max(anemic_stats['widths'])}, Mean: {np.mean(anemic_stats['widths']):.2f}")
print(f"Height - Min: {min(anemic_stats['heights'])}, Max: {max(anemic_stats['heights'])}, Mean: {np.mean(anemic_stats['heights']):.2f}")
print(f"Ukuran File (KB) - Min: {min(anemic_stats['sizes']):.2f}, Max: {max(anemic_stats['sizes']):.2f}, Mean: {np.mean(anemic_stats['sizes']):.2f}")

print("\nStatistik Dimensi Gambar Non-anemic:")
print(f"Width - Min: {min(nonanemic_stats['widths'])}, Max: {max(nonanemic_stats['widths'])}, Mean: {np.mean(nonanemic_stats['widths']):.2f}")
print(f"Height - Min: {min(nonanemic_stats['heights'])}, Max: {max(nonanemic_stats['heights'])}, Mean: {np.mean(nonanemic_stats['heights']):.2f}")
print(f"Ukuran File (KB) - Min: {min(nonanemic_stats['sizes']):.2f}, Max: {max(nonanemic_stats['sizes']):.2f}, Mean: {np.mean(nonanemic_stats['sizes']):.2f}")

print("\nFormat Gambar Anemic:")
for fmt, count in anemic_stats['formats'].items():
    print(f"{fmt}: {count}")

