Capstone Project: Netflix Data

Visualize Netflix data using Python

Completed by: Alexis G

Date: 2019-11-04

Introduction

Netflix Stock Profile

The following is a stock profile (series of studies, visualizations, and analyses that dive into different aspects a publicly traded company's data) of Netflix, completed as part of Codecademy's *Data-Science* path.

The analysis and visualizations are based on Netflix and Dow Jones data from 2017.

This presentation utilizes the following Python libraries:

- Pandas
- Matplotlib
- Seaborn

Code and visualizations prepped using jupyter notebook.

Topics Covered

Stock Health

Distribution of Netflix stock prices throughout 2017, broken down by quarter.

EPS Review

Review of estimated vs. actual earnings per share, broken down by quarter.

Revenue vs. Earnings

Review of Netflix’ influx of revenue and subsequent losses or profits throughout the fiscal year.

Stock vs. Market

Comparing Netflix’ stock growth against the Dow Jones index

Format

Visualization

Findings

Code Review

Visualization

Visualization(s) of the underlying data.

Presented in tandem with Findings.

Findings

Assertions and conclusions reached based on the aforementioned visualizations.

Presented in tandem with Visualizations.

Code Review

A section that would certainly be absent in a presentation of financial health (i.e. to non-tech users), Code Review includes any relevant notes for the purposes of fact-checking and optimization.

Stock Profile

Stock Health

Mostly steady growth throughout the year, as indicated by:

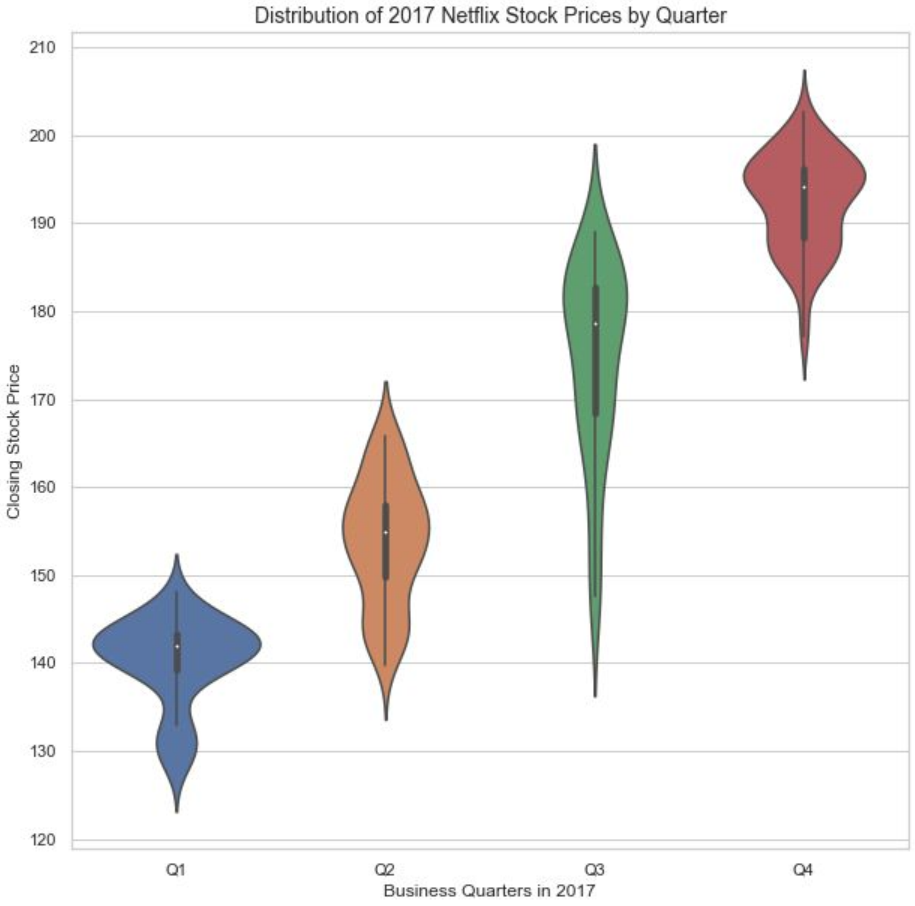
- Each quarter occupying a higher and higher position (relative to CSP)
- Most quarters occupying a relatively equal spread from lowest to highest stock value.

