Project Report

Bayesian Statistics

Eshika Jagani

Time Series Analysis of Apple Stock Prices using Gaussian Process

Topic 1: Introduction

Time Series: A data is referred to as time series data if it consists of sequence of data points distributed over regular interval of time. It helps to track changes, patterns and trends in a variable over certain time intervals. Two types of models can be prepared using time series data; namely Time Series Forecast and Time Series Analysis. Time Series Forecast is the use of model to predict future values based on previously observed values whereas, Time Series Analysis comprises methods for analyzing time series data in order to extract meaningful data. I am working on stock prices of Apple Inc. (referring it as AAPL later in report) (which is a time series data) and performing Time Series Analysis on it. Since the values of stock prices are dependent on various sentiments (like product launching, sales, offers, etc.), its future prediction cannot be done using previous (past) data. Thus, I am implementing Time Series Analysis only.

From Wikipedia information:

- A number of different notations are in use for time-series analysis. A common notation specifying a time series *X* that is indexed by the natural numbers is written as:
  - $X = (X_1, X_2, \dots)$
  - Another common notation is:
  - $Y = (Y_t : t \in T)$, where T is the index set

Topic 2: Data

The data that I am using consists of stock prices of AAPL. I am fetching this data from Yahoo Finance page. There is a package for Yahoo Finance API namely *yfinance* which I installed using pip install *yfinance* command in Anaconda Prompt. I am downloading the stock prices starting from Jan 1, 2023, to Dec 31, 2023. *Adj Close* refers to Adjusted Closing Price after adjustments for all applicable splits and dividend distributions. The stock prices are displayed based on the dates. According to New York Stock Exchange calendars, there are total 252 trading days per year. Trading days are Monday through Friday. From Wikipedia, the NYSE and NASDAQ average about 252 trading days a year. This is from 365.25 (days on average per year) * 5/7 (proportion workdays per week) - 6 (weekday holidays) - 4*5/7 (fixed date holidays) = 252.03 ≈ 252. (My code file: Project_AAPL_Analysis)

Topic 3: Model Overview

Topic 3.1: Seasonal-Trend decomposition using LOESS (STL):

- According to the documentation of STL decomposition in statsmodel:
- STL decomposes a time series into 3 components: Trend, Seasonal and Residual
- It uses LOESS (locally estimated scatterplot smoothing) to extract smooths estimates of the three components
- The key inputs into STL are:
    - Season - The length of the seasonal smoother. Must be odd. (took 13 in my case)
    - Trend - The length of the trend smoother, usually around 150% of season. Must be odd and larger than season.
    - Low_pass - The length of the low-pass estimation window, usually the smallest, odd number larger than the periodicity of the data.
- According to the information on Geeks for Geeks:
- Seasonal decomposition aids in statistical analysis by providing a clearer picture of the structure of the time series.
- The fundamental components are discussed below:
- Trend: The underlying long-term progression or direction in the data.
- Seasonal: The repeating patterns or cycles that occur at fixed intervals like daily, monthly or yearly.
- Residual: The random fluctuations or noise in the data that cannot be attributed to the trend or seasonal patterns.
- According to Wikipedia, the mathematical notation for STL decomposition is given by:
    - Time series using Additive Model: $y_t = T_t + C_t + S_t + I_t$
        - (The additive model is useful when the seasonal variation is relatively constant over time.) (from the PennState Eberly College of Science)
    - Time series using Multiplicative Model: $y_t = T_t * C_t * S_t * I_t$
        - (The multiplicative model is useful when the seasonal variation increases over time.) (from the PennState Eberly College of Science)
    - Where,
    - T_t is the Trend Component at time t, it reflects the long-term progression of the series. A trend exists when there is a persistent increasing or decreasing direction in the data. The trend component does not have to be linear
    - C_t is the Cyclical Component at time *t*, it reflects repeated but non-periodic fluctuations. The duration of these fluctuations depends on the nature of the time series.
    - S_t is the Seasonal Component at time *t*, reflecting seasonality (seasonal variation). A seasonal pattern exists when a time series is influenced by seasonal factors. Seasonality occurs over a fixed and known period (e.g., the quarter of the year, the month, or day of the week).
    - I_t is the irregular component (or "noise") at time *t*, it describes random, irregular influences. It represents the residuals or remainder of the time series after the other components have been removed.