print("\nFormat Gambar Non-anemic:")
for fmt, count in nonanemic_stats['formats'].items():
    print(f"{fmt}: {count}")
```

Ukuran gambar dalam dataset bervariasi cukup signifikan. Berdasarkan analisis terhadap 100 sampel gambar yang diambil dari masing-masing kelas, diperoleh informasi sebagai berikut:

- **Gambar Anemic:**
    - Lebar (Width): Min = 87 piksel, Max = 462 piksel, Rata-rata = 267.67 piksel

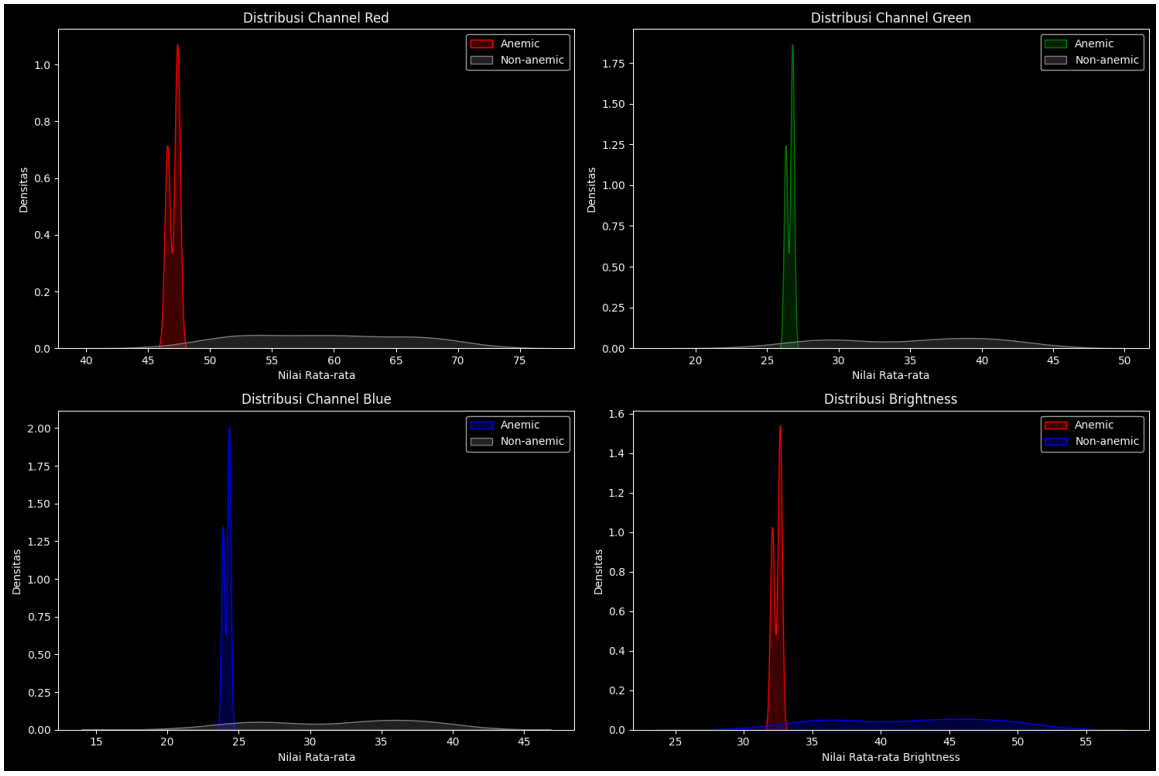
- Tinggi (Height): Min = 87 piksel, Max = 462 piksel, Rata-rata = 260.73 piksel
- Ukuran file: Min = 11.45 KB, Max = 39.99 KB, Rata-rata = 28.20 KB

- **Gambar Non-anemic:**
  - Lebar (Width): Min = 49 piksel, Max = 499 piksel, Rata-rata = 194.36 piksel
  - Tinggi (Height): Min = 49 piksel, Max = 499 piksel, Rata-rata = 193.68 piksel
  - Ukuran file: Min = 4.96 KB, Max = 58.80 KB, Rata-rata = 18.60 KB

Dari statistik ini terlihat gambar bervariasi dari 49x49 hingga 499x499 piksel, Terlihat bahwa gambar anemic memiliki ukuran rata-rata yang lebih besar dibandingkan gambar non-anemic sehingga diperlukan standardisasi ukuran sebelum pemodelan.

Selain itu, semua gambar dalam kedua kelas menggunakan format PNG, sehingga tidak diperlukan konversi format dalam preprocessing.

2.4 Analisis Warna



Hasil visualisasi :

- Gambar anemic memiliki distribusi warna yang lebih kecil dan cenderung terkonsentrasi pada rentang intensitas warna tertentu di tiap channel RGB.

- Gambar non-anemic memperlihatkan distribusi warna yang lebih luas dan menyebar secara merata di seluruh channel warna.

Hal ini memperkuat asumsi bahwa warna merupakan fitur visual yang cukup kuat dalam membedakan kedua kelas. Terutama pada tingkat kecerahan, terlihat perbedaan yang jelas di mana gambar anemic cenderung memiliki *brightness* yang lebih rendah dan homogen dibandingkan gambar non-anemic.

2.5 Insight EDA

Berdasarkan analisis eksploratori terhadap dataset gambar *inner eyelid*, diperoleh sejumlah *insight* penting yang dapat digunakan sebagai dasar pengambilan keputusan dalam preprocessing dan pemodelan:

1. Ketidakseimbangan Kelas Ringan
    - Dataset terdiri dari 60% gambar kelas anemic dan 40% non-anemic.
    - Meskipun tidak ekstrem, ketidakseimbangan ini perlu diperhatikan agar model tidak bias terhadap kelas mayoritas.
2. Perbedaan Ukuran Gambar yang Signifikan
    - Gambar anemic memiliki rata-rata dimensi lebih besar (rata-rata 267x260 piksel) dibandingkan non-anemic (rata-rata 194x193 piksel).
    - Ukuran file juga menunjukkan bahwa gambar anemic lebih berat secara storage, dengan rata-rata 28.20 KB dibandingkan 18.60 KB.
    - Hal ini menunjukkan bahwa dimensi gambar bisa menjadi indikasi tidak langsung dari kelas, namun perlu distandarisasi agar tidak mempengaruhi model secara tidak adil.
3. Format Gambar Seragam
    - Seluruh gambar dalam dataset menggunakan format PNG.
    - Konsistensi ini mempermudah tahap preprocessing karena tidak diperlukan konversi atau penyesuaian format file.
4. Distribusi Warna sebagai Fitur Pembeda
    - Gambar anemic menunjukkan pola distribusi warna yang lebih sempit dan homogen, terutama pada tingkat kecerahan.
    - Gambar non-anemic memiliki sebaran warna yang lebih luas di semua channel RGB.

- Fitur warna terbukti sangat potensial untuk digunakan dalam klasifikasi, bahkan sebelum dilakukan teknik ekstraksi fitur lanjutan.

5. Konsistensi Visual Kelas Anemic

    - Dari visualisasi sampel, gambar anemic cenderung memiliki tampilan yang seragam: warna pucat, sedikit variasi, dan bentuk sel yang relatif homogen.
    - Sebaliknya, gambar non-anemic menunjukkan lebih banyak keragaman dari segi warna dan struktur sel.
    - Hal ini membuat gambar anemic lebih mudah untuk dipelajari oleh model klasifikasi karena konsistensi visual yang lebih tinggi.

---

3. Preprocessing

Preprocessing adalah tahap krusial dalam *machine learning*, khususnya untuk data visual. Langkah ini bertujuan untuk mengubah gambar mentah menjadi representasi numerik yang konsisten dan siap diproses oleh model klasifikasi. Berikut adalah penjabaran setiap tahap preprocessing yang dilakukan:

3.1 Resizing Gambar

Langkah pertama adalah **mengubah ukuran seluruh gambar ke dimensi tetap**, yaitu 224x224 piksel. Tujuannya adalah agar semua citra memiliki dimensi seragam, sehingga mempermudah proses input ke model machine learning.

