Predicting Stock Price of Nintendo and Rockstar Games

Nathan Conger, Qiong Li, Margaret Kimbis

DATA 602: Principles of Data Science

Dr. Mohammad T. Hajiaghayi

University of Maryland, College Park

December 12th 2023

Introduction

In the financial market, stock prices are dynamic, fluctuating on a daily, monthly, and yearly basis. These patterns can be influenced by a variety of factors including age of the company, economic trends, and global events such as the COVID-19 pandemic. This paper focuses on a LSTM(Long Short-Term Memory Model) created through Python programming designed to predict the closing stock price for Nintendo and Rcokstar games in the upcoming days. This prediction uses historical data that is updated daily, and is pulled from the AlphaVantage API.

This project utilized libraries such as pandas, seaborn, matplotlib, sci-kit learn, tensorflow, and streamlit for data prediction, manipulation, visualization, and web application production. The first phase of the project involves a comparative analysis using RSI and trends in stock price between the two companies, goes into data cleaning and the creation of a LSTM model, and finishes by creating an interface that allows users to access predictions through a web application.

Why

In 2020 amidst the COVID-19 pandemic, many companies took a hit. This resulted in a steep decline in revenue for businesses which pushed many of them towards bankruptcy. Despite this being the general trend for many businesses during that time, the financial market saw a rise in stock prices for gaming companies. Since the COVID-19 pandemic, there has been a shift in the stock trends for major gaming companies.

Due to this change, as well as our own personal interest in gaming, we are interested to see how the stock prices will continue to change for both Nintendo and Rockstar games in the foreseeable future. We first wanted to conduct a preliminary analysis to determine the similarities between the two companies to ensure that the two would create a good comparison. They have a similar age, stock price, and RSI trends. Ultimately, we will use neural networks to

predict the price of both Nintendo and Rockstar games in the upcoming days to determine which of the two companies would be a better short term investment.

Preliminary Analysis

First, to ensure that Nintendo and Rockstar Games would make for a good comparison two plots were generated using matplotlib.pyplot and the stock data pulled from the AlphaVantage API. The Rockstar Games (*Figure 1.1)* and Nintendo *(Figure 1.2)* prices (open, high, low, and close) were graphed from 2000-2023. Although both companies have a wide gap between their stock values, in both *Figure 1.1* and *Figure 1.2* there is a steep increase in the stock price in the beginning of 2020, and reached a peak mid pandemic.

Figure 1.1
Rockstar (TTWO) Stock Prices

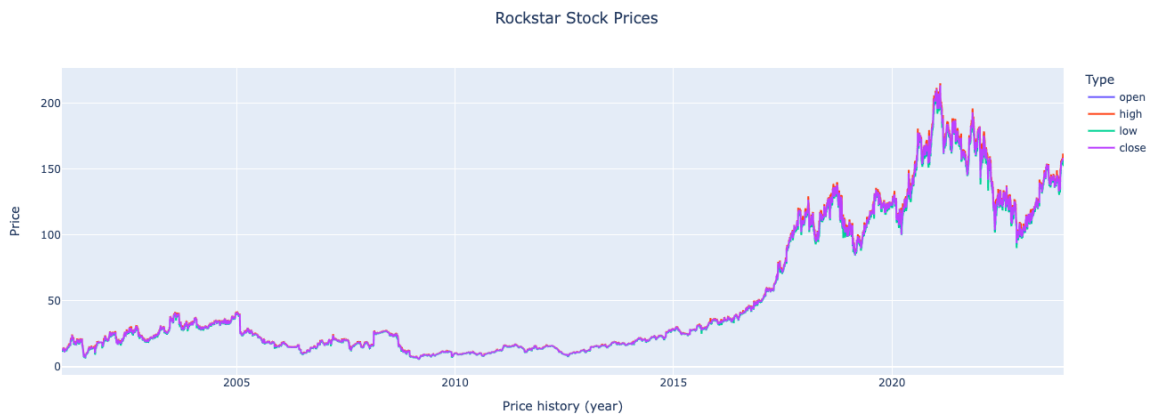
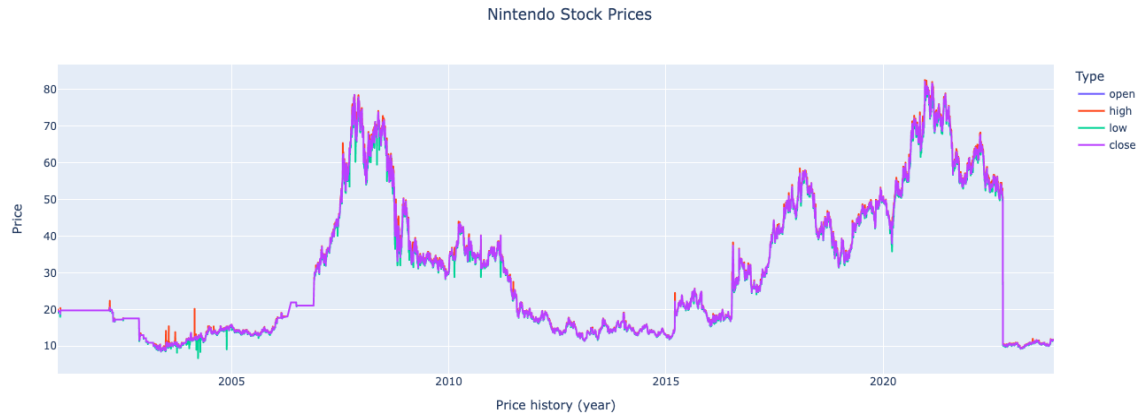


