Project 1: Predicting Startup Funding Success- A Machine Learning Approach

A copy of this notebook has also been uploaded to https://github.com/rozank/dsc680_applied_datascience/tree/main/Project-1

In [1]:
```
# import libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.graph_objects as go
import plotly.express as px
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, precision_score, f1_score, confu
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
from sklearn.tree import DecisionTreeClassifier
from sklearn.preprocessing import OrdinalEncoder
from sklearn.metrics import confusion_matrix

import re

# suppress all warnings
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
# loading dataset
df = pd.read_csv("investments.csv", encoding="ISO-8859-1")
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 54294 entries, 0 to 54293
Data columns (total 39 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   permalink             49438 non-null  object 
 1   name                  49437 non-null  object 
 2   homepage_url          45989 non-null  object 
 3   category_list         45477 non-null  object 
 4    market               45470 non-null  object 
 5    funding_total_usd    49438 non-null  object 
 6   status                48124 non-null  object 
 7   country_code          44165 non-null  object 
 8   state_code            30161 non-null  object 
 9   region                44165 non-null  object 
 10  city                  43322 non-null  object 
 11  funding_rounds        49438 non-null  float64
 12  founded_at            38554 non-null  object 
 13  founded_month         38482 non-null  object 
 14  founded_quarter       38482 non-null  object 
 15  founded_year          38482 non-null  float64
 16  first_funding_at      49438 non-null  object 
 17  last_funding_at       49438 non-null  object 
 18  seed                  49438 non-null  float64
 19  venture               49438 non-null  float64
 20  equity_crowdfunding   49438 non-null  float64
 21  undisclosed           49438 non-null  float64
 22  convertible_note      49438 non-null  float64
 23  debt_financing        49438 non-null  float64
 24  angel                 49438 non-null  float64
 25  grant                 49438 non-null  float64
 26  private_equity        49438 non-null  float64
 27  post_ipo_equity       49438 non-null  float64
 28  post_ipo_debt         49438 non-null  float64
 29  secondary_market      49438 non-null  float64
 30  product_crowdfunding  49438 non-null  float64
 31  round_A               49438 non-null  float64
 32  round_B               49438 non-null  float64
 33  round_C               49438 non-null  float64
 34  round_D               49438 non-null  float64
 35  round_E               49438 non-null  float64
 36  round_F               49438 non-null  float64
 37  round_G               49438 non-null  float64
 38  round_H               49438 non-null  float64
dtypes: float64(23), object(16)
memory usage: 16.2+ MB
```

In [3]:

```python
df.isnull().sum().sort_values(ascending=False)
```

Out[3]:
```
state_code              24133
founded_month           15812
founded_year            15812
founded_quarter         15812
founded_at              15740
city                    10972
country_code            10129
region                  10129
 market                  8824
category_list            8817
homepage_url             8305
status                   6170
name                     4857
post_ipo_debt            4856
secondary_market         4856
product_crowdfunding     4856
round_A                  4856
round_B                  4856
permalink                4856
round_C                  4856
round_D                  4856
round_E                  4856
private_equity           4856
round_F                  4856
round_G                  4856
post_ipo_equity          4856
venture                  4856
grant                    4856
angel                    4856
debt_financing           4856
convertible_note         4856
undisclosed              4856
equity_crowdfunding      4856
seed                     4856
last_funding_at          4856
first_funding_at         4856
funding_rounds           4856
 funding_total_usd       4856
round_H                  4856
dtype: int64
```

Data Preparation

Data Cleaning

These data cleaning steps address issues like leading/trailing spaces, formatting inconsistencies, and incorrect datatypes, ensuring the data is in a suitable format for further analysis.

- Removing Duplicate Data: By removing duplicate data, we can ensure that each observation in the dataset is unique, avoiding redundancy and potential biases in subsequent analysis or modeling tasks.

