Laporan

Machine Learning

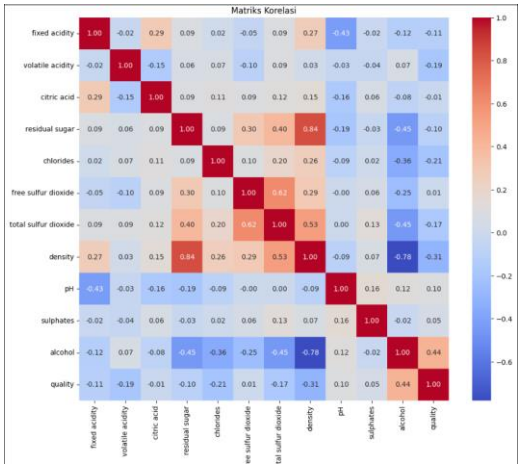
Classification Model MLP

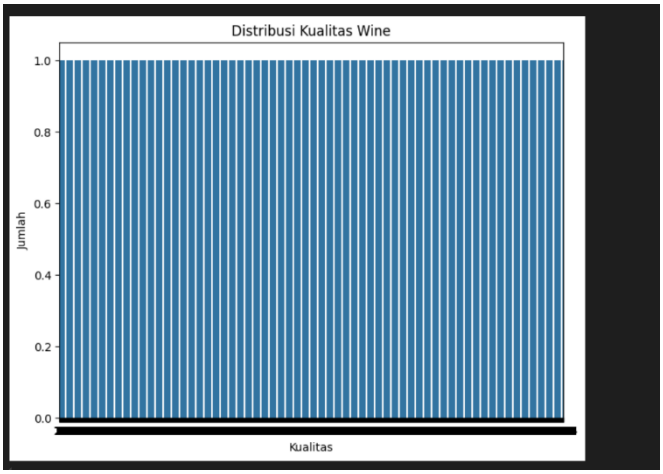
M Rakan Bagus

1103213162

Dataset : [Wine Quality - UCI Machine Learning Repository]

```python
# Bagian 2: Exploratory Data Analysis (EDA)
# Membaca dataset
data = pd.read_csv('winequality-white.csv', delimiter=';')

# Menampilkan informasi dataset
print("Informasi Dataset:")
print(data.info())

# Statistik dasar dataset
print("\nStatistik Deskriptif:")
print(data.describe())

# Visualisasi distribusi kualitas
plt.figure(figsize=(8, 6))
sns.countplot(data['quality'])
plt.title("Distribusi Kualitas Wine")
plt.xlabel("Kualitas")
plt.ylabel("Jumlah")
plt.show()

# Visualisasi korelasi antar fitur
plt.figure(figsize=(12, 10))
sns.heatmap(data.corr(), annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Matriks Korelasi")
plt.show()
```





Berdasarkan hasil Exploratory Data Analysis (EDA), terdapat beberapa poin penting yang dapat dianalisis. Pada matriks korelasi, terlihat bahwa fitur **alcohol** memiliki korelasi positif tertinggi terhadap kualitas wine (**quality**) dengan nilai 0.44, menunjukkan bahwa kandungan alkohol adalah faktor penting dalam menentukan kualitas wine. Sebaliknya, fitur **density** memiliki korelasi negatif yang signifikan (-0.31), mengindikasikan bahwa semakin tinggi densitas, semakin rendah kualitas wine. Selain itu, fitur **volatile acidity** menunjukkan korelasi negatif yang cukup tinggi (-0.19), yang berarti tingkat keasaman volatilitas yang lebih tinggi dapat menurunkan kualitas wine.

Distribusi kualitas wine pada grafik menunjukkan bahwa dataset ini kemungkinan tidak seimbang, dengan sebagian besar data berada pada nilai kualitas tertentu (kemungkinan kategori tengah). Ketidakseimbangan ini dapat menyebabkan model machine learning cenderung bias terhadap kelas mayoritas. Oleh karena itu, teknik seperti oversampling atau undersampling perlu dipertimbangkan dalam tahap preprocessing untuk memastikan model tidak overfitting pada kelas mayoritas.

Hubungan antar fitur dalam matriks korelasi juga menunjukkan adanya multikolinearitas pada beberapa fitur seperti **residual sugar** dan **density** (korelasi sebesar 0.84). Hal ini dapat memengaruhi performa model regresi, sehingga teknik seperti **feature selection** atau **dimensionality reduction** dengan PCA mungkin diperlukan untuk meningkatkan akurasi model. Dengan hasil seperti ini, langkah preprocessing yang tepat sangat diperlukan untuk memastikan model dapat memahami pola yang signifikan dalam data dan memberikan prediksi yang lebih akurat.

