Just Buy The Dip

A study using analytics and time series model to predict the right time to enter the market

1.1 Introduction:

1.1.1 Purpose:

“Buy the dip” means investors purchase an asset within a range they believe to be a good deal. Stock trading would be so much easier if investors knew when a stock reached its bottom price. However, the truth is, whenever we invest in shares of a "buy-the-dip" stock, sooner or later, we find that they drop to an even lower price.

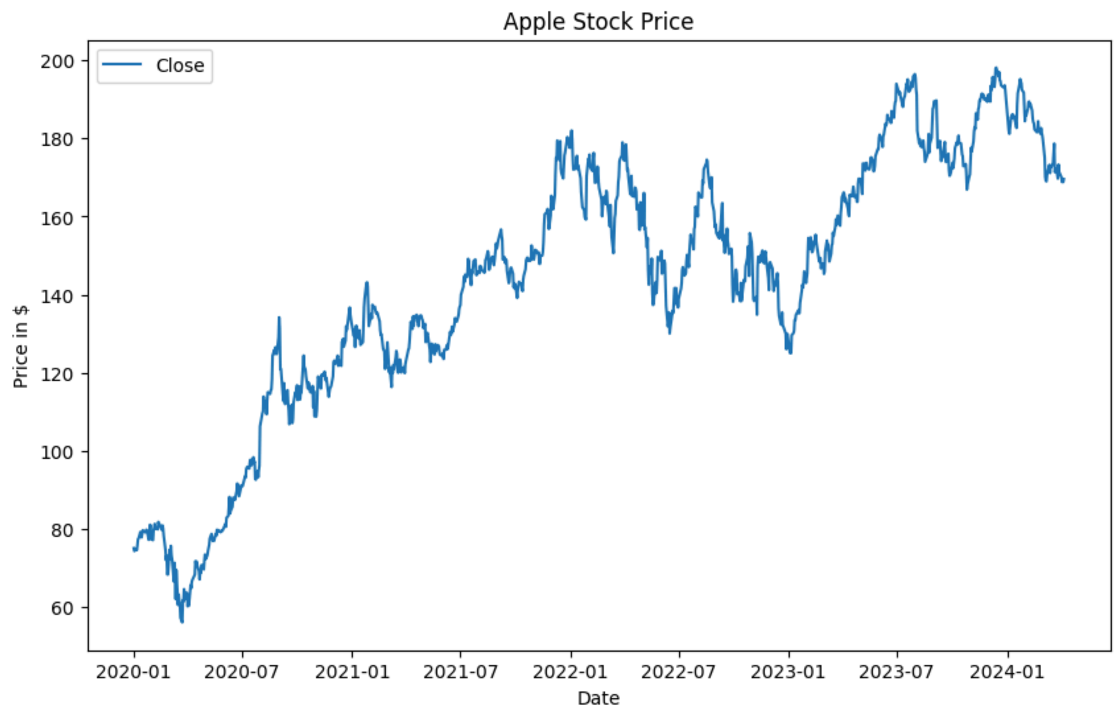
What if there were a method we could rely on to predict the dip in terms of its duration and magnitude? Well, the obvious approach is to have a deep knowledge of the company’s financial standing or the competitiveness of its products. However, researching a single company requires tremendous effort and time. You might say, 'I don’t know how to read a balance sheet,' or 'I wish I had taken Business 101 in college.' Let's leave this to a real expert in the field. Instead, we will solely use the historical market price of a stock, along with detailed data analytics and time series models, to predict the perfect time to buy the dip in a relatively long time investment.