- Since in this case, I am dealing with only Trend, Seasonality and Residual (Noise) of annual stock prices, the cyclicality can be ignored since seasonality handles the seasonal fluctuations.
- Thus, the model in this case becomes: (from the PennState Eberly College of Science)
    - $y_t = T_t + S_t + I_t$

Topic 3.2: Gaussian Process Model and its Implementation

- From the pymc.io documentations for Gaussian Process
- Sometimes an unknown parameter or variable in a model is not a scalar value or a fixed-length vector, but a *function*. A Gaussian process (GP) can be used as a prior probability distribution whose support is over the space of continuous functions. A GP prior on the function $f(x)$ is usually written as,
    - $\boldsymbol{f(x) \sim GP(m(x), k(x, x'))}$
- The function values are modeled as a draw from a multivariate normal distribution that is parameterized by the **mean function, $\boldsymbol{m(x)}$**, and the **covariance function, $\boldsymbol{k(x, x')}$**.
- Gaussian processes are a convenient choice as priors over functions due to the marginalization and conditioning properties of the multivariate normal distribution.
- Usually, the marginal distribution over $f(x)$ is evaluated during the inference step.
- The conditional distribution is then used for predicting the function values $f(x_*)$ at new points, x_*
- The joint distribution of $f(x)$ and $f(x_*)$ is multivariate normal,
    - $\begin{bmatrix} f(x) \\ f(x_*) \end{bmatrix} \sim N\left(\begin{bmatrix} m(x) \\ m(x_*) \end{bmatrix}, \begin{bmatrix} k(x, x') & k(x_*, x) \\ k(x_*, x) & k(x_*, x_*') \end{bmatrix}\right)$
- Starting from the joint distribution, one obtains the marginal distribution of $f(x)$, as $N(m(x), k(x, x'))$. The conditional distribution is
    - $f(x_*)|f(x) \sim N(k(x_*, x)k(x, x)^{-1}[f(x) - m(x)] + m(x_*),\ k(x_*, x_*) - k(x, x_*)k(x, x)^{-1}k(x, x_*))$
- Since in GP distribution, for mean $m(x)$, I am considering 0.
- And for covariance, $k(x, x')$ I am considering pm.gp.cov.ExpQuad
- It is the Exponentiated Quadratic kernel and is also referred to as the Squared Exponential, or Radial Basis Function kernel. (from pymc docs) and the notation is given as:
    - $k(x, x') = \exp\{-\frac{(x-x')^2}{2l^2}\}$
    - Where, *l* is the length scale which is passed as ls in the arguments
- **For Trend Component:**
- Since the data shows too many fluctuations, I am adding eta**2 (from GP docs)
- The product (or sum) of a covariance function with a scalar is a covariance function:
    - cov_func = eta**2 * $k(x, x')$
    - cov_func = eta**2 * pm.gp.cov.ExpQuad(…)
    - here eta is amplitude which deals with the hyperparameter tuning and smooths out the unevenness since trends represent a smoother curve than the real plots
- For Priors:

    - ls is the length scale, assigning Gamma prior
    - eta is the amplitude to smooth the curve
    - noise is the noise to add the variation the data has, assigning it half normal so as to not to deal with variance and directly fetching standard deviation
    - cov_func is the Exponential Quadratic kernel which takes ls
    - gauss is the Gaussian function which takes mean_func and cov_func. By default, mean_func is zero so not specifying, and passing cov_func kernel to cov_func
- For Likelihood:
    - Using marginal likelihood and passing the values of trend variable to it which was set after the STL decomposition and noise
- **For Seasonal Component:**
- **Fast Fourier Transform (FFT)**: Fourier analysis is a method for expressing a function as a sum of periodic components, and for recovering the signal from those components. When both the function and its Fourier transform are replaced with discretized counterparts, it is called the discrete Fourier transform (DFT). The DFT has become a mainstay of numerical computing in part because of a very fast algorithm for computing it, called the Fast Fourier Transform (FFT), which was known to Gauss (1805) and was brought to light in its current form by Cooley and Tukey.
- Using FFT for the analysis of Seasonal Component
- The FFT *y[k]* of length N of the length-*N* sequence *x[n]* is defined:
    - $$y[k] = \sum_{n=0}^{N-1} e^{-2\pi j \frac{kn}{N}} x[n] \,,$$
- The function fftfreq returns the FFT sample frequency points.
- FFT plots based on daily, weekly, monthly cycles (considering daily cycles)
- In the Gaussian Process Model, I am using the same model as for the Trends component, with some changes in the covariance function kernel
- Here, the cov_func uses eta**2 * pm.gp.cov.Periodic(…) instead of ExpQuad
- And, instead of Marginal, I'm using MarginalSparse in order to make the model run faster.

