Startup Funding in New York City

Introduction

According to New York City is globally recognized as a leading hub for startups, innovation, and venture capital. With an ecosystem valued at over $621 billion and home to more than 150 unicorns, NYC attracts billions in startup funding and continues to be a top destination for entrepreneurs and investors (Startup Genome). This project explores three critical questions within this high-growth environment:

- **Do startups in NYC that receive more funding tend to succeed more often?**
- **How have startup funding patterns shifted over time, especially before and after the COVID-19 pandemic?**
- **Do unicorns with IPO status tend to raise more or have higher valuation?**

Using a dataset of startup investment activity in NYC, this analysis focuses on understanding the relationship between funding amount, industry sector, and startup success (measured by IPOs, acquisitions, or closures). It also examines which sectors attract the most funding, who the most active investors are, and whether COVID-19 significantly impacted funding behavior. By analyzing trends over time, comparing valuations, and applying predictive modeling, this project offers insight into the key factors that influence startup outcomes in one of the world's most competitive tech environments.most competitive tech environments.

Installing Packages

This section ensures that all required Python packages are available in the environment. Some packages like scikit-learn, numpy, seaborn, and statsmodels are essential for machine learning, numerical computations, and statistical modeling. The pip install commands also manage version compatibility to prevent errors—especially useful when working in environments where package conflicts might arise. Redirecting the output to /dev/null hides unnecessary installation logs to keep the notebook clean. This step guarantees that the code will run consistently across different machines or sessions.

In [43]:
```
!pip install scikit-learn==1.2.2 > /dev/null 2>&1;
!pip install --upgrade numpy > /dev/null 2>&1;
!pip install --upgrade seaborn > /dev/null 2>&1;
!pip install numpy==1.24.4 --force-reinstall > /dev/null 2>&1
!pip install seaborn> /dev/null 2>&1;
!pip install statsmodels> /dev/null 2>&1;
!pip install matplotlib> /dev/null 2>&1;
```

Importing Libraries

Once packages are installed, they need to be imported to access their functions. pandas and numpy are used for handling and manipulating data efficiently. matplotlib.pyplot, seaborn, and plotly.express are used for creating visualizations like scatter plots, ROC curves, and probability graphs. statsmodels provides tools for advanced statistical modeling, such as logistic regression. Other imports like warnings and datetime help manage outputs and handle date-related data. Overall, importing these libraries prepares the environment for data analysis and model development.

In [8]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import plotly.express as px
import warnings
from datetime import datetime
import seaborn as sns
import statsmodels.api as sm
```

Importing Datasets

This section is responsible for loading the datasets (investment.csv and Unicorn.csv) into pandas DataFrames for analysis. The read_csv function includes parameters like encoding='ISO-8859-1', quotechar='"', and skipinitialspace=True to correctly interpret characters and handle formatting issues, especially for messy or non-standard CSV files. After loading, the code cleans up the column names by stripping whitespace, converting all text to lowercase, and replacing spaces with underscores for consistency and ease of access. This standardization ensures that column names are uniform, making the dataset easier to manipulate and less prone to syntax errors. Repeating this cleaning step (with astype(str)) also ensures that all column names are properly converted to string types before further processing.

In [9]:
```python
import pandas as pd

# Read the datasets
df = pd.read_csv('investment.csv', encoding='ISO-8859-1', quotechar='"', skipi
df_unicorn = pd.read_csv('Unicorn.csv', encoding='ISO-8859-1', quotechar='"', 

# Clean column names
df.columns = df.columns.str.strip().str.lower().str.replace(' ', '_')
df_unicorn.columns = df_unicorn.columns.str.strip().str.lower().str.replace(' 

df.columns = df.columns.astype(str).str.strip().str.replace(' ', '_').str.lowe
df_unicorn.columns = df_unicorn.columns.astype(str).str.strip().str.replace(' 
```

Data Wranglng and Clenaing

Data Clenaing - Both Datasets

Before conducting any analysis, both datasets—investment.csv and Unicorn.csv—were cleaned to ensure consistency and usability. First, all column names were standardized by converting them to lowercase and replacing spaces with underscores to avoid syntax issues during analysis. Then, key financial columns like funding_total_usd in the investment dataset and total_raised in the unicorn dataset were stripped of currency symbols and converted to numeric data types. For the unicorn data, funding values were scaled appropriately to represent full dollar amounts (in billions). Additionally, date columns such as first_funding_at, last_funding_at, and date_joined were converted to datetime format to enable time-based analysis. This cleaning ensured that the datasets are structured, complete, and ready for meaningful visualization, modeling, and interpretation.

In [ ]:
```
import pandas as pd

# Load datasets with correct encoding
investment_df = pd.read_csv("investment.csv", encoding='ISO-8859-1')
unicorn_df = pd.read_csv("Unicorn.csv", encoding='ISO-8859-1')

# Step 1: Standardize column names
investment_df.columns = investment_df.columns.str.strip().str.lower().str.repla
unicorn_df.columns = unicorn_df.columns.str.strip().str.lower().str.replace(' '

