Klasifikasi Jenis Kismis (Kecimen dan Besni) Menggunakan Model Deep Neural Network (DNN) dan TensorFlow/Keras.

Abdullah Mubarok Maspeke[1] Lukman Nur Rahman[2]
[1.2] Teknik Informatika STMIK Tazkia
[1]241552010001.maspeke@student.stmik.tazkia.ac.id
[2]241552010007.lukman@student.stmik.tazkia.ac.id

Abstrak

Kepuasan pelanggan dalam industri pangan sangat bergantung pada kualitas produk yang stabil. Namun, proses pensortiran kismis secara manual sering tidak konsisten karena pekerja bisa mengalami kelelahan. Untuk mengatasi hal ini, kami mengembangkan sistem berbasis kecerdasan buatan yang bisa membedakan jenis kismis Kecimen dan Besni lewat ciri-ciri fisiknya.

Dengan memanfaatkan data citra seperti luas, keliling, dan bentuk objek, kami menggunakan model Deep Neural Network (DNN) dan TensorFlow/Keras untuk mengenali pola pada data tersebut. Hasilnya menunjukkan bahwa sistem ini punya kemampuan yang sangat baik dalam tugas klasifikasi kismis. Ini membuktikan bahwa machine learning bisa menjadi solusi untuk kontrol kualitas otomatis, membuat produksi lebih efisien dan kualitas barang lebih terjamin.

Kata Kunci: Klasifikasi Kismis, Kontrol Kualitas, Machine Learning, Deep Neural Network, Kecerdasan Buatan.

Abstract

Customer satisfaction in the food industry relies heavily on consistent product quality. However, manual raisin sorting is often inconsistent due to worker fatigue. To address this, we developed an artificial intelligence-based system to distinguish between Kecimen and Besni raisins based on their physical characteristics.

By utilizing image data such as area, perimeter, and object shape, we employed the Deep Neural Network (DNN) and TensorFlow/Keras method to recognize patterns within the data. The results show that this system demonstrates excellent performance in the raisin classification task. This proves that machine learning can be an effective solution for automated quality control, making production more efficient and ensuring product quality.

Keywords: Raisin Classification, Quality Control, Machine Learning, Deep Neural Network, Artificial Intelligence

1.Pendahuluan

1.1 Latar Belakang

Saat ini dalam dunia industri pangan modern, kuallitas dan konsistensi adalah hal yang menentukan kepercayaan konsumen terhadap sebuah merek. Standar kualitas yang tinggi menuntut setiap produk yang sampai ke tangan pelanggan harus seragam dan bebas cacat. Namun, kenyataannya di sesi produksi seringkali menunjukan tantangan berat seperti proses penyortiran komoditas pertanian, seperti kismis, masih sangat bergantung pada inspeksi visual manual. Padahal, keterbatasan fisik manusia seperti kelelahan mata dan hilangnya konsentrasi akibat pekerjaan

berulang sering menjadi penyebab utama inkonsistensi dalam pengendalian kualitas (Cinar, Koklu, & Tasdemir, 2020).

Artificial Intelligence (AI) hadir sebagai solusi yang baru. Penelitian ini mengusulkan penerapan metode Multi-Layer Perceptron (MLP), sebuah arsitektur Jaringan Syaraf Berlapis atau Deep Neural Network (DNN) untuk menangani tugas klasifikasi ini. Tensorflow juga digunakan pada tugas klai ini.

Penelitian sebelumnya yang dilakukan oleh Cinar et al. (2020) telah membuktikan bahwa pendekatan berbasis Machine Learning mampu memberikan hasil yang menjanjikan mengunakan Raisin Dataset. Melanjutkan fondasi tersebut, penelitian ini bertujuan untuk mengimplementasikan dan mengevaluasi performa model DNN dengan tensorflow/keras dalam membedakan varietas Kecimen dan Besni.

Harapannya, sistem ini dapat menjadi bantuan bagi industri pertanian dan yang mirip dengannya dalam mengubah proses penyortiran yang subjektif dan melelahkan menjadi sistem deteksi otomatis yang cepat, akurat, dan dapat diandalkan. Seluruh proses, dari data mentah hingga menjadi sebuah model yang berfungsi, akan kami coba sampaikan secara bertahap dalam laporan penelitian ini.

1.2 Rumusan Masalah

- Bagaimana cara mengimplementasikan Deep Neural Network (DNN) dan TensorFlow/Keras untuk mengklasifikasikan jenis kismis (Kecimen vs. Besni) menggunakan 7 fitur morfologis yang tersedia?
- Bagaimana performa model DNN ini dibandingkan dengan hasil akurasi yang dilaporkan dalam penelitian terdahulu (Cinar et al., 2020)?

1.3 Tujuan Penelitian

- Membangun model klasifikasi biner menggunakan Deep Neural Network (DNN) dan TensorFlow/Keras untuk membedakan antara kismis jenis Kecimen dan Besni.
- Mengevaluasi performa model.