```
# Resizing gambar ini digunakan untuk mengubah ukuran semua gambar menjadi ukuran yang seragam. Hal ini penting untuk memastikan bahwa semua gambar memiliki dimensi yang sama sebelum diproses lebih lanjut.

def resize_image(image, target_size=(224, 224)):
    """
    Mengubah ukuran semua gambar menjadi ukuran yang seragam
    """
    return cv2.resize(image, target_size)
```

3.2 Normalisasi Nilai Piksel

Gambar digital disimpan dalam format RGB dengan nilai piksel 0–255 untuk setiap channel warna. Nilai ini perlu **dinormalisasi ke rentang 0–1** agar lebih mudah diproses oleh model, serta mencegah dominasi nilai numerik besar yang dapat menyebabkan bias dalam perhitungan bobot selama pelatihan.

fungsi yang digunakan:

```python
#
 Normalisasi gambar ini digunakan untuk mengubah nilai piksel gambar menjadi rentang 0-1. Hal ini penting untuk memastikan bahwa model pembelajaran mesin dapat belajar dengan baik dari data.
#
 Secara normal, nilai piksel dalam gambar digital memiliki rentang 0-255 di setiap channel warna (Red, Green, Blue). Nilai 0 berarti tidak ada intensitas warna (hitam), dan 255 berarti intensitas warna penuh.

def normalize_image(image):
    """
    Normalisasi nilai piksel ke rentang 0-1
    """
    return image / 255.0
```

3.3 Ekstraksi Fitur

Gambar diresize dan dinormalisasi, dilakukan proses *flattening*, yaitu mengubah representasi gambar dari format matriks 3 dimensi (64x64x3) menjadi vektor 1 dimensi. Dengan Jumlah sampel 4277 ini menghasilkan total 12.288 fitur per gambar (64×64×3).

Langkah ini penting karena model SVM konvensional seperti LinearSVC hanya menerima input dalam bentuk vektor 1D.

```
#
 Ekstraksi fitur pada kasus ini digunakan untuk mengubah gambar m
enjadi angka-angka
# yang bisa dipahami oleh model SV
 M

def simple_preprocess(anemic_folder, nonanemic_folder,
target_size=(64, 64)):
    X = []  # fitu
    y = []  #rlabe
             l
    # Proses gambar anemic
    print("Processing anemic images...")
    for filename in o .listdir(anemic_folder):
        try:       s
            file_path = o .path.join(anemic_folder, filename)
            img = cv2.imread(file_path)

            if img is not None:
                # Resize dan flatte
                img_resized = cv2.resize(img, target_size)
                img_flat = img_resized.flatten() / 255.0  #
 Normaliz
e
                X.append(img_flat)
                y.append(1)  # Anemic
        except Exception as  :
            print(f"Error peocessing {filename}: {e}")

    # Proses gambar nonanemic
    print("Processing nonanemic images...")
    for filename in o .listdir(nonanemic_folder):
        try:       s
            file_path = o .path.join(nonanemic_folder, filename)
            img = cv2.imread(file_path)

            if img is not None:
                img_resized = cv2.resize(img, target_size)
                img_flat = img_resized.flatten() / 255.0

                X.append(img_flat)
                y.append(0)  # Nonanemi
        except Exception as  :c
            print(f"Error peocessing {filename}: {e}")

    return n .array(X), n .array(y)
          p           p
# Coba dengan ukuran gambar kecil untuk mempercepat proses
import numpy as n
X, y = simple_ppeprocess('anemic', 'nonanemic', target_size=(64,
64))

print(f"Jumlah sampel: {X.shape[0]}")
print(f"Jumlah fitur: {X.shape[1]}")
```

3.4 Train-Test Split

```
# Pembagian dataset ini digunakan untuk membagi dataset menjadi dua bagian: satu untuk pelatihan
model dan satu untuk pengujian model. Hal ini penting untuk memastikan bahwa model dapat diuji pa
da data yang belum pernah dilihat sebelumnya.
# Disini kita menggunakan 80% data untuk pelatihan dan 20% untuk pengujian.