Figure 1.2
Nintendo (NTDOY) Prices



After analyzing the stock prices since 2000 for both Rockstar Games and Nintendo, we decided to look into further comparative measures as a part of our preliminary analysis. The two companies have vastly different stock prices which can be attributed to a variety of reasons. This makes it difficult for someone who has limited experience with stocks to make an informed decision on which company to invest in.

RSI is a standardized measure that can be used to compare the risk associated with purchasing a stock. An RSI of 70 or above may indicate that the stock is overbought and an RSI of 30 or below may indicate that the stock is undervalued.

The AlphaVantage API was used to pull both the RSI for Rockstar Games and Nintendo. Two URLs were constructed, within the URL, function=RSI was used to specify that we wanted to pull the RSI, interval = weekly took the RSI from each week, and datatype = csv allows the user to view and read in the data as a csv file. Following the construction of the URL, pd.read_csv was used to convert the data into a data frame. Seaborn was used to construct a graph to compare the RSI trends between Rockstar Games and Nintendo.

Using the Seaborn package, two line graphs were created based on the weekly RSI values. The data for the years 2020 and 2023 were selected from the data frame and converted into a line graph. During the pandemic, the RSI for both Nintendo and Rockstar Games had a huge jump, both reaching an RSI of 80 (*Figure 2.2)*, indicating that many people started investing in these gaming companies during that time. In the present year, Nintendo and Rockstar games have had a similar trend for buying and selling. Both companies currently have an RSI of around 70 (*Figure 2.1)* indicating that the companies may be overbought, and have a similar level of risk in terms of short term investing.