1.1.2 Dataset:

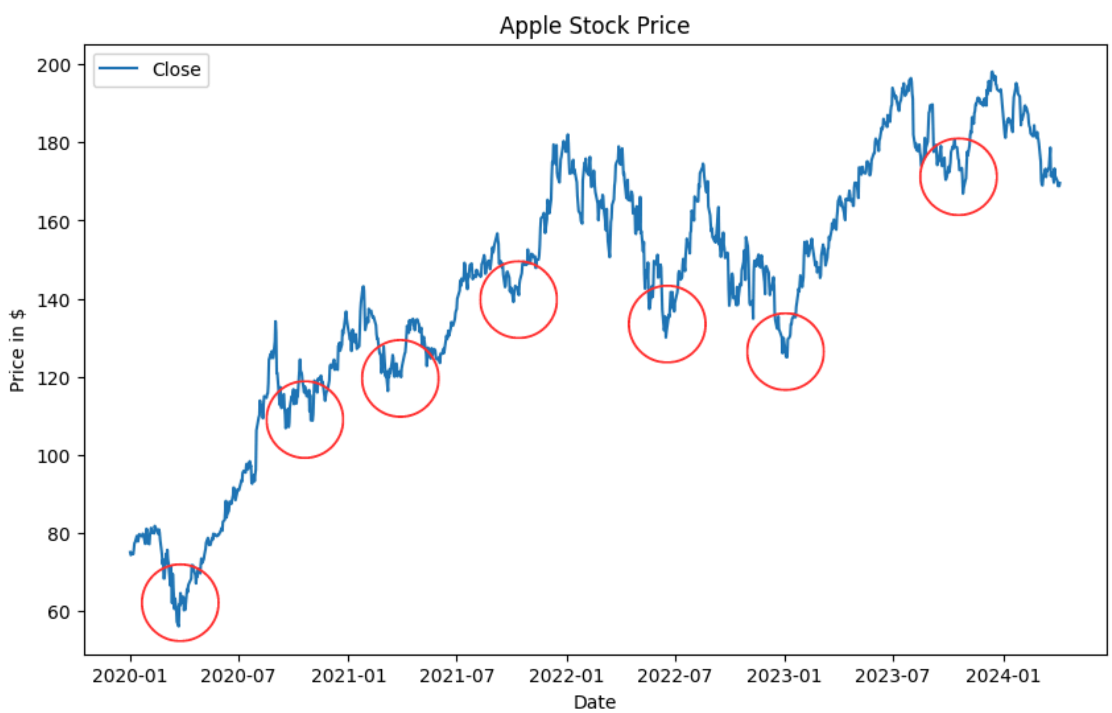
We will use Yahoo! Finance API yfinance in Python. Due to limited resources, we will look at a typical range from 2020-1-1 to 2024-4-6, excluding holidays and weekends when the market is down. In a typical dataset, the columns contain “Open”, “High”, “Low”, “Close”, “Adj Close”, “Volume” with date as index. We will mainly focus on the “Close” column.

2.1 Part 1: How Do We Identify the Dip?

Here is Apple's stock price at closing from 2020-1-1 to 2024-4-6.