Topic 3.3: Residual Component using GARCH Model:

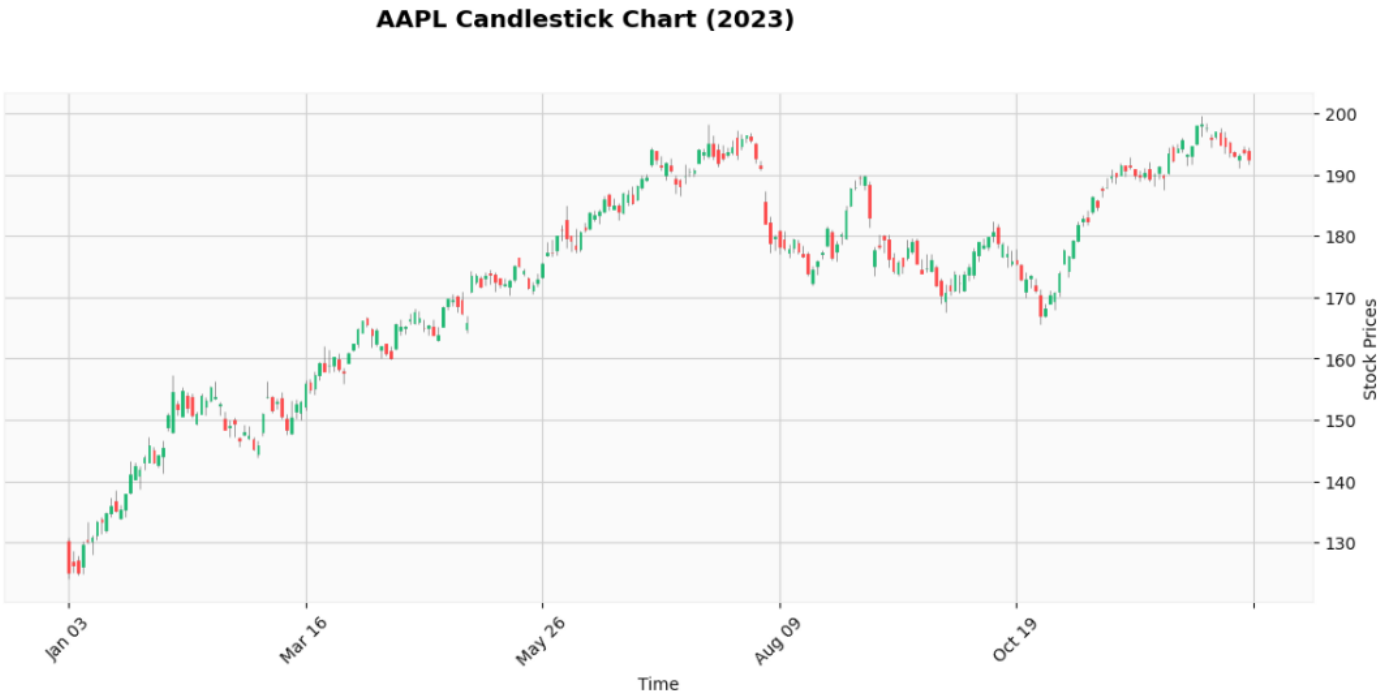
- **Generalized Autoregressive Conditional Heteroskedasticity (GARCH) Model**: (from ARCH model docs) ARCH models are a popular class of volatility models that use observed values of returns or residuals as volatility shocks. A basic GARCH model is specified as
    - $$\begin{aligned} r_t &= \mu + \epsilon_t \\ \epsilon_t &= \sigma_t e_t \\ \sigma_t^2 &= \omega + \alpha \epsilon_{t-1}^2 + \beta \sigma_{t-1}^2 \end{aligned}$$
- A complete ARCH model is divided into three components:
    - a mean model, e.g., a constant mean or an ARX
    - a volatility process, e.g., a GARCH or an EGARCH process
    - a distribution for the standardized residuals.

- In this case, using GARCH model for volatility process
- In this, I am not using Gaussian Process Model since the GARCH model does it all.
- Then I am comparing the actual Residual component with the GARCH model output using a histogram

