Clustering Startups

April 16, 2025

1 Successful Startups and Common Indicators Between Them

1.1 Purpose and Questions

Throughout our society today, startups have become a central force in driving innovation, job creation, and economic growth. The number of startups reaching billion-dollar "unicorn" valuations has grown rapidly in recent years. However, the factors that lead some startups to achieve such success while others fail remain widely debated. Many people point to location, industry, total funding, number of investors, or even the background of the founders. Despite the variety of explanations, no single factor has emerged as the definitive key to success.

Throughout this paper, I hope to explore the key characteristics that correlate with a startup reaching unicorn status and investigate whether those correlations imply causation or if deeper structural patterns are at play.

1.1.1 Questions to Answer

- Does the country or city in which a startup is founded significantly affect its likelihood of reaching unicorn status?
- Does the amount of total funding or number of funding rounds correlate with higher startup valuation?
- Do certain industries have a higher success rate in producing unicorns?

I will be using a dataset from Kaggle that includes information on over 1,000 startups valued at $1 billion or more. You can access the dataset with this link: https://www.kaggle.com/datasets/thedevastator/startups-valued-at-1-billion-or-more. I will focus on uncovering what makes these startups different and what lessons we can draw from their success.

```
[5]: #first, let's load the dataset, we'll need pandas
     import pandas as pd

     df = pd.read_csv('unicorns.csv')
```

```
[6]: df.shape #rows x columns
```

```
[6]: (1199, 11)
```

1.2 Preprocessing

Before we start visualizing data, we should evaluate the dataset and determine whether the dataset needs any cleaning done.

[8]:

```
print(df.isnull().sum())   #check for nulls in every column
print(df.isnull().sum().sum())
```

```
Updated at                       0
Company                          0
Crunchbase Url                   0
Last Valuation (Billion $)       0
Date Joined                      0
Year Joined                      0
City                            18
Country                          0
Industry                         0
Investors                        0
Company Website               1199
dtype: int64
1217
```

[9]:

```
df['City'] = df['City'].fillna('Unknown')

df = df.drop(columns=['Company Website'])
# Check if nulls are removed
print(df.isnull().sum())
```

```
Updated at                    0
Company                       0
Crunchbase Url                0
Last Valuation (Billion $)    0
Date Joined                   0
Year Joined                   0
City                          0
Country                       0
Industry                      0
Investors                     0
dtype: int64
```

[10]:

```
print(df.duplicated().sum())  # Count duplicate rows
```

```
0
```

So far, we've checked for any null values and checked for duplicates. I chose to replace missing values with the mode because the amount of nulls weren't considerable enough for a dataset with almost 7000 datapoints. At this point, I am comfortable with our preprocessing steps because it looks like this dataset is ready to go. I chose to create a binned feature called Score Category so that it would be easier for me to analyze the overll performance. I personally don't believe the difference between a 65 and a 67 is necessarily significant enough to evaluate performance.

