LAPORAN TUGAS AKHIR

”KLASIFIKASI PENYAKIT DAUN PADI MENGGUNAKAN EKSTRAKSI FITUR DENSENET201 DAN ALGORITMA XGBOOST”

MACHINE LEARNING

Disusun oleh:

Aliefian Ramadhan 22081010171

PROGRAM STUDI INFORMATIKA

FAKULTAS ILMU KOMPUTER

UNIVERSITAS PEMBANGUNAN ”VETERAN”

JAWA TIMUR

2025

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Padi (*Oryza sativa*) merupakan komoditas pangan strategis yang menjadi makanan pokok bagi lebih dari separuh populasi dunia, khususnya di Indonesia. Namun, produktivitas padi sering kali terhambat oleh serangan organisme pengganggu tanaman (OPT), terutama penyakit pada daun. Tiga penyakit utama yang kerap menurunkan kualitas dan kuantitas hasil panen adalah *Brown Spot*, *Hispa*, dan *Leaf Blast*. Deteksi dini terhadap penyakit ini sangat krusial untuk menentukan langkah pengendalian yang presisi, guna meminimalisir kerugian ekonomi yang dialami petani.

Secara konvensional, identifikasi penyakit daun padi dilakukan melalui pengamatan visual langsung oleh pakar pertanian atau penyuluh lapangan. Metode ini memiliki keterbatasan signifikan, antara lain subjektivitas pengamat, keterbatasan tenaga ahli di lapangan, serta waktu diagnosis yang relatif lama. Kesalahan diagnosis pada tahap awal dapat berakibat fatal pada penyebaran wabah penyakit yang lebih luas. Oleh karena itu, dibutuhkan sistem otomatisasi yang mampu mendiagnosis penyakit dengan cepat, objektif, dan akurat.

Perkembangan teknologi kecerdasan buatan, khususnya *Deep Learning*, telah menunjukkan performa superior dalam tugas pengenalan citra. *Convolutional Neural Network* (CNN) merupakan arsitektur yang paling dominan digunakan dalam domain ini. Salah satu varian CNN yang efisien adalah DenseNet201, yang memiliki keunggulan dalam *feature reuse* dan aliran gradien yang lebih baik melalui koneksi padat antar-*layer*. Meskipun CNN sangat handal dalam ekstraksi fitur, beberapa penelitian menunjukkan bahwa penggabungan (*hybrid*) antara ekstraktor fitur CNN dengan klasifikasi berbasis *Machine Learning* konvensional seperti XGBoost (*Extreme Gradient Boosting*) dapat meningkatkan akurasi dan efisiensi komputasi, terutama pada dataset dengan karakteristik tertentu.

Penelitian ini mengusulkan pendekatan *hybrid* dengan memanfaatkan DenseNet201 sebagai pengekstraksi fitur (*feature extractor*) dan XGBoost sebagai pengklasifikasi (*classifier*). XGBoost dipilih karena kemampuannya dalam menangani data terstruktur hasil ekstraksi fitur, mendukung regularisasi untuk mencegah *overfitting*, dan memiliki performa tinggi pada data yang tidak seimbang (*imbalanced dataset*). Selain itu, untuk meningkatkan kepercayaan pengguna terhadap model "kotak hitam" (*black-box*) ini, penelitian ini juga mengintegrasikan teknik Grad-CAM untuk memberikan visualisasi interpretabilitas area daun yang terinfeksi.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, rumusan masalah dalam penelitian ini adalah:

1. Bagaimana kinerja model *hybrid* yang menggabungkan ekstraksi fitur DenseNet201 dan klasifikasi XGBoost dalam mengidentifikasi penyakit daun padi?
2. Bagaimana pengaruh optimasi hiperparameter (*hyperparameter tuning*) pada algoritma XGBoost terhadap akurasi dan stabilitas model?
3. Bagaimana kemampuan model dalam menangani ketidakseimbangan kelas (*class imbalance*) pada dataset citra daun padi, khususnya pada kelas minoritas seperti *Hispa*?
4. Bagaimana tingkat interpretabilitas model dalam mengenali fitur visual penyakit menggunakan visualisasi Grad-CAM?

1.3 Tujuan Penelitian

Tujuan utama dari penelitian ini adalah:

1. Membangun dan mengevaluasi model klasifikasi penyakit daun padi (BrownSpot, Hispa, LeafBlast, dan Healthy) menggunakan pendekatan *Transfer Learning* dengan arsitektur DenseNet201 dan XGBoost.
2. Menerapkan teknik optimasi hiperparameter pada XGBoost untuk mendapatkan konfigurasi model terbaik yang mampu menghasilkan akurasi optimal.
3. Menganalisis performa model terhadap dataset yang tidak seimbang menggunakan metrik evaluasi yang komprehensif (F1-Score, Precision, Recall).
4. Memvisualisasikan hasil keputusan model menggunakan *heatmap* Grad-CAM untuk memvalidasi bahwa model fokus pada fitur lesi penyakit yang relevan.\

1.4 Manfaat Penelitian

Hasil penelitian ini diharapkan dapat memberikan kontribusi sebagai berikut:

1. Manfaat Teoretis:

- Memberikan kontribusi literatur mengenai efektivitas metode *hybrid* (*Deep Learning* + *Gradient Boosting*) dalam domain pertanian digital (*smart farming*).
- Menjadi referensi terkait penerapan *Explainable AI* (XAI) pada kasus klasifikasi penyakit tanaman.