Topic 4: Analyzing the Plots

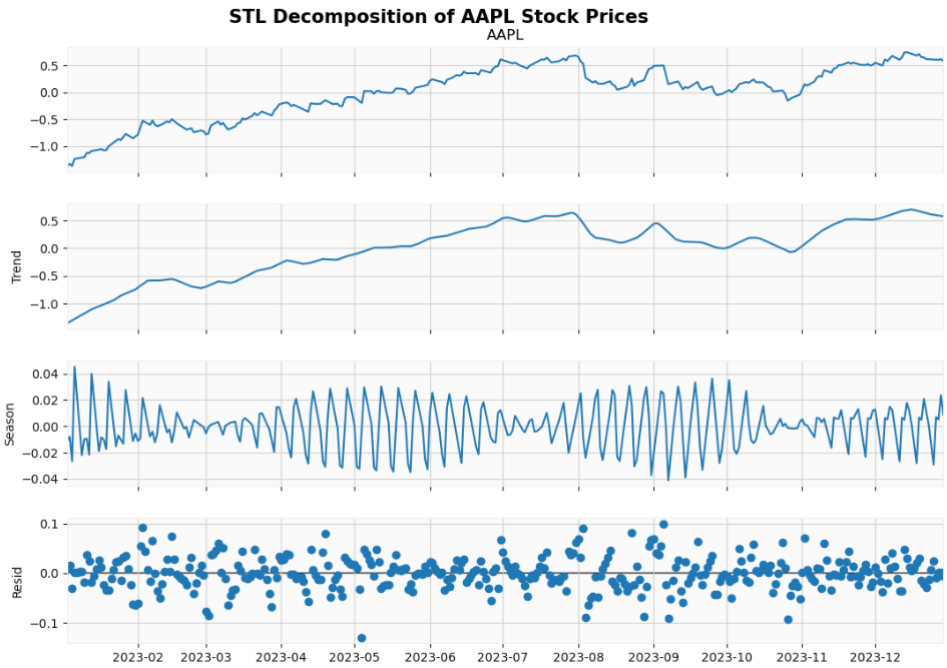
This topic consists of all the plots and its corresponding analysis

Topic 4.1: Candlestick Chart of the Original Data



- This candlestick chart is visualizing the movements in AAPL's stock prices (2023)
- There are 2 types of candlesticks: green and red
- Green candlesticks: These represent that the market is bullish
- Red candlesticks: These represent that the market is bearish
- These candlesticks are representing each day's data
- The wicks indicate the highest and lowest prices of the day.
- The bar in the candlesticks represent the starting and closing price for that day
- Overall, the January to July, the stock prices are going up and there is a down from July to August. Later, the prices are not uniform till October end. After October, again there is an increase in the stock prices.

Topic 4.2: STL Decomposition (using Standardized Data)



- STL Decomposition gives 4 plots:
  - **Original Data Plot**: this is the plot where the actual data points are plotted. In this example, the values of stock prices are rising till July end, there is a little drop till the end of October and again it increases
  - **Trend Plot**: this is the plot showing the annual trend smoothing out the spikes. It mimics the original data plot but smooths out the peaks and troughs giving a flow.
  - **Seasonal Plot:** this plot shows repeating patterns/cycles within a fixed interval of time based on the seasonal argument. Since this data consists of 12 months, I have specified 13 (as it takes only odd values).
  - **Residual Plot:** this plot gives points which represent the fluctuations or noise in the data

Topic 4.3: Gaussian Process Model using Bayesian Statistics for Trend Component

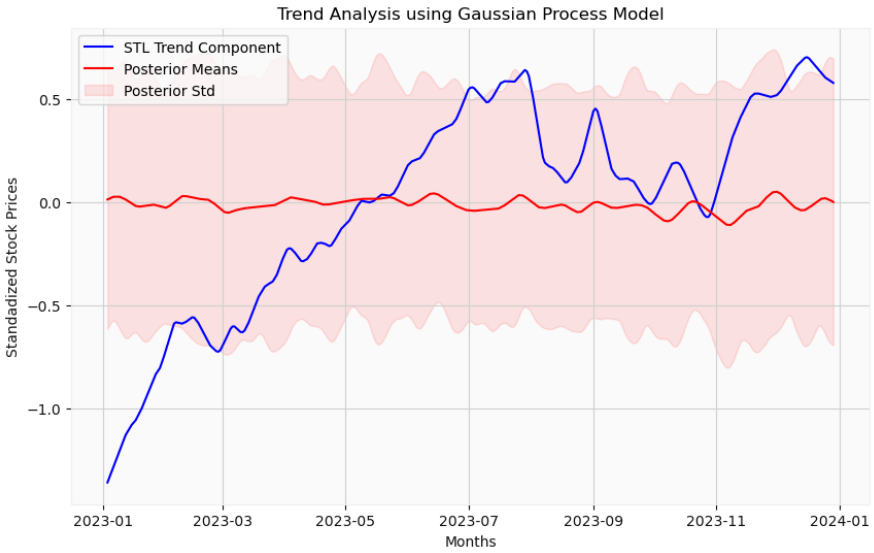
- **Summary of Trends Model**

	mean	sd	hdi_2.5%	hdi_97.5%	mcse_mean	mcse_sd	ess_bulk	ess_tail	r_hat
length_scale	6.470	0.097	6.269	6.649	0.003	0.002	1244.0	1423.0	1.0
eta	0.384	0.037	0.311	0.455	0.001	0.001	1234.0	1434.0	1.0
noise	0.001	0.000	0.001	0.001	0.000	0.000	1376.0	1094.0	1.0

  - This is the summary of implementing the trend component with Gaussian Process using PYMC (Bayesian) model.
  - This gives means, standard deviation, 95% HDI credible intervals for the three variables namely:
    - length_scale: this parameter controls the smoothness of the model.
    - eta: this parameter is the amplitude η which helps in scaling the values
    - noise: this parameter indicates the uncertainty. In this case, there is very little to no noise in the data

