MASSCHALLENGE

ML MODELS FOR UNSTRUCTURED DATA

PRESENTATION BY:

- Neha Save
- Yinliang Lyu
- Tseng Yu Lu
- Yuzhou Zheng
- Susheel Reddy
- Catherine Dommaty
- Akshay Damodar Prabhu

ABOUT THE COMPANY

MassChallenge, a global non-profit startup accelerator headquartered in Boston, supports early-stage startups (generally with less than $1 million in annual revenue and funding) with a focus on traditionally underrepresented founders. They provide free resources, including physical facilities, technical resources, structured programming, mentorship, networking opportunities, and equity-free funding.

Headquartered in Boston, MassChallenge is a non-profit global startup accelerator with operations throughout the US, Switzerland, Mexico, Israel, as well as a smaller footprint in Asia & Africa.

They support early stage startups (generally, less than $1m in annual revenue and having raised less than $1m in funding), with a particular focus on traditionally underrepresented founders.

MassChallenge aims to create a massive impact on the global startup ecosystem by empowering entrepreneurs to succeed. By providing startups with the necessary resources and a strong support network, MassChallenge helps foster innovation, economic growth, and job creation worldwide. Their vision is to build a world where anyone can access the resources they need to launch and grow a successful startup, regardless of their background or location.

DATASET

The dataset comprises information on 51,088 startups, detailing their industry, location, and online presence.

Key attributes include StartupID, Organization ID, Startup Name, Primary Industry Name, short and full elevator pitches, LinkedIn, Facebook, and website URLs, Twitter handle, Stealth Mode status, creation and founding dates, location details, and founder information.

The dataset also includes URLs to accelerator profiles and primary contact information.

This comprehensive dataset provides valuable insights into the startup ecosystem, enabling detailed analysis of trends, industry distribution, and the online footprint of emerging businesses.

TECHNICAL GUIDANCE:

No limitations on technologies or methodologies used, though preference is given to tooling which aligns with internal usage & skillsets. Preferential tooling includes:

- Python
- Jupyter Notebooks
- Can be deployed in AWS
- Containerization is nice but not required (docker, etc.)

Irrespective of tooling, solutions should be documented and all assets in a Github repository.

GOAL

Goal: Predict which industry or industries a startup is part of based on unstructured or semi-structured data. Training dataset will be provided; data includes:

·Startup short pitch: A few sentences describing the purpose/goal/mission/offering of the company
·Startup full pitch: A more detailed dive into the ‘short pitch’
·Links to third party company assets, such as their LinkedIn page, social media page(s), video pitches, Business Propositions [detailed answers to specific questions about their company- not available for all companies]
·Founders with biographical / educational background

Stretch goal:
·Create a model that does the same for individuals (data will be provided if we meet our goal with time remaining in the semester)

BUSINESS PROBLEM

One of the significant challenges MassChallenge faces is the need for accurate industry classification of startups.

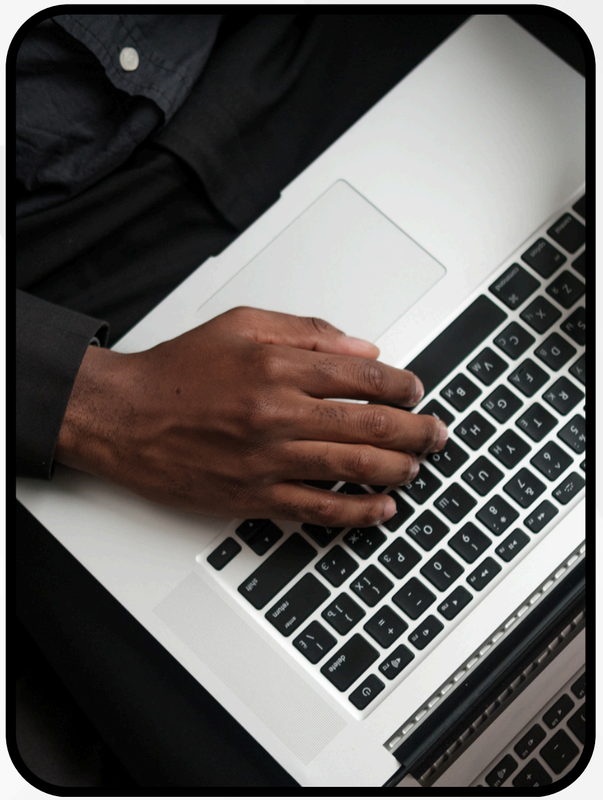
This classification is crucial for effectively matching startups with mentors, curating lists for partner collaboration and investment, and creating targeted content marketing and thought leadership materials.

The current process relies heavily on manual classification, where staff members interpret and categorize startups based on unstructured data such as pitches, social media profiles, and founder backgrounds.

This manual method is not only time-consuming and labor-intensive but also prone to inconsistencies and human error.

The inefficiency and potential inaccuracies in the current classification process hinder MassChallenge's ability to provide optimal support and resources to startups.

Inconsistent classifications can lead to mismatched mentor assignments, less effective partner collaborations, and suboptimal marketing efforts, ultimately affecting the growth and success of the startups within the accelerator program.