Figure 2.1
TTWO RSI and NDTOY RSI Over the Past Year

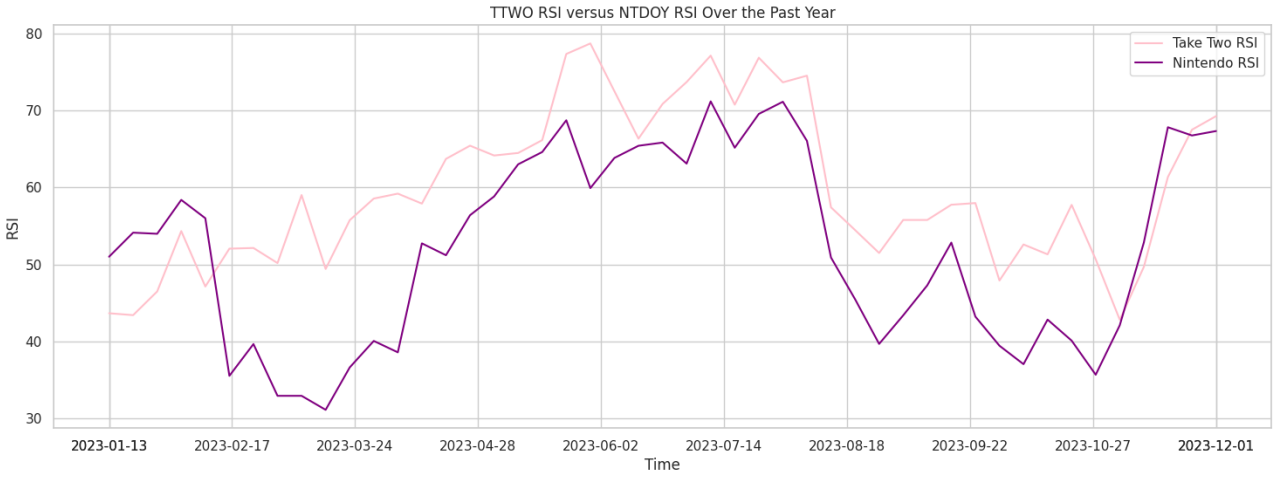
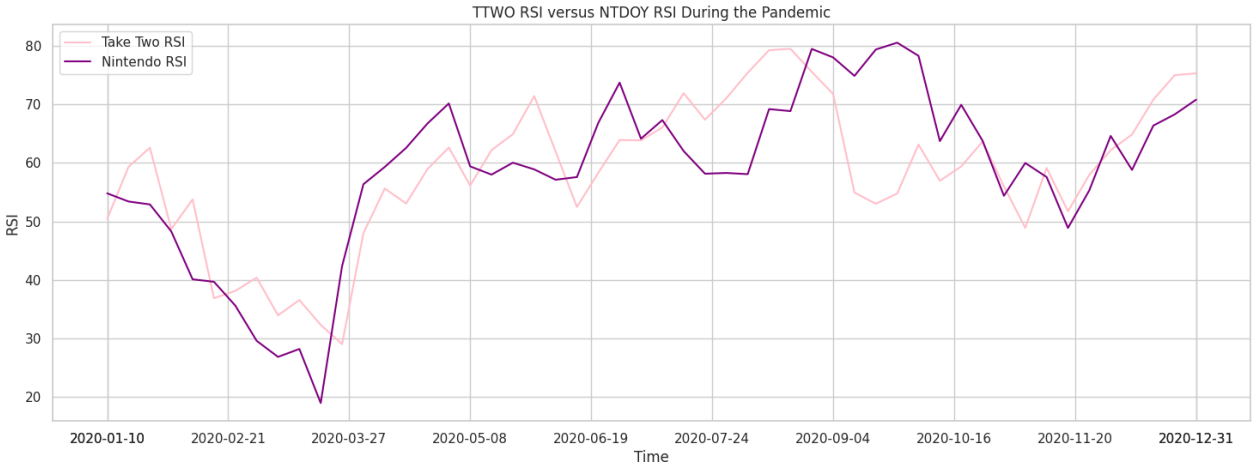


Figure 2.2
TTWO RSI versus NTDOY RSI During the Pandemic



Data Cleaning

After inspecting the graphs above it is clear that Nintendo adjusted their stock price. Nintendo split one individual share into 5 smaller shares. This 5:1 stock split which can be seen above in *Figure 1.2* when there was a significant dip in stock price from over fifty dollars down to around ten dollars. We first had to determine when this 5:1 split occurred, so that we could adjust all the data prior to this date. In order to do this, we selected all the NTDOY where the

time stamp was equal to the year 2022, and the stock prices were graphed using matplotlib *(Figure 3.1)*. From there we were able to determine that this split occurred in October of 2022.

