Investigating Netflix Movies

June 11, 2024

0.1 1. Loading your friend's data into a dictionary

Netflix! What started in 1997 as a DVD rental service has since exploded into the largest entertainment/media company by market capitalization, boasting over 200 million subscribers as of January 2021.

Given the large number of movies and series available on the platform, it is a perfect opportunity to flex our data manipulation skills and dive into the entertainment industry. Our friend has also been brushing up on their Python skills and has taken a first crack at a CSV file containing Netflix data. For their first order of business, they have been performing some analyses, and they believe that the average duration of movies has been declining.

As evidence of this, they have provided us with the following information. For the years from 2011 to 2020, the average movie durations are 103, 101, 99, 100, 100, 95, 95, 96, 93, and 90, respectively.

If we're going to be working with this data, we know a good place to start would be to probably start working with pandas. But first we'll need to create a DataFrame from scratch. Let's start by creating a Python object covered in Intermediate Python: a dictionary!

[2]:

```python
# Create the years and durations lists
sample_years = [2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019, 2020]
sample_durations = [103, 101, 99, 100, 100, 95, 95, 96, 93, 90]
 # Create a dictionary with the two lists
sample_movie_dict = {
 'years': sample_years,
 'durations': sample_durations
}
# Print the dictionary
sample_movie_dict
```

[2]:

```
{'years': [2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019, 2020],
 'durations': [103, 101, 99, 100, 100, 95, 95, 96, 93, 90]}
```

0.2 2. Creating a DataFrame from a dictionary

To convert our dictionary movie_dict to a pandas DataFrame, we will first need to import the library under its usual alias. We'll also want to inspect our DataFrame to ensure it was created correctly. Let's perform these steps now.

[3]:

```python
# Import pandas under its usual alias
import pandas as pd
# Create a DataFrame from the dictionary
sample_durations_df = pd.DataFrame(sample_movie_dict)
# Print the DataFrame
sample_durations_df
```

[3]:

```
   years  durations
0   2011        103
1   2012        101
2   2013         99
3   2014        100
4   2015        100
5   2016         95
6   2017         95
7   2018         96
8   2019         93
9   2020         90
```

0.3 3. A visual inspection of our data

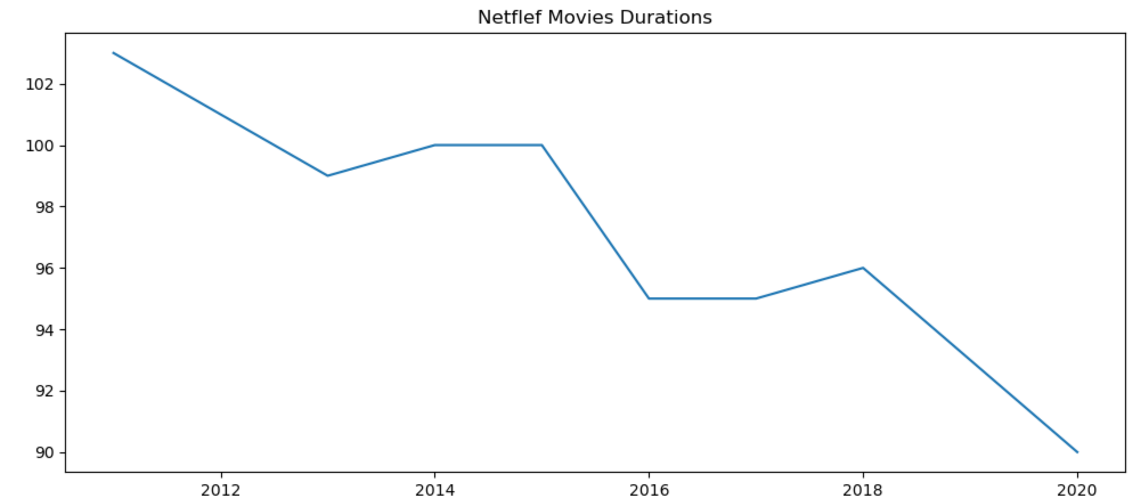
Alright, we now have a pandas DataFrame, the most common way to work with tabular data in Python. Now back to the task at hand. We want to follow up on our friend's assertion that movie lengths have been decreasing over time. A great place to start will be a visualization of the data.