OUR RECENT
STATEMENT

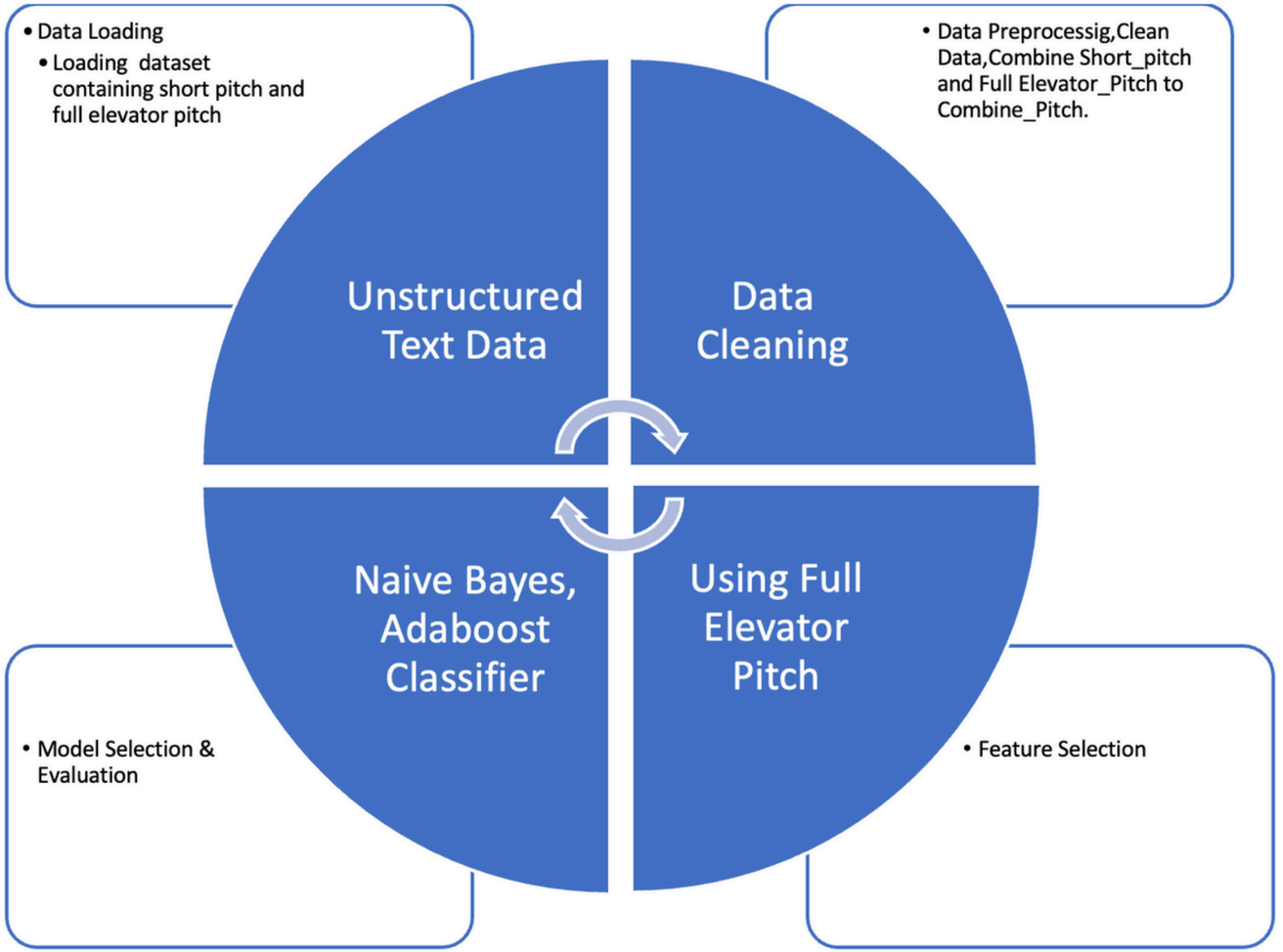
Impact: Implementing this machine learning model will streamline the industry classification process, allowing MassChallenge staff to:

- Efficiently match startups with mentors who have relevant industry expertise.
- Curate accurate lists for partners looking to collaborate with or invest in startups within specific industries.
- Enhance content marketing and thought leadership efforts by accurately categorizing startups, leading to more targeted and impactful communication.

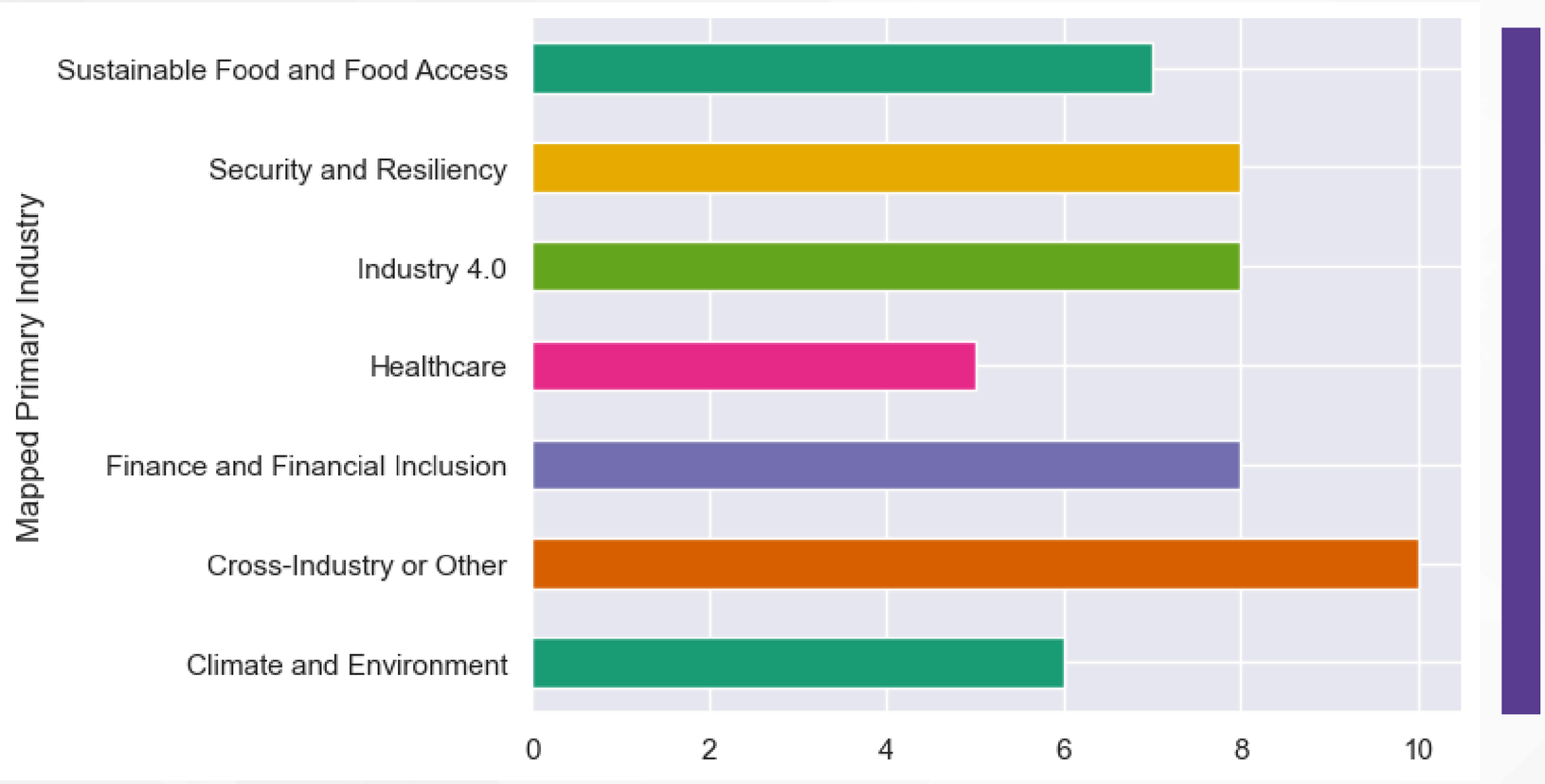
Ultimately, this model will improve the support and resources provided to startups, fostering innovation and growth within the global startup ecosystem supported by MassChallenge.