The data was then filtered for rows in which the month was equal to 10 and the closing stock price was less than or equal to 20, and was saved into a data frame. This was then graphed using the sns.barplot (*Figure 3.2)* in order to visualize the exact date in which the split occurred. From this bar chart we were able to determine that the split occurred on October 4th 2022. This date was stored as a variable and used to separate the data before the 5:1 split, after the 5:1 split. In order to standardize the stock price before the 5:1 split, all data points prior to this October 4th 2022 were divided by 5, with the goal of making our prediction more accurate. The results of this standardization were verified through the creation of another barchart using the sns.barplot of the new standardized closing price values in October, as well as ensuring the values we determined matched the values reported by Nintendo Co., Ltd. (NTDOY) on Yahoo finance. This data for NTDOY was saved as a new data frame. Next, to ensure there were no missing values .info() was used. There were no null values in our data set, so no code was created to fill in missing data.

Figure 3.1
Nintendo 2022 Stock Prices

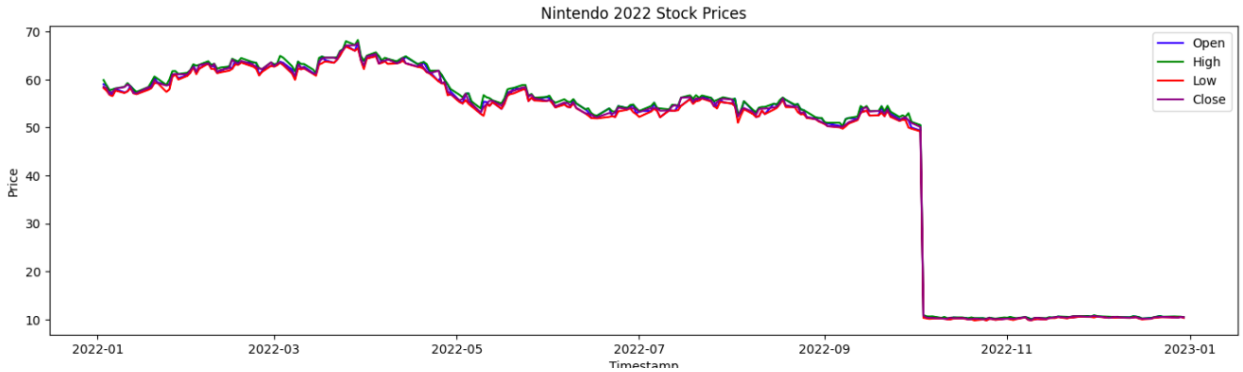
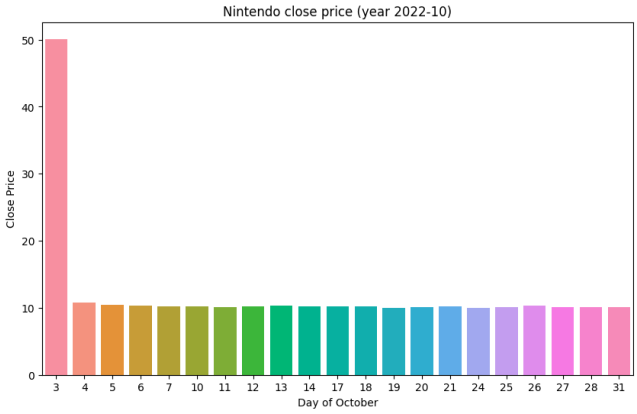


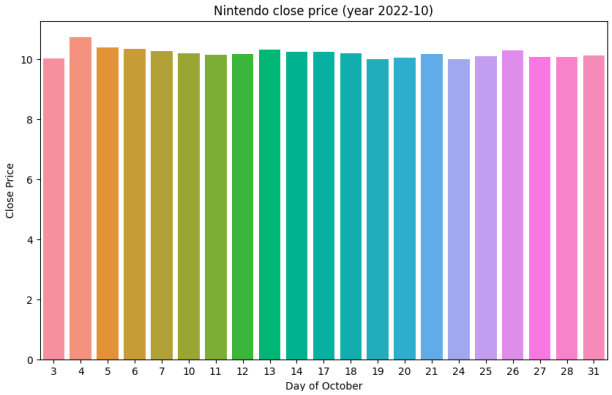
Figure 3.2

Figure 3.3

Nintendo Close Price (2022-10)