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y
)

print(f"Train set: {X_train.shape[0]} samples")
print(f"Test set: {X_test.shape[0]} samples")
```

Dataset dibagi menjadi:

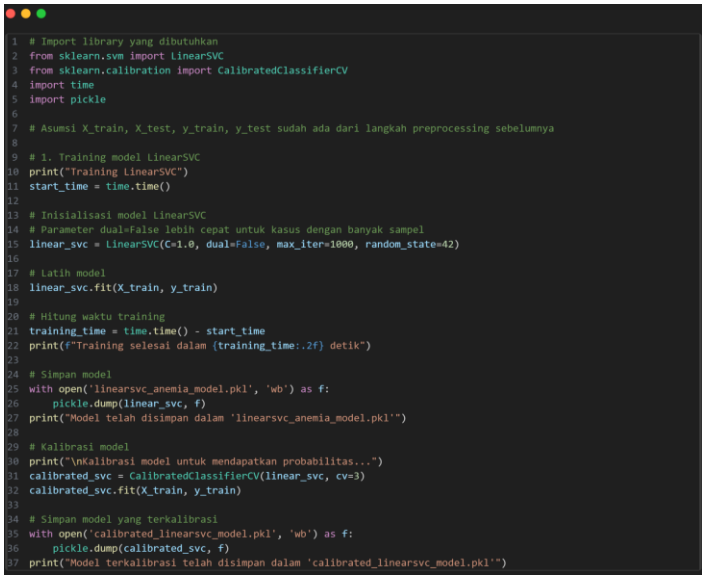
- Training set: 3421 gambar
- Testingset : 856 gambar

Pembagian dilakukan dengan *stratified sampling* agar distribusi kelas tetap seimbang .

- 1 untuk gambar **anemic**
- 0 untuk gambar **non-anemic**

---

4. Pemodelan dan Evaluasi



```python
# Import library yang dibutuhkan
from sklearn.svm import LinearSVC
from sklearn.calibration import CalibratedClassifierCV
import time
import pickle

# Asumsi X_train, X_test, y_train, y_test sudah ada dari langkah preprocessing sebelumnya

# 1. Training model LinearSVC
print("Training LinearSVC")
start_time = time.time()

# Inisialisasi model LinearSVC
# Parameter dual=False lebih cepat untuk kasus dengan banyak sampel
linear_svc = LinearSVC(C=1.0, dual=False, max_iter=1000, random_state=42)

# Latih model
linear_svc.fit(X_train, y_train)

# Hitung waktu training
training_time = time.time() - start_time
print(f"Training selesai dalam {training_time:.2f} detik")

# Simpan model
with open('linearsvc_anemia_model.pkl', 'wb') as f:
    pickle.dump(linear_svc, f)
print("Model telah disimpan dalam 'linearsvc_anemia_model.pkl'")

# Kalibrasi model
print("\nKalibrasi model untuk mendapatkan probabilitas...")
calibrated_svc = CalibratedClassifierCV(linear_svc, cv=3)
calibrated_svc.fit(X_train, y_train)