2. Metodologi

2.1. Sumber dan Modifikasi Data

Sumber penelitian ini menggunakan data sekunder yang tersedia untuk publik yang bersumber dari "Raisin Dataset" di UCI Machine Learning Repository. Dataset ini berisi data pengukuran fisik dari butiran kismis, termasuk luas area, keliling, panjang sumbu utama, dan kelengkungan. Dengan tujuan agar dapat diproses oleh komputer, kami melakukan sedikit modifikasi pada data asli. Data aslinya memuat kolom "Class" yang berisi teks "Kecimen" dan "Besni". Karena komputer tidak dapat membaca teks secara langsung untuk dihitung, kami mengubah kolom ini menjadi angka. Ini penting agar si mesin bisa membedakan mana kismis jenis A dan jenis B tanpa kebingungan.

Area	MajorAxisLength	MinorAxisLength	Eccentricity	ConvexArea	Extent	Perimeter	Class
87524	442,2460114	253,291155	0,819738392	90546	0,758650579	1184,04	Kecimen
75166	406,690687	243,0324363	0,801805234	78789	0,68412957	1121,786	Kecimen
90856	442,2670483	266,3283177	0,798353619	93717	0,637612812	1208,575	Kecimen
45928	286,5405586	208,7600423	0,684989217	47336	0,699599385	844,162	Kecimen
79408	352,1907699	290,8275329	0,56401133	81463	0,792771926	1073,251	Kecimen
49242	318,125407	200,12212	0,777351277	51368	0,658456354	881,836	Kecimen
42492	310,1460715	176,1314494	0,823098681	43904	0,665893562	823,796	Kecimen
60952	332,4554716	235,429835	0,706057518	62329	0,74359819	933,366	Kecimen
42256	323,1896072	172,5759261	0,845498789	44743	0,698030924	849,728	Kecimen
64380	366,9648423	227,7716147	0,784055626	66125	0,664375716	981,544	Kecimen
80437	449,4545811	232,3255064	0,856042518	84460	0,674235757	1176,305	Kecimen
43725	301,3222176	186,9506295	0,784258452	45021	0,697068248	818,873	Kecimen
43441	276,6108288	201,8131355	0,683882337	45133	0,690855598	803,748	Kecimen

Gambar 1: Potongan Dataset *" Raisin Dataset "*

Sumber data ini menberikan total 900 butir kismis sebagai sampel. Jumlah ini termaasuk cukup banyak, tapi jika dipikirkan kembali mungkin masih kurang banyak untuk mewakili jutaan kismis di dunia nyata. Ada juga kekhawatiran jikalau data ini diambil dari satu panen saja, sehingga mungkin hasilnya berbeda jika dipakai untuk kismis dari kebun lain. Data yang dipakai hanya 7 fitur fisik (bentuk dan ukuran) mungkin ada faktor lain seperti warna atau tekstur kulit yang juga berpengaruh ke jenisnya tapi tidak ada di data ini. Hal-hal seperti kualitas kamera saat memfoto kismisnya juga dapat berpengaruh tidak menutup kemungkinan ada bias di datanya. Karena itu, penelitian dimasa depan sebaiknya menambahkan variasi data dari berbagai sumber agar modelnya semakin baik.

2.2. Persiapan Data

Sebelum data diproses oleh model *computer* data tersebut perlu dibersihkan dan disiapkan terlebih dahulu. Dalam kode program kami, kami melakukan beberapa hal:

- Mengubah Label Jadi Angka (Encoding)
  Komputer lebih mudah memahami angka. Sehingga, label Besni dan Kecimen diubah menjadi angka 0 dan 1 dengan alat bernama LabelEncoder. Ini memudahkan komputer menghitung dan membandingkan prediksi dengan jawaban benar secara otomatis nantinya.

- Menyamakan Skala Angka (Scaling)
  Data kismis ini mempunyai rentang angka yang jauh. Misalnya dalam kolom Area angkanya dapat puluhan ribu, tapi dalam kolom Eccentricity angkanya berupa nol koma sekian. Jka langsung dimasukan komputer bisa mengira Area itu jauh lebih penting karena angkanya besar. Maka dari itu semua data numerik disamakan skalanya dengan StandardScaler.

2.3. Cara Kerja Model

Algoritma utama yang kami gunakan adalah *Deep Neural Network (DNN).* Deep Neural Network (DNN) adalah salah satu dari model machine learning yang mana ia meniru cara otak manusia dalam memproses suatu informasi. Berbeda dengan algoritma tradisional yang mengikuti aturan tetap, DNN dapat mempelajari pola dari data dan membuat prediksi yang berdasarkan pada pengalaman sebelumnya persis seperti manusia(Mercier, 2025). Di model kami ini terdapat lapisan input (data masuk), dua lapisan tersembunyi (*hidden layer*) yang isinya masing-masing 128 dan 64 neuron + ReLU, dan Output layer 1 neuron + Sigmoid untuk menghasilkan prediksi kelas 0/1 (Besni/Kecimen).

Sama halnya dengan belajar, model ini diajari dengan melihat data latih. Dia akan menebak misalnya Ini Besni!. Jika salah, dia akan dikoreksi dan dia akan mundur untuk memperbaiki bobot-bobot (nilai penting) di dalam jaringan agar prediksi berikutnya lebih mendekati benar. Proses ini dilakukan berulang selama beberapa latihan (epoch) sampai dia mampu membedkan pola bentuk kismisnya.