- Remove Null Values: By deleting rows with missing values in important columns, its ensured that the remaining data is more complete and suitable for analysis or modeling tasks.Removing rows with null values helps avoid biases or inaccuracies that could arise from imputing missing data or including incomplete observations in the analysis.

- Renaming Columns: As we can see there are lots of column names with leading and trailing spaces and Leading and they can cause issues while accessing or referencing the columns. Removing the spaces ensures easier and error-free handling of the data.

- Cleaning Funding Data: Some of the funding columns are initially read as an object (string) datatype. So removing any non numeric characters other than decimal point and replacing all the

- Cleaning Date Data: The columns 'founded_at', 'first_funding_at', 'last_funding_at', 'founded_year', and 'founded_month' are converted to datetime format using specified date format.

- Removing Outliers: The IQR (interquartile range) method is chosen over the Z-score method for the startup success dataset due to its suitability for handling the dataset's non-normal distribution and the presence of outliers. Startups experiencing exceptional success or failure can have extreme values that deviate significantly from the central distribution, and the IQR method is more resistant to such outliers as it relies on quartiles rather than mean and standard deviation. By considering the range between quartiles, the IQR method captures the spread of the central distribution, making it more meaningful for identifying outliers in the context of startup success analysis.

While the IQR itself covers 50% of the data, it is not typically used directly to identify outliers. Instead, the IQR is commonly employed in conjunction with a boxplot to identify potential outliers.

- Dropping Irrelevant Data Dropping irrelevant data is important for data preparation because it helps streamline and focus the analysis on the most relevant information.

In [4]:
```
#before cleaning
print(df.shape)
df.head()
```

```
(54294, 39)
```

Out[4]:

	permalink	name	homepage_url	
0	/organization/waywire	#waywire	http://www.waywire.com	|Entertainment|Po
1	/organization/tv-communications	&TV Communications	http://enjoyandtv.com	
2	/organization/rock-your-paper	'Rock' Your Paper	http://www.rockyourpaper.org	
3	/organization/in-touch-network	(In)Touch Network	http://www.InTouchNetwork.com	|Electronics|Guides|Coff
4	/organization/r-ranch-and-mine	-R- Ranch and Mine	NaN	|Touris

5 rows × 39 columns

In [5]:

```
df.describe()
```

Out[5]:

	funding_rounds	founded_year	seed	venture	equity_crowdfunding
count	49438.000000	38482.000000	4.943800e+04	4.943800e+04	4.943800e+04
mean	1.696205	2007.359129	2.173215e+05	7.501051e+06	6.163322e+03
std	1.294213	7.579203	1.056985e+06	2.847112e+07	1.999048e+05
min	1.000000	1902.000000	0.000000e+00	0.000000e+00	0.000000e+00
25%	1.000000	2006.000000	0.000000e+00	0.000000e+00	0.000000e+00
50%	1.000000	2010.000000	0.000000e+00	0.000000e+00	0.000000e+00
75%	2.000000	2012.000000	2.500000e+04	5.000000e+06	0.000000e+00
max	18.000000	2014.000000	1.300000e+08	2.351000e+09	2.500000e+07

8 rows × 23 columns

In [6]:

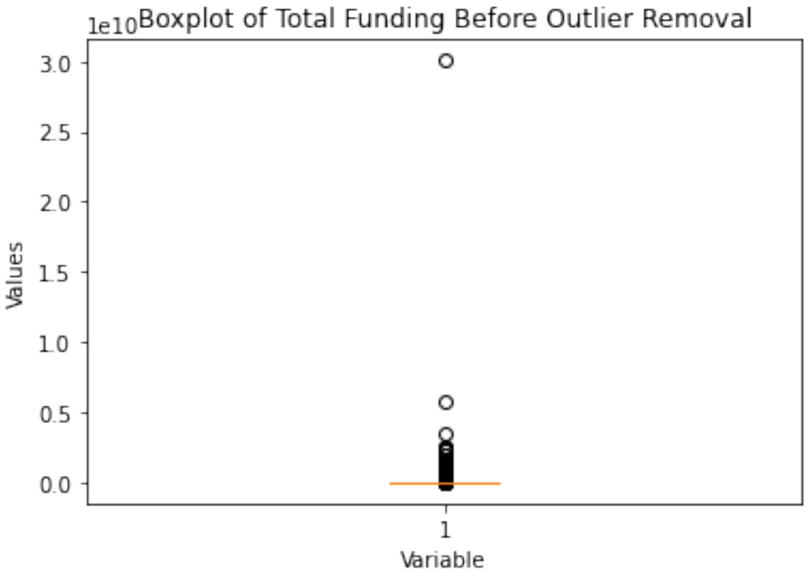
```
# setting to display all columns
pd.set_option('display.max_columns', None)
```

In [7]:
```
#Data Cleaning Functions
def clean_data(df):

    # remove leading and trailing spaces from column names
    df.columns=df.columns.str.strip()

    # remove leading and trailing spaces from row values in all columns
    df=df.applymap(lambda x: x.strip() if isinstance(x, str) else x)

    #deleting duplicate rows.
    df=df.drop_duplicates()

    #To address the high number of null values in the dataset, rows with nul
    #removed in an attempt to reduce the overall amount of missing data.
    df=df.dropna(subset=['status', 'founded_month', 'founded_year', 'market'

    return df


def clean_numeric_column_values(df, column_name):
    # remove non-numeric characters other than the decimal point from the sp
    df[column_name] = df[column_name].replace('[^\d.]', '', regex=True)

    # convert the column to numeric type and replace NaN values with 0
    df[column_name] = pd.to_numeric(df[column_name], errors='coerce').fillna
    return df

def convert_date_columns(df, date_columns, date_format='%Y-%m-%d'):
    # convert each column to datetime format using the specified date format
    for column in date_columns:
        df[column] = pd.to_datetime(df[column], format=date_format, errors='
    return df
```

In [8]:
```
# specify the date columns to be converted
y_m_d_date_columns=['founded_at', 'first_funding_at', 'last_funding_at']
m_date_columns=['founded_month']
y_date_columns=['founded_year']

# convert date columns using the function
df=convert_date_columns(df, y_m_d_date_columns,'%Y-%m-%d')
df=convert_date_columns(df, m_date_columns,'%Y-%m')
df=convert_date_columns(df, y_date_columns,'%Y')
```

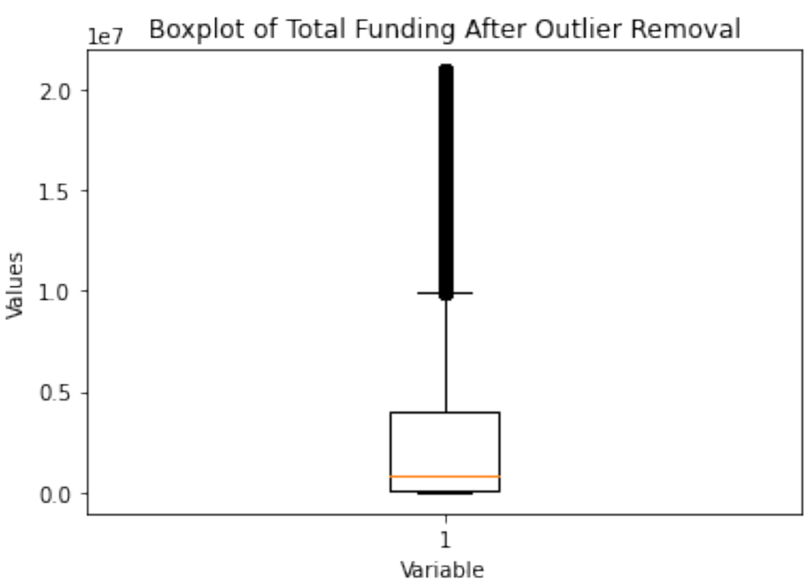
In [9]:
```
df=clean_data(df)
df=clean_numeric_column_values(df, 'funding_total_usd')
```