# Simpan model yang terkalibrasi
with open('calibrated_linearsvc_model.pkl', 'wb') as f:
    pickle.dump(calibrated_svc, f)
print("Model terkalibrasi telah disimpan dalam 'calibrated_linearsvc_model.pkl'")
```

4.1 Model: LinearSVC

Model yang digunakan adalah LinearSVC dari *scikit-learn*. Parameter yang digunakan:

- C=1.0
- dual=False
- max_iter=1000
- random_state=42

Model dikalibrasi dengan CalibratedClassifierCV untuk menghasilkan prediksi probabilitas.

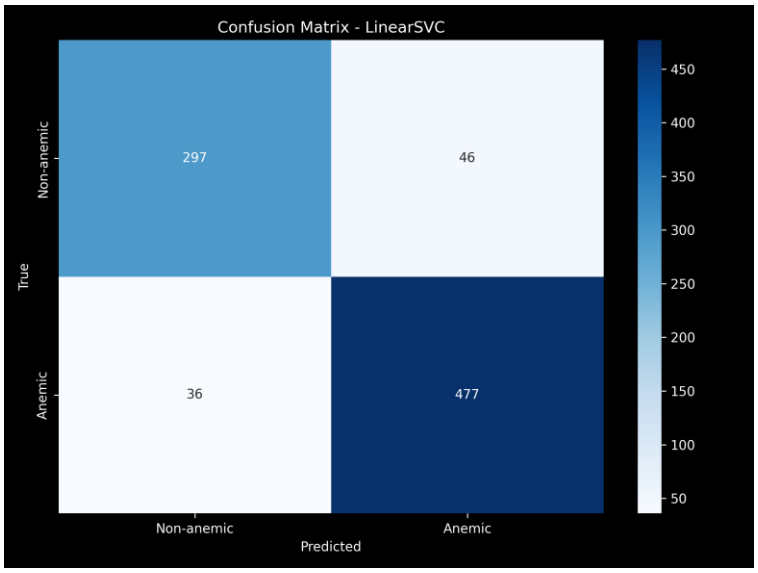
4.2 Evaluasi Model

Hasil evaluasi:

- Akurasi: **90.42%**
- Precision: *Anemic* (91%), *Non-anemic* (89%)

- Recall: *Anemic* (93%), *Non-anemic* (87%)
- F1-Score: *Anemic* (92%), *Non-anemic* (88%)

Confusion Matrix:



- True Positives (TP): 477
- True Negatives (TN): 297
- False Positives (FP): 46
- False Negatives (FN): 36

Model memiliki performa seimbang dan akurasi tinggi, dengan kemampuan yang baik untuk mengklasifikasikan kedua kelas.

---

5. Insight, Kesimpulan, dan Referensi

5.1 Insight

1. SVM linear efektif bahkan dengan fitur sederhana.
2. Distribusi warna sangat membantu klasifikasi.
3. Preprocessing sederhana (resize dan normalisasi) cukup kuat.
4. Kalibrasi model memberikan probabilitas prediksi yang bermakna.
5. Jumlah false negative lebih sedikit dari false positive — positif dalam konteks medis.

5.2 Kesimpulan

Berdasarkan seluruh tahapan analisis dan pengujian yang telah dilakukan, dapat disimpulkan bahwa pendekatan klasifikasi menggunakan LinearSVC berhasil memberikan hasil yang sangat baik dalam mendeteksi anemia melalui gambar sel darah. Dengan pendekatan sederhana namun terstruktur, kesimpulan penting sebagai berikut:

1. **Performa Model LinearSVC yang Tinggi**
   Model klasifikasi yang dibangun menggunakan algoritma Linear Support Vector Classifier (LinearSVC) mampu mencapai akurasi sebesar 90,42%. Nilai precision, recall, dan F1-score yang diperoleh menunjukkan bahwa model memiliki kapabilitas yang baik dalam membedakan antara gambar sel darah anemik dan non-anemik secara konsisten. Hal ini mencerminkan bahwa pendekatan berbasis SVM dapat diandalkan untuk tugas klasifikasi biner pada citra medis.

2. **Warna sebagai Fitur Visual Dominan**
   Hasil eksplorasi data menunjukkan bahwa distribusi warna, khususnya pada saluran RGB dan tingkat kecerahan (brightness), menjadi indikator visual utama dalam membedakan antara kedua kelas gambar. Gambar pada kelas anemik cenderung memiliki distribusi warna yang lebih sempit dan homogen, sementara gambar non-anemik lebih bervariasi. Perbedaan karakteristik ini dimanfaatkan secara efektif oleh model dalam proses klasifikasi.

3. **Potensi Model sebagai Alat Skrining Awal**
   Dengan akurasi yang tinggi dan tingkat kesalahan prediksi negatif yang relatif rendah, model ini dinilai layak untuk dikembangkan lebih lanjut sebagai sistem pendukung keputusan dalam skrining awal anemia. Kemampuan model dalam mendeteksi kasus positif (anemik) secara akurat sangat penting dalam konteks medis, terutama untuk pencegahan dan penanganan dini.

4. **Relevansi Pendekatan Klasik dalam Konteks Medis.**
   Meskipun tidak menggunakan pendekatan deep learning yang kompleks, metode klasifikasi klasik seperti SVM tetap menunjukkan kinerja yang kompetitif dan efisien, khususnya pada dataset berukuran menengah dengan struktur citra yang relatif homogen. Hal ini menegaskan bahwa pendekatan klasik masih sangat relevan dan dapat menjadi solusi yang tepat dalam pengembangan sistem berbasis machine learning pada bidang kesehatan, terutama ketika sumber daya komputasi terbatas.

5.3 Referensi

1. Github atau Dokumentasi Project : https://github.com/mandayash/ImageClassification-SVM

2. Banerjee, P. (2020). Support Vector Machines (SVM) Classifier Tutorial with Python.

Kaggle Notebooks. Available online : https://www.kaggle.com/code/prashant111/svm-classifier-tutorial/notebook#5.-Dataset-description-

3. Shanmukh, V. (2021, March 4). Image Classification Using Machine Learning-Support Vector Machine(SVM). Analytics Vidhya. Available Online : https://medium.com/analytics-vidhya/image-classification-using-machine-learning-support-vector-machine-svm-dc7a0ec92e01

4. Saksham, K. (2022). Cat And Dog Image Classification Using SVM. Kaggle Notebooks. Available online: https://www.kaggle.com/code/everydaycodings/cat-and-dog-image-classification-using-svm#Creating-the-Model