2. Manfaat Praktis:

- Menyediakan model komputasi yang dapat diadopsi menjadi sistem pendukung keputusan bagi petani atau penyuluh pertanian untuk mendeteksi penyakit secara *real-time*.

- Membantu mengurangi risiko gagal panen melalui deteksi penyakit yang lebih dini dan akurat.

3. Manfaat Teknis:

- Menghasilkan model yang siap digunakan (*deployment-ready*) yang telah disimpan dalam format yang efisien (.pkl), yang dapat diintegrasikan ke dalam aplikasi seluler atau web di masa mendatang.

BAB 2
TINJAUAN PUSTAKA

2.1 Penyakit pada Tanaman Padi

Padi (*Oryza sativa*) merupakan komoditas pangan utama di banyak negara. Namun, produktivitasnya sering terganggu oleh penyakit daun yang memiliki kemiripan visual sehingga sulit dibedakan secara kasat mata.

1. Brown Spot (*Helminthosporium oryzae*): Penyakit ini ditandai dengan bercak berbentuk oval atau bulat berwarna coklat dengan titik abu-abu di tengahnya. Penyakit ini sering dikaitkan dengan kondisi tanah yang kekurangan nutrisi dan dapat menyebabkan bulir padi menjadi hampa.
2. Hispa (*Dicladispa armigera*): Disebabkan oleh hama kumbang hispa. Gejala visualnya berupa area putih transparan memanjang pada daun karena jaringan hijau daun dimakan oleh larva atau kumbang dewasa. Kerusakan ini mengurangi area fotosintesis secara signifikan.
3. Leaf Blast (*Magnaporthe oryzae*): Penyakit blas adalah salah satu yang paling merusak. Gejalanya berupa lesi berbentuk belah ketupat (spindle) dengan pinggiran kecoklatan dan pusat berwarna putih keabu-abuan. Jika tidak ditangani, penyakit ini dapat menyebar dengan cepat dan menyebabkan gagal panen total.

2.2 Convolutional Neural Network (CNN)

Convolutional *Neural Network* (CNN) adalah arsitektur *Deep Learning* yang dirancang khusus untuk memproses data berbentuk *grid*, seperti citra digital. Berbeda dengan jaringan saraf tiruan biasa (*Artificial Neural Network*), CNN menggunakan operasi matematika konvolusi untuk mengekstrak fitur visual secara otomatis.

Komponen utama CNN meliputi:

- Convolutional Layer: Bertugas mendeteksi fitur seperti tepi, tekstur, dan bentuk objek menggunakan filter (kernel).
- Pooling Layer: Berfungsi untuk mereduksi dimensi fitur (downsampling) guna mengurangi beban komputasi dan menghindari *overfitting*.
- Fully Connected Layer: Lapisan akhir yang bertugas melakukan klasifikasi berdasarkan fitur yang telah diekstrak.

2.3 Arsitektur DenseNet201

Dalam penelitian ini, model CNN yang digunakan adalah DenseNet201 (*Densely Connected Convolutional Networks*). DenseNet diperkenalkan untuk mengatasi masalah *vanishing gradient* pada jaringan yang sangat dalam.

2.3.1 Konsep Dense Connection

Berbeda dengan arsitektur sekuensial standar, DenseNet menghubungkan setiap *layer* dengan semua *layer* berikutnya dalam satu blok (*Dense Block*).Keunggulan arsitektur ini meliputi:

1. Feature Reuse: Fitur yang dipelajari di lapisan awal dapat digunakan kembali di lapisan akhir.
2. Parameter Efficiency: Membutuhkan jumlah parameter yang lebih sedikit dibandingkan ResNet dengan kedalaman yang sama karena fitur digabungkan, bukan dijumlahkan.
3. Aliran Gradien yang Lebih Baik: Memudahkan proses pelatihan pada jaringan yang dalam (201 lapisan).

2.4 Transfer Learning

Transfer Learning adalah teknik di mana model yang telah dilatih pada dataset besar (sumber), seperti ImageNet, digunakan kembali untuk menyelesaikan masalah pada dataset lain (target) yang biasanya lebih kecil.

Terdapat dua pendekatan utama dalam *Transfer Learning*:

1. **Fine-Tuning:** Melatih ulang sebagian atau seluruh bobot model agar menyesuaikan dengan data baru.
2. **Feature Extraction:** Menggunakan model *pre-trained* sebagai pengekstrak fitur tetap (bobot dibekukan/*freeze*), kemudian hasil ekstraksi fitur dimasukkan ke klasifikasi baru (seperti SVM atau XGBoost). Penelitian ini menggunakan pendekatan *Feature Extraction*.

2.5 XGBoost (*Extreme Gradient Boosting*)

XGBoost adalah algoritma *machine learning* berbasis *ensemble* yang menggunakan prinsip *Gradient Boosting Decision Tree* (GBDT). Algoritma ini bekerja dengan membangun serangkaian pohon keputusan (*decision trees*) secara berurutan.