Flow Chart:

Data Processing and Model Training for Predicting Secondary Industry Classification



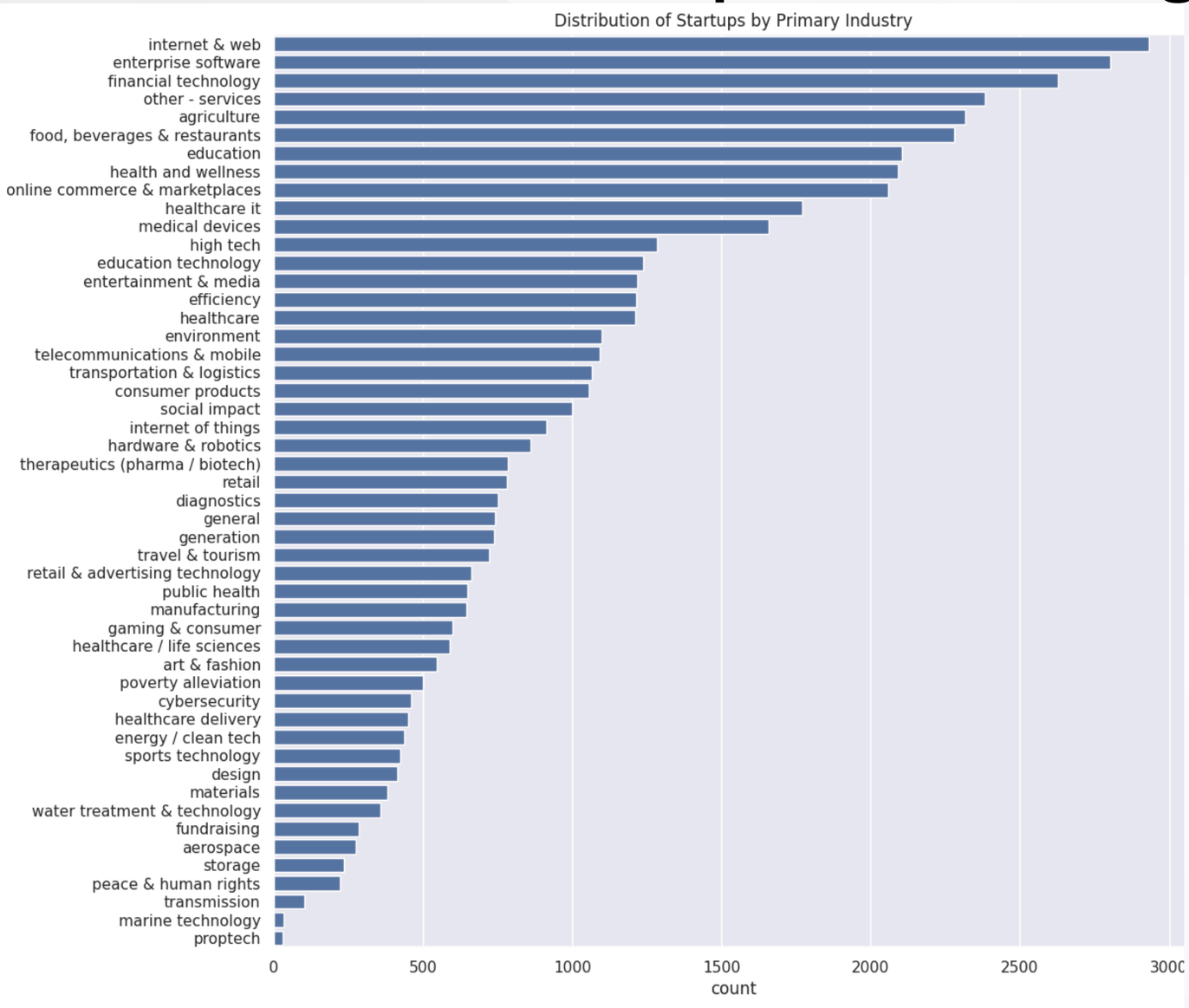
Mapping Primary to Secondary Industries:



The output displays a data frame with three columns: 'Primary Industry', 'Secondary Industry (this is what we want to predict)', and 'Mapped Primary Industry'. The data frame contains 61 rows with various industries listed. Some entries show 'NaN' (Not a Number) values, indicating missing data in certain cells. The mapping appears to successfully categorize several secondary industries under broader primary industry categories, such as 'Healthcare' and 'Cross-Industry or Other'.

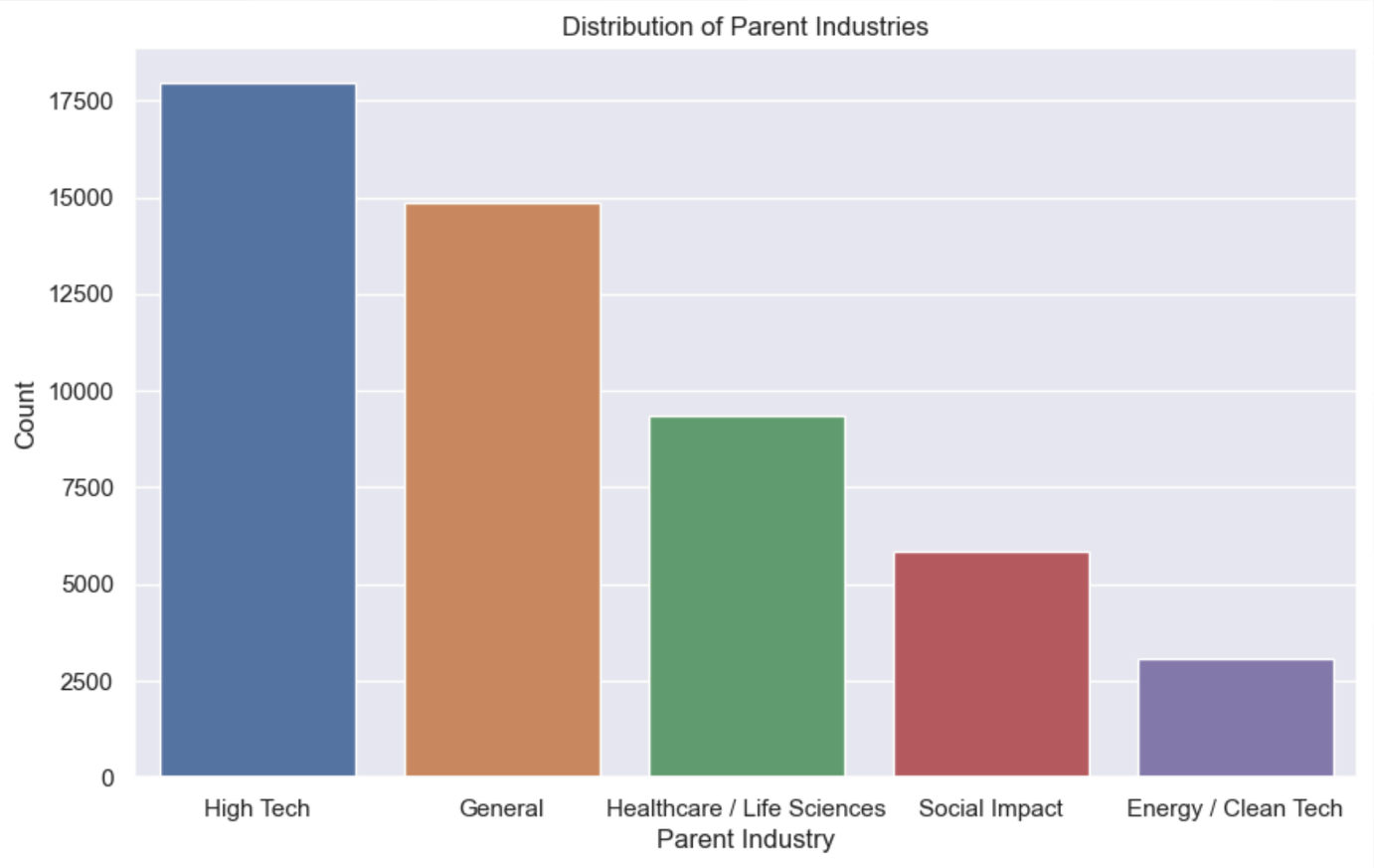
This bar chart displays the count of startups mapped to various primary industries. The Cross-Industry or Other category has the highest count, followed by Finance and Financial Inclusion, and Security and Resiliency.