2.4. Pengujian Model

Teknik yang kami terapkan untuk mengetahui kemampuan model kami termasuk umum dalam *machine learning*. Kami membagi seluruh data yang kami miliki (900 data kismis) menjadi dua bagian:

1. **Data Latihan (80%)**
   Data Ini diberikan kepada model untuk belajar. Model akan mempelajari semua pola dari data ini.
2. **Data Ujian (20%)**
   Data ini dirahasiakan dari model selama proses belajar. Setelah model selesai belajar, kami mengujinya dengan data ini untuk melihat seberapa baik performanya pada data yang belum pernah ia lihat sebelumnya.

Keberhasilan model diukur dengan metrik akurasi khusus, yaitu persentase tebakan benar yang berhasil dibuat oleh model pada data ujian.

2.5. Tools

Pembuatan model ini memakai beberapa hal yang penting diantaranya:

1. **Python**
   Kami menggunakan Python untuk menulis semua perintah yang harus dijalankan oleh komputer. Bahasa ini dipilih karena populer dan punya banyak fitur siap pakai yang mempercepat pekerjan. Selain itu bahsa ini sudah kami pelajari di kelas sehingga lebih familier.
2. **Kode Editor (VS Code & Jupyter)**
   Sebagian besar proses kami kerjakan di VS Code. Ini adalah sebuah lembar kerja digital di mana kami bisa menulis perintah, menjalankan perintah itu, dan langsung melihat hasilnya, baik berupa teks maupun gambar, di satu tempat. Ini juga sangat disarankan oleh dosen kami.
3. **Alat Bantu Tambahan (Library Python)**
   Kami juga memakai beberapa alat bantu tambahan untuk tugas-tugas spesifik agar tidak menulis dari awal:

   - scikit-learn:
     Ini adalah alat utama untuk membuat program kecerdasan buatannya. Di sinilah semua proses belajarnya si komputer terjadi.
   - TensorFlow/Keras:
     alat utama untuk membangun dan melatih model DNN
   - pandas:
     Alat ini dipakai untuk mengelola data. Tugasnya adalah membaca dan merapikan tabel data agar siap untuk dianalisis.
   - numpy:
     Berfungsi sebagai kalkulator untuk menangani semua proses hitung-hitungan yang berat.
   - matplotlib & seaborn:
     Ini adalah alat untuk menggambar. Kami menggunakannya untuk mengubah data angka menjadi grafik yang lebih mudah dilihat dan dipahami.

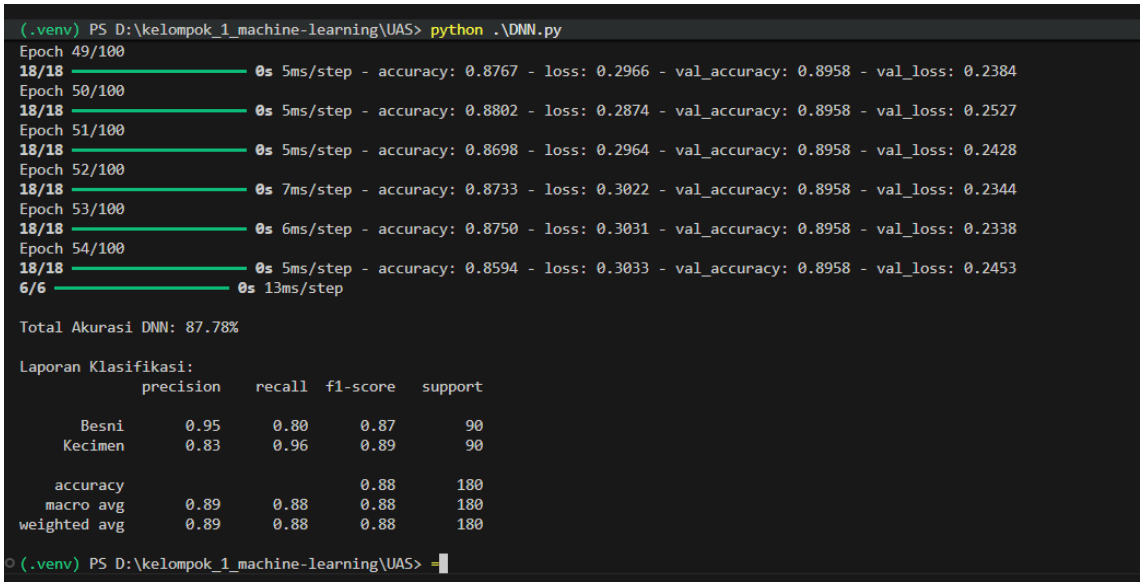
3. Hasil Dan Pembahasan

Pada bab ini, kami akan memberikan pembahasan hasil dari eksperimen klasifikasi kismis Kecimen dan Besni. Analisis difokuskan pada kinerja model DNN yang telah dilatih menggunakan data fitur morfologis kismis. Sebelum mendapatkan model terbaik, kami melakukan beberapa tahap pengujian untuk mencari pola yang cocok untuk arsitektur jaringan sarafnya.