Setiap pohon baru ditambahkan untuk memperbaiki kesalahan (*residual error*) yang dihasilkan oleh pohon sebelumnya. Keunggulan XGBoost dibandingkan algoritma lain (seperti Random Forest) adalah:

1. Regularisasi: XGBoost memiliki parameter regularisasi (L1 dan L2) yang membantu mencegah *overfitting*.
2. Kecepatan dan Efisiensi: Mendukung pemrosesan paralel dan *tree-pruning* otomatis.
3. Handling Sparse Data: Mampu menangani nilai yang hilang atau fitur dengan banyak nilai nol (seperti hasil ekstraksi fitur sparse).

Dalam penelitian ini, XGBoost bertindak sebagai *classifier* yang menerima input vektor fitur dari DenseNet201 untuk menentukan kelas penyakit daun padi.

2.6 Grad-CAM (*Gradient-weighted Class Activation Mapping*)

Untuk meningkatkan interpretabilitas model (*Explainable AI*), digunakan metode Grad-CAM. Metode ini memvisualisasikan area pada citra yang dianggap penting oleh model CNN dalam mengambil keputusan.

Grad-CAM bekerja dengan menghitung gradien dari skor kelas target terhadap *feature map* pada lapisan konvolusi terakhir. Area dengan bobot gradien besar akan ditampilkan sebagai *heatmap* berwarna merah (penting), sedangkan area yang tidak relevan berwarna biru. Ini penting untuk memvalidasi apakah model benar-benar fokus pada bercak penyakit di daun, bukan pada latar belakang (tanah/langit).

2.7 Metrik Evaluasi Kinerja

Untuk mengukur keberhasilan model klasifikasi, digunakan beberapa parameter evaluasi berdasarkan *Confusion Matrix*:

1. Akurasi (Accuracy): Rasio prediksi benar terhadap keseluruhan data.
2. Presisi (Precision): Tingkat keakuratan antara informasi yang diminta oleh pengguna dengan jawaban yang diberikan oleh sistem (seberapa banyak yang diprediksi Positif benar-benar Positif).
3. Recall (Sensitivitas): Tingkat keberhasilan sistem dalam menemukan kembali sebuah informasi (seberapa banyak data Positif aktual yang berhasil dikenali).
    - *Penting dalam kasus medis/penyakit:* Recall yang rendah berarti banyak penyakit yang terlewat (False Negative).
4. F1-Score: Rata-rata harmonis antara presisi dan recall, sangat berguna ketika dataset memiliki distribusi kelas yang tidak seimbang (*imbalanced dataset*).

BAB 3
METODOLOGI DAN HASIL PENELITIAN

3.1 Alur Penelitian

Penelitian ini dilakukan melalui beberapa tahapan sistematis untuk memastikan model klasifikasi penyakit daun padi yang dikembangkan memiliki kinerja yang optimal. Tahapan penelitian dimulai dari pengumpulan data, pra-pemrosesan data, ekstraksi fitur menggunakan *Deep Learning*, pembangunan model *Machine Learning*, hingga evaluasi kinerja model. Alur penelitian secara umum digambarkan sebagai berikut:

1. Pengumpulan Data: Mengunduh dataset citra daun padi dari repositori publik.
2. Pra-pemrosesan Data: *Resizing*, normalisasi, dan *label encoding*.
3. Ekstraksi Fitur: Menggunakan arsitektur *DenseNet201* yang telah dilatih sebelumnya (*pre-trained*) pada ImageNet.
4. Pembagian Data: Memisahkan data menjadi data latih (*training*) dan data uji (*testing*).
5. Pembangunan Model Klasifikasi: Melatih model Logistic Regression (sebagai *baseline*), Random Forest, dan XGBoost.
6. Optimasi Hiperparameter: Mencari parameter terbaik untuk model XGBoost menggunakan validasi silang (*cross-validation*).
7. Evaluasi Model: Mengukur kinerja menggunakan *Confusion Matrix*, *Accuracy*, dan *F1-Score*.

3.2 Pengumpulan Data

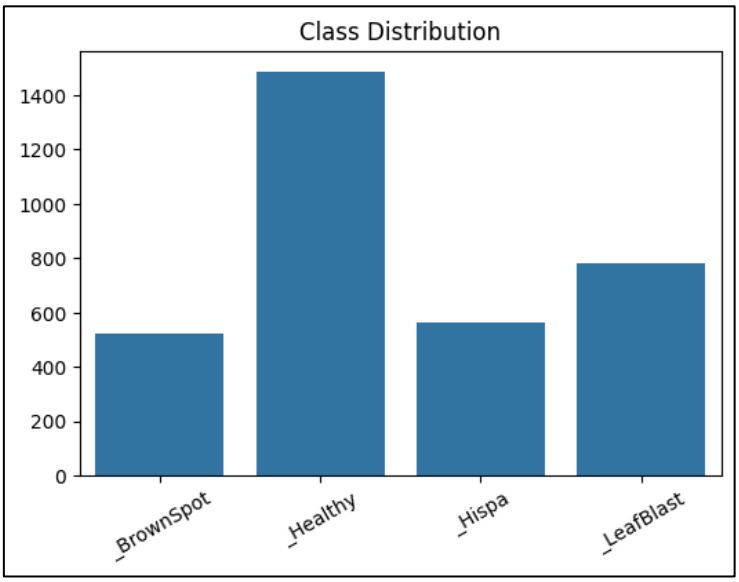
Data yang digunakan dalam penelitian ini bersumber dari dataset publik "Rice Leaf Images" yang tersedia di Kaggle (https://www.kaggle.com/datasets/nizorogbezuode/rice-leaf-images). Dataset ini terdiri dari total 3.355 citra daun padi. Data dikategorikan ke dalam empat kelas utama yang merepresentasikan kondisi kesehatan daun, yaitu:

