Time Series Analysis in Python

This repository contains a series of analysis, transforms and forecasting models frequently used when dealing with time series. The aim of this repository is to use the Consumer Price Index dataset and understand the various aspects of the inherent nature of the series so that we are better informed and take meaningful inferences and build accurate forecasts.

Dataset

The Dataset used is the Consumer Price Index (monthly, seasonally adjusted) during the calendar years 2010 to 2021 as provided by the Government of Canada. The data from the food column is pulled and pre-processed to ensure easier data handling and to save time. In this case, the series is already stationary with some next to no seasonalities which change every year.

Statistics Canada. Table 18-10-0006-01 Consumer Price Index, monthly, seasonally adjusted
DOI: https://doi.org/10.25318/1810000601-eng

Analysis and transforms

- Time series decomposition
    - Level
    - Trend
    - Seasonality
    - Noise
    - Lag

- Stationarity
    - AC and PAC plots
    - Moving Averages
    - Dickey-Fuller test

- Making our time series stationary
    - Smoothing
    - Moving averages

Models tested

ARIMA, short for ‘Auto-Regressive Integrated Moving Averages’ is actually a class of models that ‘explains’ a given time series based on its own past values, that is, its own lags and the lagged forecast errors, so that equation can be used to forecast future values.

A pure **Auto-Regressive (AR only) model** is one where Yt depends only on its own lags. That is, Yt is a function of the ‘lags of Yt’.

$$Y_t = \alpha + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + .. + \beta_p Y_{t-p} + \epsilon_1$$

Likewise a pure **Moving Average (MA only) model** is one where Yt depends only on the lagged forecast errors.

$$Y_t = \alpha + \epsilon_t + \phi_1 \epsilon_{t-1} + \phi_2 \epsilon_{t-2} + .. + \phi_q \epsilon_{t-q}$$

where the error terms are the errors of the autoregressive models of the respective lags. The errors Et and E(t-1) are the errors from the following equations :

$$Y_t = \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + .. + \beta_0 Y_0 + \epsilon_t$$

$$Y_{t-1} = \beta_1 Y_{t-2} + \beta_2 Y_{t-3} + .. + \beta_0 Y_0 + \epsilon_{t-1}$$

An ARIMA model is one where the time series was differenced at least once to make it stationary and you combine the AR and the MA terms. So the equation becomes:

$$Y_t = \alpha + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + .. + \beta_p Y_{t-p} \epsilon_t + \phi_1 \epsilon_{t-1} + \phi_2 \epsilon_{t-2} + .. + \phi_q \epsilon_{t-q}$$

Getting Started

Dependencies

- Python IDE or Python notebook (Jupyter or Google Colab)
- Libraries needed: Pandas, NumPy, MatPlotLib, Statsmodels, Pmdarima, SciPy, Sci-kit Learn

Installing

- Clone the repository
  `git clone https://github.com/microsoft/forecasting`

  `cd forecasting/`

- Run setup scripts to create conda environment. Please execute one of the following commands from the root of the Forecasting repo based on your operating system.
  - Linux

  `./tools/environment_setup.sh`

  - Windows

  `tools\environment_setup.bat`

  (Note that for Windows you need to run the batch script from Anaconda Prompt. The script creates a conda environment `forecasting_env` and installs the forecasting utility library `fclib`.)

- Start the Jupyter notebook server
  `jupyter notebook`

- Run the LightGBM single-round notebook under the `00_quick_start` folder. Make sure that the selected Jupyter kernel is `forecasting_env`.

- If you have any issues with the above setup, or want to find more detailed instructions on how to set up your environment and run examples provided in the repository, on local or a remote machine, please navigate to the Setup Guide.

Authors

Vibhuti Gandhi

Discord: ginko#5274

LinkedIn: https://www.linkedin.com/in/vibhuti-gandhi-8679ba1bb/