3.1. Eksperimen Optimasi Mencari Arsitektur Terbaik

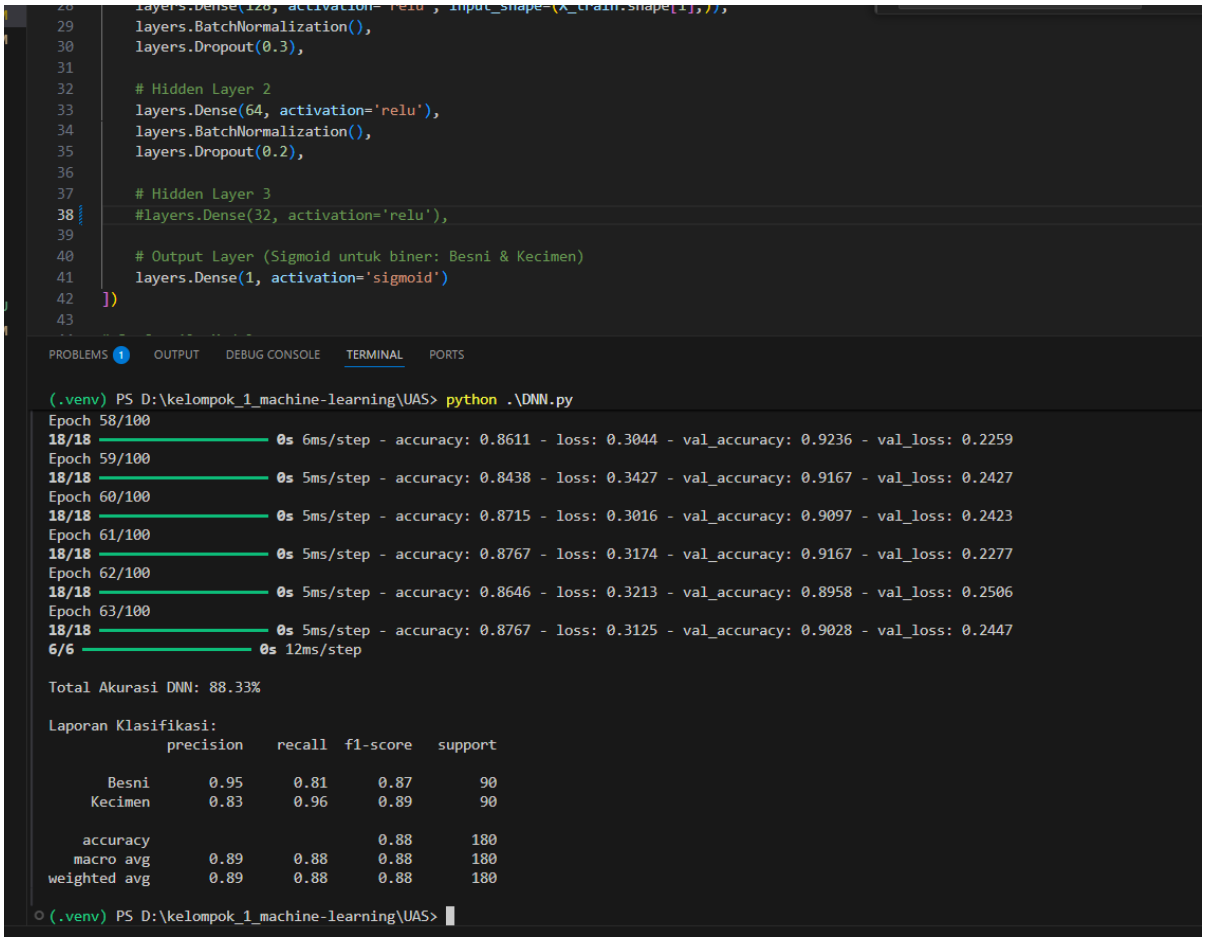
Sebelum menetapkan arsitektur final, penelitian ini melalui proses trial and error (uji coba beberapa konfigurasi). Tujuannya adalah untuk menemukan konfigurasi yang paling cocok dalam memisahkan kedua jenis kismis. Kami menguji tiga konfigurasi utama dengan pemahaman bahwa perubahan kecil pada arsitektur dapat mempengaruhi hasil.
Hasil evaluasi menunjukkan perbedaan performa yang signifikan antara ketiga percobaan:

- **Percobaan 1 (DNN lebih mendalam: 128–64–32)**
  Pada percobaan ini, kami menggunakan 3 lapisan tersembunyi (hidden layer): 128 neuron, 64 neuron, dan 32 neuron. Model dilengkapi Batch Normalization dan Dropout, serta menggunakan Early Stopping untuk mencegah model terlalu menghafal data latihan. Meskipun terhenti di 54/100 hasil pengujian menunjukkan bahwa model ini menghasilkan akurasi uji sebesar: 87.78%



Gambar 2: Potongan Hasil/output Percobaan 1

- **Percobaan 2 (DNN lebih sederhana: 128–64)**
  Pada percobaan kedua, kami mematikan layer 32 sehingga arsitektur menjadi lebih sederhana (dua hidden layer) dan mirip dengan model MLP di UTS. Tujuannya adalah menguji apakah pengurangan layer dapat meningkatkan kemampuan model saat menghadapi data uji. Meskipun ditemukan bahwa epoch berhenti di 41/100 hasil pengujian menunjukkan peningkatan akurasi menjadi: 88.33%

