CSCI 1420: Machine Learning Capstone Project

Julia Gugulski

Spring 2025

Introduction

Throughout my time at Brown, I have had the opportunity to experiment with many different types of machine learning models like linear regression, logistic regression, decision trees, and different clustering algorithms. I even had research experience with deep learning models like convolutional neural networks and variational autoencoders. One machine learning model that I was not familiar with was time series modeling. I wanted to explore how basic time series models work and how to interpret the results that we get from them.

A common application of time series modeling is forecasting stock prices. Being able to predict the future behavior of stock prices is useful for managing investments as well as leveraging their associated risks. This is personally interesting and relevant to me because in a few months I will begin working at a bank in risk management. Since time series models are very relevant in finance, banking, and risk management, I wanted to experiment with basic time series models to familiarize myself with how they work.

I chose to use a dataset that captures 24 years (2000-2024) of stock prices of crude oil. This dataset comes from Kaggle.[1] It was extracted using Python's yfinance library. Crude oil is a major building block of energy in the United States, and the rest of the world. It is the raw material that produces gasoline which is used for fuel in transportation and producing energy in power plants. Crude oil is also the building material of day to day items such as plastics, pharmaceuticals, refrigerators, asphalt, and wax. The US Energy Information Administration says that over 93% of US crude oil consumption is for transportation and industrial usages.[2] Furthermore, the US is the top consumer of crude oil worldwide, making up 20% of the consumption. Since US consumers and industries are so reliant on crude oil, the fluctuations in market price of a barrel of crude oil have a large impact.

The problem that I want to study is how we can be able to predict the future price of crude oil based on historical data. Being able to notice trends in crude oil prices and predict where they will move in the near future helps investors be aware of the market and the risks associated with the commodity. It can also further inform the average consumer about the health of the economy, and why they are seeing certain trends in prices of day to day goods.

[1] Saboor, Muhammad Hassan

[2] *Use of Oil - U.S. Energy Information Administration*

Additionally, I wanted to explore how much historical data is needed to get the best results in these predictions. The predictions can be sensitive to how much historical data we have, so I want to see how much is really needed. This will help to know if we can still make good predictions when we are unable to collect a lot of data due to any constraints.

Methodology

Background My approach to time series modeling focuses on one of the foundational time series models, the Autoregressive Integrated Moving Average (ARIMA) model. Shah, Hetvi et al. highlight that "ARIMA is a traditional statistical method" which "accounts for both autocorrelation and moving average elements within time series data." [3] It is "widely utilized in finance and economics to forecast stock market prices using past price and volume data." [4]

The ARIMA model is made up of three parts: auto regression (AR), differencing (I), and moving average (MA). The AR component "describes the linear relationship between the present observation and a set number of delayed data points" [5] and is represented by the order (parameter) p. ARIMA models assume that the time series data is stationary, which means that independent increments of the same length have the same distribution. If a time series is not stationary, it can be made stationary by differencing. This removes trends and seasonality. The number of times that the data is differenced is the order (parameter) d. This is the I component of ARIMA. Finally, the MA component "represents the dependence of the present observation on a residual error term that is divided from earlier errors."[6]

All three components are combined to form the model ARIMA(p, d, q). The parameters p, d, and q can be manually chosen using an analysis of autocorrelation and partial autocorrelation functions. The autocorrelation function visualizes the correlation between the time series data and a lagged version of itself (i.e., with its lags of different sizes). [7] It starts at finding the correlation with the data itself, which is why autocorrelation functions always begin at 1. The partial autocorrelation function is the partial correlation of the time series with its lags after removing the effects of lower-order-lags between them. [8] For both functions the input is the size of the lag, and the output is the correlation of the time series with the lagged version of itself. The ACF plot helps us judge the MA models. If the plot has a significant spike at lag q but not beyond, this suggests that the MA parameter is q. The PACF measures balance variance of the lags and helps us judge whether we should include such lag in the AR model. If the PACF has a significant spike at lag p but not beyond, this suggests an AR parameter of p.

Another metric for checking stationarity of data is a hypothesis test called the Augmented Dickey-Fuller (ADF) test. The null hypothesis is that the data has a unit root and is

[3] Shah, Hetvi, et al.

[4] Shah, Hetvi, et al.

[5] Shah, Hetvi, et al.

[6] Shah, Hetvi, et al.

[7] Leonie

[8] *How to build ARIMA models in Python for time series forecasting*

non-stationary. If we have a p-value greater than 0.05 then we fail to reject this null and can confidently say that the data is not stationary (which we want it to be for ARIMA). If the p-value is less than or equal to 0.05 then we can reject the null hypothesis and confidently say that the data does not have a unit root and is stationary.

