Senior Data Scientist Recruitment Test

1. Anomaly Detection

In a manufacturing plant, motor current data from production line equipment is being collected and stored in a real-time database. As a data scientist, you are tasked with designing a system that predicts equipment failures several days in advance and triggers alerts based on real-time streaming data. Two datasets of anomaly-related data have been shared for model development:

Dataset 1: Equipment failure occurred on December 15, 2021.

Dataset 2: Equipment failure occurred on December 30, 2021.

Tasks:

1.1. Develop a model in Python to identify anomalies and predict failures using the provided datasets.
- You may use third-party libraries but should be able to explain the inner workings of the algorithm(s) used.

1.2. Implement a scoring mechanism that ranks anomalies based on their severity or confidence level.

1.3. Implement your solution as a Python class containing methods such as fit and predict. Submit the implementation as a .py file.

1.4. Derive additional features (e.g., moving averages, rolling standard deviations, Fourier transforms) from the time-series data to enhance the anomaly detection model.Document the feature engineering process and justify the choice of features.

1.5. Demonstrate your model's performance on the given datasets.

1.6. Compare the performance of multiple anomaly detection algorithms (e.g., Isolation Forest, Autoencoders, One-Class SVM) on the datasets. Summarize the comparison in terms of accuracy, F1 score, and precision-recall balance.

2. Regression Analysis

A hydroelectric power plant maintains monthly records of water inflow to its dam. As a data scientist, you are required to forecast water inflow for the next five months based on data collected from **1999 to 2022**. Predictions must be generated every December for the subsequent five months.

Tasks:

2.1. Develop a regression model in Python to forecast water inflow.
- Use third-party libraries as needed but explain the algorithm(s) used comprehensively.

2.2. Implement at least two different approaches to the forecasting task: a traditional method (e.g., ARIMA) and a deep-learning-based method (e.g., LSTM). Compare their performance based on accuracy metrics such as RMSE, MAPE, and MAE.

2.3. Extend the model to provide prediction intervals or confidence intervals for the forecasts. Demonstrate how these intervals help in decision-making under uncertainty.

2.4. Implement your solution as a Python class containing fit and predict methods. Submit the implementation as a .py file.

2.5. Validate your model's performance on the provided dataset.