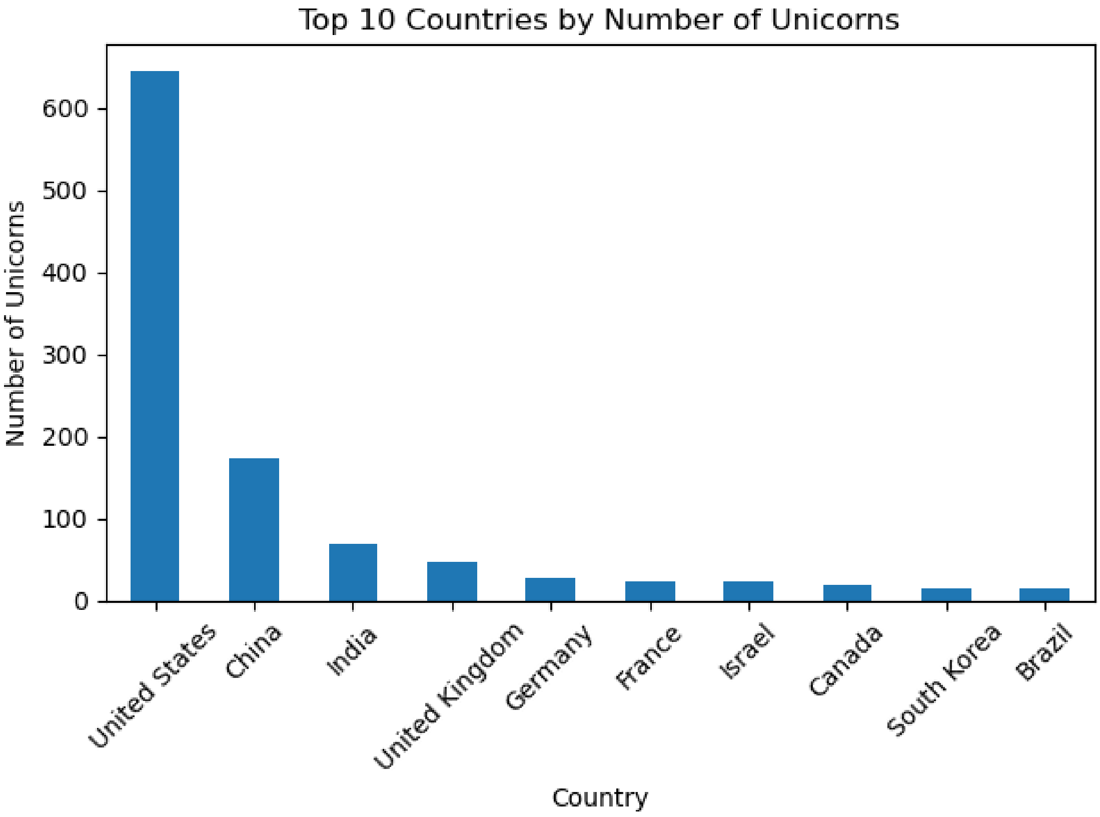
1.3 Visualizations

1.3.1 Top 10 Countries by Number of Unicorns

[14]:

```python
import matplotlib.pyplot as plt

top_countries = df['Country'].value_counts().head(10)
top_countries.plot(kind='bar')
plt.title('Top 10 Countries by Number of Unicorns')
plt.xlabel('Country')
plt.ylabel('Number of Unicorns')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



1.3.2 Unicorn Valuations Over Time

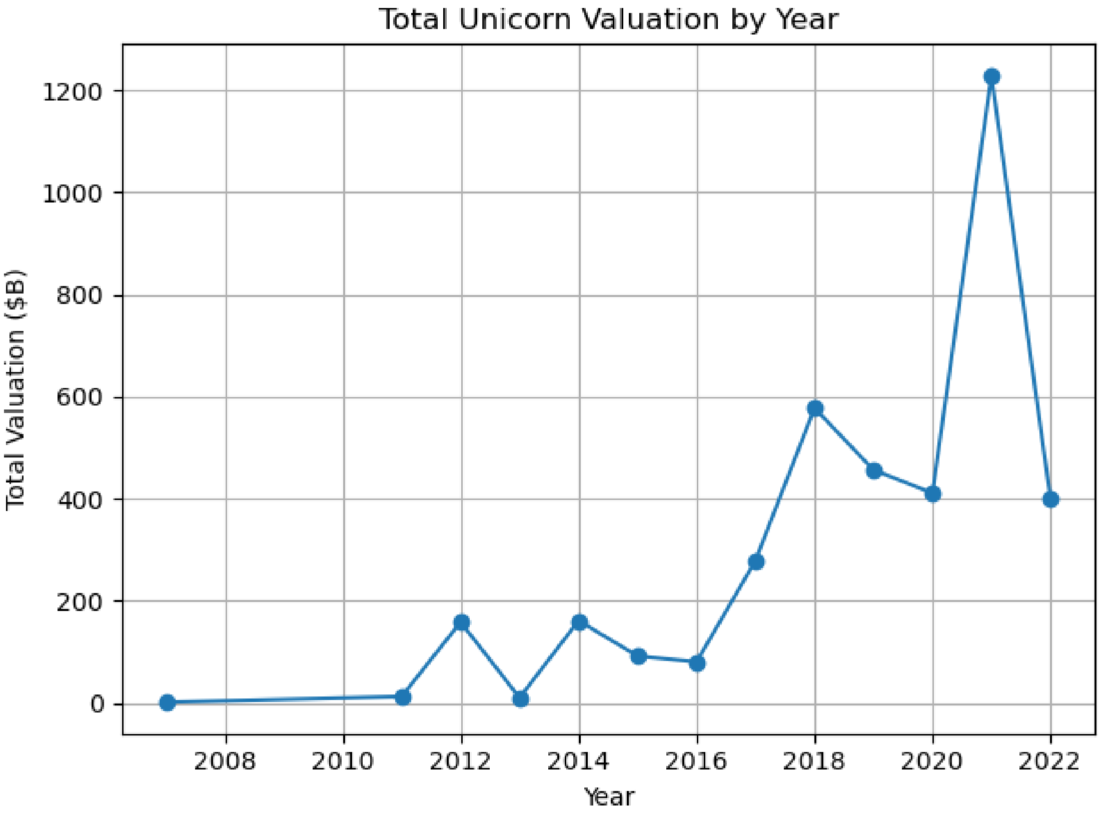
[18]:

```python
df['Date Joined'] = pd.to_datetime(df['Date Joined'], errors='coerce')
df = df.dropna(subset=['Date Joined'])

df['Year Joined'] = df['Date Joined'].dt.year
```

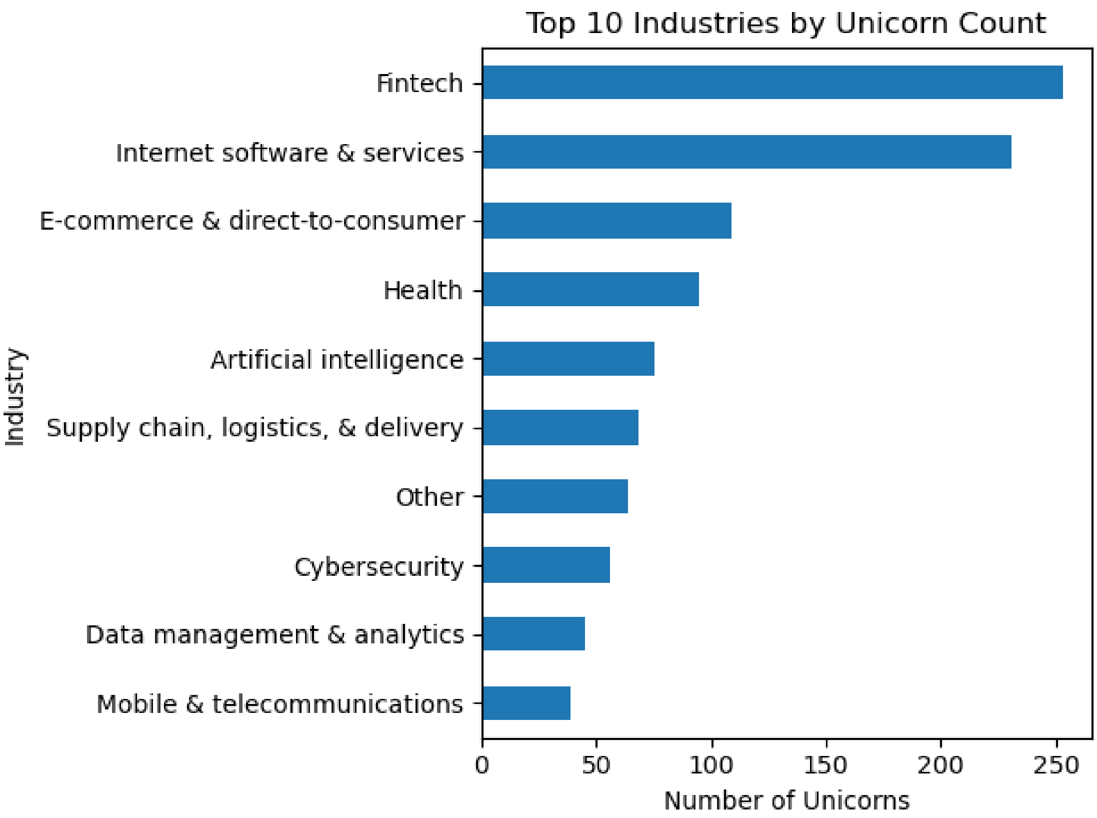
```
valuation_by_year = df.groupby('Year Joined')['Last Valuation (Billion $)'].
↪sum()