Methodology With all this background in mind, I will now explain step by step what I did to try to study my original question: how do I predict stock prices and what is the ideal amount of data to use?

First, I split up my data into 5 subsets of the original data: the full data, the most recent 20 years, the most recent 10 years, the most recent 5 years, and the most recent 3 years. In the full, 20, and 10 year subsets the testing set was 2023 and 2024 (the most recent 2 years) and the rest was the training set. In the 5 and 3 year subsets the testing set was 2024 (most recent 1 year) and the rest was the training set.

Next, I plotted the raw and log transformed data to get a sense of what the data looked like, to begin making an assessment of whether it was stationary or not, and to begin the process of manually choosing parameters based on the ACF and PACF plots.

Next, I plotted the ACF and PACF plots of the raw data subsets and tested them using the ADF test. All three pointed me to seeing that the data was not stationary. I differentiated the data and then plotted the ACF and PACF plot again and tested using ADF test. This all confirmed that the data was stationary. This manually gave me a parameter d=1 for all models. I observed all of the ACF and PACF plots and manually chose p and q parameters as well. The manually chosen parameters that I experimented with always left p non zero and q = 0, or p = 0 and q non zero. I also auto chose parameters using the pmdarima module in python.[9] So, for each subset of data, I trained three different models and compared the performance of subsets within model type.

Next, I trained the three types of models on each of the subsets and noted their results, including their error values. I plotted the error vs. subset size, which helped me visualize which subset size is ideal for our forecasting task.

Results

Because of the volume of visualizations and results that I have produced in this project, I will focus this section of discussion on the results from the subset of 10 years with autofit parameters. A comprehensive view of all results are found in the "eda and results.pdf" included in the github repository. There, you can view all the results for each model (by subset size as well as manual, and auto generated parameters).

First, I visualized the raw data of the 10 years and the logarithm transformed version of the data. These are shown side by side below.

[9] pmdarima.arima.auto_arima

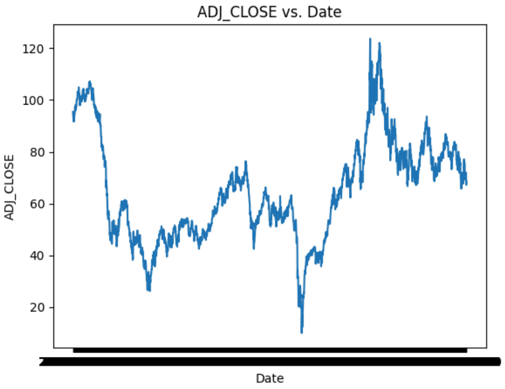

Figure 1: Raw Data Plot

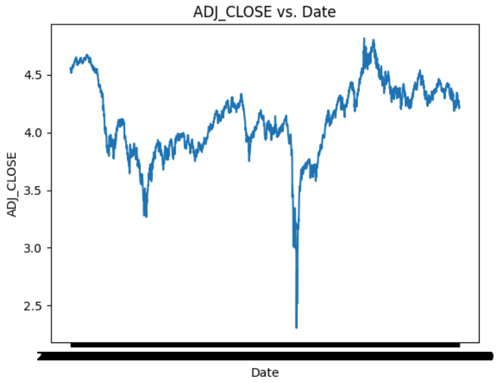

Figure 2: Log Transformation Data Plot

These two figures begin to give us a look into whether the data is stationary and if it needs differencing. To get a more concrete idea, I also plotted the ACF and PACF plots and did the ADF test on the 10 year subset. The ACF and PACF plots are shown below.

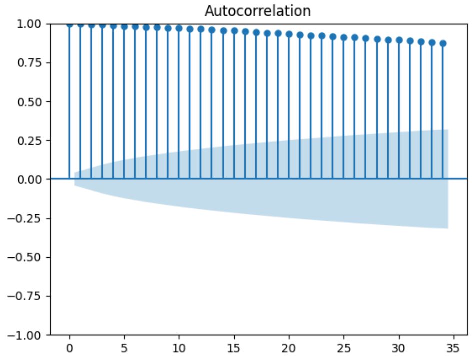

Figure 3: ACF plot of raw data

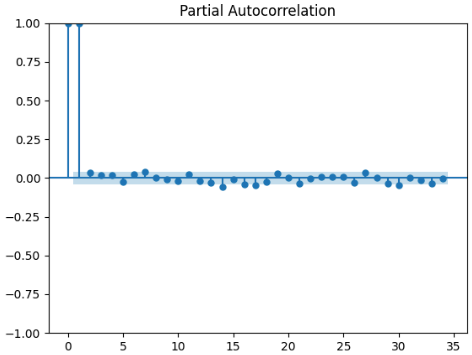

Figure 4: PACF plot of raw data

The ACF plot shows that the correlations between the data and lagged versions of itself are high and positive with a slow decline as the size of the lags gets larger. The PACF plots show a spike at 1 (the data has a correlation of 1 with itself), and then small or no spikes afterwards. These are signs of a simple random walk, which is a common time series that is not stationary.[10] Additionally, the ADF test had a p-value of 0.17, which means that we can not reject the null hypothesis; the data is not stationary.