1. BrownSpot: Bercak coklat.
2. Healthy: Daun sehat.
3. Hispa: Serangan hama hispa.
4. LeafBlast: Penyakit blas daun.

3.3 Pra-pemrosesan Data (*Data Preprocessing*)

Sebelum data dimasukkan ke dalam model, dilakukan serangkaian proses pra-pemrosesan untuk menyeragamkan format input dan meningkatkan kualitas data:

1. Pengubahan Ukuran Citra (*Resizing*): Seluruh citra diubah ukurannya menjadi dimensi 224 x224 piksel. Ukuran ini dipilih untuk menyesuaikan dengan input standar arsitektur DenseNet201.
2. Konversi Warna: Citra asli dibaca menggunakan OpenCV dalam format BGR (*Blue-Green-Red*) dan dikonversi menjadi format RGB (*Red-Green-Blue*).
3. Pelabelan Kelas (*Label Encoding*): Label kategori (teks) diubah menjadi format numerik menggunakan teknik *Label Encoder* agar dapat diproses oleh algoritma *machine learning*.
    - BrownSpot = 0
    - Healthy = 1
    - Hispa = 2
    - LeafBlast = 3
4. Normalisasi: Nilai piksel citra dinormalisasi menggunakan fungsi preprocess_input bawaan dari DenseNet untuk menyesuaikan distribusi data dengan data yang digunakan saat *pre-training* model ImageNet.



Gambar 3.1 Distribusi Kelas

3.4 Ekstraksi Fitur (*Feature Extraction*)

Penelitian ini menerapkan pendekatan *Transfer Learning* untuk ekstraksi fitur. Model yang digunakan adalah DenseNet201 dengan bobot (*weights*) ImageNet.

```
base_model = DenseNet201(
    weights="imagenet",
    include_top=False,
    pooling="avg",
    input_shape=(224, 224, 3)
)
X_features = base_model.predict(X_img, batch_size=32, verbose=1)
print("Feature shape:", X_features.shape)
```

Kode Program 3.1 *Base Model DenseNet201*

- Arsitektur: DenseNet201 dipilih karena kemampuannya dalam menangkap fitur kompleks melalui koneksi padat (*dense connections*) antar *layer*.
- Konfigurasi: Layer klasifikasi teratas (*fully connected layer*) dari DenseNet201 dihapus (include_top=False).
- Pooling: Metode *Global Average Pooling* diterapkan pada output layer konvolusi terakhir untuk mereduksi dimensi fitur spasial menjadi vektor fitur satu dimensi.
- Output: Hasil dari proses ini adalah vektor fitur yang merepresentasikan karakteristik visual dari setiap citra daun padi, yang kemudian akan menjadi input bagi algoritma klasifikasi (*classifier*).

3.5 Skenario Eksperimen dan Pemodelan

Setelah fitur diekstraksi, data dibagi menjadi data latih (80%) dan data uji (20%) menggunakan metode *Stratified Sampling* untuk menjaga proporsi kelas yang seimbang. Penelitian ini membandingkan tiga algoritma klasifikasi:

3.5.1 Baseline Model: Logistic Regression

```
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
baseline_model = LogisticRegression(
    max_iter=1000,
    class_weight="balanced"
)

baseline_model.fit(X_train_scaled, y_train)
y_pred_base = baseline_model.predict(X_test_scaled)

print("Baseline Accuracy:", accuracy_score(y_test, y_pred_base))
print(classification_report(
    y_test, y_pred_base,
    target_names=le.classes_,
    zero_division=0
))
```

Kode Program 3.2 *Baseline Model*

Sebagai pembanding dasar, digunakan Logistic Regression dengan pengaturan max_iter=1000 dan penyeimbang kelas (class_weight="balanced") untuk menangani potensi ketidakseimbangan data. Data fitur juga dinormalisasi menggunakan *StandardScaler* khusus untuk model ini.