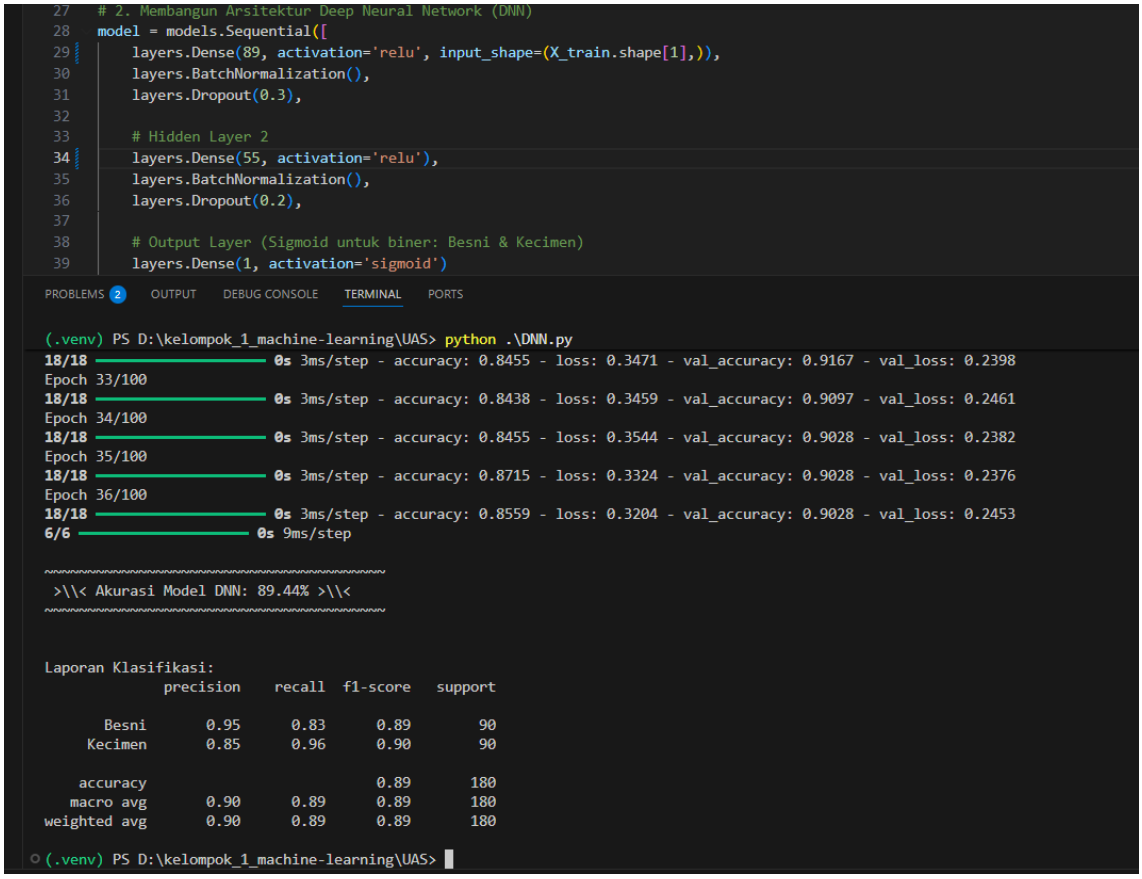


```
    layers.BatchNormalization(),
    layers.Dropout(0.3),

    # Hidden Layer 2
    layers.Dense(64, activation='relu'),
    layers.BatchNormalization(),
    layers.Dropout(0.2),

    # Hidden Layer 3
    #layers.Dense(32, activation='relu'),

    # Output Layer (Sigmoid untuk biner: Besni & Kecimen)
    layers.Dense(1, activation='sigmoid')
])
```

```
PROBLEMS 1  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

(.venv) PS D:\kelompok_1_machine-learning\UAS> python .\DNN.py
Epoch 58/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 6ms/step - accuracy: 0.8611 - loss: 0.3044 - val_accuracy: 0.9236 - val_loss: 0.2259
Epoch 59/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.8438 - loss: 0.3427 - val_accuracy: 0.9167 - val_loss: 0.2427
Epoch 60/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.8715 - loss: 0.3016 - val_accuracy: 0.9097 - val_loss: 0.2423
Epoch 61/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.8767 - loss: 0.3174 - val_accuracy: 0.9167 - val_loss: 0.2277
Epoch 62/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.8646 - loss: 0.3213 - val_accuracy: 0.8958 - val_loss: 0.2506
Epoch 63/100
18/18 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.8767 - loss: 0.3125 - val_accuracy: 0.9028 - val_loss: 0.2447
6/6 ━━━━━━━━━━━━━━━━━━━━ 0s 12ms/step

Total Akurasi DNN: 88.33%

Laporan Klasifikasi:
              precision    recall  f1-score   support

       Besni       0.95      0.81      0.87        90
     Kecimen       0.83      0.96      0.89        90

    accuracy                           0.88       180
   macro avg       0.89      0.88      0.88       180
weighted avg       0.89      0.88      0.88       180

(.venv) PS D:\kelompok_1_machine-learning\UAS>
```

Gambar 3: Potongan Hasil/output Percobaan 2

- **Percobaan 3/Final (DNN Rasio Emas Febonaci: 89-55)**
  Pada percobaan ketiga, kami merubah layer jadi 89-55 sehingga arsitektur menjadi lebih sederhana lagi dan mencoba penggunaan rasio emas dari febonaci. Tujuannya adalah menguji apakah pengurangan layer dapat meningkatkan kemampuan model saat menghadapi data uji. Karena ditemukan bahwa epoch berhenti di 36/100 maka seterusnya epoch di kode akan di ubah jadi 36. Hasil pengujian menunjukkan peningkatan akurasi menjadi: 89.44%



Gambar 4: Potongan Hasil/output Percobaan 3/final

3.2. Analisis Kinerja Model

Demi melihat dan mendapatkan perilaku model terbaik pada setiap jenis kismis (Besni dan Kecimen). Percobaan ketiga dengan arsitektur (89, 55) ditetapkan sebagai model final karena menghasilkan Akurasi tertinggi yaitu 89.44%.