Stock's Closing Price mostly sat between $130 and $200, with extremes at:

- Lowest: $127.48
- Highest: $202.67

While most quarters were relatively steady, Q3 saw the greatest degree of volatility, indicated by:

- Wide distribution of Closing Stock Prices

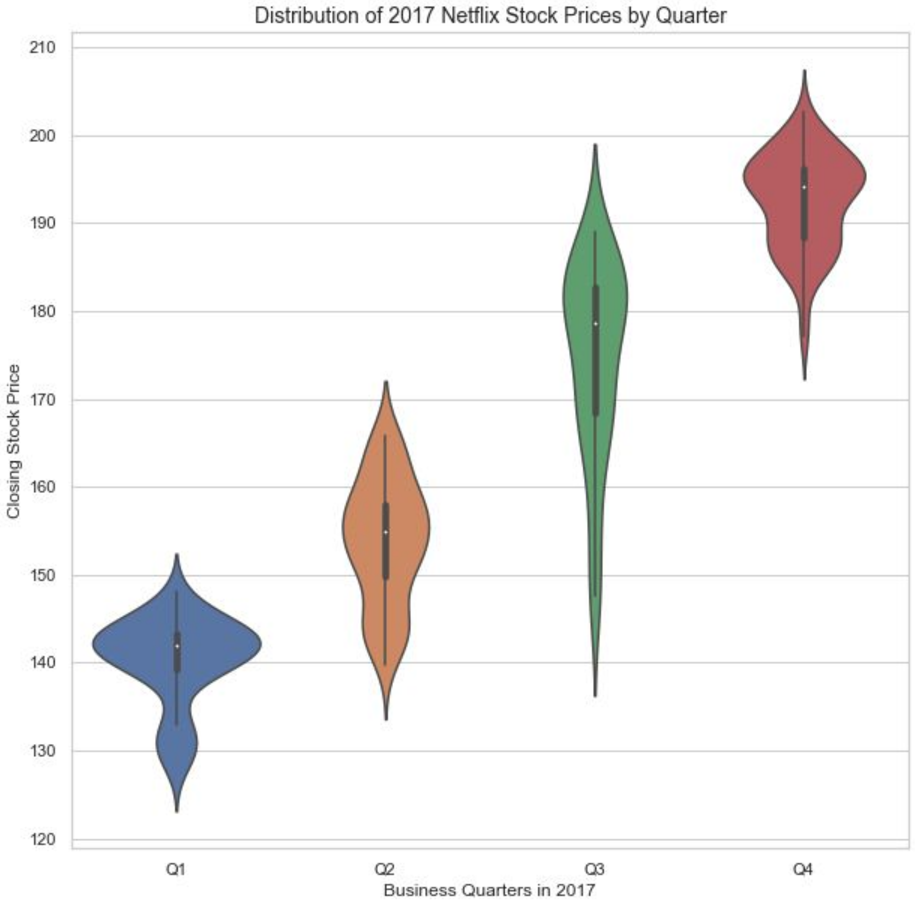


Stock Health: Code

Main changes:

- Applying whitegrid to better tie later quarters to CSP
- Upping the number of yticks for additional granularity

```
plt.figure(figsize=(10, 10))
sns.set(style="whitegrid")
ax =
sns.violinplot(x=netflix_stocks_quarterly[
"Quarter"],
y=netflix_stocks_quarterly["Price"],
data=netflix_stocks_quarterly)
ax.set_title("Distribution of 2017 Netflix
Stock Prices by Quarter", fontsize=14)
ax.set_xlabel("Business Quarters in 2017")
ax.set_ylabel("Closing Stock Price")
ax.set_yticks(list(range(120, 220, 10)))
plt.savefig('Distribution of Netflix Stock
Prices by Quarter.png')
plt.show()
```