Distribution of startups across a range of primary industries



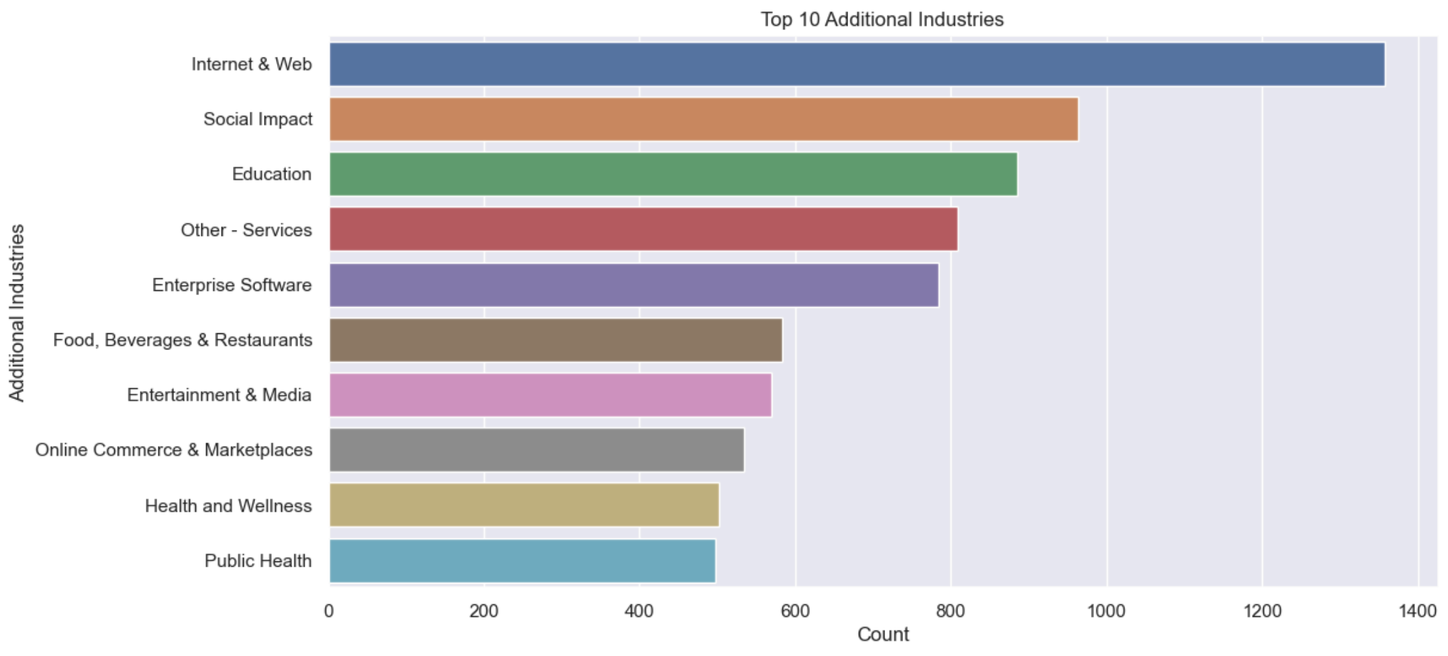
This horizontal bar chart shows the distribution of startups across a range of primary industries. Internet & Web and Enterprise Software are the most common industries, while Proptech is the least represented.