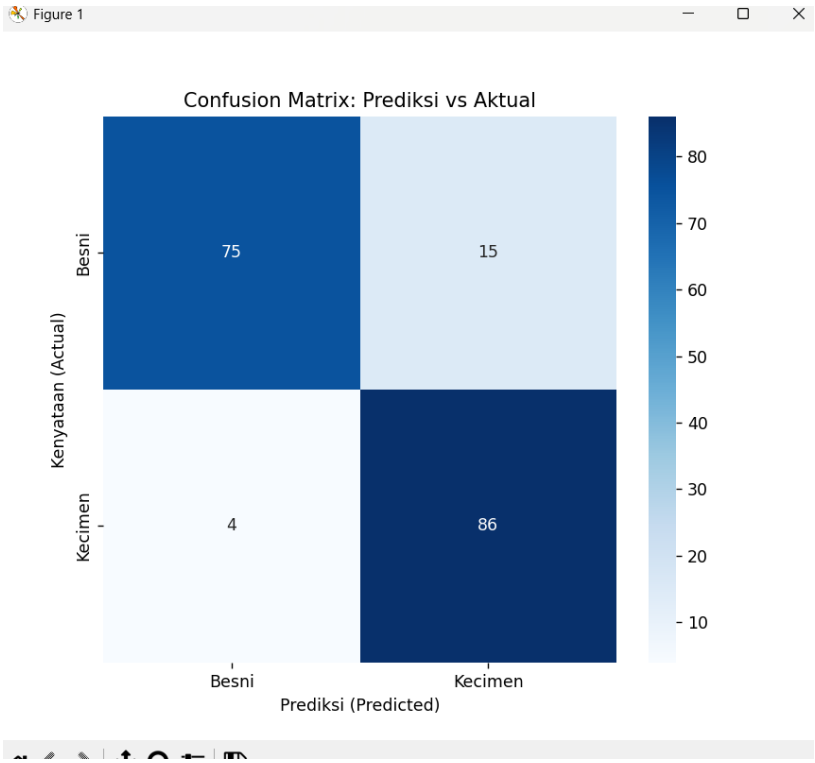
Akurasi dan Laporan Klasifikasi Percobaan ketiga. Akurasi Model Final: 89.44%. Akurasi ini menunjukkan bahwa model berhasil mengklasifikasikan 161 kismis dengan benar dari total 180 sampel uji sekitar (161/180 = 0.8944).

3.3 Analisis Kesalahan (Confusion Matrix)
Berdasarkan Matriks Kebingungan/confusion matrix model final (akurasi 89.44%), diperoleh hasil sebagai berikut:

- True Besni (Besni ke Besni) = 75
- Besni salah menjadi Kecimen (Besni ke Kecimen) = 15
- Kecimen salah menjadi Besni (Kecimen ke Besni) = 4
- True Kecimen (Kecimen ke Kecimen) = 86

Kesalahan model dapat dijelaskan dengan:

- False Positive (FP = 15): 15 kismis Besni salah diklasifikasikan sebagai Kecimen.
- False Negative (FN = 4): 4 kismis Kecimen salah diklasifikasikan sebagai Besni.



Gambar 5: Hasil/output Grafik model final dengan arsitektur (89,55)

Kesalahan terbesar tetap berasal dari Besni menjadi Kecimen (15 data), namun jumlahnya lebih rendah dibanding konfigurasi sebelumnya (yang error Besni dianggap Kecimen lebih tinggi). Penurunan error ini menjadi alasan utama mengapa akurasi meningkat menjadi 89.44%.

Tingginya angka *False Positives* (15) merupakan penyebab utama mengapa Akurasi tidak lebig dari 89.44%. Ini juga menunjukkan bahwa beberapa kismis Besni memiliki kemiripan fitur yang signifikan dengan Kecimen yang dapat mengecoh model.

3.4. Analisis Proses Pelatihan (Loss Curve dan EarlyStopping)

Proses pelatihan pada model final/percobaan 3 menunjukkan pelatihan berhenti lebih cepat dari 100 epoch karena EarlyStopping saat performa validasi (val_loss) tidak membaik lagi. Ini membantu mencegah model terlalu menghafal data latihan dan menjaga performa pada data uji.

Hasil penelitian ini menunjukkan bahwa model *DNN* dengan akurasi 89.44% karena menggunakan model dengan konfigurasi (89, 55) dan iterasi 36(diubah sesuai percobaan 3 dengan akurasi tertinggi) memiliki potensi sebagai teknologi pendukung/bantuan bagi industri pertanian dalam mengubah proses penyortiran yang melelahkan menjadi sistem deteksi otomatis yang cepat, akurat, dan dapat diandalkan. Namun, analisis kinerja mengungkap akurasi hanya 89.44% yang masih memiliki ruang pengembangan kedepannya. Komposisi ygang cocok dapat membawa model ini menuju akurasi yang lebih baik lagi.