In [10]:
```
plt.boxplot(df['funding_total_usd'])
plt.xlabel('Variable')
plt.ylabel('Values')
plt.title('Boxplot of Total Funding Before Outlier Removal')
plt.show()
```



In [11]:
```
# Removing outliers from Funding amount using the IQR (Interquartile Range)

Q1 = df['funding_total_usd'].quantile(0.25)  # Calculate the first quartile
Q3 = df['funding_total_usd'].quantile(0.75)  # Calculate the third quartile
IQR = Q3 - Q1  # Calculate the interquartile range

# Define the lower and upper bounds for outlier detection
lower = (Q1 - 1.5 * IQR)
upper = (Q3 + 1.5 * IQR)

# Filter the dataset to include data points within the defined bounds
df = df[(df['funding_total_usd'] >= lower) & (df['funding_total_usd'] <= upp
```

In [12]:
```
plt.boxplot(df['funding_total_usd'])
plt.xlabel('Variable')
plt.ylabel('Values')
plt.title('Boxplot of Total Funding After Outlier Removal')
plt.show()
```



Explanation:

Even thought not completely removed, the before and after box plot shows a significant removal of outliers using IQR.

Exploratory Data Analysis (EDA)

In [13]:
```
# The code below calculates the age of a company at the time of its first fu
# its founding/start date and the first funding date.

df['age_first_funding'] = (df['first_funding_at'] - df['founded_at']).dt.day
mean_first_funding=df['age_first_funding'].mean()

print(f"Explanation:\nThe average duration between a company started and fun
```

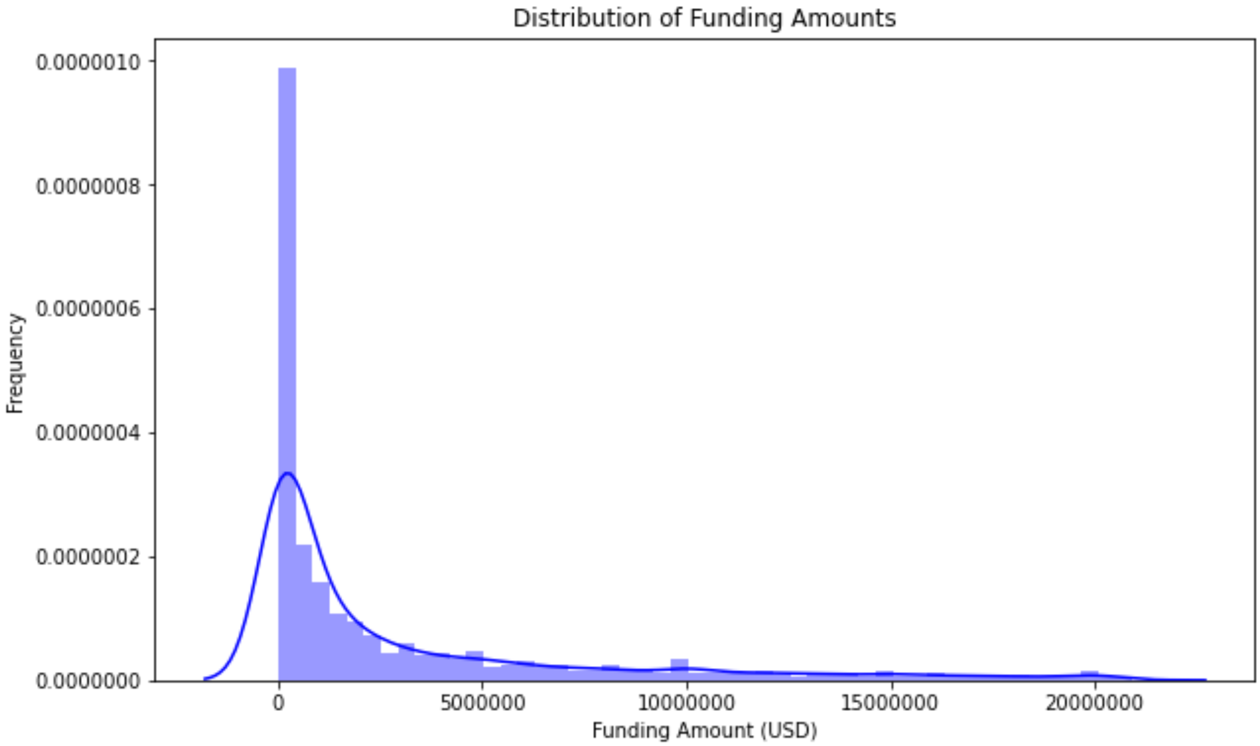
```
Explanation:
The average duration between a company started and funded is 40.57 months.
```

