Anticipating Market Prices of HPQ market

1. Introduction

a) Problem statement

We are finding ways to track market prices. We want to anticipate the flow of the HPQ market and the market in general.

b) Background

HPQ, previously known as Hewlett-Packard, is a super famous tech company that has made major contributions to the industry. It all began in 1939 when HPQ was all about electronic test and measurement equipment. But they didn't stop there! Throughout the years, they expanded their product line to include computers, printers, and all the interesting stuff. One of their big moments was the merger with Compaq in 2002, which made HPQ a big shot worldwide. Lately, they've been all about innovation, like 3D printing and cloud computing. The company's stock has had its ups and downs, influenced by money problems and what's going on in the market. HPQ's history shows how they've grown as a tech leader, adapted to changes, and stayed focused on providing awesome solutions to their customers.

c) Goal

The goal is to anticipate the price market of HPQ. We want to find the best time to buy and sell. By looking at price, we can determine the direction of the market

Dataset

The data we use comes from the Kaggle website. The website link: https://www.kaggle.com/datasets/qks1lver/amex-nyse-nasdaq-stock-histories. The dataset is composed of various stocks symbols. I chose HPQ.

3. Data Cleaning and Data Wrangling

We have 12741 rows of data in our collection.
Our data sources starts with seven columns:
date: The date of the price change.
volume: The volume amount during price change.
open: The opening price during that day.
close: The closing price during that day.
high: The highest price during that day.

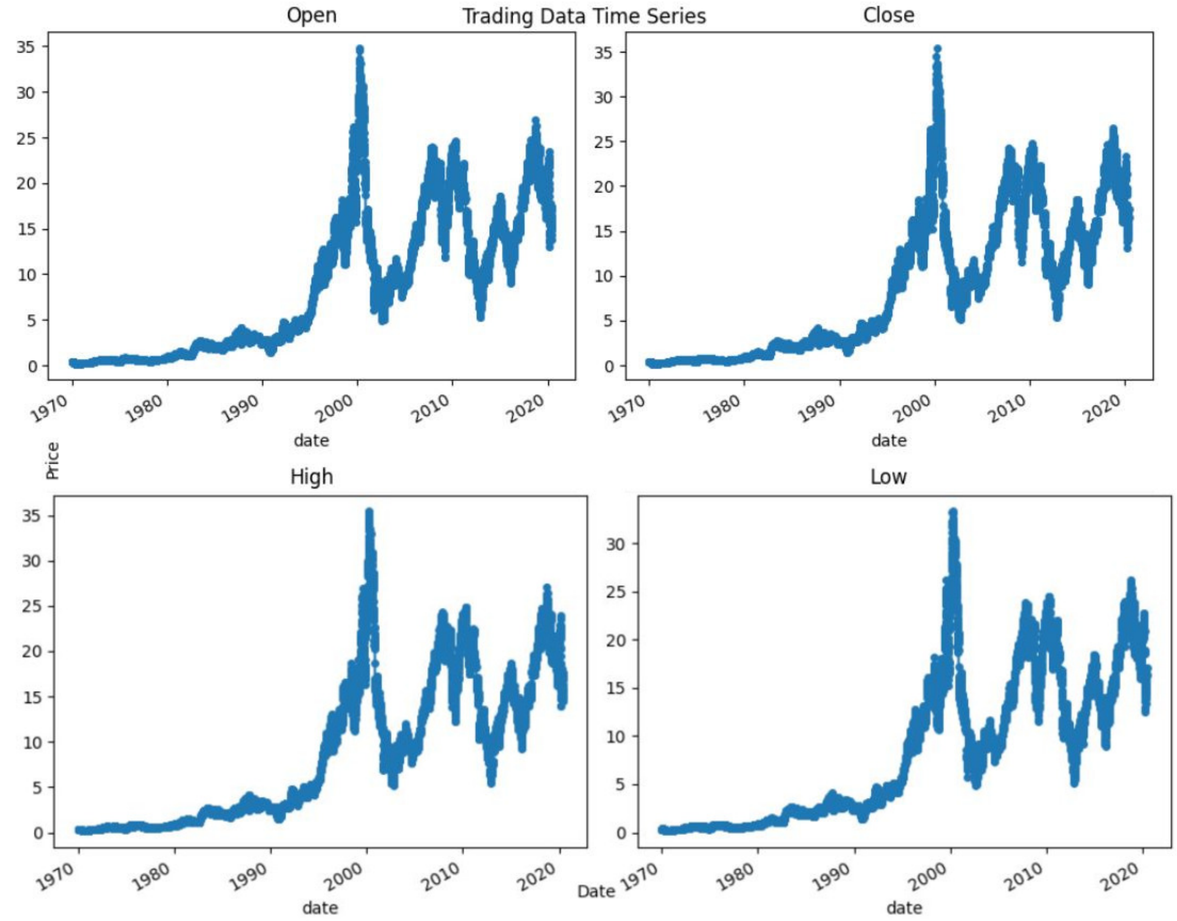
low: The lowest price during that day.
adjclose: The adjusted close price during that day