In order to make the data stationary, which is assumed by the ARIMA model, we should difference the data once, then plot ACF and PACF and run ADF test with the differenced data. The figures resulting after differencing are below.

[10] *How to build ARIMA models in Python for time series forecasting*

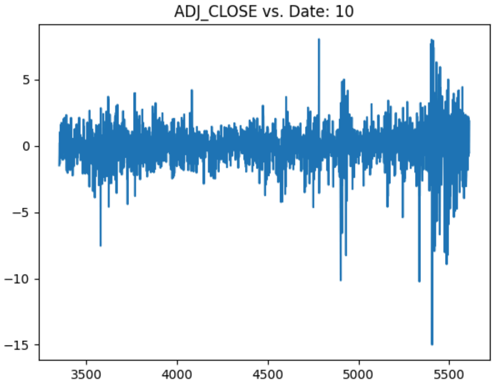

Figure 5: Differenced data plot

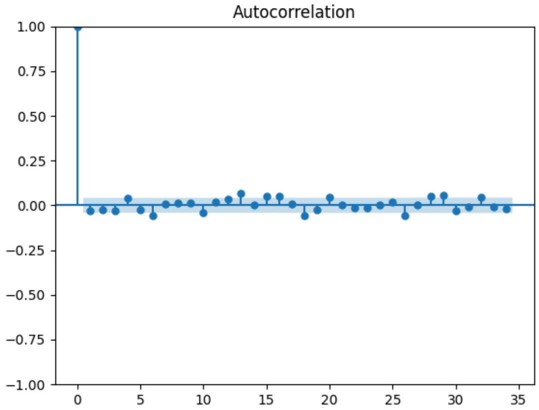

Figure 6: ACF plot of differenced data

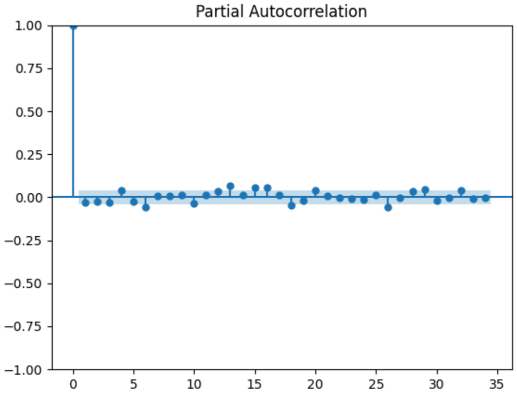

Figure 7: PACF plot of differenced data

Additionally, the p-value from the ADF test for the differenced data was 1.71 x 10^{-15}. The differenced data plot being centered around zero, the ACF and PACF plots being more steady, and a small p-value from the ADF test all point to the differenced data being stationary. This means that we should expect the (I) parameter, d, to be equal to 1, since we only had to difference the data once in order to get stationary data.

The auto generated parameters using pmdarima auto_arima gives us the parameters ARIMA (2, 1, 2). The full summary of this model can be seen in the “eda and results.pdf” file in the github repo, along with all other results. The model defined by these parameters and trained on the 10 year subset forecasts the following orange line for the 2 years of testing (on next page)

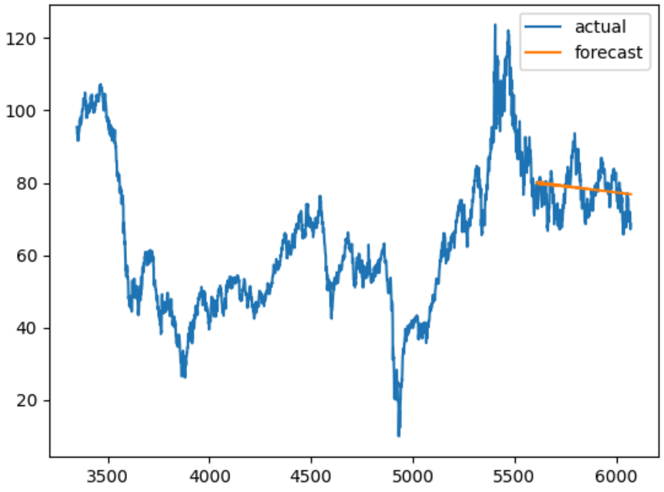

Figure 8: 10 years raw data plot with last two years forecasted by auto generated ARIMA parameters

The residual mean squared error of this forecast is 5.63. Here, we see that the prediction is some sort of linear regression through the test set. This makes sense as ARIMA is a good model for discovering linear relationships within a time series data set.