Adjusted Nintendo Close Price (2022-10)



Model

An LSTM model was implemented to create a stock forecast for NTDOY and TTWO. The model was developed using sci-kit learn and the tensorflow keras packages. First, we set our target variable to the closing price. Our training features consisted of the opening, high, low, and closing price alongside trading volume from historical stock data obtained from the AlphaVantage API from both companies. Both our target variable and training features were scaled using StandardScaler which is based off of the following formula:

$$Standardized\ Feature\ =\ \frac{Original\ Feature - Mean}{Standard\ Deviation}$$

Following this, TimeSeriesSplit was used to split the dataset into training and testing sets for time series data.The architecture of this stock price prediction application is implemented using the Sequential API provided by Keras which enables creation of a linear stack of layers. The first layer is an LSTM layer with 50 units. This is followed by a dense layer consisting of 1 unit and a linear activation function which takes the features listed above and converts them into a single output that represents price prediction. Before training the model, compilation was performed using lstm.compile and we set “loss” equal to the mean squared error loss function and “optimizer” to the Adam optimizer. MSE minimizes the squared differences between predictions and true values, while the Adam optimizer adjusts the model's parameters to find the optimal solution during training.

Our LSTM model then trains the data using lstm.fit for 10 epochs. Once the data is trained, the model is used to predict stock prices on the test set and the predicted values are then compared with the true values using root mean squared error (RMSE) and mean absolute percentage error (MAPE) as evaluation metrics to ensure our models accuracy. Our model produced an RMSE value of 0.04 and an MAPE value of 0.42.

User Interface

In order to create a web application where users can access and create predictions, the streamlit package was used. Using pip install streamlit, the package was downloaded and imported into the environment. Streamlit is a package that builds web applications, and is utilized for data science and machine learning projects. Due to the nature of our API, there is a 25 inquiry limit per day, if the inquiry limit is exceeded the application will not run. This issue can be solved by purchasing a premium key.

We created a title using st.title, and a drop down menu using st.selectbox so that users could select which stock they want a prediction for (*Figure 5.1)*. Next, we defined a function that allowed us to fetch data for NTDOY and TTWO from the API. A table of the stock data collected from the most recent 5 days was displayed on our application *(Figure 5.2)* and labeled using st.write and choose_dataset.head(). Below the table, an interactive line chart was created from the data set using plotly express and displayed using st.plotly_chart.

The next step was to create two sliding bars so that users were able to select how many training days they wanted to select with a minimum of 10 days and a maximum of 100 days, and the number of days into the future they wanted to predict the stock price with a minimum of 3 days and a maximum of 10 days *(Figure 5.5)*. This feature was created using st.slider. A progress bar was designed to show the training progress of the LSTM model we described in the model section, my_bar = st.progress(0) ensures that the training bar starts at 0 percent. The epoch was set to 10, and the progressbar increases by 1/10th every time the training finishes

one epoch. After training is complete my_bar.empty() clears the bar so it can restart again at 0 percent. A line graph showing the true versus predicted values *(Figure 5.4)* was added by defining a function that creates a line plot using px.line to visualize our test versus predicted values. This graph is then displayed on the application using st.plotly_chart() and is accompanied by our model evaluation which includes the RMSE and MAPE values *(Figure 5.4)*. The RMSE and MAPE values were displayed using st.write.

The final component to our user interface was a line chart for the predicted stock price values for however many days in the future the user wishes to predict stock price. My_bar was used once again to develop a progress bar for the forecasting progress. The forecast graph was developed by creating a function that takes the number of forecasted days, and predicted stock price for that number of days and converts it into a data frame. Plotly express was used to turn the new data frame into a line chart, and then st.plotly_chart() displays the chart on the web application *(Figure 5.6)*.