Given that the data is continuous, a line plot would be a good choice, with the dates represented along the x-axis and the average length in minutes along the y-axis. This will allow us to easily spot any trends in movie durations. There are many ways to visualize data in Python, but matploblib.pyplot is one of the most common packages to do so.

Note: In order for us to correctly test your plot, you will need to initalize a matplotlib.pyplot Figure object, which we have already provided in the cell below. You can continue to create your plot as you have learned in Intermediate Python.

[4]:

```python
# Import matplotlib.pyplot under its usual alias and create a figure
import matplotlib.pyplot as plt
fig = plt.figure(figsize=(12, 5))
# Draw a line plot of release_years and durations
plt.plot(sample_durations_df['years'], sample_durations_df['durations'])
# Create a title
plt.title('Netflef Movies Durations')
# Show the plot
plt.show()
```



0.4 4. Loading the rest of the data from a CSV

Well, it looks like there is something to the idea that movie lengths have decreased over the past ten years! But equipped only with our friend's aggregations, we're limited in the further explorations we can perform. There are a few questions about this trend that we are currently unable to answer, including:

What does this trend look like over a longer period of time?

Is this explainable by something like the genre of entertainment?

Upon asking our friend for the original CSV they used to perform their analyses, they gladly oblige and send it. We now have access to the CSV file, available at the path "datasets/netflix_data.csv". Let's create another DataFrame, this time with all of the data. Given the length of our friend's data, printing the whole DataFrame is probably not a good idea, so we will inspect it by printing only the first five rows.

```
[5]: # Read in the CSV as a DataFrame
     netflix_df = pd.read_csv('./datasets/netflix_data.csv')
     # Print the first five rows of the DataFrame
     netflix_df.head()
```

```
[5]:   show_id     type  title           director  \
     0      s1  TV Show     3%                NaN
     1      s2    Movie   7:19  Jorge Michel Grau
     2      s3    Movie  23:59       Gilbert Chan
     3      s4    Movie      9        Shane Acker
     4      s5    Movie     21     Robert Luketic


                                                    cast        country  \
     0  João Miguel, Bianca Comparato, Michel Gomes, R…         Brazil
```

```
1  Demián Bichir, Héctor Bonilla, Oscar Serrano, …         Mexico
2  Tedd Chan, Stella Chung, Henley Hii, Lawrence …      Singapore
3  Elijah Wood, John C. Reilly, Jennifer Connelly…  United States
4  Jim Sturgess, Kevin Spacey, Kate Bosworth, Aar…  United States

          date_added  release_year  duration  \
0    August 14, 2020          2020         4
1  December 23, 2016          2016        93
2  December 20, 2018          2011        78
3  November 16, 2017          2009        80
4    January 1, 2020          2008       123

                                         description             genre
0  In a future where the elite inhabit an island …  International TV
1  After a devastating earthquake hits Mexico Cit…            Dramas
2  When an army recruit is found dead, his fellow…     Horror Movies
3  In a postapocalyptic world, rag-doll robots hi…            Action
4  A brilliant group of students become card-coun…            Dramas
```

[6]:

```
netflix_df.isnull().sum()
 # .isnull tells that in which columns are empty rows
 # .sum tells the number of empty rows
```

[6]:

```
show_id            0
type               0
title              0
director        2389
cast             718
country          507
date_added        10
release_year       0
duration           0
description        0
genre              0
dtype: int64
```

- We have 11 columns in which 2 columns are numerical and remaining are categorical columns
- 4 columns have null values

[8]:

```
# Create the years and durations lists
years = netflix_df.release_year.to_list()
durations = netflix_df.duration.to_list()
# Create a dictionary with the two lists
movie_dict = {
    'years': years,
    'durations': durations
}
```

[10]:

```
# Create a DataFrame from the dictionary
durations_df = pd.DataFrame(movie_dict)
# Print the DataFrame
durations_df.head(10)
```

[10]:

```
   years  durations
0   2020          4
1   2016         93
2   2011         78
3   2009         80
4   2008        123
5   2016          1
6   2019         95
7   1997        119
8   2019        118
9   2008        143
```