3.5. Perbandingan Kinerja dengan Studi Acuan/ Paper Referensi

Untuk mengevaluasi penelitian ini maka perbandingan dilakukan dengan hasil model yang dilaporkan dalam studi acuan/ paper referensi yang menggunakan *dataset* yang sama. Perbandingan ini memberikan 4 temuan penting:

1. MLP (UTS) Lebih Unggul dari Paper Acuan
   Model MLP yang dikembangkan dalam bagian UTS memberikan akurasi 87.78% berhasil mengungguli model MLP yang dilaporkan dalam studi acuan/ paper referensi yang memberikan akurasi 86.33%. Peningkatan kinerja ini menunjukkan bahwa penggunaan fungsi aktivasi ReLU dan algoritma *solver* Adam pada arsitektur yang lebih ramping (128, 64) lebih efektif dalam mempelajari pola fitur morfologis kismis. Ini juga menunjukkan pentingnya komposisi dari sebuah model.

2. MLP (UTS) Melampaui Metode Terbaik dalam Studi Acuan
   Akurasi 87.78% dalam model yang dikembangkan ini juga melampaui akurasi tertinggi yang dicapai oleh metode lain dalam studi acuan, yaitu akurasi model SVM dengan 86.44%. Hal ini menginformasikan bahwa setelah proses yang tepat, model *Multi-Layer Perceptron* adalah salah satu dari metode klasifikasi yang efektif untuk *dataset* kismis ini.

3. DNN (UAS) Melampaui Metode Terbaik dalam Studi Acuan
   Akurasi 89.44% dalam model UAS yang dirancang ini juga sukses melampaui akurasi tertinggi yang dicapai oleh metode lain dalam studi acuan, yaitu akurasi model SVM dengan 86.44%. Dengan kenaikan 3% hal ini menunjukkan bahwa setelah proses yang tepat, model *DNN* adalah metode yang paling efektif untuk *dataset* kismis ini.

4. DNN (UAS) Melampaui Metode MLP (UTS)
   Dengan hasil akurasi DNN (UAS) 89.44%, maka ini menunjukkan bahwa ada kenaikan 1.66 % dibandingkan dengan akurasi MLP(UTS) 87.78%

3.6. kode python

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

```
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
import tensorflow as tf
from tensorflow.keras import layers, models, callbacks

# 1. Load & Preprocess
# Pastikan file 'Raisin.csv' berada di folder yang sama dengan script ini
df = pd.read_csv('Raisin.csv')
X = df.drop('Class', axis=1)
y = df['Class']

le = LabelEncoder()
y_encoded = le.fit_transform(y)

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

X_train, X_test, y_train, y_test = train_test_split(
    X_scaled, y_encoded, test_size=0.2, random_state=42,
stratify=y_encoded
)

# 2. Membangun Arsitektur Deep Neural Network (DNN)
model = models.Sequential([
    layers.Dense(89, activation='relu', input_shape=(X_train.shape[1],)),
    layers.BatchNormalization(),
    layers.Dropout(0.3),

    # Hidden Layer 2
    layers.Dense(55, activation='relu'),
    layers.BatchNormalization(),
    layers.Dropout(0.2),

    # Output Layer (Sigmoid untuk biner: Besni & Kecimen)
    layers.Dense(1, activation='sigmoid')
])

# 3. Compile Model
model.compile(
    optimizer='adam',
    loss='binary_crossentropy',
```

```
    metrics=['accuracy']
)

# 4. Training dengan Early Stopping
early_stopping = callbacks.EarlyStopping(
    monitor='val_loss',
    patience=10,
    restore_best_weights=True
)

print("Memulai Pelatihan DNN...")
history = model.fit(
    X_train, y_train,
    epochs=36,
    batch_size=32,
    validation_split=0.2,
    callbacks=[early_stopping],
    verbose=1
)

# 5. Evaluasi
y_pred_prob = model.predict(X_test)
y_pred = (y_pred_prob > 0.5).astype(int).flatten()

accuracy = accuracy_score(y_test, y_pred) * 100

print("\n" + "~" * 40)
print(f" >\\\< Akurasi Model DNN: {accuracy:.2f}% >\\\< ")
print("~" * 40 + "\n")

print("\nLaporan Klasifikasi:")
print(classification_report(y_test, y_pred, target_names=le.classes_))

# --- 6. VISUALISASI HASIL ---
# Plot Confusion Matrix
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(7, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
            xticklabels=le.classes_, yticklabels=le.classes_)
plt.title('Confusion Matrix: Prediksi vs Aktual')
plt.xlabel('Prediksi (Predicted)')
plt.ylabel('Kenyataan (Actual)')
plt.show()
```

4. Penutup

4.1 Kesimpulan

Berdasarkan hasil eksperimen dan analisis kinerja model klasifikasi kismis Kecimen dan Besni menggunakan Deep Neural Network (DNN) dengan TensorFlow/Keras ditemukan beberapa kesimpulan:

- Akurasi Klasifikasi Tercapai
  Model DNN dengan arsitektur final (89, 55) berhasil mengklasifikasikan varietas kismis pada data uji dengan akurasi 89.44%. Angka ini menunjukkan bahwa model yang dirancang ini mampu membedakan kedua jenis kismis dengan tingkat kebenaran yang cukup tinggi pada data yang belum pernah dilihat saat pelatihan.