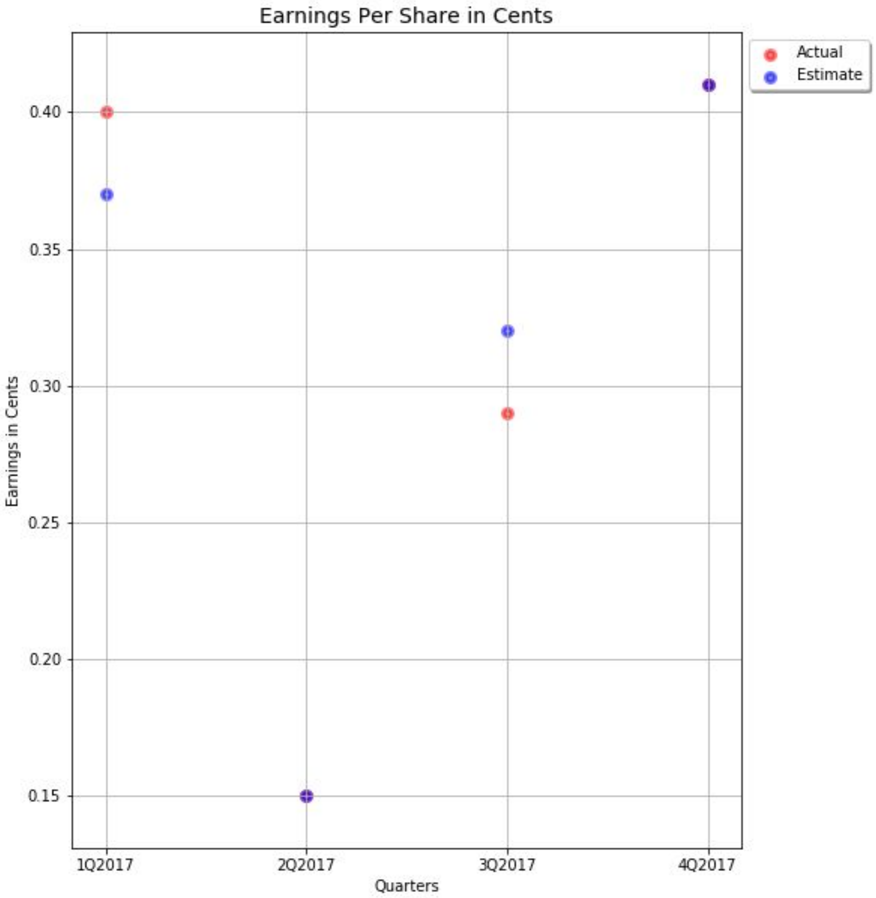


EPS Review

EPS, as predicted by Yahoo, was accurate in 2/4 quarters (indicated with purple dot):

- Q2
- Q4

Netflix EPS outperformed estimates in Q1, but underperformed in Q3.



EPS Review: Code

Main changes:

- Plotting a grid in conjunction with two scatter plots
- Taking the axes position and utilizing that position as a reference point to make the plot 20% narrower.
    - This is done to accommodate a legend outside of the plot itself (while retaining that legend within the figure itself)
- Placing the legend outside of the plot for readability, by passing in the `bbox_to_anchor` keyword.

```
plt.figure(figsize=(10,10))
x_positions = [1, 2, 3, 4]
chart_labels = ["1Q2017","2Q2017","3Q2017","4Q2017"]
earnings_actual =[.4, .15,.29,.41]
earnings_estimate = [.37,.15,.32,.41 ]

ax = plt.subplot()
plt.scatter(x_positions, earnings_actual, c='red', alpha=0.5,
linewidths=3)
plt.scatter(x_positions, earnings_estimate, c='blue', alpha=0.5,
linewidths=3)
plt.grid(b=True)
plt.xticks(x_positions, chart_labels)
ax.set_title("Earnings Per Share in Cents", fontsize=14)
ax.set_xlabel("Quarters")
ax.set_ylabel("Earnings in Cents")
box = ax.get_position()
ax.set_position([box.x0, box.y0, box.width * 0.8, box.height]) #
Bringing in the width by 20% to fit the legend.
ax.legend(["Actual", "Estimate"], shadow=True, fancybox=True,
bbox_to_anchor=(1.2, 1))
plt.savefig('Earnings Per Share in Cents.png')
plt.show()
```