Streamlit cloud was used to deploy our web application. The final code was uploaded to a github repository alongside a requirements.txt file that contained the packages we needed to run the code alongside the version of the package that was used in our code. For example, to use tensorflow on a streamlit cloud we needed to include tensorflow==2.7.0 in our requirements.txt. Both the .py file and .txt file were located in the root of the github repository. The link to the .py file was imported into streamlit cloud, and the app was successfully deployed and made public so that users could access the website. Our website can aid users who want to make a short-term investment in either gaming company.

Figure 5.1`
Example 1 from User Interface: Drop Down Menu

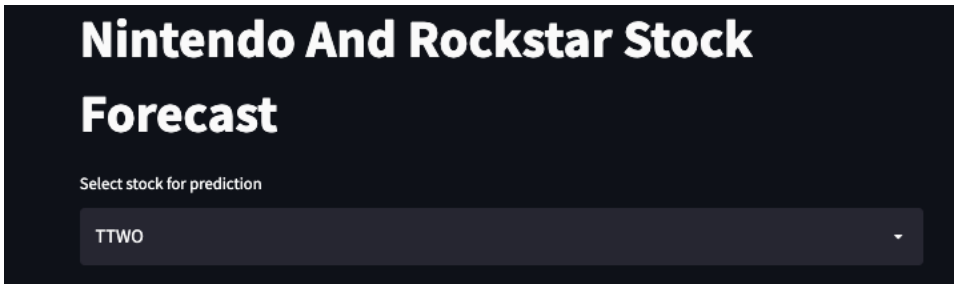


Figure 5.2
Example 2 from User Interface: Recent Five Days Data Table

Recent five days

	timestamp	open	high	low	close	volume
0	2023-12-08T00:00:00	154.3800	156.7225	154.0800	155.3200	2078756
1	2023-12-07T00:00:00	155.1000	155.8500	153.5550	154.2100	2437241
2	2023-12-06T00:00:00	157.5000	158.8100	156.5450	157.3200	2544042
3	2023-12-05T00:00:00	154.2600	157.9800	152.1200	156.7600	3596960
4	2023-12-04T00:00:00	157.3500	158.4500	154.5000	157.5600	1878865

Note: *The data displayed on our application in this table was pulled on 12/10/2023 at 8:53 am and is showing the price for TTWO.*

Figure 5.3
User Interface: Interactive Stock Price

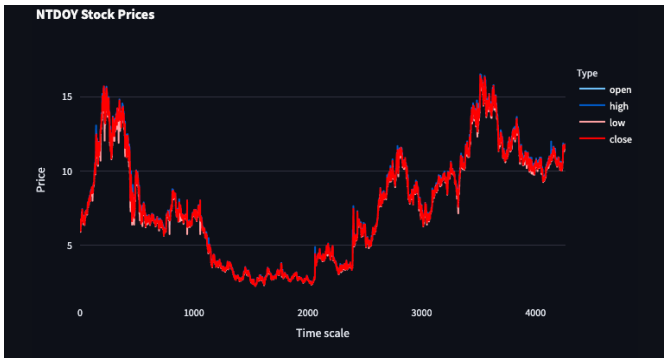


Figure 5.4
User Interface: True versus Predicted

Figure 5.5
Bar for Forecasting and Training

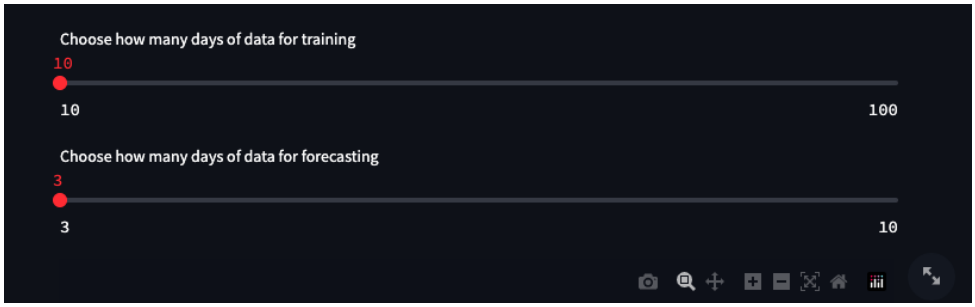


Figure 5.6
Stock Forecast