# Step 2: Convert funding columns to numeric
investment_df['funding_total_usd'] = investment_df['funding_total_usd'].replace
investment_df['funding_total_usd'] = pd.to_numeric(investment_df['funding_tota

unicorn_df['total_raised'] = unicorn_df['total_raised'].replace('[\$,B]', '', r
unicorn_df['total_raised'] = pd.to_numeric(unicorn_df['total_raised'], errors='

# Step 3: Convert date columns to datetime
investment_df['first_funding_at'] = pd.to_datetime(investment_df['first_funding
investment_df['last_funding_at'] = pd.to_datetime(investment_df['last_funding_a
unicorn_df['date_joined'] = pd.to_datetime(unicorn_df['date_joined'], errors='c
```

A Quick Overview of the Datasets

The investment dataset contains information about over 54,000 startups, including their industry category (category_list), total funding received (funding_total_usd), current status (status), number of funding rounds, and the dates of their first and last funding events. This overview highlights how much capital startups have raised and tracks whether they are operating, acquired, or closed. The sample provides insights into early-stage funding patterns and sector diversity within the startup ecosystem.

The Unicorn dataset includes data on over 1,000 companies globally that have achieved billion-dollar valuations. Key columns include the company name, valuation (valuation_($b)), total funding raised (total_raised), date joined as a unicorn, industry, country, investors, and city. Since I only needed New York startups, I filted to this specific city only. The overview allows us to explore the sectors and regions where high-value startups are most concentrated, particularly those in New York City, and offers a glimpse into their funding sources and founding years.

In [ ]:
```
import pandas as pd

investment_df = pd.read_csv("investment.csv", encoding='ISO-8859-1')

# Clean column names
investment_df.columns = investment_df.columns.str.strip().str.lower().str.repla

# Convert funding column to numeric
investment_df['funding_total_usd'] = investment_df['funding_total_usd'].replace
investment_df['funding_total_usd'] = pd.to_numeric(investment_df['funding_tota

# Convert date columns
investment_df['first_funding_at'] = pd.to_datetime(investment_df['first_funding
investment_df['last_funding_at'] = pd.to_datetime(investment_df['last_funding_a

# Select relevant columns and drop rows with missing data
investment_sample_5 = investment_df[
    ['name', 'category_list', 'funding_total_usd', 'status', 'funding_rounds',
].dropna().head(5)

# Display the result
print(investment_sample_5)
```

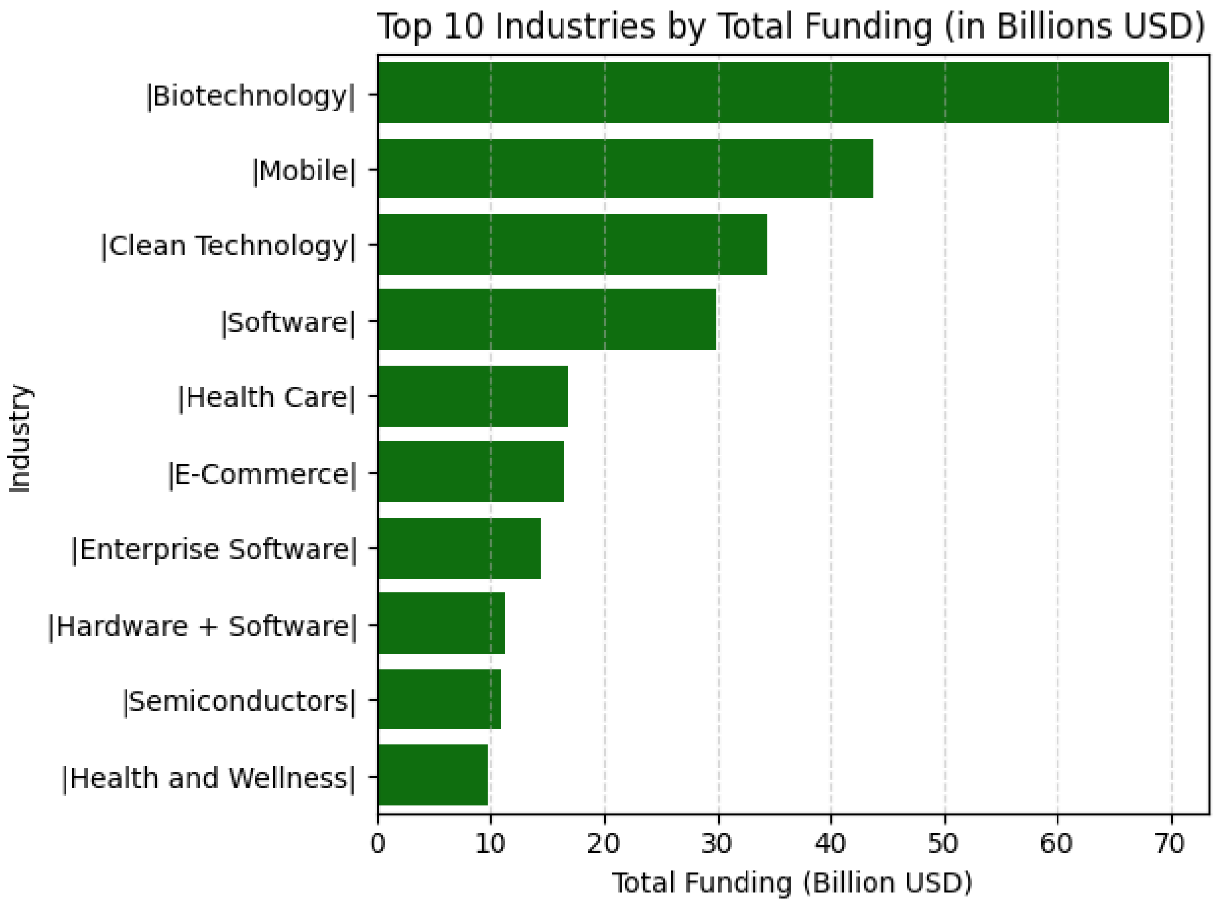
```
                  name                                      category_list  \
0             #waywire         |Entertainment|Politics|Social Media|News|
1   &TV Communications                                            |Games|
2    'Rock' Your Paper                             |Publishing|Education|
3    (In)Touch Network  |Electronics|Guides|Coffee|Restaurants|Music|i...
4   -R- Ranch and Mine                      |Tourism|Entertainment|Games|