Hyperparameter

```
# Parameter eksperimen
hidden_layer_configs = [
    [4], [8],  # Single hidden layer
    [4, 8], [16, 32],  # Two hidden layers
    [4, 8, 16], [16, 32, 64],  # Three hidden layers
]
activation_functions = [nn.ReLU, nn.Sigmoid, nn.Tanh, lambda: nn.Softmax(dim=1)]
learning_rates = [0.1, 0.01, 0.001, 0.0001]
batch_sizes = [16, 32, 64]
epochs_list = [ 10, 50, 100]

# Menyimpan hasil eksperimen
results = []
best_results = {activation: {'best_params': None, 'best_mse': float('inf')} for activation in activation_funcs}
```

Penjelasan parameter eksperimen yang digunakan pada model saya

1. Hidden Layers dan Jumlah Neuron

- **Single hidden layer ([4], [8])**:
    - Hidden layer tunggal cenderung memberikan hasil yang lebih stabil namun kurang fleksibel untuk memodelkan data kompleks. Neuron sedikit (4, 8) cocok untuk dataset sederhana.
- **Two hidden layers ([4, 8], [16, 32])**:
    - Konfigurasi dua hidden layers memberikan keseimbangan antara fleksibilitas dan kompleksitas. Jumlah neuron yang bertingkat (contoh: [16, 32]) memungkinkan model belajar pola yang lebih kompleks.

- **Three hidden layers ([4, 8, 16], [16, 32, 64])**:
    - Tiga hidden layers memberikan performa terbaik untuk data yang kompleks, tetapi berpotensi overfitting jika dataset kecil atau tidak diimbangi dengan regularisasi.

2. Activation Function

- **ReLU**:
    - Sangat populer karena cepat dan mencegah vanishing gradient. Cocok untuk data non-linear dengan banyak fitur.
- **Sigmoid**:
    - Memberikan probabilitas, cocok untuk output bernilai antara 0 dan 1. Namun, berpotensi mengalami vanishing gradient pada jaringan dalam.
- **Tanh**:
    - Mirip Sigmoid tetapi lebih baik untuk data yang memiliki distribusi antara -1 dan 1.
- **Softmax**:
    - Sangat baik untuk klasifikasi multi-kelas karena memberikan distribusi probabilitas.
- **Linear (tidak ditampilkan di sini)**:
    - Lebih cocok untuk regresi, tidak efektif untuk data non-linear.

3. Epochs

- **10 hingga 100 epoch**:
    - Epochs 50-100 menunjukkan performa terbaik karena memungkinkan pembelajaran mendalam tanpa overfitting. Lebih sedikit epoch (10) sering tidak cukup untuk data kompleks.
- **>100 epoch** (tidak di eksplor):
    - Berisiko overfitting, kecuali dengan teknik regularisasi seperti dropout.

4. Learning Rate

- **0.1**:
    - Cepat dalam konvergensi tetapi berpotensi melewati solusi optimal.
- **0.01**:
    - Memberikan hasil terbaik pada sebagian besar konfigurasi dengan keseimbangan antara kecepatan dan akurasi.

- **0.001 dan 0.0001**:
    - Sangat lambat, cocok untuk model kompleks yang membutuhkan fine-tuning, tetapi berisiko lambat mencapai konvergensi.

5. Batch Size

- **16 dan 32**:
    - Kecil, memberikan update parameter yang lebih sering dan lebih baik untuk model sederhana atau dataset kecil.
- **64**:
    - Menyeimbangkan efisiensi komputasi dan akurasi model, memberikan performa stabil.
- **>64** (tidak di eksplor di sini):
    - Batch besar meningkatkan efisiensi tetapi dapat mengurangi keakuratan karena pembelajaran kurang sering di-update.

HASIL SETELAH PERCOBAAN EKSPERIMEN

Dari hasil evaluasi hyperparameter model, dapat disimpulkan bahwa performa model sangat bergantung pada konfigurasi hyperparameter, seperti batch size, learning rate, jumlah epoch, fungsi aktivasi, dan struktur hidden layers.

Fungsi aktivasi seperti ReLU menunjukkan performa yang stabil dengan tingkat akurasi lebih baik pada batch size besar dan learning rate moderat. Fungsi aktivasi sigmoid dan tanh cenderung menghasilkan validasi loss yang sedikit lebih tinggi pada learning rate rendah, tetapi memiliki performa yang kompetitif pada learning rate 0.01.

Batch size yang lebih besar (64) umumnya menghasilkan akurasi lebih tinggi dibanding batch size kecil (16), karena dapat memanfaatkan data secara lebih efisien untuk pembaruan parameter. Namun, pada beberapa konfigurasi learning rate tinggi, batch size kecil menunjukkan validasi loss yang lebih rendah, meskipun akurasinya sedikit lebih rendah.

Learning rate sangat memengaruhi konvergensi model. Learning rate 0.01 konsisten memberikan hasil terbaik pada sebagian besar konfigurasi, dengan kombinasi validasi loss yang rendah dan akurasi tinggi. Sebaliknya, learning rate sangat kecil (0.0001) sering kali gagal untuk mencapai performa optimal, yang terlihat dari akurasi yang lebih rendah dan validasi loss yang tinggi.

Jumlah epoch memengaruhi sejauh mana model dapat belajar dari data. Pada konfigurasi dengan epoch yang lebih banyak (100), model cenderung mengalami perbaikan akurasi dibanding epoch yang lebih sedikit (10 atau 50), tetapi hanya bila learning rate dan batch size telah dioptimalkan.