0.5 3. A visual inspection of our data

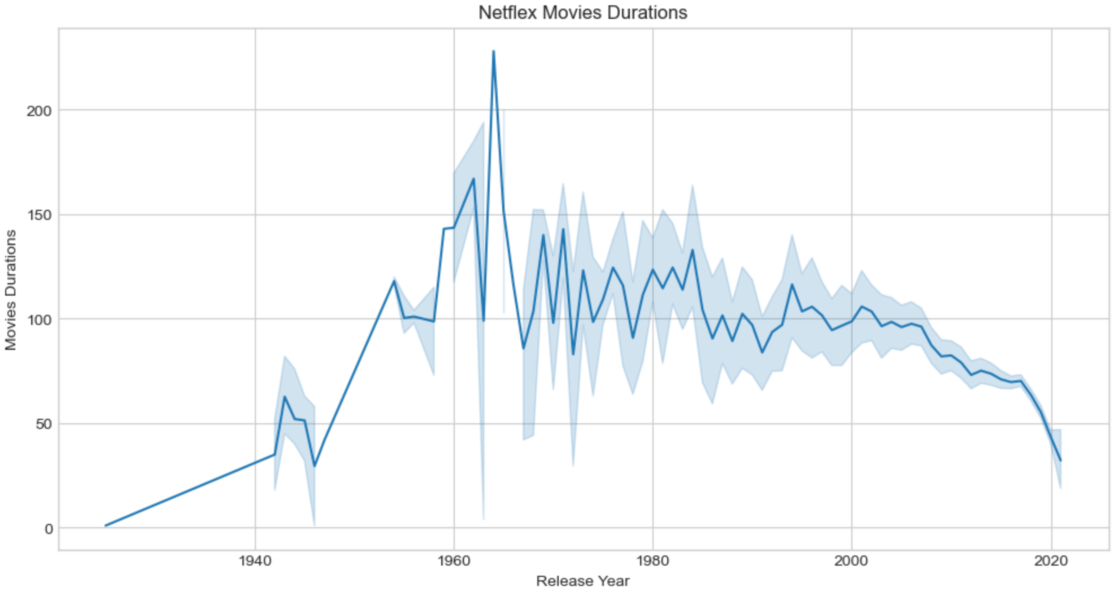
Alright, we now have a pandas DataFrame, the most common way to work with tabular data in Python. Now back to the task at hand. We want to follow up on our friend's assertion that movie lengths have been decreasing over time. A great place to start will be a visualization of the data.

Given that the data is continuous, a line plot would be a good choice, with the dates represented along the x-axis and the average length in minutes along the y-axis. This will allow us to easily spot any trends in movie durations. There are many ways to visualize data in Python, but matploblib.pyplot is one of the most common packages to do so.

Note: In order for us to correctly test your plot, you will need to initalize a matplotlib.pyplot Figure object, which we have already provided in the cell below. You can continue to create your plot as you have learned in Intermediate Python.

[13]:

```
# To remove all jupyter warnings
import warnings
warnings.filterwarnings('ignore')
# Import seaborn under its usual alias and create a figure
import seaborn as sns
plt.figure(figsize=(12, 6))
plt.style.use('seaborn-v0_8-whitegrid')
# Draw a line plot of release_years and durations
sns.lineplot(x='years', y='durations', data=durations_df)
# Create a title
plt.title('Netflex Movies Durations')
# Add x label
plt.xlabel('Release Year')
# Add ylabel
plt.ylabel('Movies Durations')
# show the plot
plt.show()
```



0.6 5. Filtering for movies!

Okay, we have our data! Now we can dive in and start looking at movie lengths.

Or can we? Looking at the first five rows of our new DataFrame, we notice a column type. Scanning the column, it's clear there are also TV shows in the dataset! Moreover, the duration column we planned to use seems to represent different values depending on whether the row is a movie or a show (perhaps the number of minutes versus the number of seasons)?

Fortunately, a DataFrame allows us to filter data quickly, and we can select rows where type is Movie. While we're at it, we don't need information from all of the columns, so let's create a new DataFrame netflix_movies containing only title, country, genre, release_year, and duration.