- All the 3 variables have very less standard deviation indicating that the variation is very less
- HDI credible intervals of length_scale and eta do not contain 0 in it indicating that the data is statistically significant. But that for noise is close to zero showing very little significance

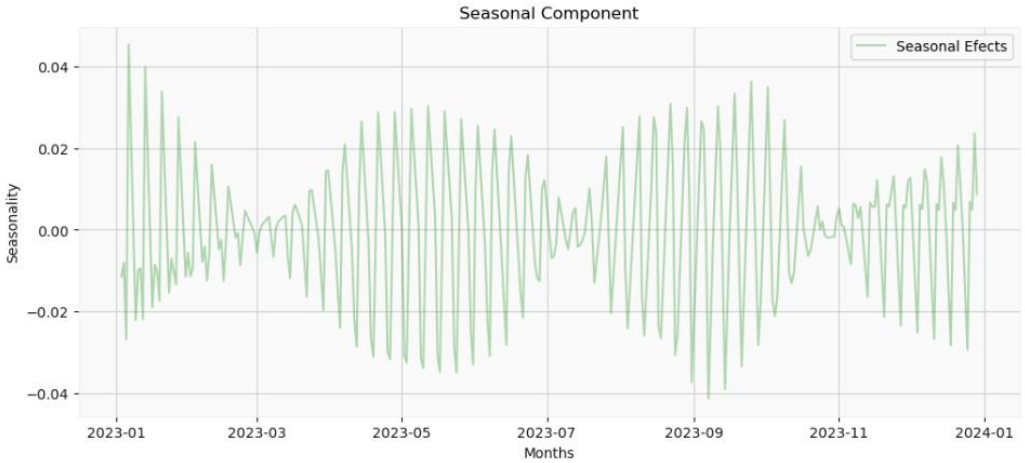
- **Plotting the Posterior Means and Standard Deviations vs Trend Component**



- This is the chart plotting the posterior means (red line) and posterior standard deviations (the shaded region) with the original data
- This chart, initially the prices start show rise for a long time, then around July the values get close to the upper bound of the standard deviation, then get close to mean and again rise.
- For data sets like INR=X (foreign currency), GC=F (Gold Prices), etc. unlike this data, show movements around the mean and show less deviation.

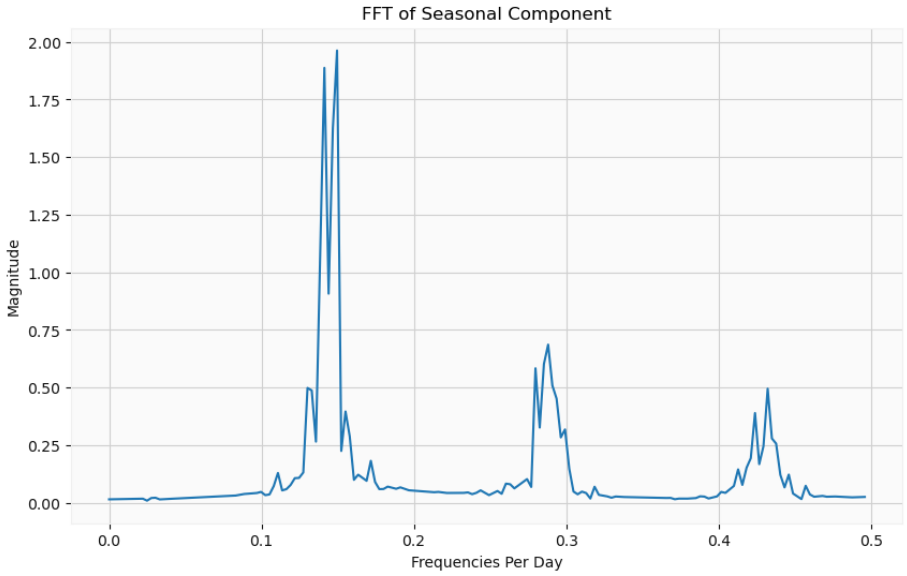
Topic 4.4: Gaussian Process Model using Bayesian Statistics for Seasonal Component

- **Seasonal Component Chart**



- This is the plot for the seasonal component from the STL decomposition
- This chart shows the seasonal effects on the stock prices over time and generates repetitive cycles.
- The first cycle is from Jan to end of Feb, the second is from around mid of March to July, third is from around end of Aug to around end of Oct, and the last one is from early Nov onwards.
- In between the cycles reduce and are around the (horizontal) center