I applied the same process outlined above for the 4 other subsets of data. I found the residual mean squared error of the model for each subset, and then plotted the errors against the sizes of the subsets.

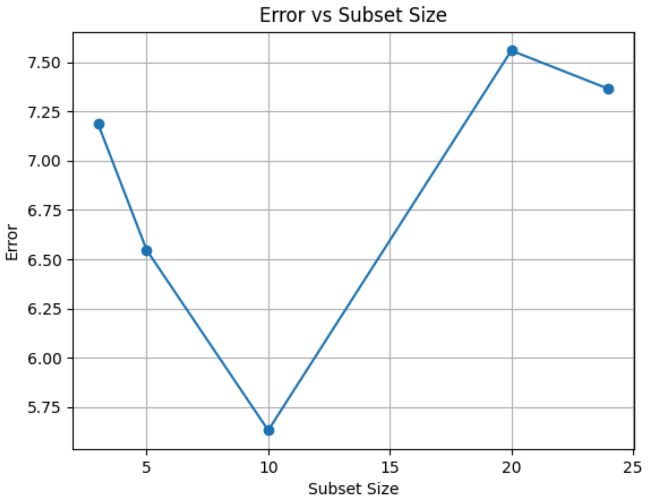

Figure 9: Error vs Subset Size for auto models with auto generated parameters

For the subset sizes that I experimented with, 10 years is ideal.

Takeaways and Conclusion

According to the results of my experiments in this project, using an ARIMA model for time series forecasting of crude oil prices captures linear relationships in the data and works the best when we have 10 years of historical data. The auto generated parameters worked better than my manually chosen parameters, which makes sense because it is difficult to choose AR and MA parameters from just looking at ACF and PACF plots. The hidden computation that the pmdarima module does for me would be interesting to go into and see why they chose the parameters that they did. The auto generated parameters made the model learn useful things while keeping it still quite general and not too complex.

There are many additions that I would consider adding to this project to improve it if I had more time to continue working. I would like to learn more about how to manually choose parameters and really understand how changing the AR and MA parameters changes the model. Right now, that is still kind of a mystery to me. I also would like to learn more about other types of time series models and compare their performance to the ARIMA model that I had here. I would also like to flesh out how much historical data is really ideal. Here, 10 years worked well, but I wonder if any subset around the size of 10 also works well, like 9 or 12 years. I also wonder if the size of training data that is ideal is task dependent, or if certain time series models work better with certain training set sizes.

Works Cited

“How to build ARIMA models in Python for time series forecasting.” YouTube, uploaded by Lianne and Justin, 7 September 2022, https://www.youtube.com/watch?v=-aCF0_wfVwY

Leonie. “Time Series: Interpreting ACF and PACF.” *Kaggle*, Kaggle, 15 Mar. 2022, www.kaggle.com/code/iamleonie/time-series-interpreting-acf-and-pacf#Order-of-AR,-MA,-and-ARMA-Model.

“pmdarima.arima.auto_arima.” pmdarima Documentation, alkaline-ml, https://alkaline-ml.com/pmdarima/modules/generated/pmdarima.arima.auto_arima.html

Saboor, Muhammad Hassan. “Crude Oil Stock Dataset 2000-2024.” *Kaggle*, 16 Nov. 2024, www.kaggle.com/datasets/mhassansaboor/crude-oil-stock-dataset-2000-2024.

Seabold, Skipper, and Josef Perktold. “statsmodels: Econometric and statistical modeling with python.” *Proceedings of the 9th Python in Science Conference.* 2010.

Shah, Hetvi, et al. “A Neoteric Technique Using ARIMA-LSTM for Time Series Analysis on Stock Market Forecasting.” *Mathematical Modeling, Computational Intelligence Techniques and Renewable Energy*, vol. 1405, Springer, 2021, pp. 381–92, https://doi.org/10.1007/978-981-16-5952-2_33.

“U.S. Energy Information Administration - EIA - Independent Statistics and Analysis.” *Use of Oil - U.S. Energy Information Administration (EIA)*, 22 Aug. 2023, www.eia.gov/energyexplained/oil-and-petroleum-products/use-of-oil.php#:~:text=We%20use%20petroleum%20products%20to,intermediate%20and%20end%2Duser%20goods.