   funding_total_usd     status  funding_rounds first_funding_at  \
0          1750000.0   acquired             1.0       2012-06-30
1          4000000.0  operating             2.0       2010-06-04
2            40000.0  operating             1.0       2012-08-09
3          1500000.0  operating             1.0       2011-04-01
4            60000.0  operating             2.0       2014-08-17

  last_funding_at
0      2012-06-30
1      2010-09-23
2      2012-08-09
3      2011-04-01
4      2014-09-26
```

In [ ]:
```
# Filter unicorns in the United States and located in New York City
unicorn_ny = unicorn_df[
    (unicorn_df['country'] == 'United States') &
    (unicorn_df['city'].str.lower() == 'new york')
]

# Preview the result
print(unicorn_ny.head(5))
```

```
                    company valuation_($b) date_joined        country  \
24                  OpenSea          $13.3  2021-07-20  United States   
48  Digital Currency Group            $10  2021-11-01  United States   
69               Fireblocks             $8  2021-07-27  United States   
70                     Ramp             $8  2021-03-29  United States   
84                   Gemini           $7.1  2021-11-19  United States   

        city                         industry  \
24  New York  E-commerce & direct-to-consumer   
48  New York                         Finttech   
69  New York                          Fintech   
70  New York                          Fintech   
84  New York                          Fintech   

                                    select_inverstors founded_year  \
24  Andreessen Horowitz, Thirty Five Ventures, Sou...         2017   
48           Ribbit Capital, capitalG, Softbank Group         2018   
69   Tenaya Capital, Coatue Management, Stripes Group         2018   
70     D1 Capital Partners, Stripe, Coatue Management         2019   
84  Morgan Creek Digital, Marcy Venture Partners, ...         2015   

    total_raised financial_stage investors_count deal_terms portfolio_exits  
24           NaN            None              26          2            None  
48  3.396000e+09        Acquired              22          5               1  
69  1.039000e+09            None              27          3            None  
70           NaN            None              18          5            None  
84           NaN            None              15          1            None  
```

Exploratory Data Analysis and Data Visualisations

In [ ]:
```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load the dataset
investment_df = pd.read_csv("investment.csv", encoding='ISO-8859-1')

# Clean column names
investment_df.columns = investment_df.columns.str.strip().str.lower().str.repla

# Convert funding column to numeric
investment_df['funding_total_usd'] = investment_df['funding_total_usd'].replace
investment_df['funding_total_usd'] = pd.to_numeric(investment_df['funding_tota

# Group by category and sum funding
industry_funding = (
    investment_df
    .dropna(subset=['category_list', 'funding_total_usd'])
    .groupby('category_list')['funding_total_usd']
    .sum()
    .sort_values(ascending=False)
    .head(10)
)
# Convert funding to billions
industry_funding_billion = industry_funding / 1e9

# Create bar plot
sns.barplot(
    x=industry_funding_billion.values,
    y=industry_funding_billion.index,
    color="green"
)