Distribution of startups across different parent industries



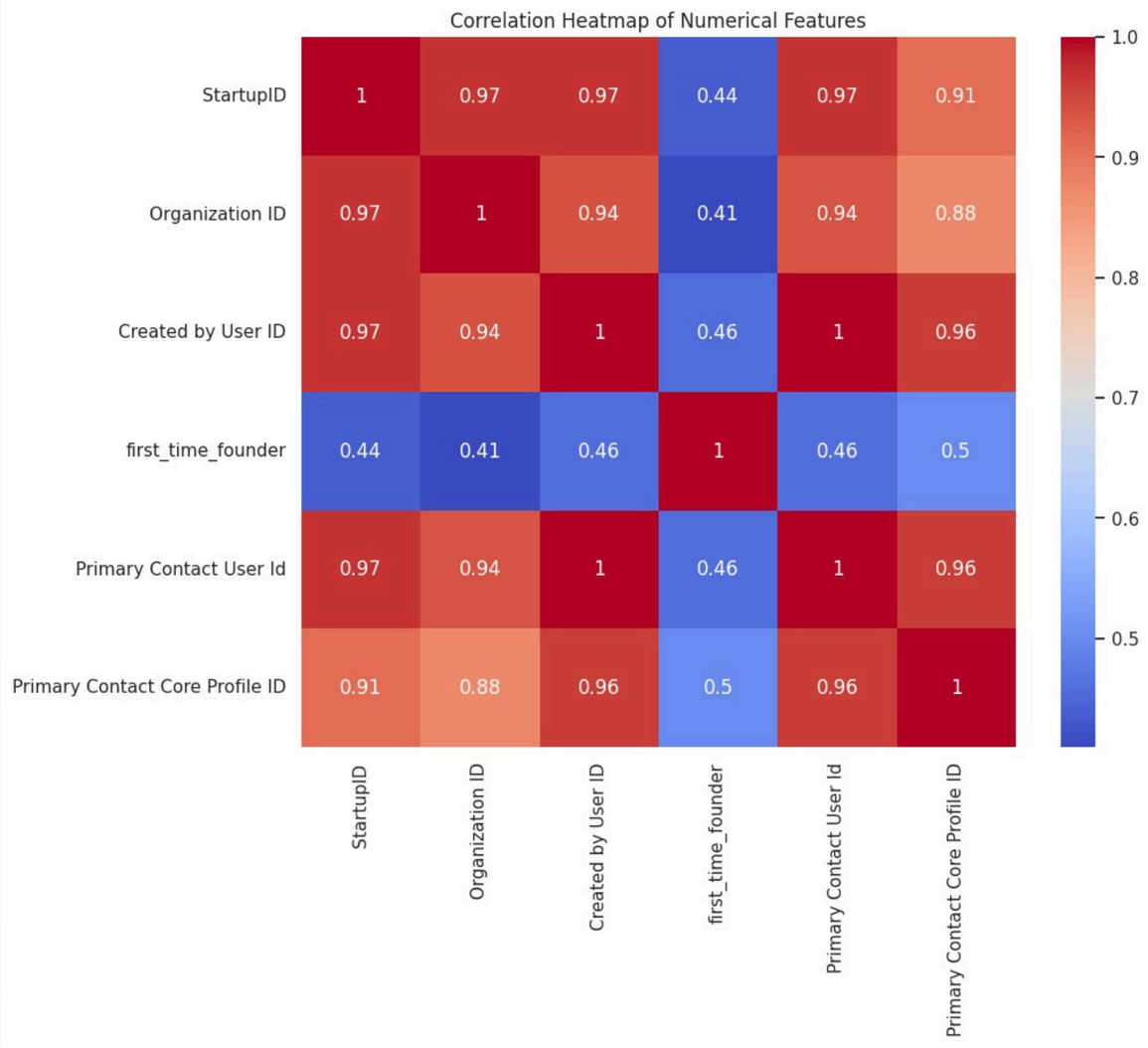
This bar chart represents the distribution of startups across different parent industries. High Tech has the largest count, followed by General, Healthcare/Life Sciences, Social Impact, and Energy/Clean Tech.

Illustrates the top 10 additional industries for startups



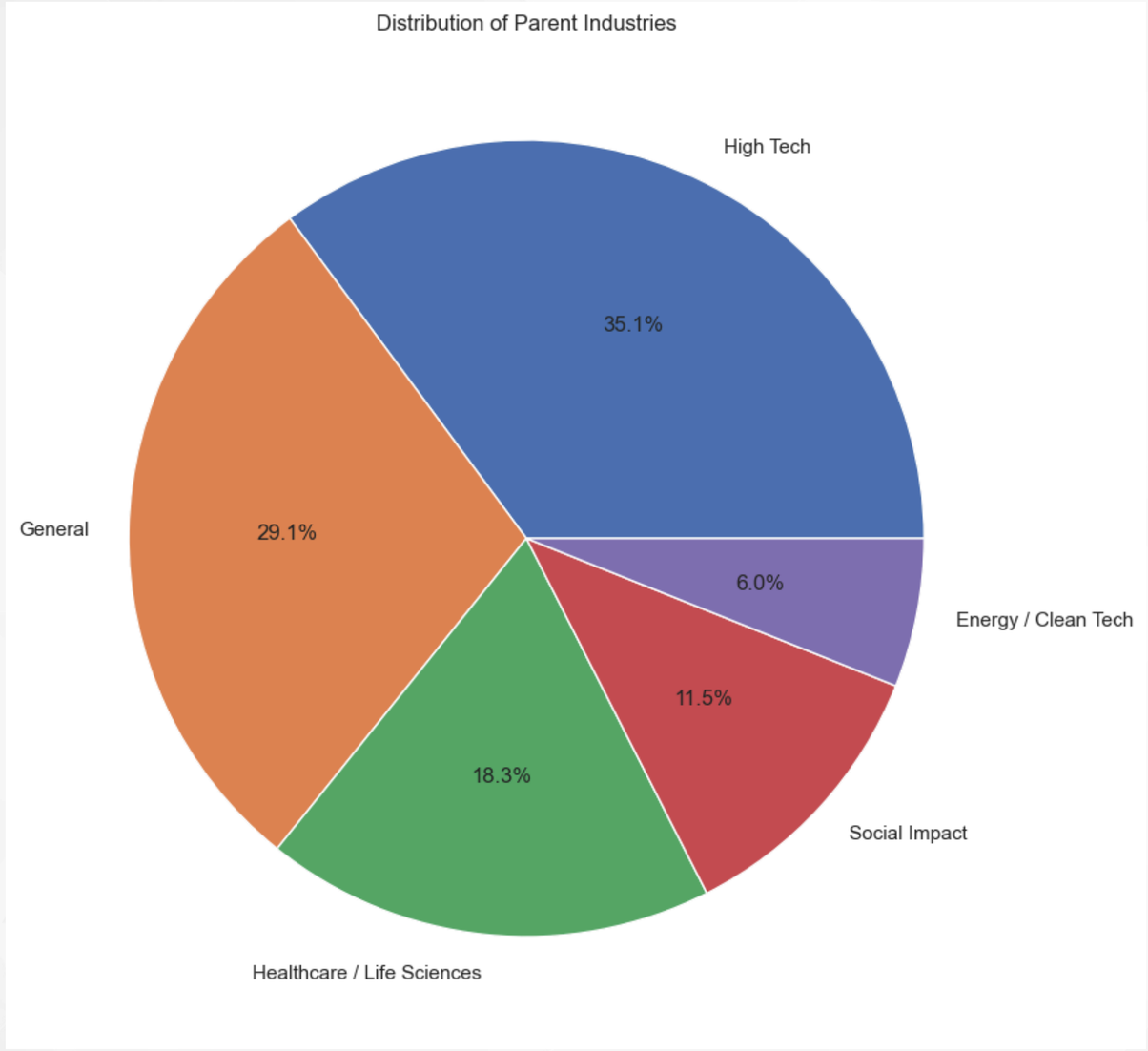
This horizontal bar chart illustrates the top 10 additional industries for startups. Internet & Web leads with the highest count, followed by Social Impact and Education.