Let's put our data subsetting skills to work!

```
[14]: # Subset the DataFrame for type "Movie" only
      netflix_movies_df = netflix_df[netflix_df['type'] == 'Movie']

      # Select only the columns of interest
      netflix_movies_col_subset = netflix_movies_df[['title', 'country', 'genre',␣
        ↪'release_year', 'duration']]

      # Print the first five rows of the new DataFrame
      netflix_movies_col_subset.head()
```

```
[14]:    title        country          genre  release_year  duration
      1   7:19         Mexico         Dramas          2016        93
      2  23:59      Singapore  Horror Movies          2011        78
      3      9  United States         Action          2009        80
```

```
4      21   United States          Dramas           2008        123
6     122           Egypt   Horror Movies           2019         95
```

0.7 6. Creating a scatter plot

Okay, now we're getting somewhere. We've read in the raw data, selected rows of movies, and have limited our DataFrame to our columns of interest. Let's try visualizing the data again to inspect the data over a longer range of time.

This time, we are no longer working with aggregates but instead with individual movies. A line plot is no longer a good choice for our data, so let's try a scatter plot instead. We will again plot the year of release on the x-axis and the movie duration on the y-axis.

Note: Although not taught in Intermediate Python, we have provided you the code fig = plt.figure(figsize=(12,8)) to increase the size of the plot (to help you see the results), as well as to assist with testing. For more information on how to create or work with a matplotlib figure, refer to the documentation.

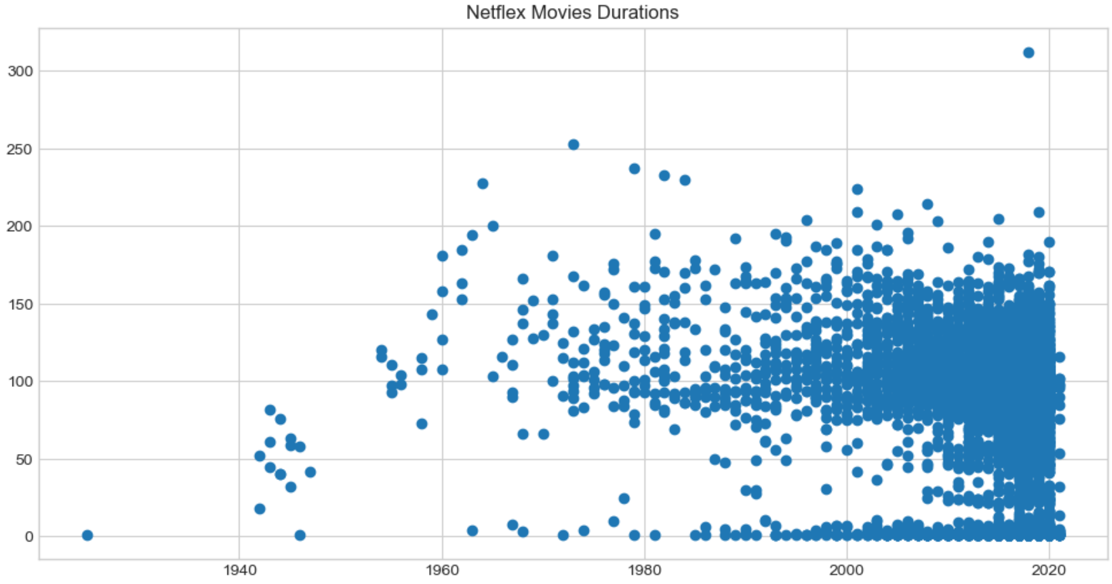
[15]:

```
# Create a figure and increase the figure size
fig = plt.figure(figsize=(12,6))

# Create a scatter plot of duration versus year
plt.scatter(x='years', y='durations', data=durations_df)

# Create a title
plt.title('Netflex Movies Durations')

# Show the plot
plt.show()
```



0.8 7. Digging deeper

This is already much more informative than the simple plot we created when our friend first gave us some data. We can also see that, while newer movies are overrepresented on the platform, many short movies have been released in the past two decades.

Upon further inspection, something else is going on. Some of these films are under an hour long! Let's filter our DataFrame for movies with a duration under 60 minutes and look at the genres. This might give us some insight into what is dragging down the average.