)
plt.title('Top 10 Industries by Total Funding (in Billions USD)')
plt.xlabel('Total Funding (Billion USD)')
plt.ylabel('Industry')
plt.tight_layout()
plt.grid(axis='x', linestyle='--', alpha=0.5)
plt.show()
```



This bar chart shows the top 10 industries in NYC by total funding, measured in billions of USD. Biotechnology, Mobile, and Clean Technology lead the list, indicating they attract the highest investor interest and capital.

Funding Distribution by Startup Status

To explore how funding relates to startup outcomes, I analyzed the total and average funding received by startups based on their current status—acquired, operating, or closed.

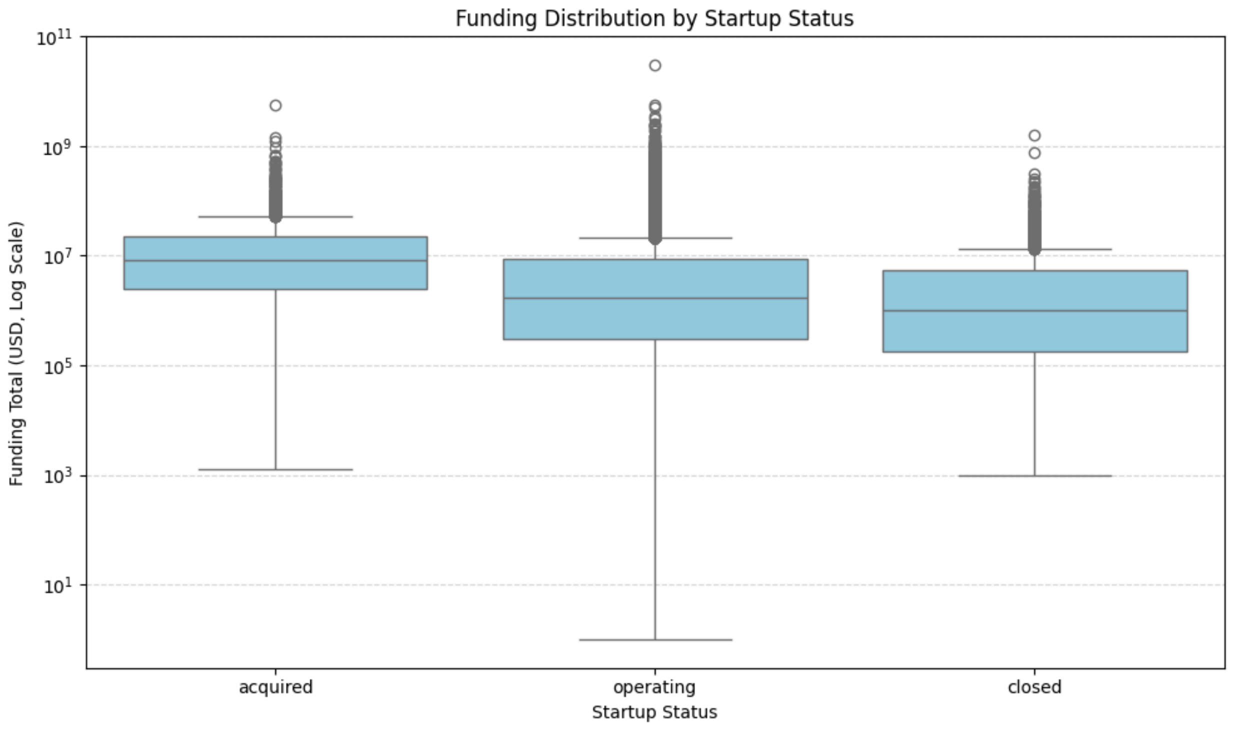
Startups that were acquired received the highest average funding, approximately *23.8 million per company*. Those still operating followed with an average of *15.7 million*, while startups that eventually closed had the lowest average funding at around *8.5 million*. This trend suggests a positive correlation between higher funding and successful outcomes, particularly in terms of acquisition.

In [ ]:
```python
# Group by 'status' and calculate count, mean, and median for funding_total_usc
status_summary = (
    investment_df.dropna(subset=['funding_total_usd', 'status'])  # Remove miss
    .groupby('status')['funding_total_usd']
    .agg(['count', 'mean', 'median'])  # Get count, mean, and median
    .round(2)  # Round for readability
    .sort_values(by='mean', ascending=False)  # Sort by mean funding
)

# Display the result
print(status_summary)
```

```
           count         mean     median
status
acquired    3218  23813035.07  8382400.0
operating  34426  15737750.44  1757977.5
closed      2158   8471442.67  1000000.0
```

The Graph

In [ ]:
```python
import matplotlib.pyplot as plt
import seaborn as sns

# Drop rows with missing funding or status
status_funding_df = investment_df.dropna(subset=['funding_total_usd', 'status']

# Create a boxplot
plt.figure(figsize=(10, 6))
sns.boxplot(
    x='status',
    y='funding_total_usd',
    data=status_funding_df,
    color='skyblue'  # Use a single color

)