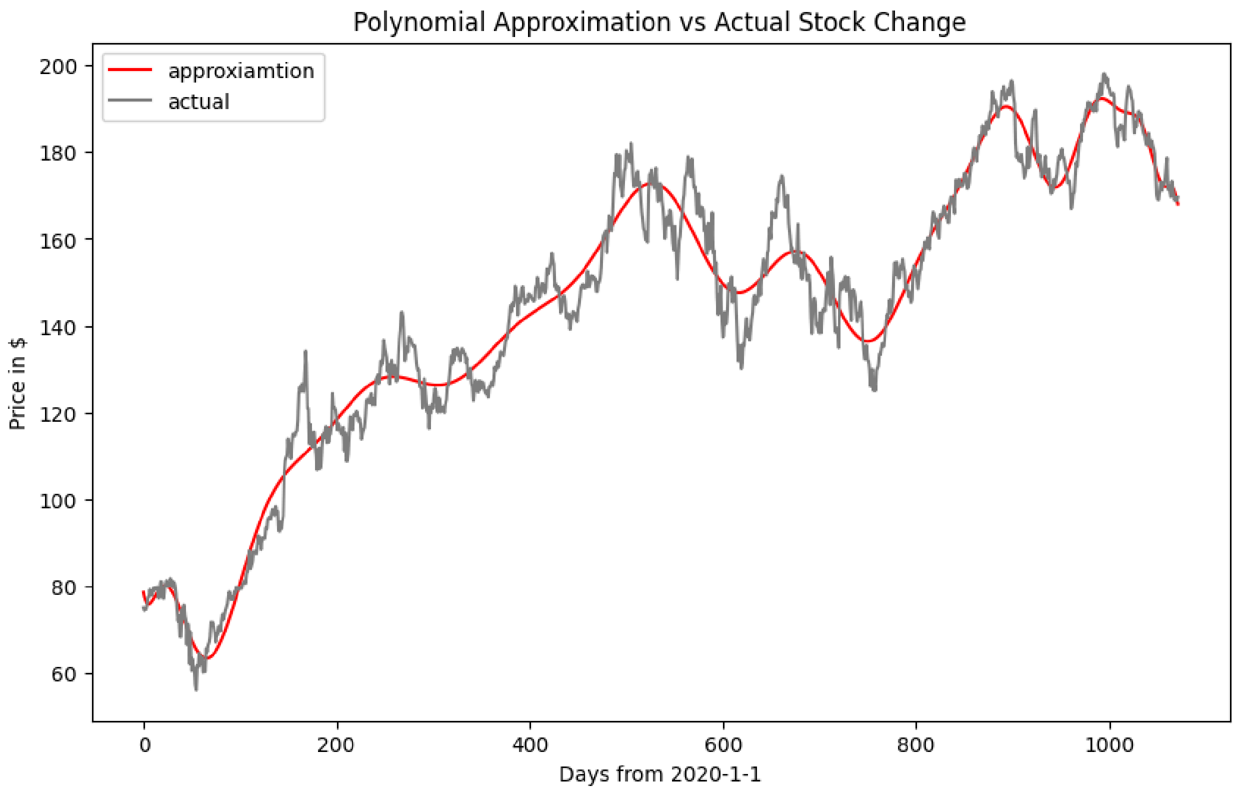


The line plot is no way to be called smooth due the dynamic nature of stock price and the market closed days. You can still identify some obvious dips in certain intervals.



2.1.1 Curve Fitting

This is not helpful and too complicated for us to measure the duration of each dip. So our goal is to smooth the line. To achieve this, we utilize the method polyfit to fit a 1-d polynomial for our line plot.