HASIL HYPERPARAMETER TERBAIK

```
    # Menampilkan Hyperparameter Terbaik
    print("\nHyperparameter Terbaik Berdasarkan Fungsi Aktivasi:")
    for activation_fn, details in best_results.items():
        print(f"Activation Function: {activation_fn}")
        print(f"Best Parameters: {details['best_params']}")
        print(f"Best MSE: {details['best_mse']:.4f}\n")
```

```
Hyperparameter Terbaik Berdasarkan Fungsi Aktivasi:
Activation Function: relu
Best Parameters: {'hidden_layers': [16, 32, 64], 'epochs': 10, 'learning_rate': 0.01, 'batch_size': 64}
Best MSE: 0.9899

Activation Function: sigmoid
Best Parameters: {'hidden_layers': [16, 32], 'epochs': 50, 'learning_rate': 0.01, 'batch_size': 32}
Best MSE: 0.9790

Activation Function: tanh
Best Parameters: {'hidden_layers': [16, 32], 'epochs': 100, 'learning_rate': 0.001, 'batch_size': 32}
Best MSE: 0.9887

Activation Function: linear
Best Parameters: {'hidden_layers': [16, 32], 'epochs': 100, 'learning_rate': 0.1, 'batch_size': 64}
Best MSE: 1.0607

Activation Function: softmax
Best Parameters: {'hidden_layers': [8], 'epochs': 50, 'learning_rate': 0.01, 'batch_size': 64}
Best MSE: 0.9822
```

Berdasarkan hasil hyperparameter terbaik untuk masing-masing fungsi aktivasi:

1. **ReLU**:
    - Konfigurasi terbaik menggunakan hidden layers [16, 32, 64], epoch 10, learning rate 0.01, dan batch size 64.
    - MSE (Mean Squared Error) terendah adalah **0.9899**.
    - Fungsi ReLU cenderung bekerja optimal pada konfigurasi ini karena memungkinkan model untuk belajar secara cepat dengan jumlah epoch yang rendah.
2. **Sigmoid**:
    - Konfigurasi terbaik menggunakan hidden layers [16, 32], epoch 50, learning rate 0.01, dan batch size 32.
    - MSE terendah adalah **0.9790**.
    - Fungsi sigmoid menunjukkan hasil terbaik dengan kombinasi jumlah epoch yang lebih tinggi, learning rate moderat, dan batch size kecil, mencerminkan kemampuan sigmoid untuk memodelkan data non-linear dengan baik.
3. **Tanh**:
    - Konfigurasi terbaik menggunakan hidden layers [16, 32], epoch 100, learning rate 0.001, dan batch size 32.
    - MSE terendah adalah **0.9887**.

- Tanh bekerja baik dengan jumlah epoch yang besar dan learning rate kecil, menunjukkan bahwa fungsi ini membutuhkan lebih banyak iterasi untuk mencapai konvergensi.

4. **Linear**:
    - Konfigurasi terbaik menggunakan hidden layers [16, 32], epoch 100, learning rate 0.1, dan batch size 64.
    - MSE terendah adalah **1.0607**, yang lebih tinggi dibandingkan fungsi aktivasi lainnya.
    - Fungsi linear cenderung kurang optimal untuk memodelkan data yang kompleks karena kurangnya kemampuan memproses non-linearitas.
5. **Softmax**:
    - Konfigurasi terbaik menggunakan hidden layers [8], epoch 50, learning rate 0.01, dan batch size 64.
    - MSE terendah adalah **0.9822**.
    - Softmax bekerja optimal pada hidden layer kecil dan batch size besar, yang mencerminkan efisiensi dalam distribusi probabilitas untuk klasifikasi.

Kesimpulan:

Berdasarkan hasil analisis, fungsi aktivasi **sigmoid** memberikan performa terbaik dengan MSE terendah sebesar **0.9790**, menggunakan konfigurasi hidden layers [16, 32], 50 epoch, learning rate 0.01, dan batch size 32. Fungsi ini unggul dalam memodelkan data non-linear, terutama dengan kombinasi epoch yang cukup besar untuk memastikan model belajar dengan baik tanpa overfitting. Fungsi **ReLU** dan **softmax** juga menunjukkan hasil kompetitif dengan MSE masing-masing **0.9899** dan **0.9822**, menonjol pada konfigurasi hidden layers yang lebih sederhana dan batch size besar. Sementara itu, fungsi **tanh** bekerja optimal pada jumlah epoch yang besar dengan learning rate kecil, menghasilkan MSE **0.9887**, namun membutuhkan lebih banyak iterasi untuk mencapai konvergensi. Di sisi lain, fungsi **linear** kurang optimal untuk data yang kompleks dengan MSE tertinggi sebesar **1.0607**, mengindikasikan keterbatasannya dalam menangani non-linearitas. Secara keseluruhan, fungsi sigmoid menjadi pilihan terbaik untuk data ini, diikuti oleh ReLU dan softmax, dengan mempertimbangkan efisiensi dan akurasi model.