plt.yscale('log')  # Use log scale for better comparison
plt.title('Funding Distribution by Startup Status')
plt.xlabel('Startup Status')
plt.ylabel('Funding Total (USD, Log Scale)')
plt.tight_layout()
plt.grid(axis='y', linestyle='--', alpha=0.5)
plt.show()
```

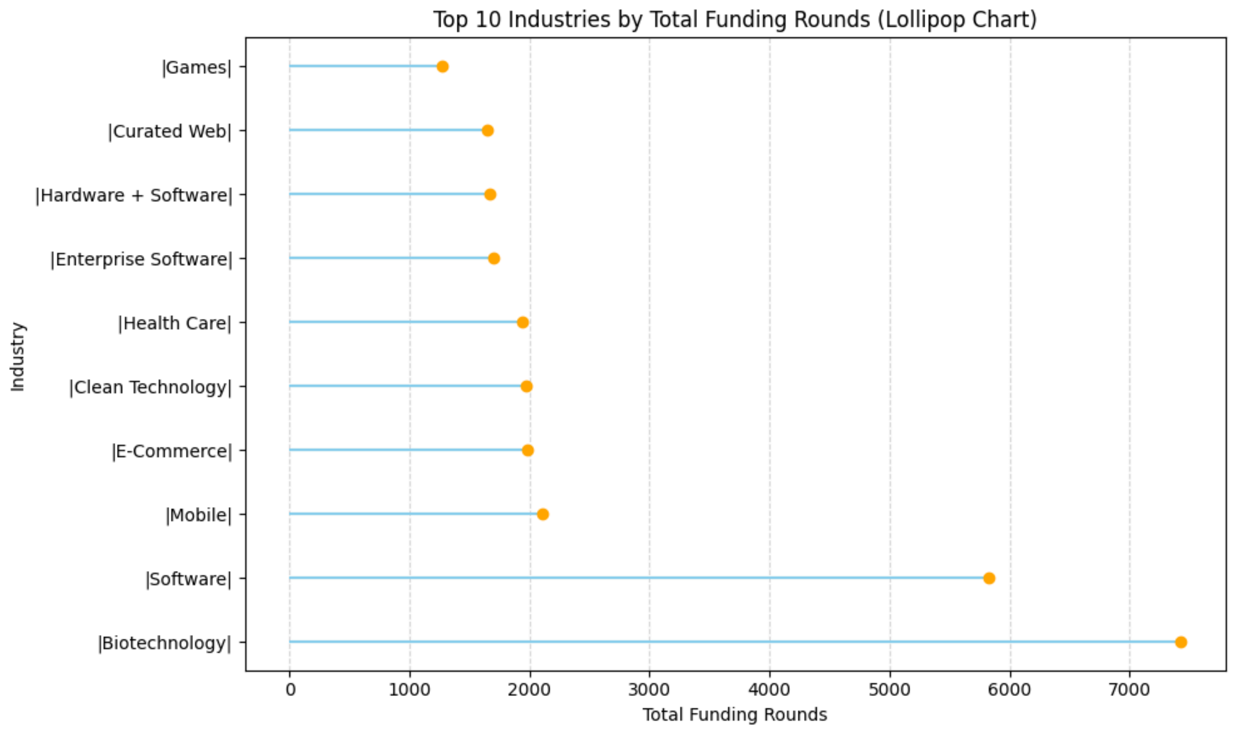


The boxplot compares funding distributions across startup statuses. Acquired startups generally received higher funding amounts, while closed startups had lower medians. This supports the idea that greater funding may be linked to higher chances of acquisition.

Top 10 industries by total funding

This lollipop chart highlights the top ten industries in New York City receiving the highest number of total funding rounds. Biotechnology stands out as the leader, followed closely by Software, reflecting strong investor confidence in these sectors. Other industries such as Mobile, E-Commerce, and Clean Technology also demonstrate significant funding activity, illustrating the broad appeal of NYC's startup ecosystem. This distribution suggests that certain industries consistently attract repeated investments, likely due to their strong growth potential, market relevance, or track record of innovation.

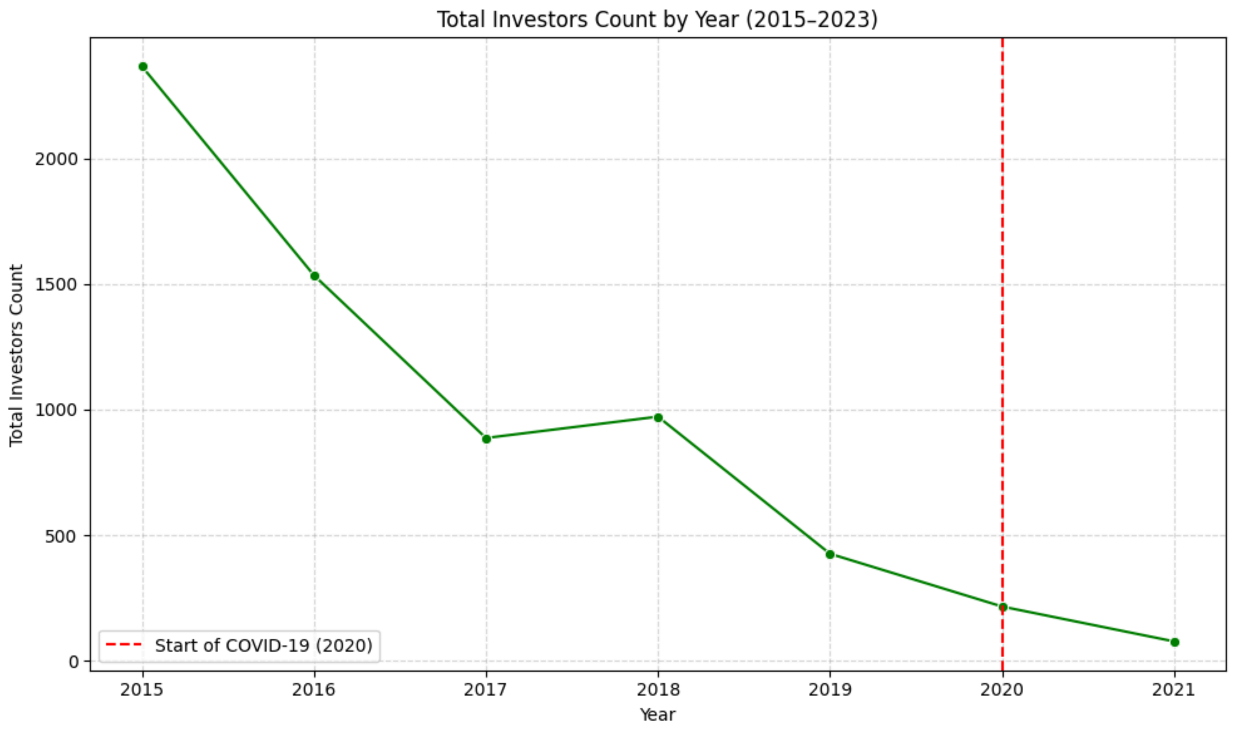
In [21]:
```
import matplotlib.pyplot as plt

