Bitcoin Modeling & Simple Moving Average

Contents

[Environment Setup 1](#_Toc137464659)

[Data 2](#_Toc137464660)

[Simple Moving Average 2](#_Toc137464661)

# Environment Setup

Jupyter Lab

* Start Anaconda Promp (anaconda3)
* Run command: Jupyter lab --notebook-dir="D:\projects"
  + This will start the Jupyter environment
* Required libraries for notebooks:
  + Import pandas as pd
  + Import numpy as np
  + Import bokeh
  + From bokeh.models import HoverTool
  + From bokeh.models import ColumnDataSource
  + Import matplotlib.pyplot as plt

A screenshot of a computer

Description automatically generated

File Structure

* Previously saved data tables are saved as Pickle files (.pkl)
  + Use name convention btc\_xxx.pkl
* Btc\_pickle.ipynb notebook for creating pickle files
* Data Analysis notebook named as btcX.ipynb
* All notebooks should be saved in D:\projects\crypto
* Raw data is saved in folder D:\projects\crypto\data

# Data

Bitcoin historical data can be obtained from finance.yahoo.com. Data ranges from Sep 15, 2014 to Jun 11, 2023.

A picture containing text, line, plot, screenshot

Description automatically generated

# Simple Moving Average

Objective is to run an automated scheduled task. The task should provide an analysis of whether the price of Bitcoin has “significantly” increased on dropped. If a “significant” change is detected, the task should send an e-mail alert to the subscriber.

A significant change can be measured by using the simple moving average. For example, a 30-day moving average change graph shows points where significant increases or decreases occur within a 30-day average.

btc\_df["ROL\_DIFF"] = btc\_df["Close"] - btc\_df["SMA30"]

A picture containing text, line, plot, screenshot

Description automatically generated

By looking at the largest increases, we can see which days saw the biggest increase from the 30-day moving average.

A screenshot of a computer screen

Description automatically generated with medium confidence

By looking at the largest decreases, we can see which days saw the biggest decreases from the 30-day moving average.