- Optimalisasi Arsitektur Berhasil
  Proses trial and error (Percobaan 1, 2, dan 3 menunjukkan bahwa penyesuaian arsitektur jaringan saraf dapat meningkatkan performa. Penyederhanaan jumlah neuron dari konfigurasi sebelumnya (lebih besar) menuju konfigurasi yang lebih ringan (89, 55) terbukti meningkatkan kemampuan model dalam melakukan menilai pada data uji.

- Identifikasi Sumber Kesalahan
  Analisis kinerja menemukan adanya kelemahan signifikan pada kemampuan Analisis confusion matrix menunjukkan bahwa kesalahan terbesar berasal dari kasus Besni yang tertukar menjadi Kecimen. Pada hasil final, terdapat 15 data Besni yang salah diklasifikasikan sebagai Kecimen, sedangkan kesalahan Kecimen menjadi Besni itu hanya 4 data. Pola ini mengindikasikan bahwa sebagian sampel Besni memiliki kemiripan fitur morfologi yang cukup tinggi dengan Kecimen, sehingga masih berpotensi mengecoh model.

- **Model DNN Memiliki Performa Lebih Tinggi Dibandingkan Studi Acuan**
  Jika dibandingkan dengan studi acuan/paper referensi, hasil penelitian ini menunjukkan performa yang lebih tinggi. Studi acuan melaporkan akurasi MLP sebesar 86.33% dan metode terbaik SVM sebesar 86.44%, sedangkan model DNN dalam penelitian ini mencapai 89.44% lebih tinggi 3% dari akurasi terbaik di paper acuan..

4.2 Saran (Future Work)

Meskipun model DNN sudah mencapai akurasi **89.44%**, hasilnya masih bisa ditingkatkan supaya **lebih stabil** dan lebih siap dipakai untuk **penyortiran kualitas (quality control)** di dunia nyata. Berikut saran untuk penelitian selanjutnya:

1. Uji dengan cara yang sama dengan paper acuan
   Penelitian ini hanya mengetes sekali dengan pembagian 80/20. Lebih bagus kalau dilakukan tes berulang seperti di paper acuan (misalnya dibagi 10 bagian lalu dites bergantian) atau dijalankan beberapa kali agar dapat nilai rata-rata. Ini membuat hasil lebih dapat dipercaya dan tidak bergantung pada kebetulan di satu kali pembagian data.

2. Bandingkan dengan model lain yang mungkin lebih cocok untuk data tabel atau data kecil. Karena data Raisin ini bentuknya tabel angka, ada beberapa metode lain yang mungkin dapat dipercobakan, misalnya SVM, Random Forest, atau XGBoost. Sebaiknya dibandingkan dengan cara uji yang sama dengan paper acuan agar dapat menilai metode mana yang paling cocok untuk kasus kismis ini.

Ucapan Terima Kasih

Dengan penuh rasa terima kasih dan rasa hormat, kami menyampaikan apresiasi kepada Bapak Hendri Karisma, S.Kom., M.T. atas segala pengajaran, bimbingan, motivasi, serta saran yang telah membantu dalam penyusunan penelitian ini dan selama pembelajaran. Penulis juga meyampaikan rasa terima kasih dan rasa hormatnya kepada pihak penyedia dataset melalui platform archive.ics.uci.edu yang telah memfasilitasi tersedianya data publik sebagai dasar penelitian ini. Penulis juga meyampaikan rasa terima kasih ke semua pihak yang telah menyediakan artikel online yang sangat membantu dalam penyusunan penelitian ini.

Daftar Pustaka

Cinar, I., Koklu, M., & Tasdemir, S. (2020). Classification of raisin grains using machine vision and artificial intelligence methods. Gazi Journal of Engineering Sciences, 6(3), 200–209. https://dergipark.org.tr/tr/download/article-file/1227592

Cinar, I., Koklu, M., & Tasdemir, S. (2020). Raisin [Data set]. UCI Machine Learning Repository. https://archive.ics.uci.edu/dataset/850/raisin

Mercier, M. (2025). Deep neural network. Botpress.com. https://botpress.com/id/blog/deep-neural-network

Mubarok Maspeke, A., & Nur Rahman, L. (2024). Proposal: Klasifikasi jenis kismis (Kecimen dan Besni) menggunakan model Multi-Layer Perceptron (MLP). Sekolah Tinggi Manajemen Informatika dan Komputer Tazkia. https://docs.google.com/document/d/17sDLy8k7twdD-8YO3RJXZJIxI9H0_cclU2xIHtMmCec/edit?tab=t.0#heading=h.2otviyliemv0

Mubarok Maspeke, A., & Nur Rahman, L. (2024). Klasifikasi jenis kismis (Kecimen dan Besni) menggunakan model DNN. Sekolah Tinggi Manajemen Informatika dan Komputer Tazkia. https://github.com/STMIK-Tazkia/kelompok_1_machine-learning/blob/master/UAS/DNN.py