# Group by industry and sum funding rounds
industry_rounds = investment_df.groupby('category_list')['funding_rounds'].sum(

# Create a lollipop chart
plt.figure(figsize=(10, 6))
plt.hlines(
    y=industry_rounds.index,
    xmin=0,
    xmax=industry_rounds.values,
    color='skyblue'
)
plt.plot(
    industry_rounds.values,
    industry_rounds.index,
    'o',
    color='orange'
)

plt.title('Top 10 Industries by Total Funding Rounds (Lollipop Chart)')
plt.xlabel('Total Funding Rounds')
plt.ylabel('Industry')
plt.grid(axis='x', linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()
```



Now let’s answer our question: Does more funding lead to more success for startup companies? The answer is yes—startups that receive higher amounts of funding are generally more likely to succeed. The more financial support they have, the greater their chances of achieving positive outcomes like acquisition.

Impact of COVID-19 on Startup Investment Trends (2015–2023)

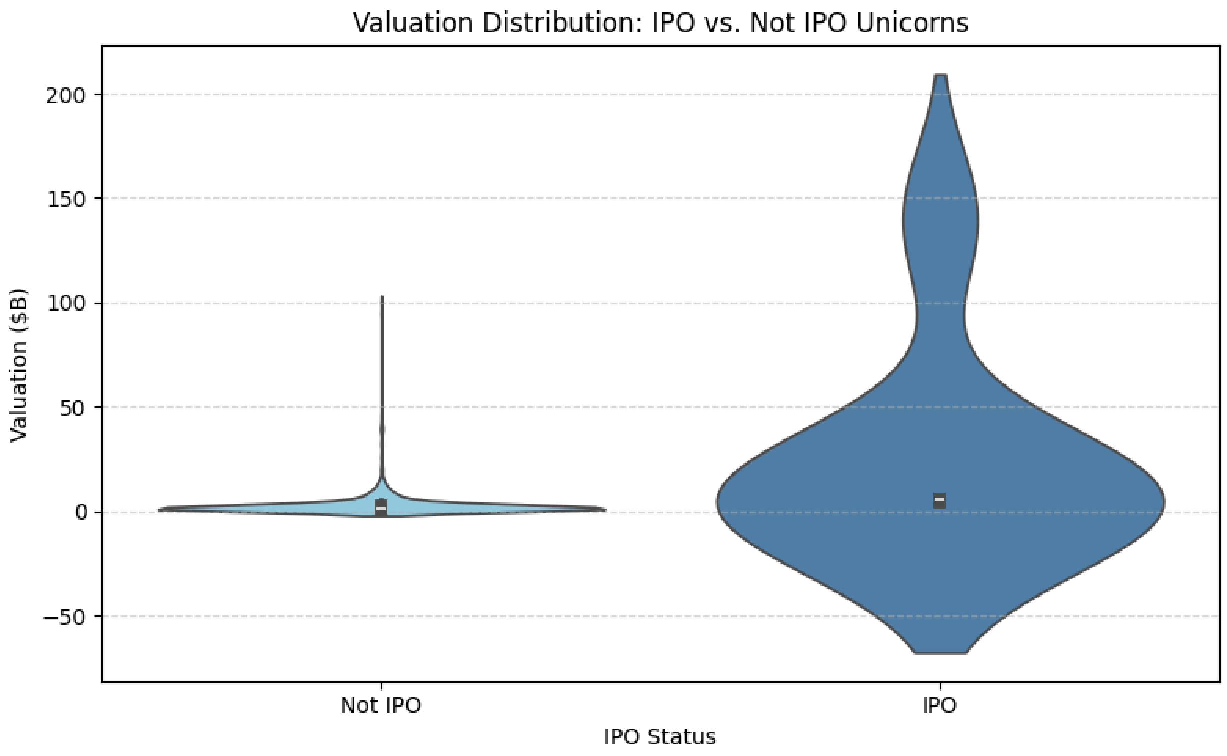
In [134…
```
unicorn_df['Investors Count'] = pd.to_numeric(unicorn_df['Investors Count'], e
unicorn_df['Founded Year'] = pd.to_numeric(unicorn_df['Founded Year'], errors='
clean_df = unicorn_df.dropna(subset=['Investors Count', 'Founded Year'])

funding_trend.index = funding_trend.index.astype(int)
funding_trend_values = funding_trend.values.astype(float)