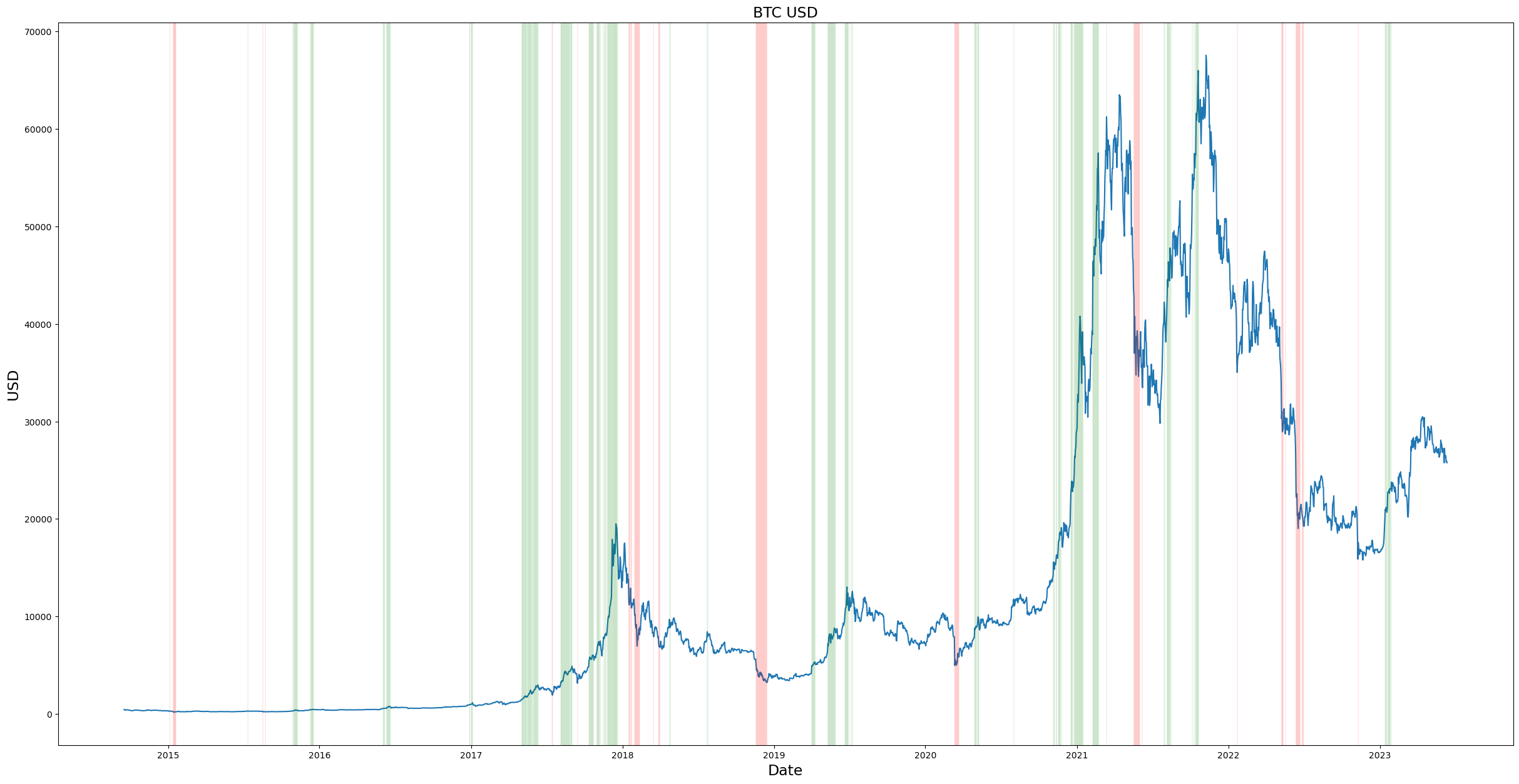
A screenshot of a computer

Description automatically generated with medium confidence

Green lines show dates that had a significant increase relative to the past 30 days.

Red lines show dates that had significant decrease relative to the past 30 days.

\*Significant change is defined here as +/- 20%



# Data Scraping

Data can be obtained from Yahoo! Finance following this guide: [3 Simple Ways To Obtain Cryptocurrency Data In Python | by Jason LZP | Geek Culture | Medium](https://medium.com/geekculture/3-simple-ways-to-obtain-cryptocurrency-data-in-python-f45b9d603a97)

Install yfinance module in anaconda (in base environment):

pip install yfinance

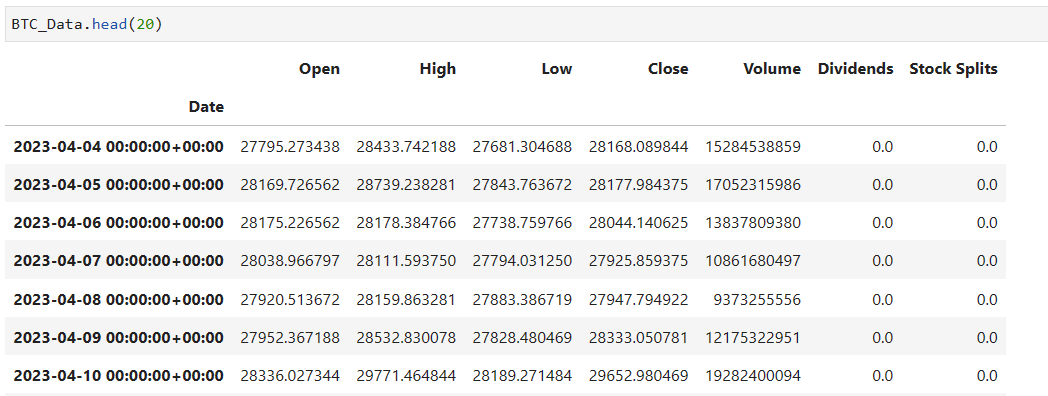
To use Yahoo! Finance, import yfinance in the notebook:

import yfinance as yf

A dataframe can be obtained by using Yahoo!’s ticker name:

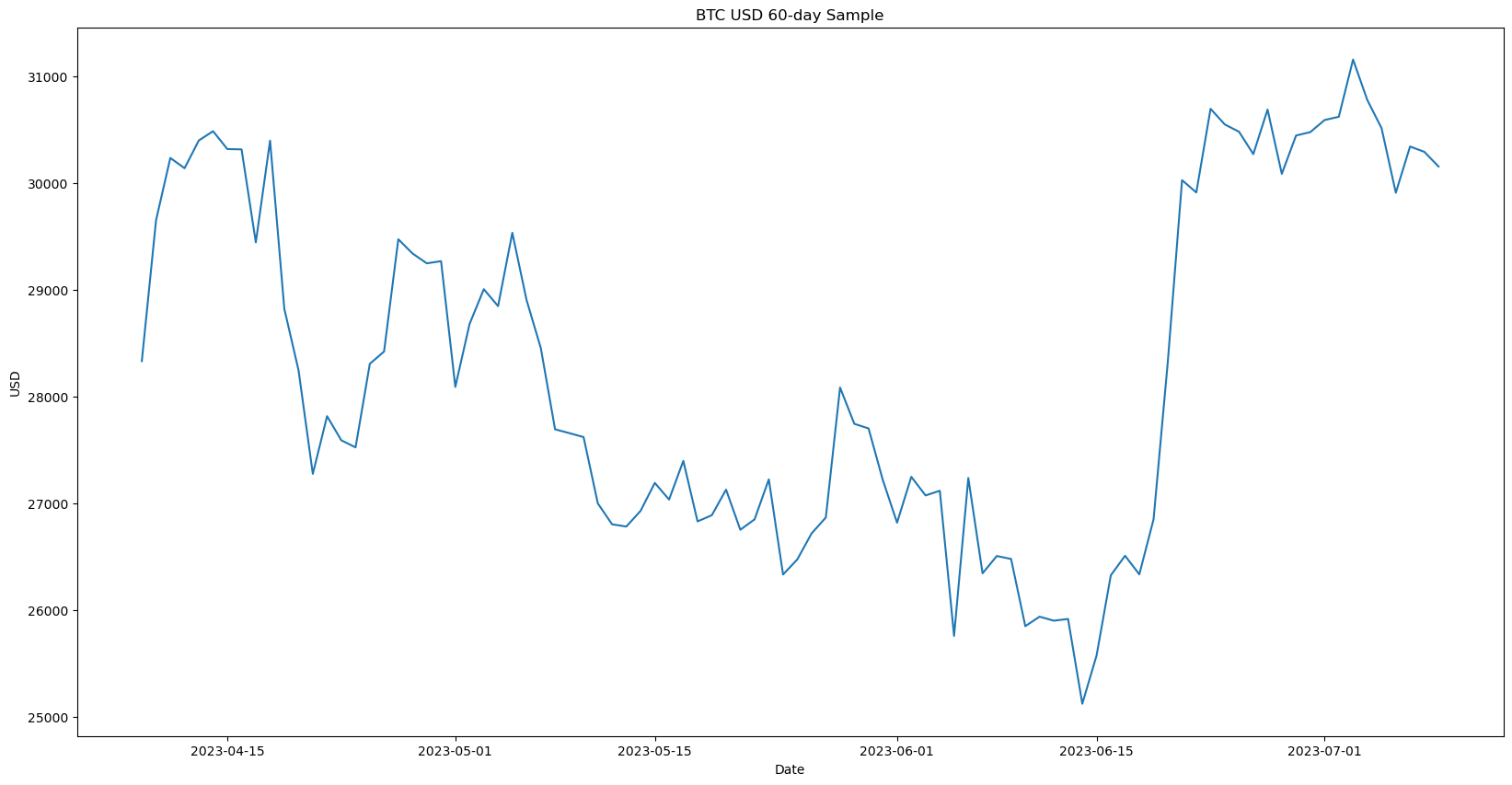
BTC\_Ticker = yf.Ticker("BTC-USD")

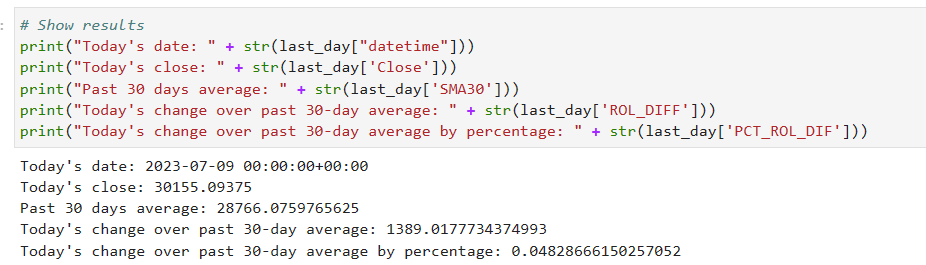
BTC\_Data = BTC\_Ticker.history(period="3mo")



This data needs to be cleaned.