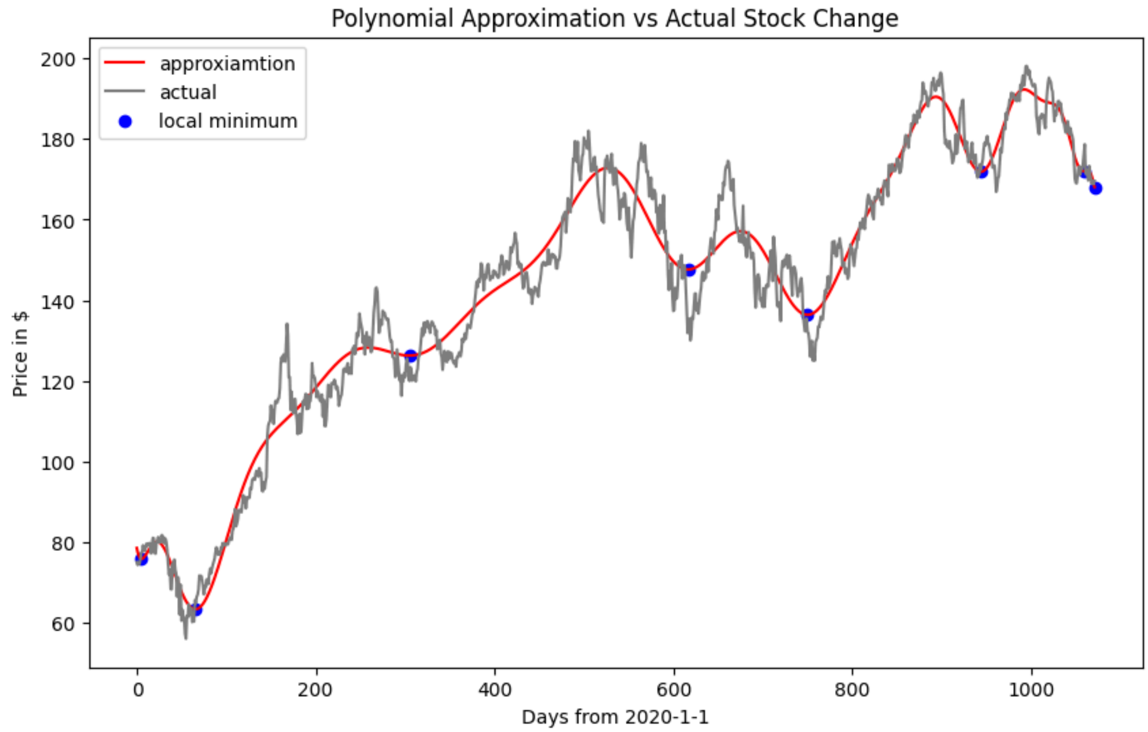


2.1.2 Local Minimum

We can observe that many dips disappear in our polynomial approximate line plot, which is okay as we are focusing on long-term investment. By calculating local minima, we can obtain the (x, y) of the dip.

```
local_min_indices = argrelextrema(poly_curve, np.less_equal, order=5)[0]
local_min_indices
```

```
array([   5,   65,  305,  617,  750,  944, 1059, 1071])
```



There are 8 local minimums notated on the graph. However, not all of them are obvious dips, especially the ones at the endpoints. Let's eliminate them to ensure a more accurate result of dip duration. Here, we use the concept of curvature to determine whether a dip is obvious enough.

$$\kappa = \frac{|x'y'' - y'x''|}{(x'^2 + y'^2)^{3/2}}.$$

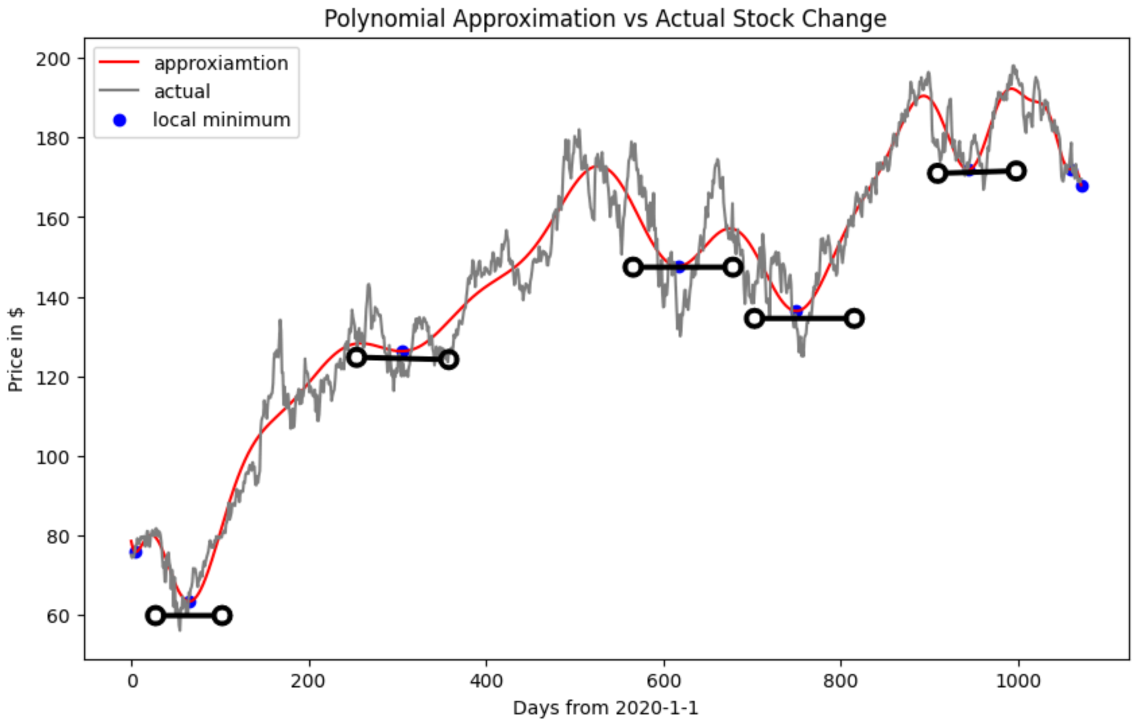
```
#first derivatives
dx= np.gradient(local_min_indices)
dy = np.gradient(p100(local_min_indices))

#second derivatives
d2x = np.gradient(dx)
d2y = np.gradient(dy)

#calculation of curvature from the typical formula
curvature = np.abs(dx * d2y - d2x * dy) / (dx * dx + dy * dy)**1.5
curvature
```

```
array([1.47054151e-02, 3.90096959e-04, 1.97992643e-04, 2.77177872e-04,
       3.30374141e-04, 5.06549463e-05, 3.22842844e-03, 1.13945127e-01])
```

