Analisis Kode dan Output

1. **Pengolahan Data dan Normalisasi**

   Dataset winequality-white.csv telah diproses dengan benar, di mana fitur (X) dan target (y) dipisahkan dan dinormalisasi. Proses normalisasi ini membantu dalam meningkatkan stabilitas pelatihan model, mengurangi bias yang mungkin timbul akibat skala fitur yang berbeda. Pembagian data menjadi data pelatihan dan pengujian (80:20) memastikan bahwa model dilatih dan diuji pada set data yang berbeda, yang penting untuk mengevaluasi kemampuannya dalam generalisasi.

2. **Model MLP dan Eksperimen Hyperparameter**

   Kode ini mengimplementasikan model MLP dengan konfigurasi eksperimen yang sangat variatif. Eksperimen ini melibatkan pengujian berbagai kombinasi parameter seperti jumlah lapisan tersembunyi, fungsi aktivasi, learning rate (LR), ukuran batch, dan jumlah epoch. Tujuan dari eksperimen ini adalah untuk menemukan kombinasi parameter yang menghasilkan Test Loss terendah, yang menunjukkan bahwa model dapat memprediksi target dengan akurasi yang lebih baik. Fungsi aktivasi yang digunakan mencakup sigmoid, relu, tanh, softmax, dan linear, yang mempengaruhi cara model mempelajari hubungan antara fitur dan target. Hasil eksperimen menunjukkan bahwa penggunaan fungsi aktivasi sigmoid, tanh, dan softmax memberikan hasil yang lebih stabil dan cenderung lebih baik dalam hal Test Loss dibandingkan dengan relu dan linear.

3. **Analisis Output dan Performa Model**

   Dari output yang diberikan, dapat terlihat bahwa **learning rate**, **ukuran batch**, dan **jumlah epoch** memiliki dampak besar terhadap performa model. Sebagai contoh, pada konfigurasi Hidden Layers: [32, 64], Activation: tanh, Epochs: 250, LR: 0.1, dan Batch Size: 256, hasil Test Loss adalah 0.7534, yang lebih rendah dibandingkan dengan kombinasi lain dengan learning rate yang lebih tinggi, seperti LR: 10, yang menghasilkan Test Loss yang jauh lebih tinggi. Hal ini menunjukkan bahwa penggunaan learning rate yang lebih tinggi dapat menyebabkan model kesulitan dalam mencapai konvergensi yang stabil. Kombinasi **tanh** sebagai fungsi aktivasi juga menunjukkan performa yang lebih baik dengan loss yang lebih rendah pada berbagai ukuran batch. Di sisi lain, kombinasi dengan fungsi aktivasi **linear** seperti pada Hidden Layers: [4, 8, 16], Activation: linear, menghasilkan loss yang sangat besar, yang mengindikasikan bahwa konfigurasi tersebut mungkin tidak cocok untuk tugas regresi pada dataset ini. Secara umum, fungsi aktivasi seperti sigmoid dan tanh memberikan hasil yang lebih efektif dan efisien, terutama dengan jumlah epoch yang lebih tinggi dan learning rate yang lebih moderat.

4. **Efek Ukuran Batch dan Epoch**

Analisis lebih lanjut mengungkapkan bahwa **ukuran batch** dan **epoch** memainkan peran penting dalam menentukan kecepatan konvergensi dan hasil akhir dari model. Ukuran batch yang lebih kecil, seperti 16 atau 32, sering kali memberikan hasil yang lebih stabil dan akurat pada data uji, meskipun waktu pelatihan lebih lama. Hal ini disebabkan oleh pembaruan bobot yang lebih sering, yang memungkinkan model untuk lebih cepat beradaptasi terhadap pola dalam data. Sebaliknya, ukuran batch yang besar (seperti 256 atau 512) dapat mempercepat pelatihan tetapi terkadang menyebabkan model terjebak pada solusi lokal atau tidak mampu mengeksplorasi seluruh ruang solusi dengan cukup baik. Penggunaan **250 epoch** pada beberapa eksperimen memungkinkan model untuk belajar lebih lama dan dengan demikian mencapai hasil yang lebih optimal, seperti yang terlihat pada konfigurasi Activation: tanh dengan Test Loss yang lebih rendah dibandingkan dengan 1 atau 25 epoch.

5. **Kesimpulan**

Secara keseluruhan, eksperimen ini menunjukkan bahwa pengaturan yang hati-hati terhadap **learning rate**, **ukuran batch**, dan **fungsi aktivasi** sangat mempengaruhi kualitas prediksi model. **Fungsi aktivasi** seperti tanh dan sigmoid memberikan hasil yang lebih stabil dan akurat, terutama dalam kombinasi dengan learning rate yang moderat dan ukuran batch yang kecil. Untuk tugas regresi pada dataset ini, **epoch yang lebih banyak** dapat meningkatkan performa, namun pengaturan yang optimal harus tetap memperhatikan parameter lainnya seperti learning rate dan ukuran batch untuk menghindari overfitting atau konvergensi yang tidak stabil. Hasil ini menunjukkan pentingnya **eksperimen parameter** yang ekstensif untuk menemukan konfigurasi terbaik dalam melatih model.