3.5.2 Random Forest Classifier

```
rf_model = RandomForestClassifier(
    n_estimators=300,
    max_depth=None,
    min_samples_split=2,
    class_weight="balanced",
    random_state=42,
    n_jobs=-1
)

rf_model.fit(X_train, y_train)

y_pred_rf = rf_model.predict(X_test)

print("Random Forest Accuracy:", accuracy_score(y_test, y_pred_rf))
print(classification_report(
    y_test,
    y_pred_rf,
    target_names=le.classes_,
    zero_division=0
))
```

Kode 3.3 ***Kode Program RandomForest Model***

Model berbasis *ensemble* ini dilatih dengan 300 *estimators* (n_estimators=300) dan *class weight* seimbang. Model ini digunakan untuk melihat kinerja algoritma berbasis pohon keputusan tanpa *boosting*.

3.5.3 Model Utama: XGBoost (eXtreme Gradient Boosting)

XGBoost dipilih sebagai model utama karena kemampuannya yang tinggi dalam menangani data terstruktur hasil ekstraksi fitur.

```
learning_rates = [0.05, 0.1]
cv = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)

best_score = 0
best_params = {}

for lr in learning_rates:
    print(f"\nTesting learning_rate={lr}")
    fold_scores = []

    for fold_counter, (train_idx, val_idx) in enumerate(cv.split(X_train,
y_train), 1):
        X_tr, X_val = X_train[train_idx], X_train[val_idx]
        y_tr, y_val = y_train[train_idx], y_train[val_idx]

        model = XGBClassifier(
            n_estimators=300,
            max_depth=6,
            learning_rate=lr,
            subsample=0.8,
            colsample_bytree=0.8,
            objective="multi:softprob",
            eval_metric="mlogloss",
            random_state=42,
            n_jobs=-1,
            tree_method="gpu_hist",
            predictor="gpu_predictor"
        )

        model.fit(X_tr, y_tr)
        y_val_pred = model.predict(X_val)
        f1 = f1_score(y_val, y_val_pred, average="macro")
        fold_scores.append(f1)

        print(f"  Fold {fold_counter} F1-macro: {f1:.4f}")

    mean_f1 = np.mean(fold_scores)
    print(f"Mean F1-macro for learning_rate={lr}: {mean_f1:.4f}")

    if mean_f1 > best_score:
        best_score = mean_f1
        best_params = {"learning_rate": lr}
```

Kode Program 3.4 ***Hyperparameter Tuned XGBoost Model***

- Parameter Awal: *Objective function* multi:softprob digunakan untuk klasifikasi multi-kelas.
- Optimasi Hiperparameter: Dilakukan pencarian parameter terbaik (*Grid Search*) secara manual dengan skema *Stratified K-Fold Cross Validation* (k=5). Parameter yang diuji meliputi learning_rate: [0.05, 0.1]
- Metrik Optimasi: Parameter terbaik dipilih berdasarkan nilai rata-rata *F1-Score Macro* tertinggi.

3.6 Evaluasi Model

Kinerja model dievaluasi menggunakan data uji yang tidak pernah dilihat oleh model selama proses pelatihan. Metrik evaluasi yang digunakan meliputi:

1. Akurasi (*Accuracy*): Persentase prediksi yang benar dari total data.
2. Confusion Matrix: Untuk memvisualisasikan kesalahan prediksi (misklasifikasi) antar kelas.
3. Classification Report: Meliputi *Precision*, *Recall*, dan *F1-Score* untuk setiap kelas.
4. K-Fold Cross-Validation: Pada tahap akhir, model terbaik divalidasi ulang menggunakan *5-Fold Stratified Cross Validation* untuk memastikan stabilitas dan keandalan model terhadap variasi data.

3.7 Lingkungan Pengembangan

Penelitian ini diimplementasikan menggunakan perangkat lunak dan spesifikasi lingkungan sebagai berikut:

- Platform: Google Colab (GPU T4 Runtime).
- Bahasa Pemrograman: Python 3.
- Pustaka Utama:
    - *TensorFlow/Keras:* Untuk memuat model DenseNet201.
    - *Scikit-learn:* Untuk pra-pemrosesan, pembagian data, dan evaluasi.
    - *XGBoost:* Untuk algoritma klasifikasi utama.
    - *OpenCV:* Untuk pemrosesan citra digital.
    - *Matplotlib & Seaborn:* Untuk visualisasi data.

3.8 Hasil Implementasi dan Pengujian

Bagian ini memaparkan data hasil eksperimen yang diperoleh dari pengujian tiga skenario model: *Logistic Regression* (Baseline), *Random Forest*, dan *XGBoost*. Evaluasi dilakukan berdasarkan metrik *Accuracy*, *Precision*, *Recall*, dan *F1-Score*.

3.8.1 Hasil Pengujian Model Baseline (Logistic Regression)

Model *Logistic Regression* digunakan sebagai tolak ukur (*baseline*).

Baseline Accuracy: 0.7049180327868853

	precision	recall	f1-score	support
_BrownSpot	0.73	0.74	0.74	104
_Healthy	0.78	0.78	0.78	298
_Hispa	0.50	0.51	0.51	113
_LeafBlast	0.70	0.68	0.69	156
accuracy			0.70	671
macro avg	0.68	0.68	0.68	671
weighted avg	0.71	0.70	0.71	671

Gambar 3.2 *Classification Report Baseline*

- Akurasi Baseline: 70,49% (0.7049).
- Analisis: Model linear ini memberikan kinerja awal yang cukup baik. Namun, nilai *Recall* pada kelas *Hispa* hanya mencapai 0.51, yang mengindikasikan bahwa model masih kesulitan membedakan separuh dari total sampel penyakit Hispa.

