Mahindra University

**Computational sequential modeling (AI4101)**

Assignment

This assignment is mandatory work as an essentiaal learning activity. You may choose either one of the two. Option 1 OR Option 2

**Option 1:** Long-Term Forecasting with Hybrid Time Series Models for Financial Data

Objective

The objective of this assignment is to build a **hybrid forecasting model** that combines **ARIMA** (Auto-Regressive Integrated Moving Average) and **LSTM** (Long Short-Term Memory) to handle both short-term and long-term dependencies within a time series. This model is particularly suited to financial or volatile data, such as stock prices, due to the combination of ARIMA's capacity for trend capture and LSTM’s strength in handling complex, nonlinear patterns over extended periods.

Problem Statement

Design a forecasting model for predicting daily stock prices. Financial time series data often shows nonstationary trends, seasonality, and long-term dependencies. A hybrid ARIMA-LSTM model can help capture these distinct patterns. Your task is to:

-->Preprocess and split the time series data.

-->Build an ARIMA model to capture immediate patterns.

-->Build an LSTM model to capture long-term dependencies.

-->Combine both models for enhanced accuracy in forecasting.

-->Evaluate and compare this hybrid model with standalone ARIMA and LSTM models.

**Assignment steps:**

1. **Data collection and preprocessing:**

**i. Data Source**: Download historical daily stock prices for a selected company (e.g., from Yahoo Finance, source code for the same is provided at the end).

**ii. Data Cleaning**: Check for missing values, outliers, and perform any necessary transformations (like log transformations for nonstationarity).

**iii. Stationarity Check**: Use tests like the **Augmented Dickey-Fuller** (ADF) test to check for stationarity in the series. If nonstationary, differencing can be applied to remove trends.

1. **Building the ARIMA Model for short-term dependencies**
2. **ARIMA Model Setup**:
   * Use the ACF (Auto-Correlation Function) and PACF (Partial Auto-Correlation Function) plots to identify potential values for the AR (Auto-Regressive) and MA (Moving Average) parameters.
   * Build and train the ARIMA model on the preprocessed series.

**ii. Model Evaluation**:

* + Forecast the next few time steps and visualize the ARIMA model’s predictions against actual values.
  + Document performance metrics like Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) for the ARIMA model.

1. **Building the LSTM Model for Long-Term Dependencies**
2. **Data Preparation for LSTM**:
   * Split the data into a rolling window format with inputs and outputs to train the LSTM model effectively.
   * Normalize the data to ensure smooth gradient updates.

**ii. LSTM Model Setup**:

* + Build an LSTM network (with potential layers like Dense or Dropout to reduce overfitting).
  + Train the LSTM model using the training data, tuning the model’s hyperparameters (e.g., number of epochs, batch size, hidden units).

**iii. Model Evaluation**:

* + Evaluate LSTM on the test set and track similar metrics (MAE, RMSE) to enable later comparison with ARIMA.

1. **Combining ARIMA and LSTM Outputs**
2. **Hybrid Model Integration**:
   * Use ARIMA to predict the short-term component and LSTM for long-term predictions.
   * Possible methods of integration:
     + **Additive**: Combine ARIMA’s prediction for short-term trends with LSTM’s prediction for long-term effects by averaging or summing their outputs.
     + **Weighted Combination**: Assign weights to ARIMA and LSTM based on their respective prediction accuracies or use a linear regression to optimally blend them.
     + ARIMA residuals fed into LSTM

**ii. Hybrid Model Evaluation**:

* + Use the same test set to evaluate the hybrid model and compare its performance against standalone ARIMA and LSTM models.

1. Visualization and Analysis
2. **Visualize Results**:
   * Plot the predictions from ARIMA, LSTM, and the hybrid model alongside actual values to compare forecast quality visually.

**ii. Quantitative Analysis**:

* + Report final performance metrics (MAE, RMSE) and discuss which model performed best and why.
  + Explain how the hybrid model improved (or didn’t) forecasting accuracy compared to standalone models.

1. Deliverables:
   1. **Code and Model Files**: Submit the code for each model (ARIMA, LSTM, Hybrid) and any model files generated including **Plots and Visualizations (**Visualizations of the time series, ACF/PACF plots, and the performance of each model).
   2. **Report**:
      1. A detailed report that includes data preprocessing steps, model architectures, hyperparameter choices, and evaluation metrics for each model.
      2. An explanation of how the ARIMA and LSTM models complement each other and the observed effect in the hybrid model.

**Option 2:** Power Demand Forecasting with Integrated Weather Data Using Hybrid Models

**Problem Statement**

This assignment explores power demand forecasting using multivariate time series models by incorporating weather data. The Open Power System Data (OPSD) dataset contains hourly power demand and weather variables for France (FR), Germany (DE), Spain (ES), Italy (IT), Netherlands (NL), and Poland (PL). The link to the datasets is provided below:

NOTE: You only must consider data (power and weather) for these six countries (France (FR), Germany (DE), Spain (ES), Italy (IT), Netherlands (NL), and Poland (PL).) in your analysis.

Your task is to develop and evaluate **hybrid models** that combine classical, machine learning, and deep learning techniques for accurate forecasting of daily power demand. You will analyze historical data (January 1, 2015, to December 31, 2019) and create models to forecast demand for December 2019. You will also compare hybrid models' performance against standalone classical, machine learning, and deep learning models.

**Your report and implementation must include:**

1. Problem Understanding:

* Define the role of power demand forecasting in energy planning and management.
* Explain how weather variables (e.g., temperature, solar radiation) affect power demand.
* State the goal: Develop a hybrid model for forecasting daily power demand for December 2019.