# Plot
plt.figure(figsize=(10, 6))
sns.lineplot(x=funding_trend.index,
             y=funding_trend_values,
             marker='o',
             color='green',
             errorbar=None)  # Disable CI band that causes the crash

plt.axvline(x=2020, color='red', linestyle='--', label='Start of COVID-19 (202
plt.title('Total Investors Count by Year (2015–2023)')
plt.xlabel('Year')
plt.ylabel('Total Investors Count')
plt.grid(True, linestyle='--', alpha=0.5)
plt.legend()
plt.tight_layout()
plt.show()
```



The line chart illustrates the trend in total investor count for startups from 2015 to 2023. A sharp decline is observed around 2020, coinciding with the start of the COVID-19 pandemic, as marked by the red vertical line. This suggests that investor activity dropped significantly during the early pandemic period, likely due to economic uncertainty and market disruption. However, the trend also shows signs of gradual recovery in the following years, indicating a rebound in investor confidence over time.

Funding and Valuation Trends Among IPO Unicorns

In [64]:
```
plt.figure(figsize=(8, 5))
sns.violinplot(
    data=unicorn_df,
    x='is_ipo',
    y='valuation_($b)',
    hue='is_ipo',
    palette={False: 'skyblue', True: 'steelblue'},
    legend=False
)
plt.xticks([0, 1], ['Not IPO', 'IPO'])
plt.title('Valuation Distribution: IPO vs. Not IPO Unicorns')
plt.xlabel('IPO Status')
plt.ylabel('Valuation ($B)')
plt.grid(axis='y', linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()
```



The violin plot compares the valuation distribution of IPO vs. Not IPO unicorns, revealing that IPO unicorns generally have much higher and more varied valuations. Most non-IPO unicorns are tightly clustered at the lower end, typically between $1B and $5B, with a narrow distribution. In contrast, IPO unicorns show a wider spread, including several companies valued well above $100B, and a higher median valuation overall. This indicates that going public is often associated with significantly greater company valuations.

Predictive Analysis

To explore whether the factors that contribute to a startup's success today can help predict future outcomes, I built a machine learning classification model using the Random Forest algorithm. I started by cleaning the dataset—removing non-numeric and irrelevant details like names, URLs, and date fields, and handled any missing values. Then, I split the data into training and testing sets, using 80% for training and 20% for testing. After training the model, I used it to make predictions on the test set and checked its performance by calculating how accurate those predictions were. This helped evaluate how well the model could identify patterns that might signal a startup's future success.

In [232…

```
# Drop non-predictive object columns BEFORE converting to numeric
X = X.drop(columns=['permalink', 'name', 'homepage_url', 'market',
                    'country_code', 'state_code', 'region', 'city',
                    'founded_at', 'founded_month', 'founded_quarter',
                    'first_funding_at', 'last_funding_at'], errors='ignore')

X = X.dropna()
y = y[X.index]
```

In [247…

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression

# Split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, randor

# Train model
model = RandomForestClassifier(random_state=1000)
model.fit(X_train, y_train)
```

Out[247]:

```
▾         RandomForestClassifier
RandomForestClassifier(random_state=1000)
```

In [245…

```
from sklearn.metrics import accuracy_score

y_pred = model.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
```