3.8.2 Hasil Pengujian Random Forest

Random Forest Accuracy: 0.6721311475409836

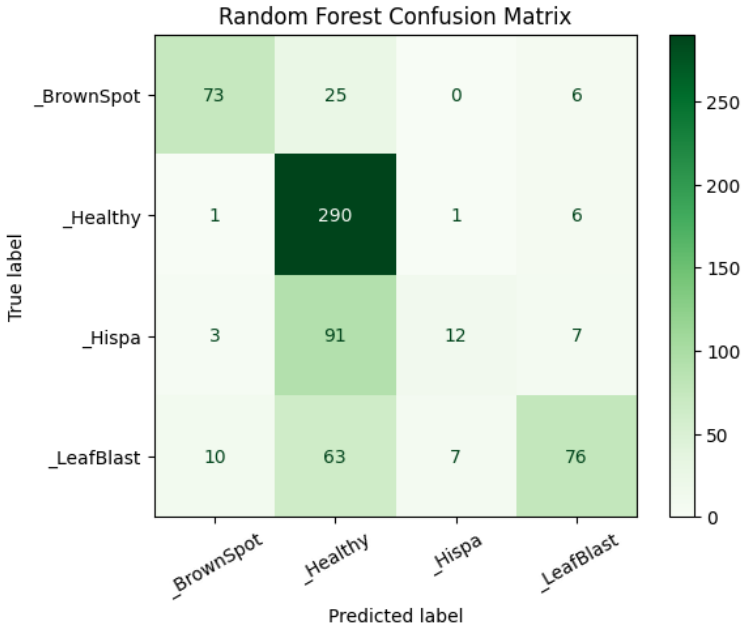
	precision	recall	f1-score	support
_BrownSpot	0.84	0.70	0.76	104
_Healthy	0.62	0.97	0.76	298
_Hispa	0.60	0.11	0.18	113
_LeafBlast	0.80	0.49	0.61	156
accuracy			0.67	671
macro avg	0.71	0.57	0.58	671
weighted avg	0.69	0.67	0.63	671

Gambar 3.3 *Classification Report RandomForest*

Pengujian selanjutnya menggunakan algoritma *Random Forest*.

- Akurasi Random Forest: 67,21% (0.6721).

- Perbandingan dengan Baseline: Kinerja model ini justru menurun sekitar 3,2% dibandingkan *baseline*.
- Analisis: Penurunan ini terutama disebabkan oleh kegagalan model dalam mengklasifikasikan kelas *Hispa*. Nilai *Recall* untuk *Hispa* sangat rendah, yaitu 0.11, yang berarti model hanya mampu mengenali 11% dari total kasus Hispa dengan benar. Sebagian besar data Hispa (91 citra) salah diprediksi sebagai *Healthy*.



Gambar 3.4 *Confusion Matrix RandomForest*

3.8.3 Hasil Pengujian dan Optimasi XGBoost

Eksperimen utama dilakukan menggunakan XGBoost dengan penyetelan hiperparameter (*hyperparameter tuning*).

a. Optimasi Hiperparameter : Proses pencarian parameter terbaik berfokus pada learning_rate. Berdasarkan hasil validasi silang (*cross-validation*), diperoleh hasil rata-rata *F1-macro* sebagai berikut:

- learning_rate = 0.05 menghasilkan F1-macro: 0.6835

```
Testing learning_rate=0.05
  Fold 1 F1-macro: 0.6821
  Fold 2 F1-macro: 0.6774
  Fold 3 F1-macro: 0.7066
  Fold 4 F1-macro: 0.6531
  Fold 5 F1-macro: 0.6984
Mean F1-macro for learning_rate=0.05: 0.6835
```

Gambar 3.5 Hasil learning_rate 0.05

- learning_rate = 0.1 menghasilkan F1-macro: 0.6877

```
Testing learning_rate=0.1
  Fold 1 F1-macro: 0.6839
  Fold 2 F1-macro: 0.6921
  Fold 3 F1-macro: 0.7047
  Fold 4 F1-macro: 0.6594
  Fold 5 F1-macro: 0.6985
Mean F1-macro for learning_rate=0.1: 0.6877
```

Gambar 3.6 Hasil learning_rate 0.1

Parameter terbaik yang dipilih adalah **learning_rate: 0.1** dengan skor validasi terbaik 0.6877.

b. Hasil Akhir (Tuned XGBoost) : Setelah diterapkan parameter terbaik, model menghasilkan kinerja final:

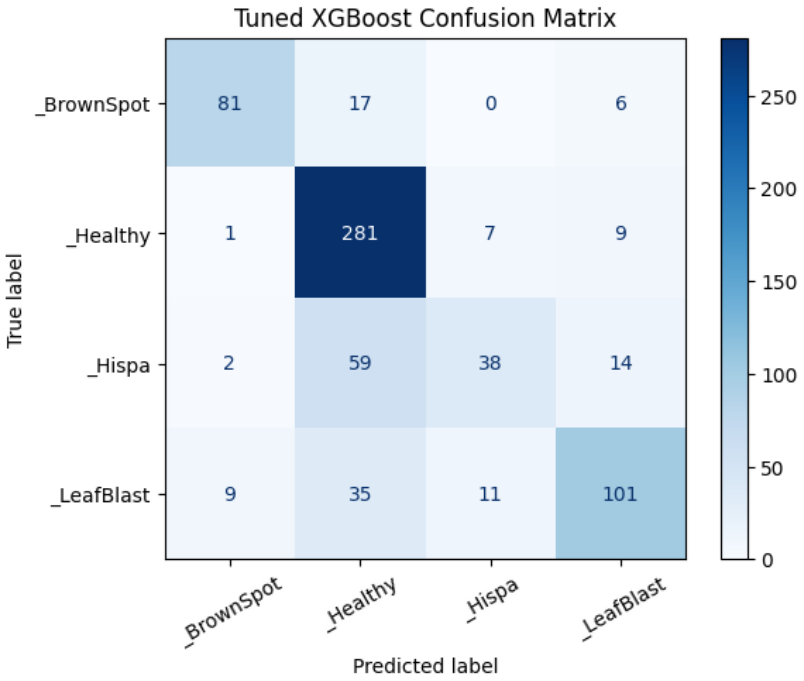
```
Tuned XGBoost Accuracy: 0.7466467958271237
              precision    recall  f1-score   support

  _BrownSpot       0.87      0.78      0.82       104
    _Healthy       0.72      0.94      0.81       298
      _Hispa       0.68      0.34      0.45       113
  _LeafBlast       0.78      0.65      0.71       156

    accuracy                           0.75       671
   macro avg       0.76      0.68      0.70       671
weighted avg       0.75      0.75      0.73       671
```

Gambar 3.7 *Classification Report Tuned XGBoost*

- Akurasi Final: 74,66% (0.7466).
- Peningkatan: Terjadi peningkatan performa yang signifikan dibandingkan *Random Forest* (67,21%) dan lebih tinggi dari *Baseline* (70,49%).

3.9 Pembahasan

3.9.1 Analisis Perbandingan Algoritma



Gambar 3.8 *Confusion Matrix Tuned XGBoost*

Berdasarkan rekapitulasi pengujian, algoritma XGBoost terbukti memberikan hasil terbaik dengan akurasi 74,66%. Keunggulan XGBoost dibandingkan Random Forest terletak pada kemampuannya menangani kelas yang sulit (*hard samples*) melalui mekanisme *boosting*. Meskipun Random Forest gagal total dalam mengenali kelas *Hispa* (hanya 12 prediksi benar), XGBoost mampu memperbaikinya dengan meningkatkan prediksi benar menjadi 38 citra, meskipun masih belum optimal.

3.9.2 Analisis Kesalahan (*Confusion Matrix*)

Berdasarkan visualisasi *Confusion Matrix* pada model terbaik (XGBoost), pola kesalahan prediksi dapat dianalisis sebagai berikut:

1. Kelas dengan Prediksi Terbaik: Kelas Healthy memiliki kinerja paling stabil dengan 281 prediksi benar dari 298 data (Recall 94%).
2. Miskasifikasi Dominan (Kasus Hispa): Tantangan terbesar dalam penelitian ini adalah mendeteksi penyakit *Hispa*.
    - Dari 113 data aktual *Hispa*, sebanyak 59 citra salah diprediksi sebagai *Healthy*.
    - Analisis Penyebab: Tingginya kesalahan prediksi *Hispa* menjadi *Healthy* menunjukkan adanya kemiripan fitur visual yang diekstraksi oleh DenseNet201 antara daun yang terserang Hispa (mungkin pada tahap awal) dengan daun

sehat. Selain itu, dominasi jumlah data kelas *Healthy* (imbalance dataset) membuat model cenderung memprediksi kelas mayoritas ketika fitur tidak cukup distingtif.

3.10 Interpretabilitas Model Menggunakan Grad-CAM

Selain mengukur kinerja model menggunakan metrik kuantitatif (akurasi dan F1-score), penelitian ini juga menerapkan teknik Grad-CAM (*Gradient-weighted Class Activation Mapping*) untuk menganalisis aspek interpretabilitas (*explainability*) dari model *Deep Learning* (DenseNet201) yang digunakan sebagai ekstraktor fitur.