[17]:

```
# Filter for durations shorter than 60 minutes
short_movies = netflix_movies_df[netflix_movies_df['duration'] < 60]

# Print the first 7 rows of short_movies
short_movies.head(7)
```

[17]:

```
    show_id   type                                               title  \
35      s36  Movie                                           #Rucker50
55      s56  Movie                      100 Things to do Before High School
67      s68  Movie  13TH: A Conversation with Oprah Winfrey & Ava …
101    s102  Movie                                    3 Seconds Divorce
146    s147  Movie                                      A 3 Minute Hug
162    s163  Movie  A Christmas Special: Miraculous: Tales of Lady…
171    s172  Movie                           A Family Reunion Christmas

                  director                                              cast  \
35   Robert McCullough Jr.                                               NaN
55                     NaN  Isabela Moner, Jaheem Toombs, Owen Joyner, Jac…
67                     NaN                        Oprah Winfrey, Ava DuVernay
101           Shazia Javed                                               NaN
146      Everardo González                                               NaN
162         Thomas Astruc  Cristina Vee, Bryce Papenbrook, Keith Silverst…
171      Robbie Countryman  Loretta Devine, Tia Mowry-Hardrict, Anthony Al…

           country         date_added  release_year  duration  \
35   United States   December 1, 2016          2016        56
55   United States   November 2, 2019          2014        44
67             NaN   January 26, 2017          2017        37
101         Canada      June 15, 2019          2018        53
146         Mexico   October 28, 2019          2019        28
162         France  December 20, 2016          2016        22
171  United States   December 9, 2019          2019        29

                                          description          genre
35   This documentary celebrates the 50th anniversa…  Documentaries
55   Led by seventh-grader C.J., three students who…  Uncategorized
67   Oprah Winfrey sits down with director Ava DuVe…  Uncategorized
101  A Muslim women's activist group in India prote…  Documentaries
```

```
146  This documentary captures the joy and heartbre…  Documentaries
162  Parisian teen Marinette transforms herself int…  Uncategorized
171  M'Dear and her sisters struggle to keep their …  Uncategorized
```

0.9 8. Marking non-feature films

Interesting! It looks as though many of the films that are under 60 minutes fall into genres such as “Children”, “Stand-Up”, and “Documentaries”. This is a logical result, as these types of films are probably often shorter than 90 minute Hollywood blockbuster.

We could eliminate these rows from our DataFrame and plot the values again. But another interesting way to explore the effect of these genres on our data would be to plot them, but mark them with a different color.

In Python, there are many ways to do this, but one fun way might be to use a loop to generate a list of colors based on the contents of the genre column. Much as we did in Intermediate Python, we can then pass this list to our plotting function in a later step to color all non-typical genres in a different color!

Note: Although we are using the basic colors of red, blue, green, and black, matplotlib has many named colors you can use when creating plots. For more information, you can refer to the documentation here!

[20]:
```
# Define an empty list
colors = []

# Iterate over rows of netflix_movies_col_subset
for rows, columns in netflix_movies_col_subset.iterrows() :
    if columns['genre'] == 'Children':
        colors.append('red')
    elif columns['genre'] == 'Stand-Up':
        colors.append('green')
    elif columns['genre'] == 'Documentaries':
        colors.append('blue')
    else:
        colors.append('black')

 # Inspect the first 10 values in your list
colors[:10]
```

[20]:
```
['black',
 'black',
 'black',
 'black',
 'black',
 'black',
 'black',
 'black',
 'black',
 'blue']
```

[24]:

```
netflix_df_movies_only.columns
```

[24]:

```
Index(['show_id', 'type', 'title', 'director', 'cast', 'country', 'date_added',
       'release_year', 'duration', 'description', 'genre'],
      dtype='object')
```

0.10 9. Plotting with color!

Lovely looping! We now have a colors list that we can pass to our scatter plot, which should allow us to visually inspect whether these genres might be responsible for the decline in the average duration of movies.

This time, we'll also spruce up our plot with some additional axis labels and a new theme with plt.style.use(). The latter isn't taught in Intermediate Python, but can be a fun way to add some visual flair to a basic matplotlib plot. You can find more information on customizing the style of your plot here!