- **Chart showing FFT for the Seasonal Component**



- This is a plot using Fast Fourier Transform for the seasonal component
- It shows the dominant peaks over time (in this example per day)
- This plot shows 3 significant peaks indicating 3 period of times where the magnitude rises
- From left to right, the peaks show a decreasing strength.
- Higher the peak, higher the seasonality and drastic changes in the prices
- And lower the peak, lower the seasonality and lesser the changes in the prices

- **Summary of Seasonal Model**

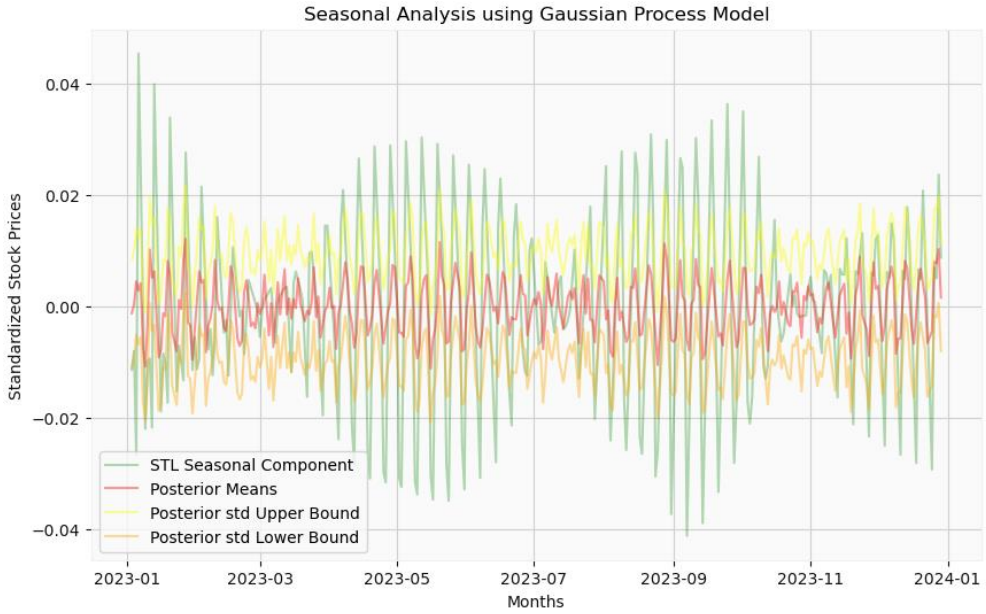
	mean	sd	hdi_2.5%	hdi_97.5%	mcse_mean	mcse_sd	ess_bulk	ess_tail	r_hat
eta	1.663	1.355	0.012	3.877	0.654	0.498	5.0	31.0	2.82
period	2.920	2.267	1.049	6.710	1.111	0.848	5.0	34.0	2.86
length_scale	3.152	1.343	1.511	5.788	0.555	0.414	8.0	58.0	1.69
noise	0.014	0.001	0.013	0.016	0.000	0.000	7.0	20.0	1.55

- This is the summary of implementing the trend component with Gaussian Process using PYMC (Bayesian) model.
- This gives means, standard deviation, 95% HDI credible intervals for the three variables namely:

- eta: this parameter is the amplitude η which helps in scaling the values
- period: this parameter is to include the duration (in this example 12 months)
- length_scale: this parameter controls the smoothness of the model.
- noise: this parameter indicates the uncertainty. In this case, there is very little to no noise in the data

- All the 4 variables have very less standard deviation indicating that the variation is very less
- HDI credible intervals of length_scale, period and eta do not contain 0 in it indicating that the data is statistically significant. But that for noise is close to zero showing very little significance

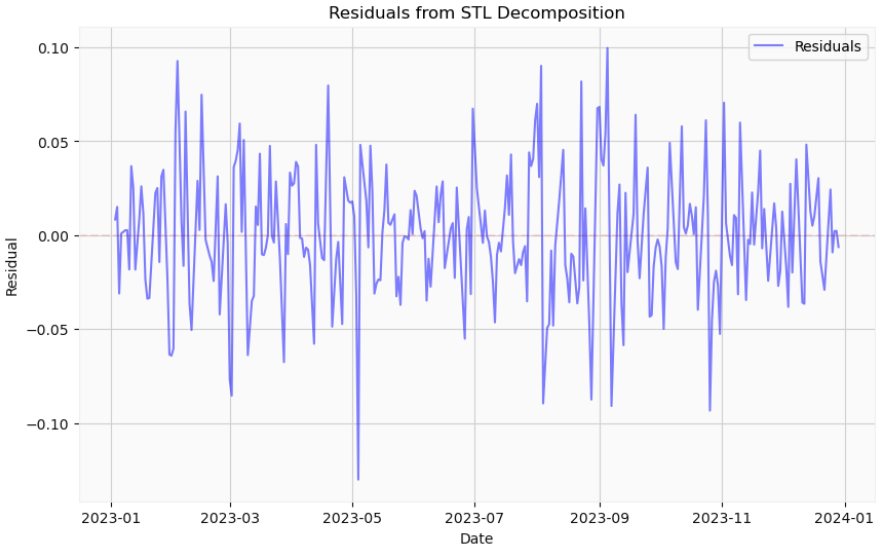
- **Plotting the Posterior Means and Standard Deviations vs Seasonal Component**