valuation_by_year.plot(kind='line', marker='o')
plt.title('Total Unicorn Valuation by Year')
plt.xlabel('Year')
plt.ylabel('Total Valuation ($B)')
plt.grid(True)
plt.tight_layout()
plt.show()
```



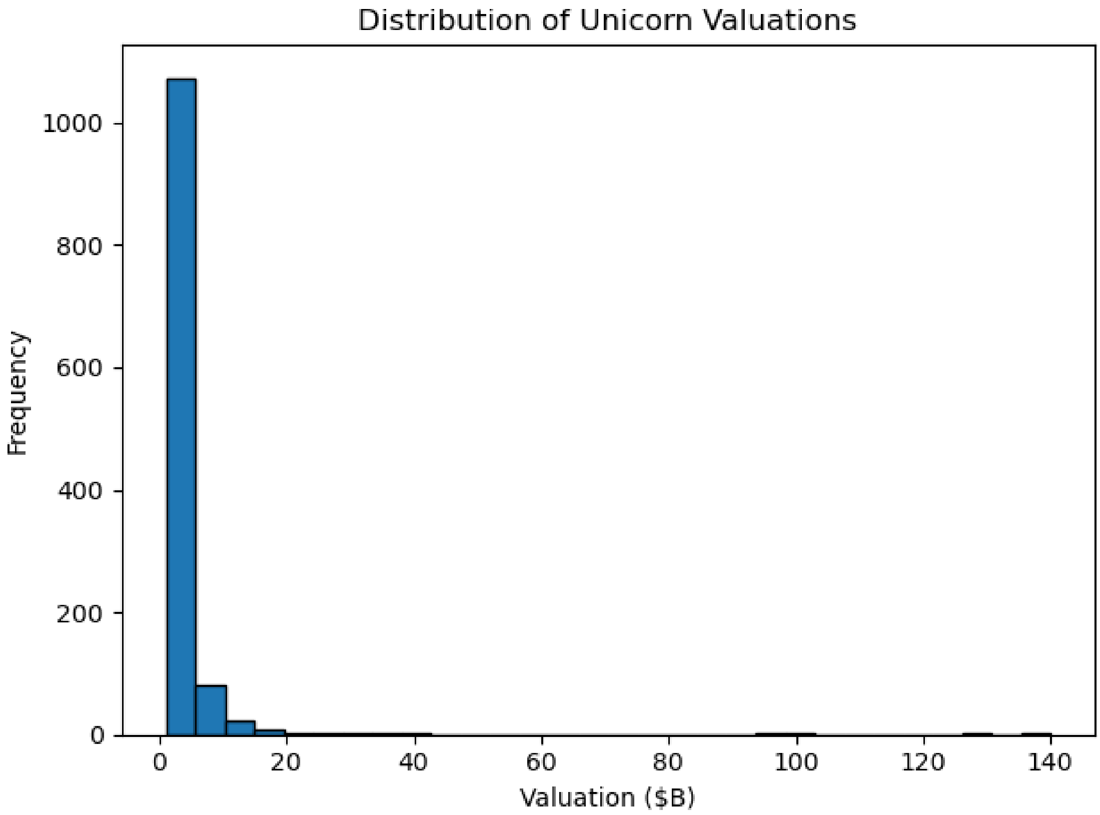
1.3.3 Top 10 Industries with Most Unicorns

[21]:
```
top_industries = df['Industry'].value_counts().head(10)
top_industries.plot(kind='barh')
plt.title('Top 10 Industries by Unicorn Count')
plt.xlabel('Number of Unicorns')
plt.gca().invert_yaxis()
plt.tight_layout()
plt.show()
```



1.3.4 Valuation Distribution

[24]:
```
df['Last Valuation (Billion $)'].plot(kind='hist', bins=30, edgecolor='black')
plt.title('Distribution of Unicorn Valuations')
plt.xlabel('Valuation ($B)')
plt.ylabel('Frequency')
plt.tight_layout()
plt.show()
```



1.3.5 Correlation Heatmap for Numerical Features

[ ]:

[ ]:

1.3.6 s

Ty

[ ]:

School Type

[ ]:

1.3.7

[ ]:

[ ]:

1.4 Summary

1.4.1 Bias and Limitations

1.5 References