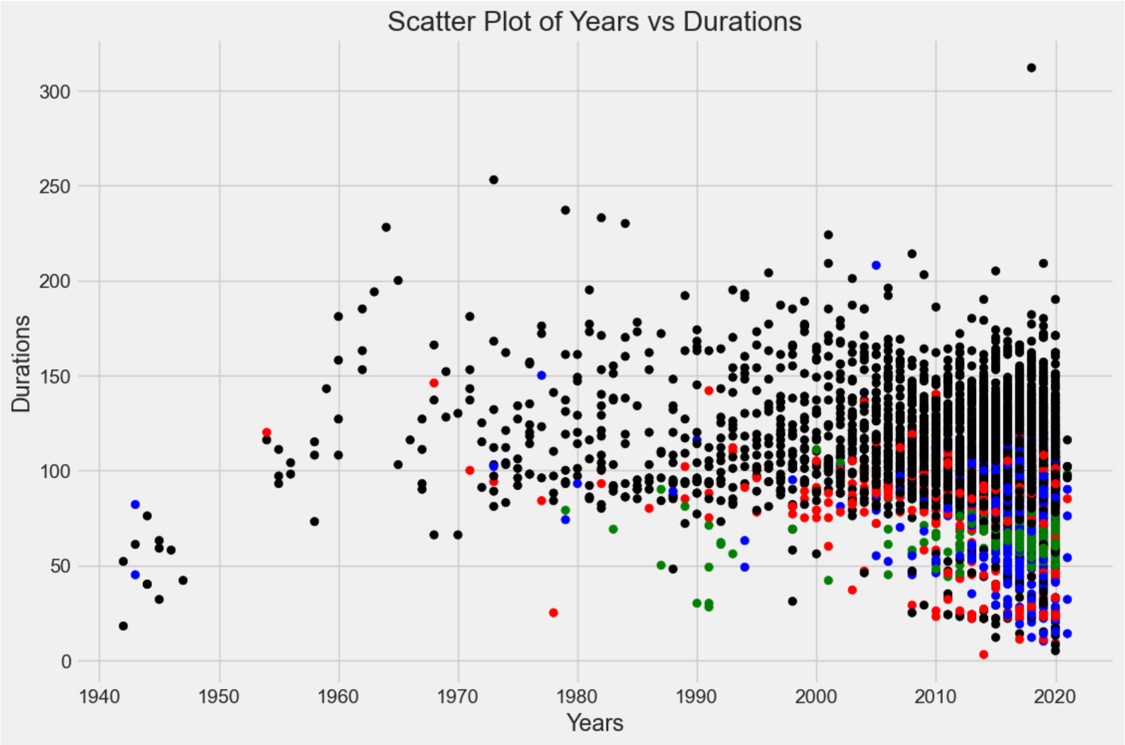
[21]:

```python
# Set the figure style and initalize a new figure
plt.style.use('fivethirtyeight')
fig = plt.figure(figsize=(12,8))

# Create a scatter plot of duration versus release_year
plt.scatter(x='release_year', y='duration', data=netflix_movies_df, c=colors)

# Create a title and axis labels
plt.title('Scatter Plot of Years vs Durations')
plt.xlabel('Years')
plt.ylabel('Durations')

# Show the plot
plt.show()
```



0.11 10. What next?

Well, as we suspected, non-typical genres such as children's movies and documentaries are all clustered around the bottom half of the plot. But we can't know for certain until we perform additional analyses.

Congratulations, you've performed an exploratory analysis of some entertainment data, and there are lots of fun ways to develop your skills as a Pythonic data scientist. These include learning how to analyze data further with statistics, creating more advanced visualizations, and perhaps most importantly, learning more advanced ways of working with data in pandas. This latter skill is covered in our fantastic course Data Manipulation with pandas.

We hope you enjoyed this application of the skills learned in Intermediate Python, and wish you all the best on the rest of your journey!

```
[ ]: # Are we certain that movies are getting shorter?
     are_movies_getting_shorter = 'Yes the movies are the getting shorter by each␣
      ↪year.'
```