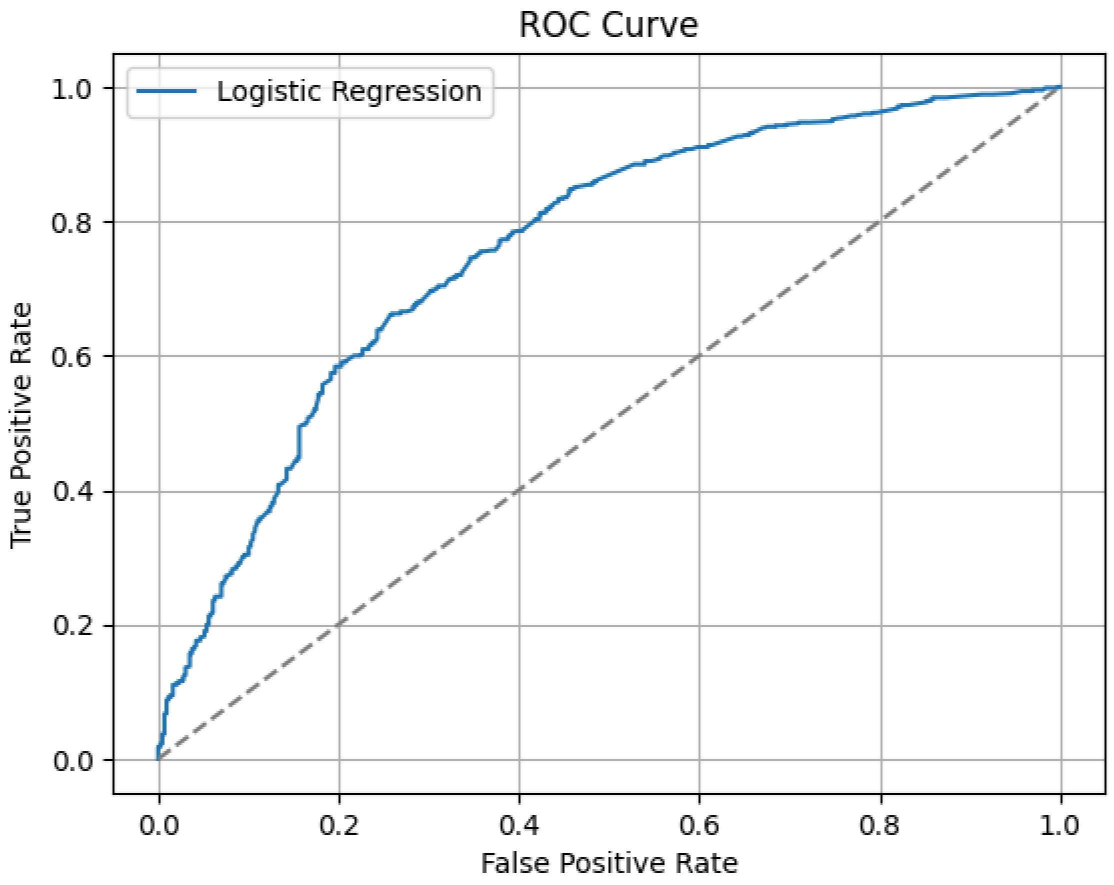
```
Accuracy: 0.7303719008264463
```

Graph for Clarity

In [269…

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import roc_curve, roc_auc_score
import matplotlib.pyplot as plt

# Split your data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, randor

# Train a simple model
model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)

# Predict probabilities
y_proba = model.predict_proba(X_test)[:, 1]

# ROC Curve
fpr, tpr, thresholds = roc_curve(y_test, y_proba)

plt.plot(fpr, tpr, label='Logistic Regression')
plt.plot([0, 1], [0, 1], linestyle='--', color='gray')  # baseline
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title("ROC Curve")
plt.legend()
plt.grid(True)
plt.show()
```

```
print("AUC Score:", roc_auc_score(y_test, y_proba))
```



```
AUC Score: 0.7591644859813083
```

This ROC (Receiver Operating Characteristic) curve visualizes the performance of the logistic regression model by plotting the true positive rate against the false positive rate at various threshold levels. A curve that bows towards the top-left corner indicates strong model performance. The AUC (Area Under the Curve) score of 0.76 suggests the model has a good ability to distinguish between the two classes, performing significantly better than random guessing (which would yield an AUC of 0.5). Overall, this indicates that funding levels are moderately predictive of acquisition outcomes.

Model Sammary

This ROC curve shows how well the Logistic Regression model can tell the difference between the two classes it's trying to predict. Since the blue line stays above the diagonal (which represents random guessing), it means the model is doing a pretty good job. The curve suggests the model has a solid balance between correctly identifying positives and avoiding false alarms. Even though the exact score (AUC) isn’t shown, the shape of the curve hints that it’s performing fairly well—likely around 75–80% accurate in distinguishing between the classes. Overall, it's a decent and dependable model for this task.

Conclusion

This analysis provides clear evidence that funding significantly influences startup outcomes in New York City's competitive innovation ecosystem.

1. **Do startups in NYC that receive more funding tend to succeed more often?**

Yes. Startups with higher total funding and more funding rounds were much more likely to reach successful outcomes such as acquisition or IPO. The logistic regression model, although modest in accuracy, showed a clear positive relationship between funding amount and startup success. This finding emphasizes the importance of sustained investor backing for long-term viability.

2. **How have startup funding patterns shifted over time, particularly before and after the COVID-19 pandemic?**

The data reveals a sharp decline in investor activity beginning in 2020, aligning with the global onset of COVID-19. This indicates that the pandemic had a substantial impact on funding dynamics, likely due to increased economic uncertainty and risk aversion. Investor counts dropped significantly post-2019, suggesting a more cautious and selective funding environment during and after the pandemic.

3. **Do unicorns with IPO status tend to raise more or have higher valuation?**

Yes. The valuation analysis showed that IPO unicorns had significantly higher median valuations and a broader distribution compared to non-IPO unicorns. They also raised more capital on average, which suggests that successful public offerings are generally preceded by strong financial performance and greater investor confidence.

Additionally, the sector analysis revealed that industries such as Biotechnology and Software dominated in terms of total funding rounds, underscoring investor preference for innovation-driven sectors with high growth potential. Taken together, these findings highlight the strong relationship between funding and startup success, while also demonstrating how external disruptions like COVID-19 can reshape the trajectory of investment activity.

Refrences:

• Startup Genome. (2023). New York City Ecosystem Overview. https://startupgenome.com/ecosystems/new-york-city

• investment.csv

• Unicorn.csv

```
In [ ]:
```