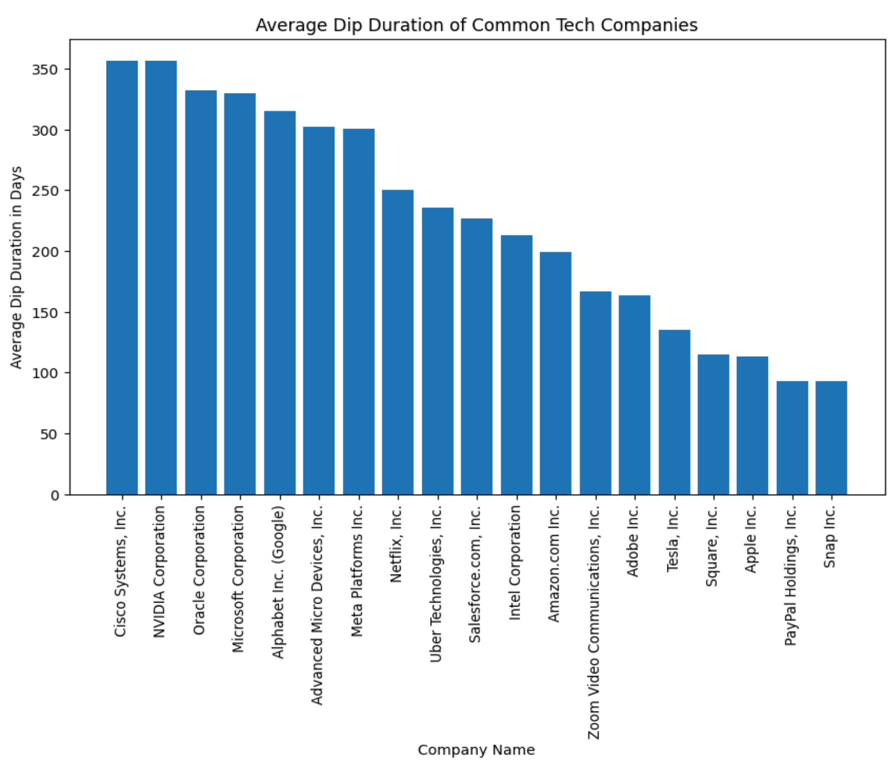
Limit certain numerical values for the curvatures. In the meanwhile, we need to calculate the duration of this dip.



2.1.3 Calculating the duration

To achieve this, our logic is to calculate the local maximums between the local minimums, pick the shortest distance in terms of days between one of the local maximums to the local minimum and multiply by two. Then we can calculate the average dip duration in days for each dip.

Here are average dip durations for some common tech companies.