Heatmap of correlation



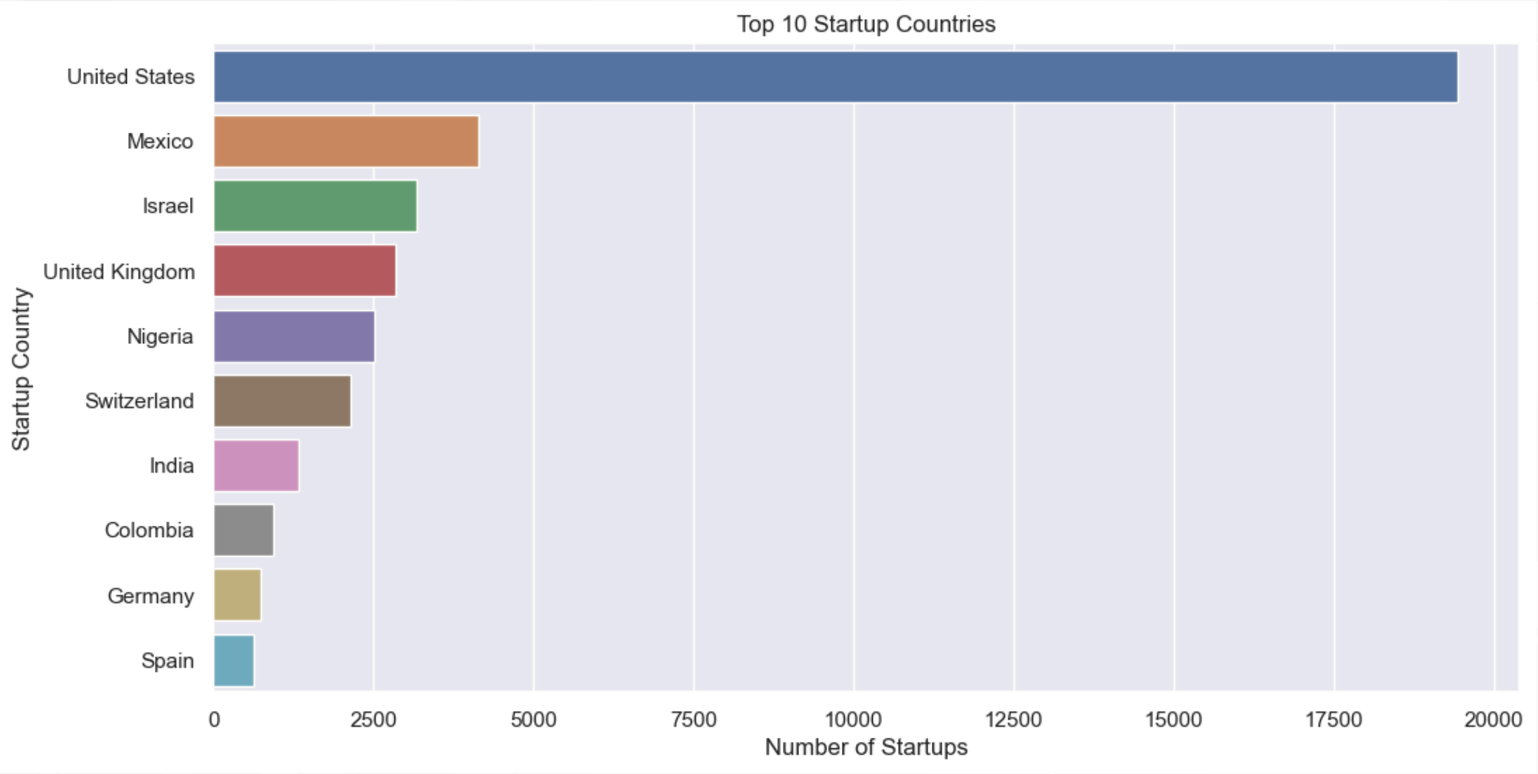
The heatmap shows the correlation between various numerical features related to startups. High correlations are visible among StartupID, Organization ID, and related features, indicating strong associations.or sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore

Percentage distribution of startups across parent industries



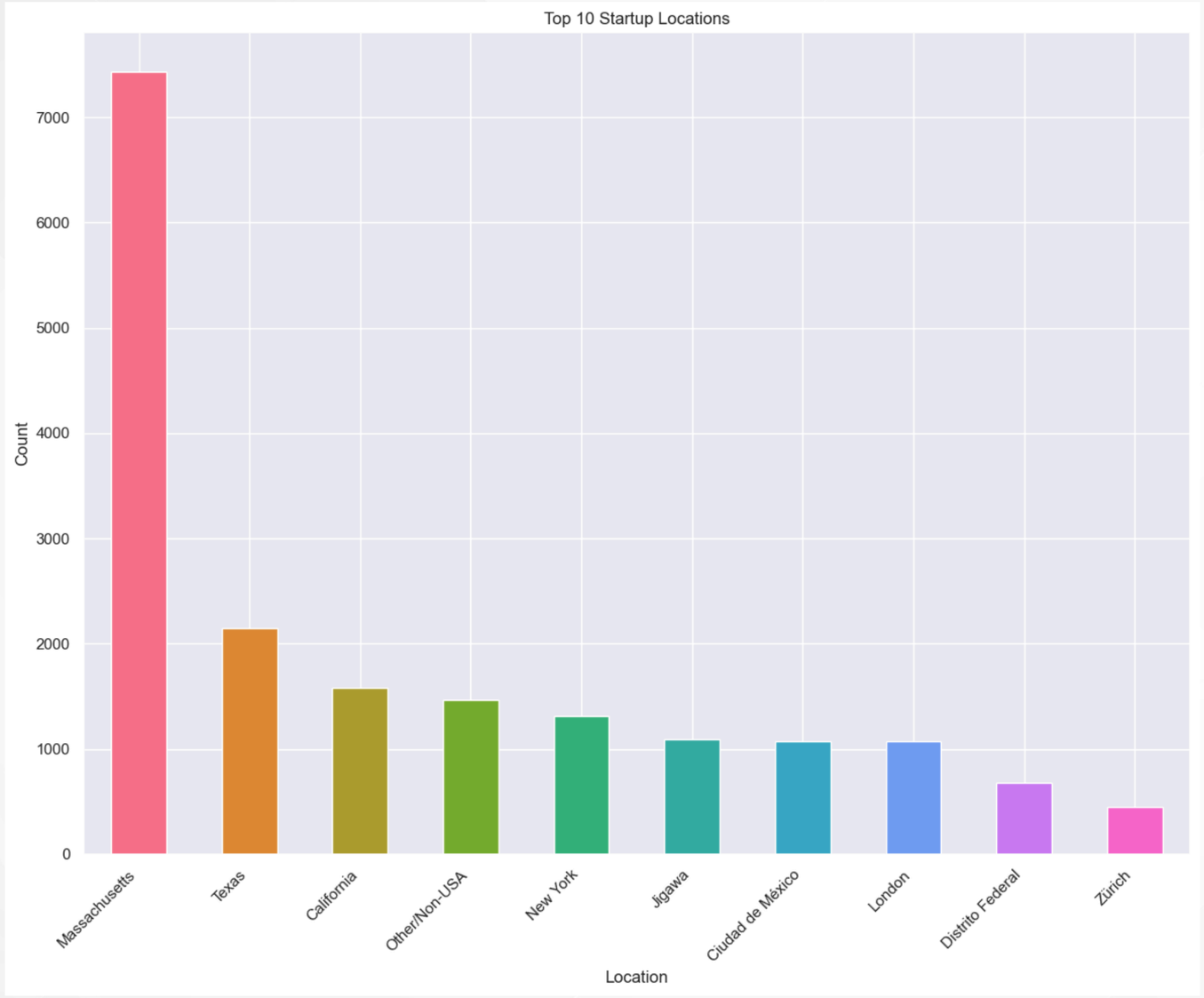
This pie chart displays the percentage distribution of startups across parent industries. High Tech holds the largest share, followed by General, Healthcare/Life Sciences, Social Impact, and Energy/Clean Tech.