2. Data Preparation:

* Load the OPSD dataset, focusing on the following:
  + Power demand for France (FR), Germany (DE), Spain (ES), Italy (IT), Netherlands (NL), and Poland (PL).
  + Link to power dataset: [time\_series\_60min\_singleindex\_filtered.csv](https://mahindraecolecentrale-my.sharepoint.com/:x:/g/personal/neeraj_choudhary_mahindrauniversity_edu_in/EUSKAAUtoy1Jt0_QrCr03BoBmy6YF3X6-B2yayd35Bzjag?e=PzlcEB)
  + Link to weather dataset: [weather\_data.csv](https://mahindraecolecentrale-my.sharepoint.com/:x:/g/personal/neeraj_choudhary_mahindrauniversity_edu_in/Ef1Sc-smkEVOnC5PJ9yyT0gBwmkiQ3QPq15tLl83ejj0qQ?e=4tFcyb)
  + In case above links do not open (<https://data.open-power-system-data.org/weather_data/2020-09-16>)
  + Time series: https://data.open-power-system-data.org/time\_series/2020-10-06
  + Weather variables such as temperature, solar radiation.
  + Filter data to the range **January 1, 2015, to December 31, 2019**.
  + Preprocess the data:
    - Handle missing values.
    - Normalize features for better model performance.
    - Split data into training (2015–2018) and validation/testing sets (2019) or based on your understandings.

3. Exploratory Data Analysis (EDA):

* Analyze trends and seasonality in power demand and weather data.
* Explore the relationship between power demand and weather variables using:
  + Correlation heatmaps.
  + Scatterplots and time series plots.
* Present insights on how weather impacts power demand in different countries.

4. Feature Engineering:

* Create additional features:
  + Lagged variables for power demand and weather.
  + Aggregated features (e.g., rolling averages).
  + Categorical features such as holidays and weekends.
* Highlight the importance of these features for forecasting.

5. Model Development:

* Implement and compare the following models:
  + **Classical Models**:
    - ARIMAX (ARIMA with weather data as exogenous variables).
  + **Machine Learning Models**:
    - Random Forest Regressor, SVR
    - Gradient Boosting (e.g., XGBoost or LightGBM).
  + **Deep Learning Models**:
    - Long Short-Term Memory (LSTM).
    - Gated Recurrent Units (GRU).
  + **Hybrid Models**:
    - Combine classical models (e.g., ARIMA residuals) with machine learning models.
    - Build hybrid models integrating deep learning with machine learning techniques (e.g., feature extraction with LSTM and prediction with Gradient Boosting) or combine classical models with machine learning models (e.g., ARIMA residuals fed into Random Forest/SVR/LSTM)

6. Model Evaluation

* Forecast daily power demand for December 2019 using each model.
* Compare the models using:
  + **Root Mean Squared Error (RMSE)**
  + **Mean Absolute Error (MAE)**
  + **Mean Absolute Percentage Error (MAPE)**
* Discuss how weather data improves forecasting accuracy.

7. Report:

Prepare a comprehensive report with:

1. Problem understanding and objectives.
2. Description of preprocessing and feature engineering.
3. Insights from EDA.
4. Model architecture and training.
5. Performance comparison of models, highlighting hybrid models.
6. Conclusions and potential improvements.

8. Deliverables:

1. **Code**: Submit a Jupyter Notebook or Python script that includes all steps, from data preparation to evaluation including **Plots and Visualizations**.
2. **Report**: Submit a PDF report (6–10 pages) summarizing the assignment.
3. **Comparison Table**: Include a table showing evaluation metrics for all models.

Bonus Challenges:

1. Experiment with ensemble models that combine predictions from different model types.
2. Analyze how removing certain weather variables impacts model performance.

Content of **GroupNumber.zip** file (Yellow highlights are folders; others are files). For OneDrive submission, upload to

<https://mahindraecolecentrale-my.sharepoint.com/:f:/g/personal/neeraj_choudhary_mahindrauniversity_edu_in/Enh-H2FhafVPutr9TpvQjWUBIAenJ05jDAvN_V_l_PY1_Q>

+--- **GroupNumber**

| +--- Code

| | +--- all\_codes\_R\_py\_Matlab\_here.py (Can be multiple files too)

| | +--- **GroupNumber** \_code.ipynb

| +--- Report

| | +--- **GroupNumber** \_report.pdf

| | +--- **GroupNumber**\_report\_source\_docx\_latex/word file

| | +---**README** file stating the contribution of each group member in percentage terms

**This work is due 28th November 23:59 PM (firm deadline). (Thursday)**

Sample code to download dataset. You can try other stock datasets too.

###############################################

#download data from yahoo! finance

###############################################

##data source: http://finance.yahoo.com/

"""

from pandas\_datareader import data as pdr

from datetime import datetime

#download data

ibm = pdr.DataReader('IBM', 'yahoo', start=datetime(2014, 8, 1), end=datetime(2016, 11, 30))

aapl = pdr.DataReader('AAPL', 'yahoo', start=datetime(2014, 8, 1), end=datetime(2016, 11, 30))

fb = pdr.DataReader('FB', 'yahoo', start=datetime(2014, 8, 1), end=datetime(2016, 11, 30))

googl = pdr.DataReader('GOOGL', 'yahoo', start=datetime(2014, 8, 1), end=datetime(2016, 11, 30))

#print first few lines of data

print(ibm.head())

print(aapl.head())

print(fb.head())

print(googl.head())

#export and save as csv files

ibm.to\_csv('IBM\_stock.csv', sep=',')

aapl.to\_csv('Apple\_stock.csv', sep=',')

fb.to\_csv('Facebook\_stock.csv', sep=',')

googl.to\_csv('Google\_stock.csv', sep=',')