- The model performs very well and gives the mean and standard deviation
- The green line indicates the actual seasonal component, red line indicates the mean, yellow line indicates the upper bound for the posterior standard deviation and orange line indicates the lower bound for the posterior standard deviation.
- The mean corresponds the horizontal center and also has cycles just like the seasonal component
- The upper and lower bounds of standard deviation are not exact but almost close to the seasonal component showing a good performance
- Thus, the model performs well with the seasonal component

Topic 4.5: Residual Analysis using GARCH Model

- **Residual Component Chart**



  - This is plot of the residual variable from the STL decomposition
  - The residuals are the noise part of the data, and thus its plot also looks the noise in the stream of audio.
  - The STL decomposition has set the mean to zero and the residuals tend to be vary around the mean.
  - The variations look closely packed indicating the changes in very short period of time

- **GARCH Model Summary**

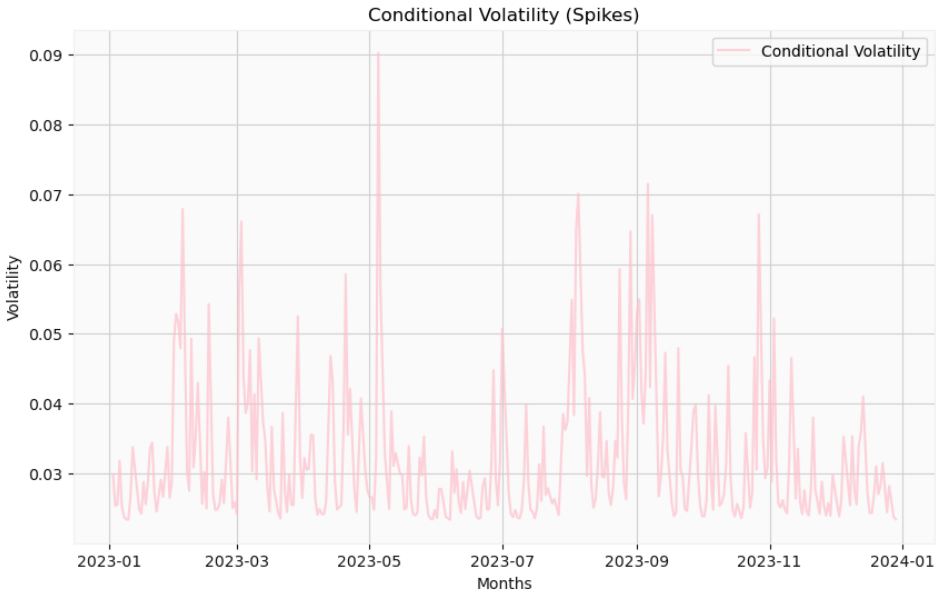
```
                     Constant Mean - GARCH Model Results
==============================================================================
Dep. Variable:                  resid   R-squared:                       0.000
Mean Model:             Constant Mean   Adj. R-squared:                  0.000
Vol Model:                      GARCH   Log-Likelihood:                740.949
Distribution:                  Normal   AIC:                           -1473.90
Method:            Maximum Likelihood   BIC:                           -1458.34
                                        No. Observations:                  361
Date:                Tue, Nov 26 2024   Df Residuals:                      360
Time:                        21:49:37   Df Model:                            1
                            Mean Model
==============================================================================
                 coef    std err          t      P>|t|       95.0% Conf. Int.
------------------------------------------------------------------------------
mu         8.4955e-04  1.859e-03      0.457      0.648 [-2.794e-03,4.493e-03]
                          Volatility Model
==============================================================================
                 coef    std err          t      P>|t|      95.0% Conf. Int.
------------------------------------------------------------------------------
omega      4.2487e-04  1.141e-04      3.723  1.968e-04 [2.012e-04,6.485e-04]
alpha[1]       0.4363      0.155      2.823  4.754e-03     [  0.133,  0.739]
beta[1]        0.2170      0.149      1.457      0.145  [-7.490e-02,  0.509]
==============================================================================

Covariance estimator: robust
```

  - The summary gives various information regarding the GARCH model like:
    - Mean model is constant mean, volatility model is GARCH, distribution is normal, values for log-likelihood, AIC, BIC, number of observation etc.
  - There are 2 models:
    - Mean model: showing estimate of coefficients, standard errors etc. for mu