Revenue vs. Earnings

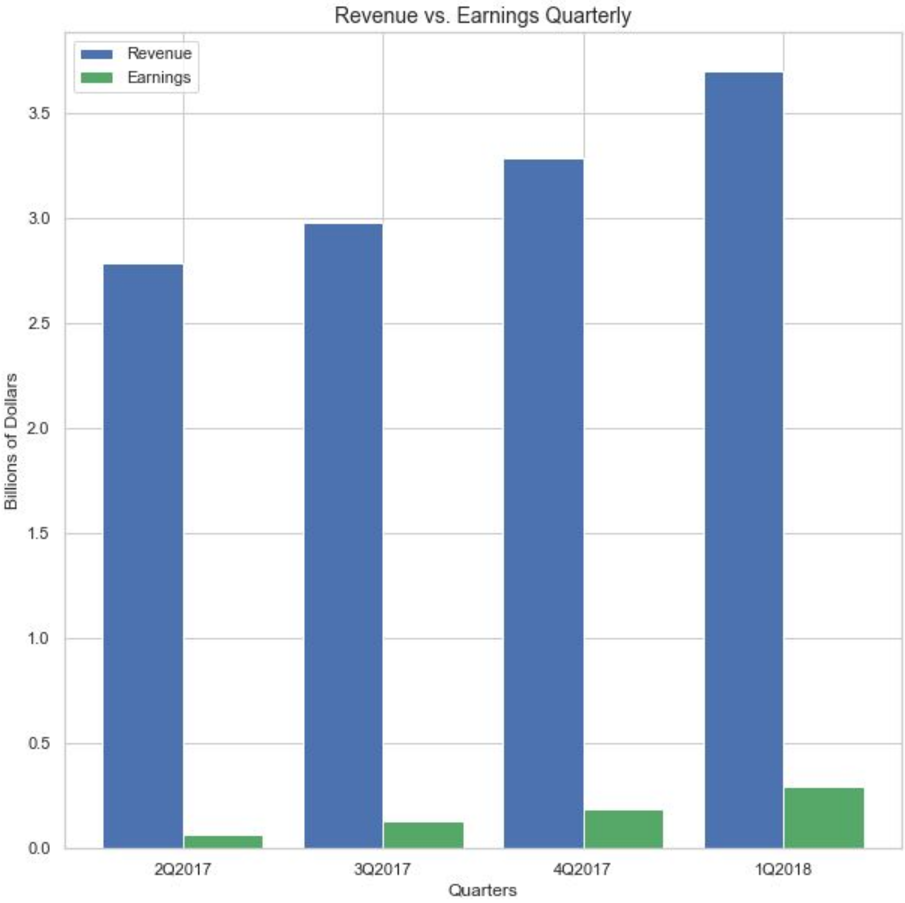
Both revenue and earnings saw a consistent upward trend throughout the fiscal year:

- Start:
  - Revenue: 2.79 (billion)
  - Earnings: 0.065 (billion)
- End:
  - Revenue: 3.7 (billion)
  - Earnings: 0.29 (billion)

The ratio of earnings against revenue also increased:

- Q1: 2.3%
- Q2: 4.3%
- Q3: 5.6%
- Q4: 7.8%

Meaning: Netflix increased their footprint while making themselves ultimately leaner and more profitable.



Revenue vs. Earnings: Code

Main changes:

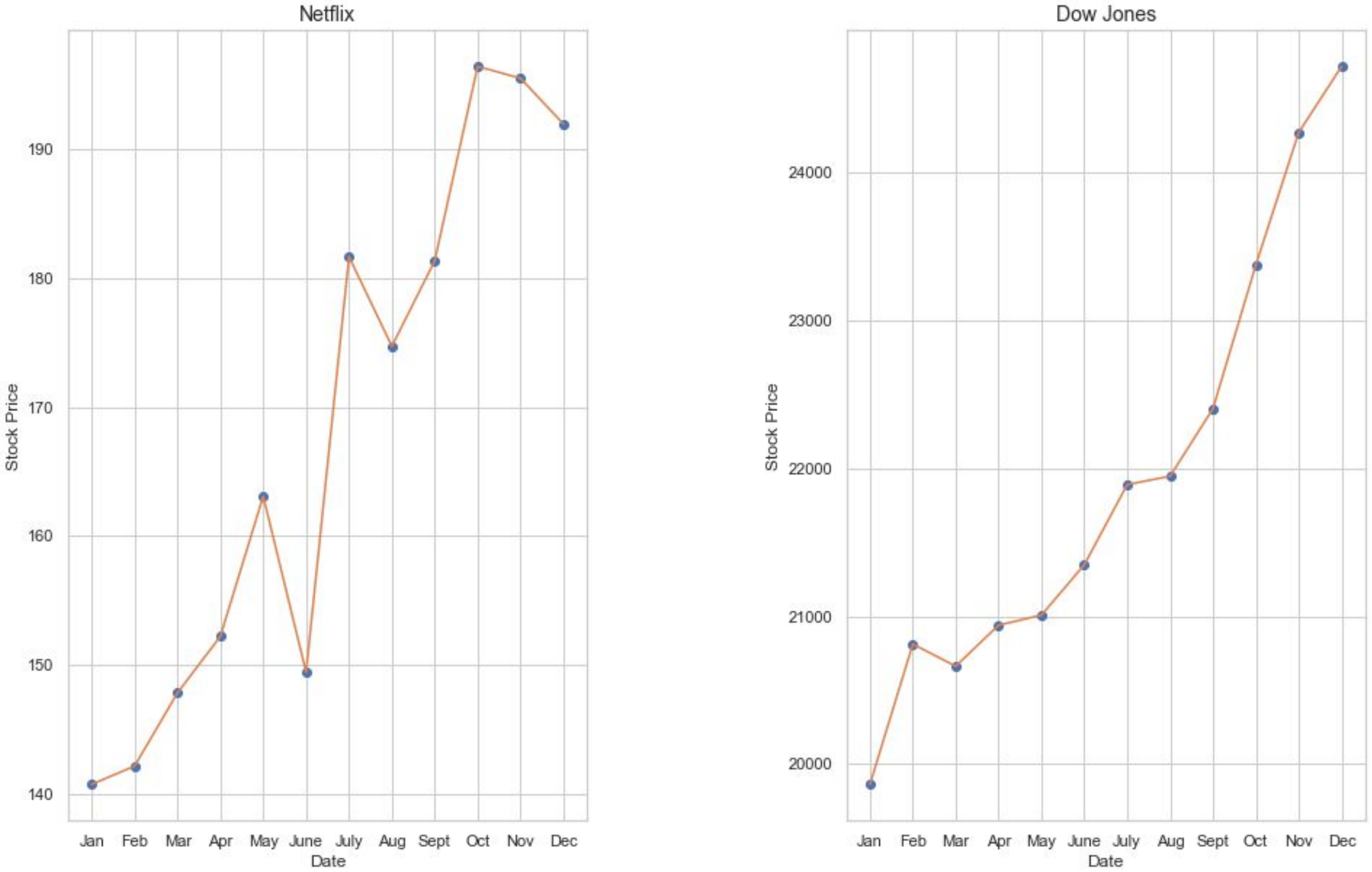
- Retained whitegrid to easily compare datasets against metrics
- Changed earnings (i..e profit) color to green to highlight profit to company.

```
# The metrics below are in billions of dollars
revenue_by_quarter = [2.79, 2.98,3.29,3.7]
earnings_by_quarter = [.0656,.12959,.18552,.29012]
quarter_labels = ["2Q2017","3Q2017","4Q2017", "1Q2018"]

# Bar offset formulae and variable assignment omitted for brevity

middle_x = [ (a + b) / 2.0 for a, b in zip(bars1_x, bars2_x)]
labels = ["Revenue", "Earnings"]

plt.figure(figsize=(10,10))
ax = plt.subplot()
plt.bar(bars1_x, revenue_by_quarter, color='b')
plt.bar(bars2_x, earnings_by_quarter, color='g')
ax.set_title("Revenue vs. Earnings Quarterly", fontsize=14)
ax.set_xlabel("Quarters")
ax.set_ylabel("Billions of Dollars")
plt.xticks(middle_x, quarter_labels)
plt.legend(labels)
plt.savefig('Revenue vs Earnings Quarterly.png')
plt.show()
```

Code used to determine ratio between revenue and earnings for each quarter

```
ratios = []
for i in range(len(revenue_by_quarter)):
    ratios.append(earnings_by_quarter[i] / revenue_by_quarter[i] *
100)
print(ratios)
print("{:.2f}%".format(sum(ratios) / len(ratios)))
```

How did Netflix do relative to the Dow Jones Index?