Number of startups across the top 10 startup countries



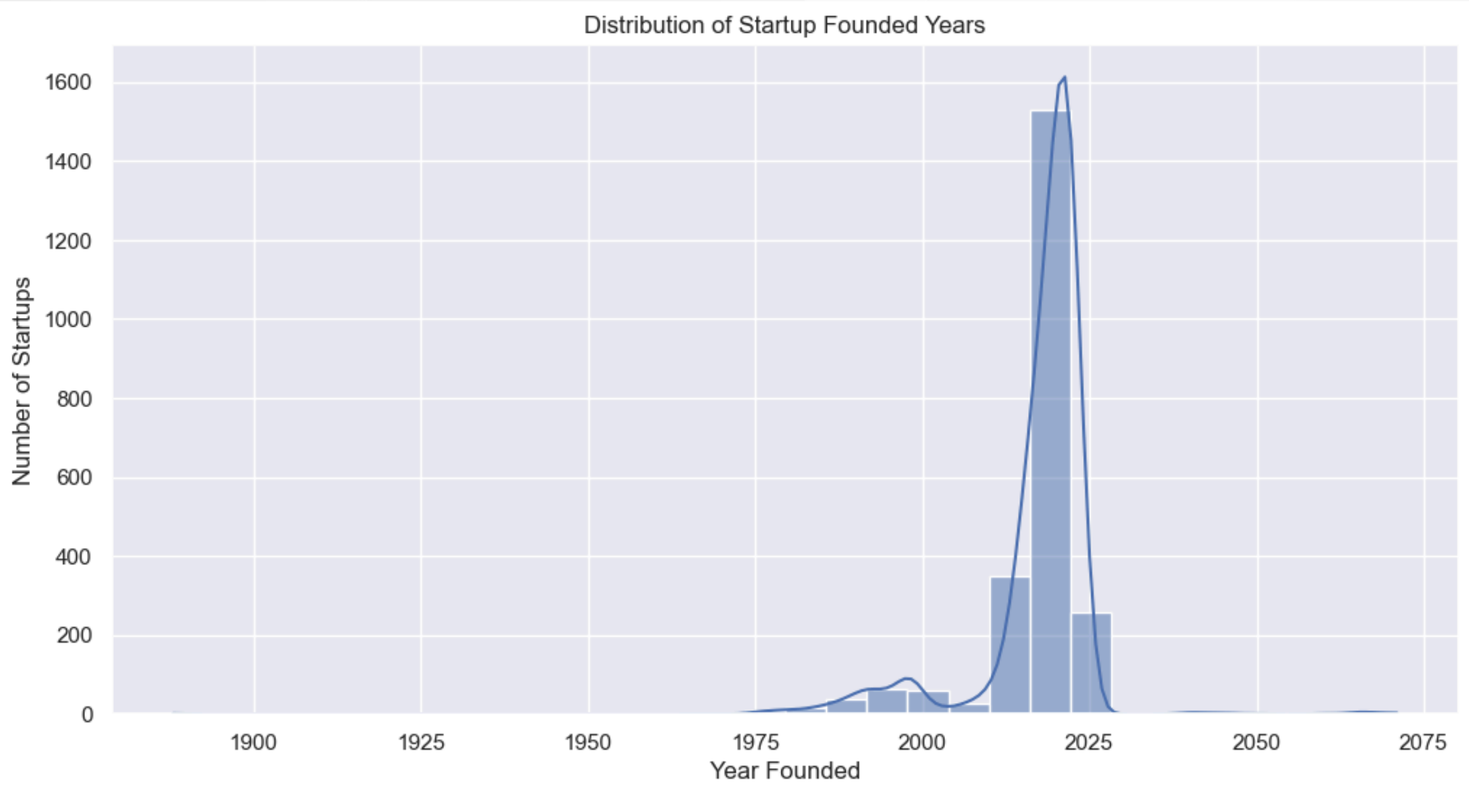
the bar chart shows the number of startups across the top 10 startup countries. The United States has the highest count, significantly leading over Mexico, Israel, and other countries.

Top 10 startup locations by count



This bar chart represents the top 10 startup locations by count. Massachusetts is the leading location, with Texas, California, and New York also having substantial counts.

Distribution of startup founding years



The histogram shows the distribution of startup founding years, with a significant peak around the 2000s. This indicates a surge in startup formations during that period.

In [28]:

```python
# Clean text function
def clean_text(text):
    if pd.isnull(text):
        return ""
    text = text.lower()  # Lowercase text
    text = ''.join([char for char in text if char not in string.punctuation])  # Remove punctuation
    tokens = word_tokenize(text)  # Tokenize text
    tokens = [word for word in tokens if word not in stopwords.words('english')]  # Remove stopwords
    return ' '.join(tokens)

# Apply clean_text function to the relevant columns
df_company['clean_short_pitch'] = df_company['short_pitch'].apply(clean_text)
df_company['clean_full_pitch'] = df_company['full_elevator_pitch'].apply(clean_text)

# Handle missing values
df_company.fillna('', inplace=True)
```

After defining the function, it is applied to specific columns in a data frame called df_company. The code cleans the 'short_pitch' and 'full_elevator_pitch' columns, creating new columns with the cleaned text.

Finally, it fills any missing values in the data frame with empty strings.

This code appears to be part of a data preprocessing pipeline, for natural language processing or text analysis tasks on company descriptions or pitches.