- Volatility model: showing estimate of coefficients, standard errors etc. for omega, alpha and beta values

- **Conditional Volatility Chart**



  - This plot shows the conditional volatility derived by the GARCH model
  - This plot has many spikes, some are high indicating high volatility, and some are low indicating low volatility
  - This chart plays a crucial role in when it comes to analyzing market risks
  - When there is a change in sentiments like product failure would lead to drastic drop in stock prices which would show very high spikes
  - Similarly, in case of new product launch with the best ever features, the stocks are going to shoot up. In this case as well, the chart will show high spikes
  - In this example, spikes in May are high and comparing it with original chart, this indicates huge growth in the prices. Thus, the GARCH model fits well with the Residual component of the data

Topic 5: Conclusion

At first, fetching the data from the Yahoo Finance API. Then I am plotting a Candlestick Chart for the original data. The data being too huge, I am standardizing the data to avoid any divergences occurring after the model is ran (this is totally optional). After standardizing, I am performing STL decomposition. Then, I am performing analysis on the 3 components from the STL decomposition, namely Trend, Seasonal and Residual. For trend component the Gaussian Process model performs good and gives posterior mean and standard deviation. But the Gaussian Process model perform very well enough with the Seasonal component and gives posterior mean and standard deviation which fits well with the actual data. For the Residual component, GARCH model fits well with the data.

References

1. PYMC docs:
    a. https://www.pymc.io/projects/docs/en/latest/api/distributions/generated/pymc.HalfNormal.html
    b. https://www.pymc.io/projects/docs/en/stable/api/generated/pymc.sample.html
2. Gaussian Process:
    a. Gaussian Processes: https://www.pymc.io/projects/docs/en/latest/learn/core_notebooks/Gaussian_Processes.html
    b. pm.gp.cov.ExpQuad: https://www.pymc.io/projects/docs/en/latest/api/gp/generated/pymc.gp.cov.ExpQuad.html
    c. pm.gp.Marginal (a very good example code where I got the inspiration from!): https://www.pymc.io/projects/docs/en/stable/api/gp/generated/pymc.gp.Marginal.html
    d. pm.gp.Marginal.marginal_likelihood: https://www.pymc.io/projects/docs/en/latest/api/gp/generated/classmethods/pymc.gp.Marginal.marginal_likelihood.html
    e. pm.HalfNormal: https://www.pymc.io/projects/docs/en/latest/api/distributions/generated/pymc.HalfNormal.html
    f. pm.gp.MarginalSparse: https://www.pymc.io/projects/examples/en/latest/gaussian_processes/GP-SparseApprox.html
    g. pm.gp.cov.Periodic: https://www.pymc.io/projects/docs/en/stable/api/gp/generated/pymc.gp.cov.Periodic.html
3. STL:
    a. STL Decomposition: https://www.statsmodels.org/dev/examples/notebooks/generated/stl_decomposition.html
    b. STL Decomposition formula:
        i. https://online.stat.psu.edu/stat510/lesson/5/5.1#:~:text=The%20following%20two%20structures%20are,%3A%20%3D%20Trend%20*%20Seasonal%20*%20Random
        ii. https://en.wikipedia.org/wiki/Decomposition_of_time_series
    c. statsmodels.tsa.seasonal.STL: https://www.statsmodels.org/dev/generated/statsmodels.tsa.seasonal.STL.html
    d. to deal with missing values: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.asfreq.html
    e. Information: https://www.geeksforgeeks.org/seasonal-decomposition-of-time-series-by-loess-stl/

4. pm.sample_posterior_predictive: https://www.pymc.io/projects/docs/en/latest/api/generated/pymc.sample_posterior_predictive.html
5. Fast Fourier Transform:
    a. https://en.wikipedia.org/wiki/Fast_Fourier_transform
    b. ftt, fttfreq: https://docs.scipy.org/doc/scipy/tutorial/fft.html
6. GARCH Model: https://arch.readthedocs.io/en/stable/univariate/introduction.html
7. Yahoo Finance:
    a. https://pypi.org/project/yfinance/
    b. https://finance.yahoo.com/
    c. Apple Stock Prices: https://finance.yahoo.com/quote/AAPL/
8. New York Stock Exchange:
    a. https://www.nyse.com/markets/hours-calendars
    b. https://en.wikipedia.org/wiki/Trading_day#:~:text=The%20NYSE%20and%20NASDAQ%20average,Day%2C%20Martin%20Luther%20King%20Jr.
9. data.columns.get_level_values: https://pandas.pydata.org/docs/reference/api/pandas.Index.get_level_values.html
10. Matplotlib:
    a. https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.fill_between.html
    b. https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.plot.html
11. Course Videos and Slides