We emulate some data cleaning and transformation on our dataset. We remove some null values. Null data will interfere with the machine learning process. We did some type conversion such as converting the “date” column to a date time object. We did some statistical analysis and looked at the mean, standard deviation, etc. of each column. We also check for duplicates in the data. We also wanted to remove outliers from data. We made a comparison between data without outliers and data with outliers, but decided to keep the data with outliers. We made use of the Talib libraries and generated some extra columns. These column are candlestick pattern. Each candlestick will have value of 100 if it have bullish pattern, -100 if have a bearish pattern and a value of 0 if it doesn't have a pattern.

We also calculated the support and resistance levels and made columns of those too. We also calculated the RSI, MACD, MFI, VWAP, Fibonacci, and exponential moving average. We realize that we have too many columns. So we decided to remove some columns. We calculate the sum of each candlestick pattern row and if it is greater than 10, we keep it. During the process of creating a new column, we create some new null value. So we drop some more values. That ends the data cleaning process.

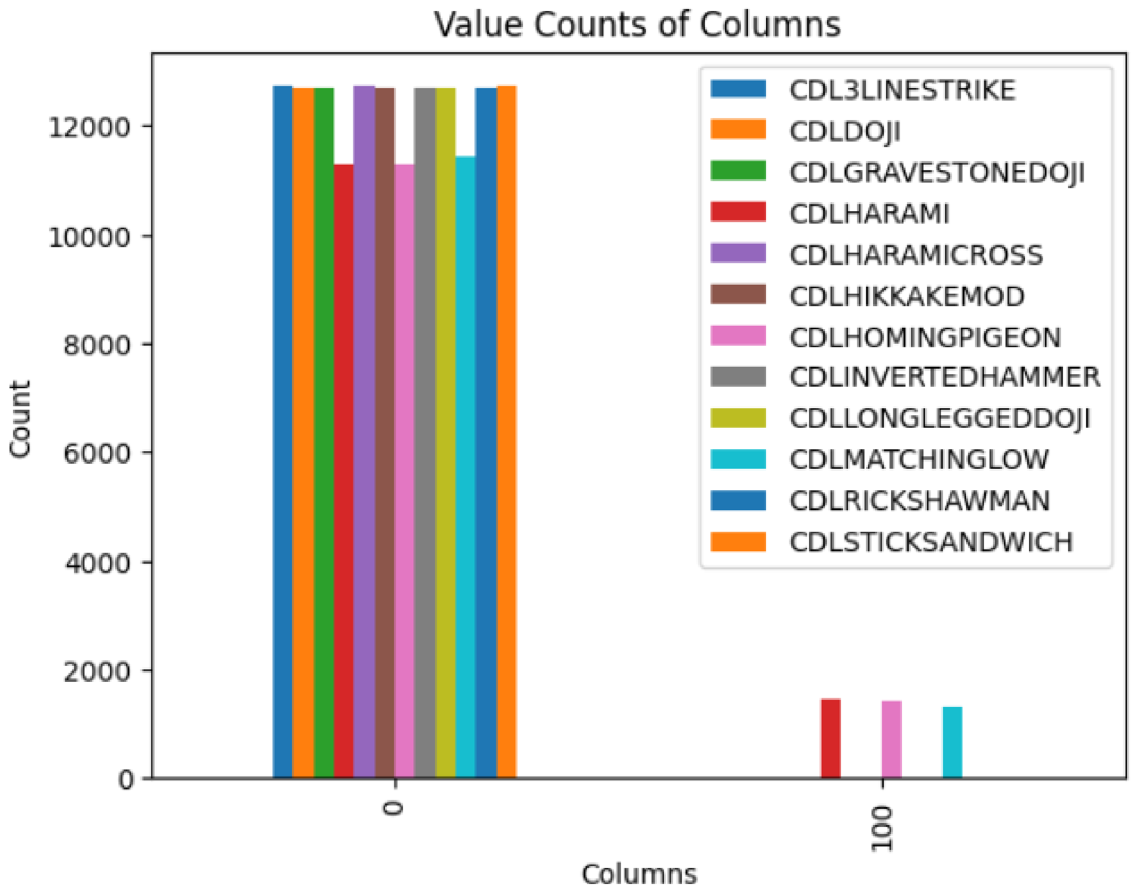
Data exploratory analysis:

We decide to look at some graphs, so we can know how the data look like. We first observe the relation between open, close, high, and low.