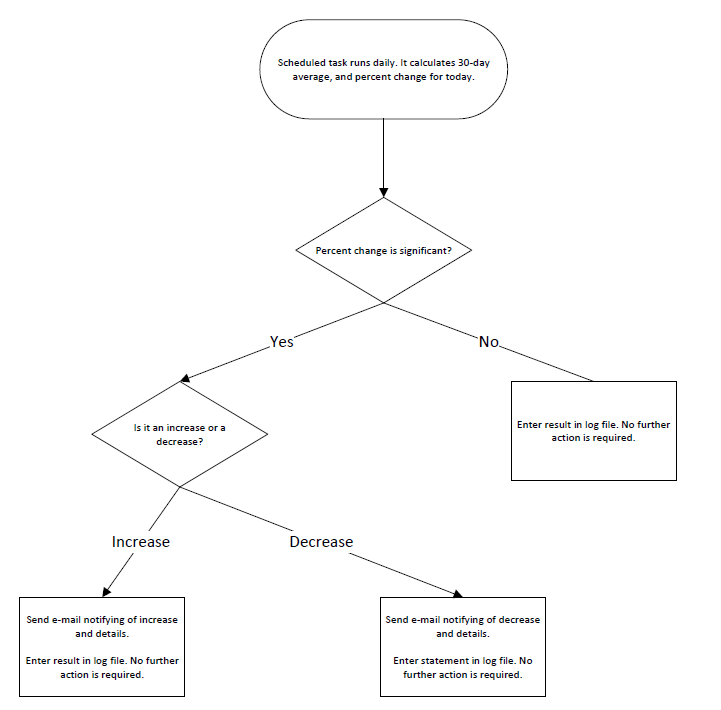
Here is a sample 60-day data from 2023-07-09:





## Automated Calculation

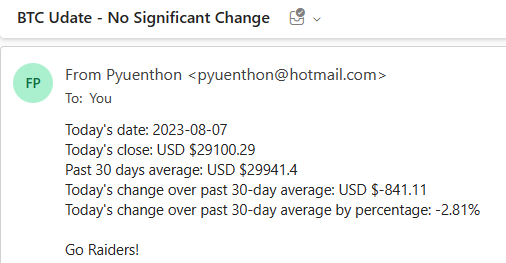
The objective of automated calculation is to scrape data and calculate the current date’s price with the average of the previous period. If this can be a scheduled task to run daily, it can prompt the user when a significant change is detected.



This is the process that happens when the task is run:

1. Download the data from Yahoo Finance
2. Clean date to “datetime” format
3. Make calculations on data
   1. Create 30 day moving average column
   2. Calculate difference of each day to the moving average
   3. Calculate the percent difference of each day to the moving average
4. Create variables for data to be sent in e-mail
5. Define functions
   1. Calculate result of percentage change (significant or not)
   2. Create the formatted SMTP message
   3. Send SMTP message
6. Run the task

The result should be an e-mail message to the receiver



The scheduler runs the task at 8:00am daily. It will run as long as Jupyter Lab is active.

The Jupyter-lab scheduled task will only run if Jupyter-lab is running. To make sure of this, a Windows scheduled task runs daily 30 minutes before the Jupyter-lab task to confirm Jupyter-lab is running.

$jupyter = Get-Process Jupyter-lab -ErrorAction SilentlyContinue

if ($jupyter) {

# do nothing

Write-Output "Jupyter-lab is running"

} else {

Write-Output "Jupyter-lab not running, starting now..."

C:\Users\a\_yuen\anaconda3\Scripts\activate2.bat C:\Users\a\_yuen\anaconda3

}

Note: The scheduled task will run Jupyter-lab in the background context. The Anaconda prompt and the web UI will not show in the local user context. However, the web UI can be reached directly at:

<http://localhost:8888/lab>