Stock vs DJIA

Overall, both Netflix stock value and the Dow Jones rose throughout 2017.

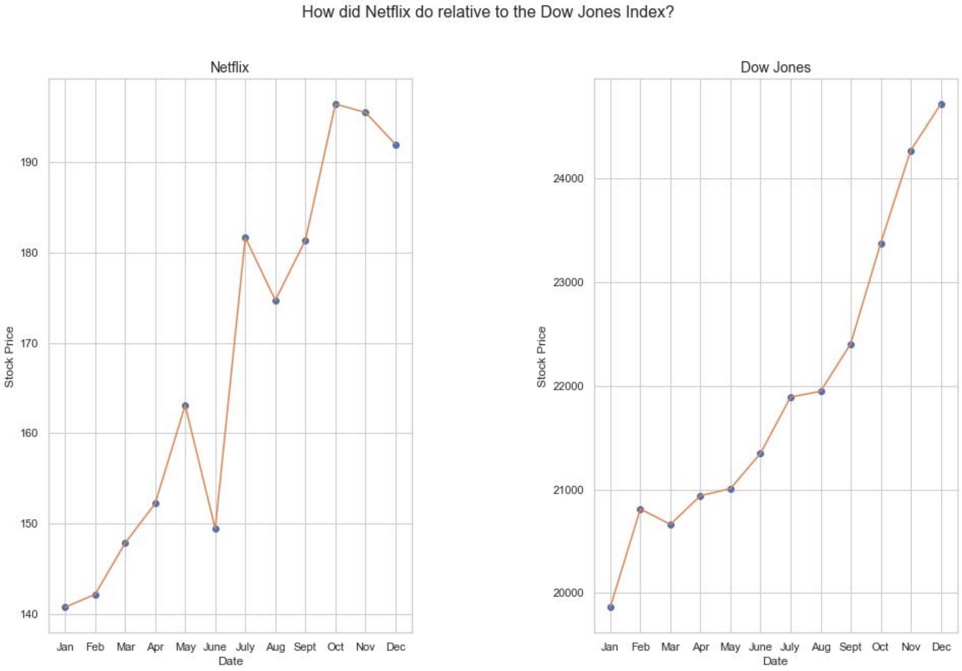
However, Netflix stock was far more volatile, seeing downturns in Q2, Q3, and Q4.

- In comparison, the Dow Jones only saw a brief downturn in Q1.

This means:

- Netflix stock overall was more volatile than the general stock market.
- The stock's volatility was likely not caused by the market.

Netflix accounts for less than 1% of the Dow Jones stock price, suggesting it has little material impact on the overall stock market.



Stock vs. DJIA: Code

Main changes:

- Replaced the yyyy-mm-dd dates from the underlying data set and utilizing a list of month names (abbreviated) and the `set_xticklabels` method for immediate readability.

- Passing in overlapping datasets with different linestyles in each `plt.plot()` call to generate a line with points at each month to highlight price points at regular intervals.

```
# Prepping figure
compfig = plt.figure(figsize=(16,10))
# Making the figure more readable with month names and applying a
supertitle
month_names = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'June', 'July',
'Aug', 'Sept', 'Oct', 'Nov', 'Dec']
compfig.suptitle("How did Netflix do relative to the Dow Jones
Index?", fontsize=16)

# Left plot Netflix
ax1 = plt.subplot(1, 2, 1)
plt.plot(netflix_stocks["Date"], netflix_stocks["Price"], 'o',
netflix_stocks["Date"], netflix_stocks["Price"], '-')
ax1.set_title("Netflix", fontsize=14)
ax1.set_xlabel("Date")
ax1.set_ylabel("Stock Price")
ax1.set_xticklabels(month_names)

# Right plot Dow Jones
ax2 = plt.subplot(1, 2, 2)
plt.plot(dowjones_stocks["Date"], dowjones_stocks["Price"], 'o',
dowjones_stocks["Date"], dowjones_stocks["Price"], '-')
ax2.set_title("Dow Jones", fontsize=14)
ax2.set_xlabel("Date")
ax2.set_ylabel("Stock Price")
ax2.set_xticklabels(month_names)

plt.subplots_adjust(wspace=0.5)
plt.savefig('Netflix and Dow Jones Stocks 2017.png')
plt.show()
```