ADVANCE MACHINE LEARNING AND NEURAL NETWORKS
Assignment 2

INTRODUCTION

In this assignment, I worked with a synthetic dataset generated from a Multimodal Gaussian distribution. My goal was to explore various regression models to understand how they fit this complex data and to evaluate their performance in terms of accuracy and computational time. The models I tested include Piecewise linear regression, Piecewise polynomial regression, and Polynomial regression, as well as Ridge and LASSO regression for comparison. Throughout this process, I used several important libraries and functions to implement and evaluate the models, which I'll describe below.

Problem 1: Generating and Visualizing Multimodal Gaussian Data

To start, I generated a dataset consisting of three clusters using the NumPy library, specifically the np.random.multivariate_normal() function. This function allowed me to create three different Gaussian distributions, each with a different mean and covariance, resulting in a multimodal dataset.

After generating the data, I visualized it using Matplotlib with plt.scatter() to plot the data points and plt.contour() to illustrate the contours of the distributions. This plot gave a clear view of how the data was distributed across multiple clusters, which is crucial for the next steps in regression analysis.

Key Libraries and Functions:
- **NumPy**
- **Matplotlib**

Problem 2: Piecewise Linear Regression

Next, I applied piecewise linear regression to fit different line segments to the data. The idea behind this method is to break the data into segments and fit straight lines to each segment, rather than using a single line for the entire dataset.

To identify where to split the data (called "knots"), I used the SciPy library and its curve_fit() function, which finds the best-fitting parameters for each segment.

I identified two knots in the data, at x0 = 0.38 and x1 = 2.90, which marked the points where the regression line changes its slope.

Key Libraries and Functions:
- **SciPy** (from scipy.optimize import curve_fit)

ADVANCE MACHINE LEARNING AND NEURAL NETWORKS
Assignment 2

Problem 3: Piecewise Polynomial Regression

In this problem, I took the same approach as in Problem 2 but replaced the straight lines with quadratic (polynomial) curves. This is more flexible and can capture more complex relationships within each segment of the data.

I reused the same knots (x0 = 0.38 and x1 = 2.90) and fit quadratic curves to each segment using the LSQUnivariateSpline function from SciPy.

This method resulted in a smoother fit compared to the linear model, as quadratic equations can account for more curvature in the data.

Key Libraries and Functions:
- **SciPy** (from scipy.interpolate import LSQUnivariateSpline)

Problem 4: Model Performance Evaluation

To compare the accuracy of the models, I calculated key performance metrics: RMSE, R^2, and the F-statistic. These metrics help assess how well each model fits the data.

I used the scikit-learnlibrary to calculate RMSE and R^2 with the mean_squared_error() and r2_score() functions.

Here are the results:

- **Piecewise Linear Regression**: RMSE = 1.3040, R^2 = 0.7431
- **Piecewise Polynomial Regression**: RMSE = 1.4227, R^2 = 0.6942
- **Polynomial Regression (Degree 3)**: RMSE = 1.7844, R^2 = 0.5190

From these results, I concluded that the **piecewise linear regression model offered the best fit for this dataset**, with the lowest RMSE and the highest R^2 value.

Key Libraries and Functions:
- **scikit-learn** (from sklearn.metrics import mean_squared_error, r2_score)

Problem 5: Modelling the Multimodal Gaussian Distribution with Polynomial Regression

I modelled the multimodal Gaussian distribution using a single polynomial regression. Instead of segmenting the data, I fit a third-degree polynomial to the entire dataset using NumPy's Polynomial.fit() function. This approach allowed for a smooth fit over the entire dataset, capturing the overall trend without breaks.

After fitting the model, I visualized the results by plotting the polynomial curve alongside the original data points. This gave a clear indication of how well the polynomial regression approximated the multimodal distribution.

However, **compared to the piecewise models, the single polynomial regression did not perform as well in terms of accuracy.**

The **RMSE** was **1.7844**, and the **R^2** value was **0.5190**, indicating that this model struggled to capture the complexities of the dataset.

Key Libraries and Functions Used:
- Polynomial.fit(): To fit the polynomial to the dataset.

Problem 6: Execution Time Comparison

To evaluate the efficiency of each model, I measured the execution time using Python's time module. This gave me an idea of how long each model took to fit the data, which is important when considering models for larger datasets.

Here are the results:
- **Piecewise Linear Regression**: 0.009199 seconds
- **Piecewise Polynomial Regression**: 0.001140 seconds
- **Polynomial Regression (Degree 3)**: 0.001791 seconds

Surprisingly, **the polynomial regression models were faster than the piecewise linear regression.** This could be due to the way the algorithms are implemented, with more complex models not always requiring more time.

Key Libraries and Functions:
- **time**

Problem 7: Ridge and LASSO Regression

Lastly, I applied Ridge and LASSO regression to the dataset. These are regularization techniques that help prevent overfitting by adding a penalty for large coefficients. Using the scikit-learn library, I fit both models and compared their performance to the polynomial regression model.

Here's a comparison of the results:

- **Ridge Regression**: RMSE = 2.6322, R^2 = -0.0022
- **LASSO Regression**: RMSE = 2.6315, R^2 = -0.0017
- **Polynomial Regression (Degree 3)**: RMSE = 1.7844, R^2 = 0.5190

Both Ridge and LASSO regression performed worse than the polynomial regression in this case, possibly because the dataset required more flexibility than these regularization models could provide.

Key Libraries and Functions:

- **scikit-learn** (from sklearn.linear_model import Ridge, Lasso)

CONCLUSION

In this assignment, I explored how different regression models perform on a complex, multimodal dataset. The **piecewise linear regression model stood out as the best in terms of accuracy, while the quadratic polynomial regression provided a smoother fit.** I also learned that the execution time for more complex models doesn't necessarily increase, which was an interesting takeaway. Finally, Ridge and LASSO regression were less effective for this dataset but remain valuable techniques for preventing overfitting in other contexts.