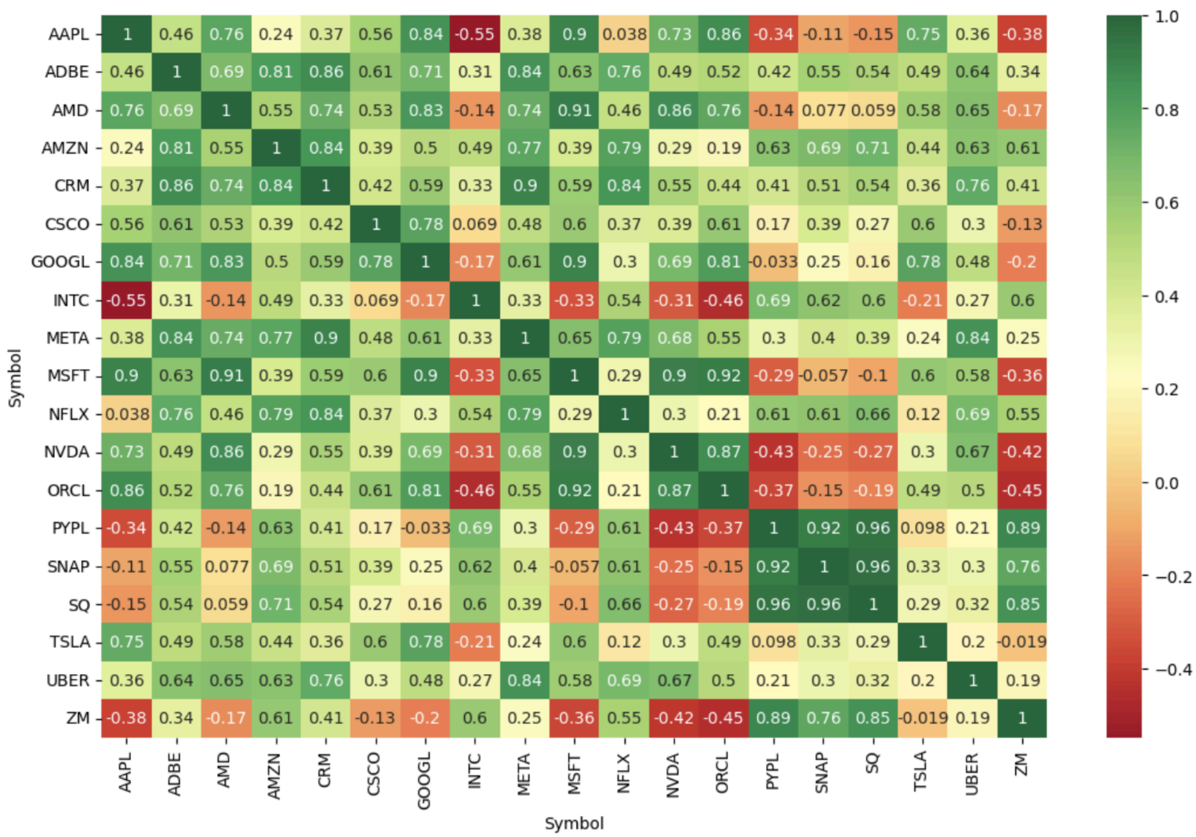
In conclusion for this part, understanding the average dip duration could assist investors in identifying the investing cycle for certain companies if they aim to buy low. It is more sensible to invest in companies with shorter dip durations, as this results in a shorter waiting period.

2.2 Part 2: Correlation between stocks

While focusing on the specific stocks you are interested in it is important to observe other stocks who share the high correlation. This might become a potential feature in our model so let's investigate it.

Once we download all the pricing data of each stock, we utilize the function pivot and corr to determine the correlation between each stock. We can draw a heatmap to visualize it.

Here is the example of tech companies that I analyzed in part 1:



The closer to 1, two stocks are supposed to be more correlated, meaning their price movement could be similar. As for Apple, we can see from the graph that AMD, GOOGL, MSFT, ORCL, NVDA, TSLA all share high correlation (>=0.7). We can use this method to analyze all the stocks in the market to figure out the top 10 most correlated stocks with Apple.

3.1 Predict the Buy Time

3.1.1 Model Choice

ARIMA vs. LSTM vs. Informer