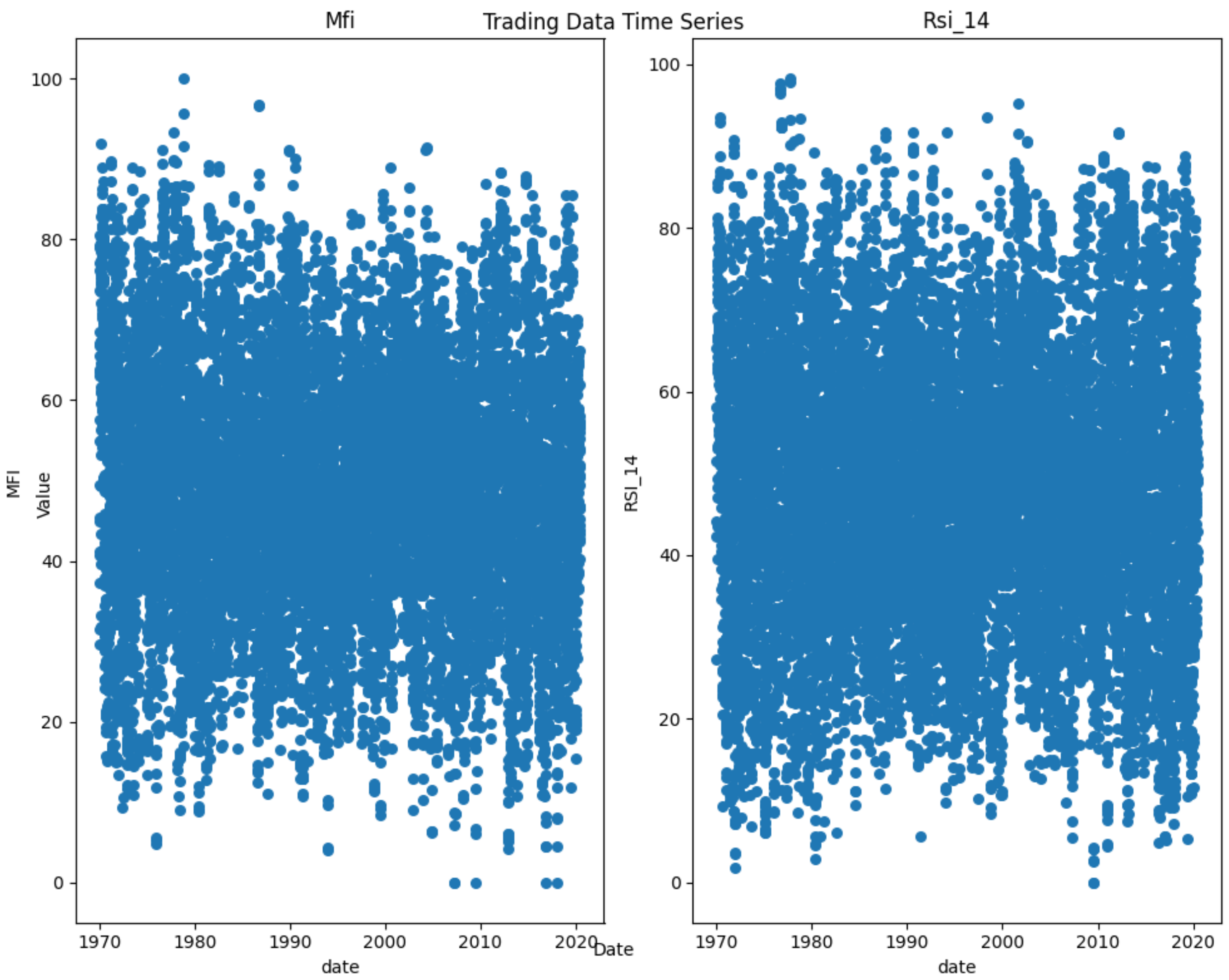
All four graphs share a similar structure, but the maximum for low is at 30, while the rest of the chart stops at 35. This is normal because the low is supposed to be lower than high. I found it interesting that close and open is as high as the high. I can assume that we are more likely to close at a high than a low.

We also create a graph observing the amount of times that a pattern is detected.



