Assignment: Deep Learning

Task 1: Outlier detection with Autoencoder

Objective: The goal of this assignment is to detect outliers in credit card transactions using an autoencoder. You will identify a small set of transactions with a high probability of being fraudulent in a completely unsupervised manner.

Dataset: Use the following dataset: Credit Card Transactions Dataset. This dataset is highly imbalanced, with only a small fraction of transactions labeled as fraudulent.

Key concept: An autoencoder trained to reproduce the original transaction records will struggle to accurately reconstruct rare or unusual (potentially fraudulent) transactions. These transactions will have a high mean squared error (MSE) between their actual details and the details reconstructed by the autoencoder. Note that the autoencoder training is unsupervised.

Specific instructions:

a. Data Preparation:
- Load the dataset and perform an initial exploration.
- Split the data into training and test sets.
- Ensure that the values are appropriately scaled if necessary (e.g., using StandardScaler).

2. Autoencoder Model Definition and Training:
- Define an autoencoder model using the Keras framework.
- Compile the model using an appropriate loss function and optimizer.
- Train the model using the training set.
  - Use a validation split to monitor the model's performance.
  - Use an appropriate number of epochs and batch size.
- Save model and train history into files to avoid retraining the model multiple times.

3. Loss History Plot:
- Plot the training and validation loss histories.
- Confirm that the loss curves are converging and the model is not overfitting.
- Explain the plot results

4. Anomaly Detection:
- Apply the autoencoder to the test set and compute the reconstruction error (MSE) for each transaction.
- Identify transactions with high reconstruction errors as outliers.
- Present these transactions in the output

5. Setting a Threshold:
- Determine a threshold for anomaly detection.
  - Use methods such as the 95th or 99th percentile of the reconstruction errors to define an initial threshold.

    - Experiment with different threshold values.
6. Model Evaluation and Validation:
    - Measure the number of fraudulent transactions detected as the threshold changes.
    - Analyze how the detection rate and precision (fraction of actual frauds among suspected frauds) change with varying thresholds.
    - Plot graphs to show:
        - Detection Rate vs. Threshold
        - Precision vs. Threshold
        - Explain the results you got from these plots
7. Identifying the Optimal Threshold:
    - Use the plots to determine the optimal threshold that balances detecting a high number of frauds while keeping false positives low.
    - Explain what helped you decide on the optimal threshold
8. Discussion:
    - Discuss the advantages and deficiencies of using an autoencoder for fraud detection in credit card transaction data.

Task 2: Energy Consumption Forecasting

Background

In this assignment, you will act as a Data Scientist for a Smart Home analytics startup. Your goal is to build a robust predictive engine that forecasts future energy consumption based on historical usage patterns. You will utilize LSTM (Long Short-Term Memory) deep learning model.

Dataset:

You will use the Individual Household Electric Power Consumption dataset. It contains measurements of electric power consumption in one household with a one-minute sampling rate over a period of almost 4 years.
Key Variable: You should predict Global_active_power, the household global minute-averaged active power in kilowatts.

Objective:

Your primary task is to develop, train, and evaluate an LSTM model. Unlike standard regression tasks, time-series forecasting requires careful consideration of time horizons, data granularity, and model staleness.

Experiments:

Please address the following in your analysis:

- Data resolution. Raw sensor data often arrives at high frequencies, which may contain noise or be too granular for forecasting. Experiment with different time resolutions. Compare the

model's performance when training on the raw 1-minute data versus aggregating the data into hourly or daily averages.

- Look-back window: How does the size of the input window (the amount of past history fed into the model) affects performance?
- Prediction horizon: Configure your model to predict different future horizons (e.g. 1 or 7 days) and at different resolutions (e.g. hourly or daily). Note that the input resolution does not need to match the resolution of the predicted time series.
- Retraining: Simulate a production environment to test how model performance degrades over time. Based on this degradation, determine the optimal retraining schedule (e.g., weekly vs. monthly).

Report:

Conclude this assignment with a concise report providing clear performance analysis, visualizations of your forecasts, and strategic recommendations based on your experiments.