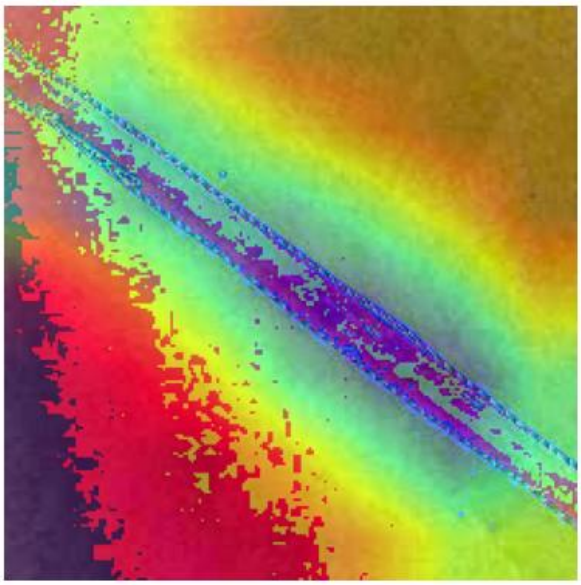
3.10.1 Implementasi Grad-CAM

Grad-CAM digunakan untuk memvisualisasikan area pada citra yang dianggap penting oleh model dalam mengambil keputusan klasifikasi. Teknik ini bekerja dengan menghitung gradien dari kelas prediksi terhadap *feature map* pada lapisan konvolusi terakhir.
Dalam implementasi ini, Grad-CAM diterapkan pada arsitektur DenseNet201 dengan konfigurasi sebagai berikut:

- Target Layer: conv5_block32_concat. Lapisan ini dipilih karena merupakan lapisan konvolusi terakhir pada DenseNet201 yang mengandung informasi spasial paling kaya dan abstrak sebelum masuk ke *pooling layer*.
- Proses Generasi Heatmap: Nilai gradien dirata-ratakan (*global average pooling*) untuk mendapatkan bobot pentingnya setiap *feature map*. Bobot ini kemudian dikalikan dengan *feature map* untuk menghasilkan *heatmap*.
- Visualisasi: *Heatmap* dinormalisasi dan diubah ukurannya agar sesuai dengan citra asli (224×224), kemudian diterapkan skema warna (*colormap*) 'JET' dan ditumpuk (*superimposed*) di atas citra asli dengan rasio transparansi tertentu.

3.10.2 Analisis Visual Heatmap

Hasil visualisasi Grad-CAM pada salah satu sampel citra daun padi dapat dilihat pada Gambar di bawah ini.



Gambar 3.9 Visualisasi *Grad-CAM* pada Citra Daun Padi

Analisis: Berdasarkan visualisasi pada Gambar 3.1, area yang berwarna merah hingga kuning menunjukkan area dengan aktivasi tinggi (sangat berpengaruh), sedangkan area biru menunjukkan aktivasi rendah.

1. Fokus Model: Terlihat bahwa *heatmap* menyoroti area sepanjang permukaan daun dan tekstur spesifik pada daun tersebut.
2. Validasi Fitur: Hal ini mengindikasikan bahwa ekstraktor fitur (DenseNet201) berhasil memfokuskan perhatiannya pada objek daun padi itu sendiri, bukan pada latar belakang (seperti tanah atau gangguan visual lainnya). Ini memvalidasi bahwa klasifikasi yang dilakukan oleh XGBoost didasarkan pada fitur morfologi daun yang relevan, bukan pada bias latar belakang (*background noise*).

3.11 Penyimpanan Model Akhir (*Model Deployment Preparation*)

```
joblib.dump(best_xgb, "xgb_best_model.pkl")
print("Best XGBoost model saved!")
```

Kode Program 3.5 Penyimpanan Model Terbaik

Tahap terakhir dalam alur penelitian ini adalah penyimpanan model terbaik untuk keperluan penggunaan di masa depan (*deployment*). Berdasarkan hasil evaluasi pada sub-bab sebelumnya, model XGBoost yang telah melalui proses *hyperparameter tuning* (learning_rate=0.1) ditetapkan sebagai model terbaik.

Model tersebut disimpan menggunakan pustaka joblib ke dalam format file biner .pkl (*pickle*).

- Nama File: xgb_best_model.pkl
- Tujuan: Penyimpanan ini memungkinkan model untuk dimuat kembali (*loaded*) tanpa perlu melakukan pelatihan ulang (*retraining*), sehingga dapat langsung digunakan untuk memprediksi data baru dalam aplikasi nyata atau sistem berbasis web.

3.11 Implementasi Streamlit

1. Halaman Home: Berisi deskripsi proyek, gambar sawah, dan metodologi (DenseNet201 + XGBoost) serta referensi dataset Kaggle.
2. Halaman Upload & Preview:
   - Mengakomodasi permintaan Upload CSV (Tab 2) jika user memiliki data fitur mentah.
   - Menambahkan Upload Gambar (Tab 1) yang lebih relevan untuk kasus ini, di mana gambar di-preprocess, diubah ukurannya menjadi 224x224, dan diekstrak fiturnya menggunakan DenseNet201 secara otomatis.
3. Halaman Run Model:
   - Memilih model XGBoost.
   - Melakukan prediksi probabilitas.
   - Fitur interaktif untuk memasukkan "Label Asli" secara manual guna menghasilkan Confusion Matrix dan Accuracy Score secara *real-time* (karena data upload tidak memiliki label ground truth bawaan).
4. Halaman Explainability:
   - Menggunakan library SHAP (shap.TreeExplainer) untuk menjelaskan prediksi XGBoost.
   - Menampilkan grafik batang (Bar Plot) fitur teratas yang berkontribusi terhadap prediksi kelas tertentu.
5. Halaman Download:
   - Tombol st.download_button untuk mengunduh hasil prediksi (file_name, prediction, confidence, probs) dalam format CSV.