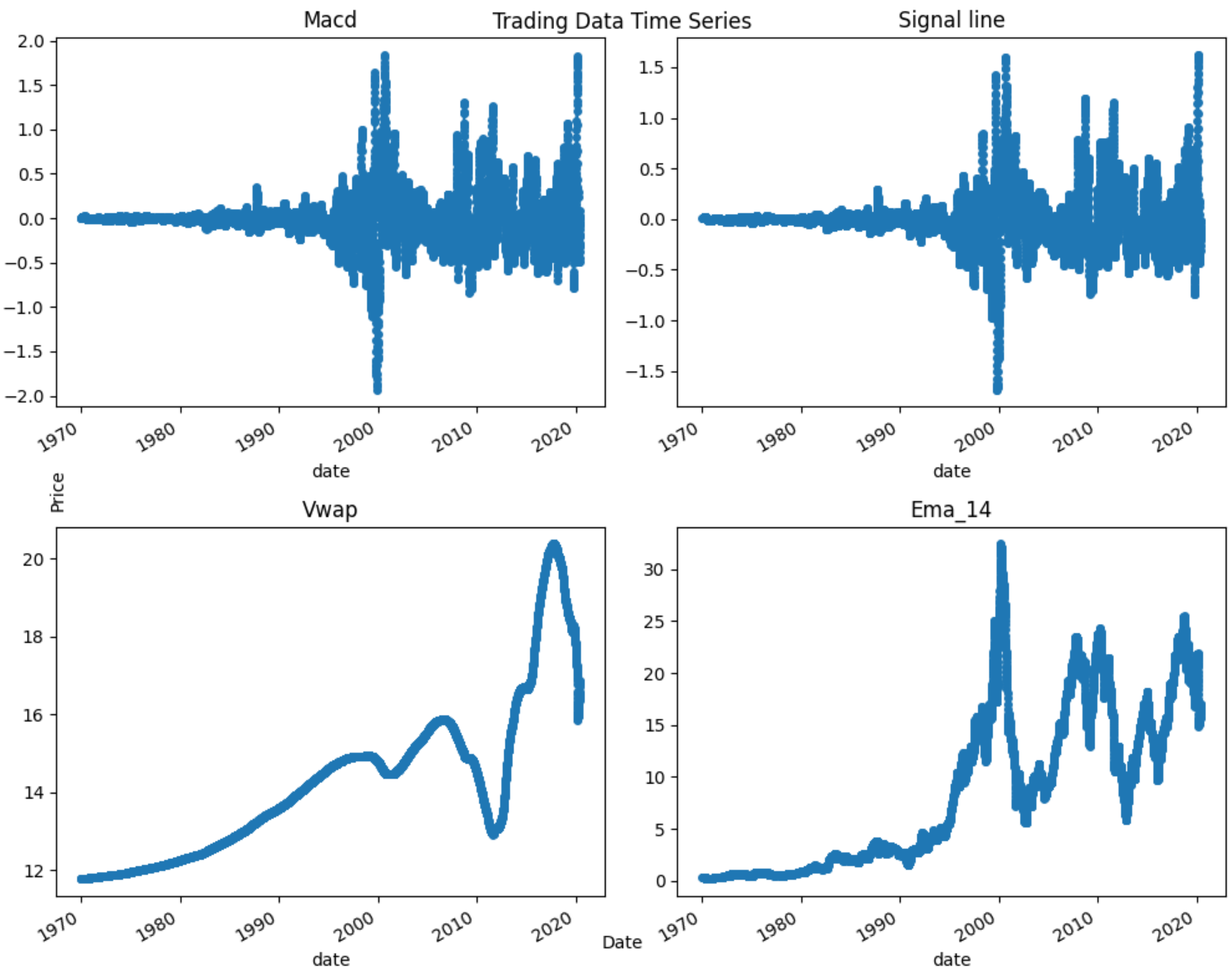
We realize that there is not much pattern going on. I might have to remove these columns. They don't have much purpose.

We also made some comparison between MFI and RSI.



We can say that most of the RSI and MFI are between 10 and 80.

We also did analysis on MACD, signal line, Vwap, and EMA.



The signal line and MACD share similar structure which is understandable because the signal line are use to calculate the MACD. The EMA seems to share a similar structure to the high, low, open, and close graphs. The Vwap shows a different structure. It seems to be at its highest during 2018.

Deep analysis

I realize we don't have a dependent variable. So I created a new column called “Price_Change”. This calculates the price change as a percentage of each row. We decide to do normalization to the data. We normalize the following columns: 'volume', 'open', 'high', 'low', 'close','adjclose', 'Support_Level','Resistance_Level', 'MFI', 'VWAP', 'Swing_High', 'Swing_Low','Fib_Retracement_0.382', 'Fib_Retracement_0.618', 'MACD', 'Signal Line','RSI_14', and 'EMA_14'.Then

Next we emulate some training on the data. We first tried some regression without optimization and the results weren't too bad. I first tried linear regression and got a R2 score of 0.46. We then try out the decision tree and got a negative R2 score. So we decided to try another model, which supports vector regression and got another negative r2 score. So I decided to use gridsearch to find the most optimal models. I did a gridsearch on the following model : ridge, polynomial, support vector regression, decision tree regression, and finally MLPRegressor. All regression gives me a bad R2 score except support vector regression. I was surprised that MLPregressior gave me no good result.

The Support Vector Regression gave me a r2 of 0.67, which is not too bad. I decided to save these results.