In [29]:

```
# Vectorize the text data using TF-IDF
vectorizer = TfidfVectorizer(stop_words='english', max_features=1000)
X = vectorizer.fit_transform(df_company['clean_full_pitch'])
y = df_company['Additional Industries']


# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Print the shapes of the new splits
print("X_train shape:", X_train.shape)
print("X_val shape:", X_test.shape)
print("y_train shape:", y_train.shape)
print("y_val shape:",y_test.shape)
```

```
X_train shape: (40870, 1000)
X_val shape: (10218, 1000)
y_train shape: (40870,)
y_val shape: (10218,)
```

This code snippet shows the process of preparing text data for machine learning. It uses TF-IDF (Term Frequency-Inverse Document Frequency) vectorization to convert the 'clean_full_pitch' text from the df_company DataFrame into numerical features.

The vectorizer is set to use English stopwords and create a maximum of 1000 features. The target variable y is set as the 'Additional Industries' column.

The data is then split into training and testing sets using train_test_split, with 20% of the data reserved for testing and a random state of 42 for reproducibility.

KNN Model

```
In [30]: from sklearn.neighbors import KNeighborsClassifier
         from sklearn.metrics import accuracy_score, classification_report
         from sklearn.preprocessing import LabelEncoder
         # Initialize and train the KNN model
         knn_classifier = KNeighborsClassifier(n_neighbors=5)
         knn_classifier.fit(X_train, y_train)

         # Predict and evaluate
         y_pred = knn_classifier.predict(X_test)
         accuracy = accuracy_score(y_test, y_pred)
         report = classification_report(y_test, y_pred)

         print(f'K-Nearest Neighbors Classifier')
         print(f'Accuracy: {accuracy}')
         print(f'Classification Report:\n{report}')
```

```
K-Nearest Neighbors Classifier
Accuracy: 0.3096496378939127
Classification Report:

precision    recall  f1-score   support


0.33      0.94      0.48      3317

Aerospace       0.00      0.00      0.00        11

Aerospace; Agriculture       0.00      0.00      0.00         1

Aerospace; Agriculture; Efficiency; Energy / Clean Tech; Hardware & Robotics       0.00      0.00      0.00         1

Aerospace; Agriculture; Efficiency; Generation; Manufacturing       0.00      0.00      0.00         1

Aerospace; Agriculture; Environment; Financial Technology; Gaming & Consumer       0.00      0.00      0.00         1
```

This output shows the performance of a K-Nearest Neighbors (KNN) classifier on the test data. The overall accuracy of the model is about 31%, which is relatively low, indicating that the model struggles to correctly classify most samples.

The classification report provides more detailed metrics, but only one class is shown (likely due to truncation). For this class, the precision is 0.33, meaning that when the model predicts this class, it is correct 33% of the time.

The recall is high at 0.94, suggesting the model correctly identifies 94% of the actual instances of this class. The F1-score, which is the harmonic mean of precision and recall, is 0.48.

Naive Bayes Model

In [31]:
```
# Vectorize the text data using TF-IDF
vectorizer = TfidfVectorizer(stop_words='english', max_features=1000)
X = vectorizer.fit_transform(df_company['clean_full_pitch'])
y = df_company['Additional Industries']

# Label Encoding the target variable
label_encoder = LabelEncoder()
y_encoded = label_encoder.fit_transform(y)

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y_encoded, test_size=0.2, random_state=42)

# Print the shapes of the new splits
print("X_train shape:", X_train.shape)
print("X_test shape:", X_test.shape)
print("y_train shape:", y_train.shape)
print("y_test shape:", y_test.shape)

# Initialize and train the Naive Bayes model
nb_classifier = MultinomialNB()
nb_classifier.fit(X_train, y_train)

# Predict and evaluate
y_pred = nb_classifier.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)

# Ensure target_names matches the labels in y_test
unique_labels = np.unique(y_test)
target_names = label_encoder.inverse_transform(unique_labels)

report = classification_report(y_test, y_pred, target_names=target_names, zero_division=1)

print(f'Naive Bayes Classifier')
print(f'Accuracy: {accuracy}')
print(f'Classification Report:\n{report}')
```

```
Naive Bayes Classifier
Accuracy: 0.32481894695635155
Classification Report:

precision    recall  f1-score   support


0.33      1.00      0.49      3317
```

The Naive Bayes classifier applied to this dataset achieves an accuracy of approximately 32.48%. This indicates that the model correctly predicts the 'Additional Industries' category for about one-third of the samples in the test set.

The classification report shows metrics for one class (likely due to truncation). For this class, the precision is 0.33, meaning 33% of the predictions for this class are correct.

The recall is 1.00, indicating the model identifies all actual instances of this class. The F1-score, balancing precision, and recall, is 0.49. The support of 3317 suggests this many samples belong to this class in the test set.

AdaBoost Model

```
X = vectorizer.fit_transform(df_company['clean_full_pitch'])
y = df_company['Additional Industries']

# Encode the labels
label_encoder = LabelEncoder()
y = label_encoder.fit_transform(y)

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Print the shapes of the new splits
print("X_train shape:", X_train.shape)
print("X_test shape:", X_test.shape)
print("y_train shape:", y_train.shape)
print("y_test shape:", y_test.shape)


# AdaBoost
ada_model = AdaBoostClassifier(n_estimators=100, random_state=42)
ada_model.fit(X_train, y_train)
ada_predictions = ada_model.predict(X_test)
ada_accuracy = accuracy_score(y_test, ada_predictions)
print(f'AdaBoost Accuracy: {ada_accuracy}')
print(classification_report(y_test, ada_predictions))
```

```
X_train shape: (40870, 1000)
X_test shape: (10218, 1000)
y_train shape: (40870,)
y_test shape: (10218,)
AdaBoost Accuracy: 0.32423174789587006
              precision    recall  f1-score   support

           0       0.32      1.00      0.49      3317
```

- The AdaBoost model achieved an accuracy of 32.42%.
- High recall (1.00) but low precision (0.32) indicates the model is highly sensitive but not very specific.
- The weighted average F1-score of 0.16 suggests overall model performance needs improvement.

CONCLUSION

This project aimed to predict the secondary industry classification for startups using unstructured data provided by MassChallenge, an organization supporting high-impact startups.

The task involved two datasets: one containing detailed startup information and another outlining primary to secondary industry mappings.

The startup data included attributes such as location, short and full pitches, additional industries, and founding year, presenting challenges due to its unstructured nature.

The industry mapping data provided essential relationships between primary and secondary industries, further complicating the task.

We began with extensive exploratory data analysis (EDA) to understand the distributions, patterns, and anomalies within the datasets.

This analysis highlighted significant variation in industry distribution and revealed trends in startup founding years, locations, and social media presence.

Following the EDA, we undertook substantial data cleaning and preprocessing. Missing values were handled by filling in placeholders, and text data was standardized and cleaned to improve consistency. This involved converting text to lowercase, removing punctuation, tokenizing text, and eliminating stopwords.

THANK YOU