Creating a new column called 'funding_status' by examining the 'funding_total_usd' column. If the 'funding_total_usd' value is 0 or less, it indicates that the startup was not funded.

In [14]:
```
# Create a new column 'funding_status' based on 'total_funding_usd'
df['funding_status'] = np.where(df['funding_total_usd'] == 0, 'notfunded', '
counts=df.funding_status.value_counts()

print(f"Explanation:\nCounts of startups based on their funding status\n{cou
```

```
Explanation:
Counts of startups based on their funding status
funded       23726
notfunded     4860
Name: funding_status, dtype: int64.
```

In [15]:
```
# Plot the distribution of funding amounts
#sns.set_palette('viridis')
plt.figure(figsize=(10, 6))
sns.distplot(df['funding_total_usd'], color='blue')

# Add comments and labels
plt.title('Distribution of Funding Amounts')
plt.xlabel('Funding Amount (USD)')
plt.ylabel('Frequency')
plt.ticklabel_format(style='plain', axis='both')
# Display the plot
plt.show()
```



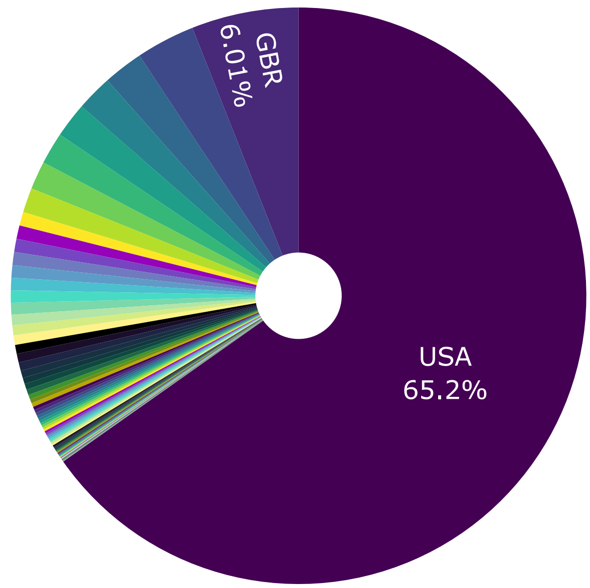
Explanation:

The distribution of funding amounts shows that most startups receive funding between $1 million - 10$ million. There are a small number of startups that receive funding amounts of over $100 million. This suggests that a large number of startups are funded at a relatively small amount, and a small number of startups are funded at a substantial amount.

In [16]:

```
fig = px.pie(df, names='country_code',
             title="Startup by Countries",
             color_discrete_sequence=px.colors.sequential.Viridis,
             hole=0.15,
             width=700,
             height=450).update_layout(uniformtext_minsize=10, uniformtext_m

fig.show()
```

Startup by Countries

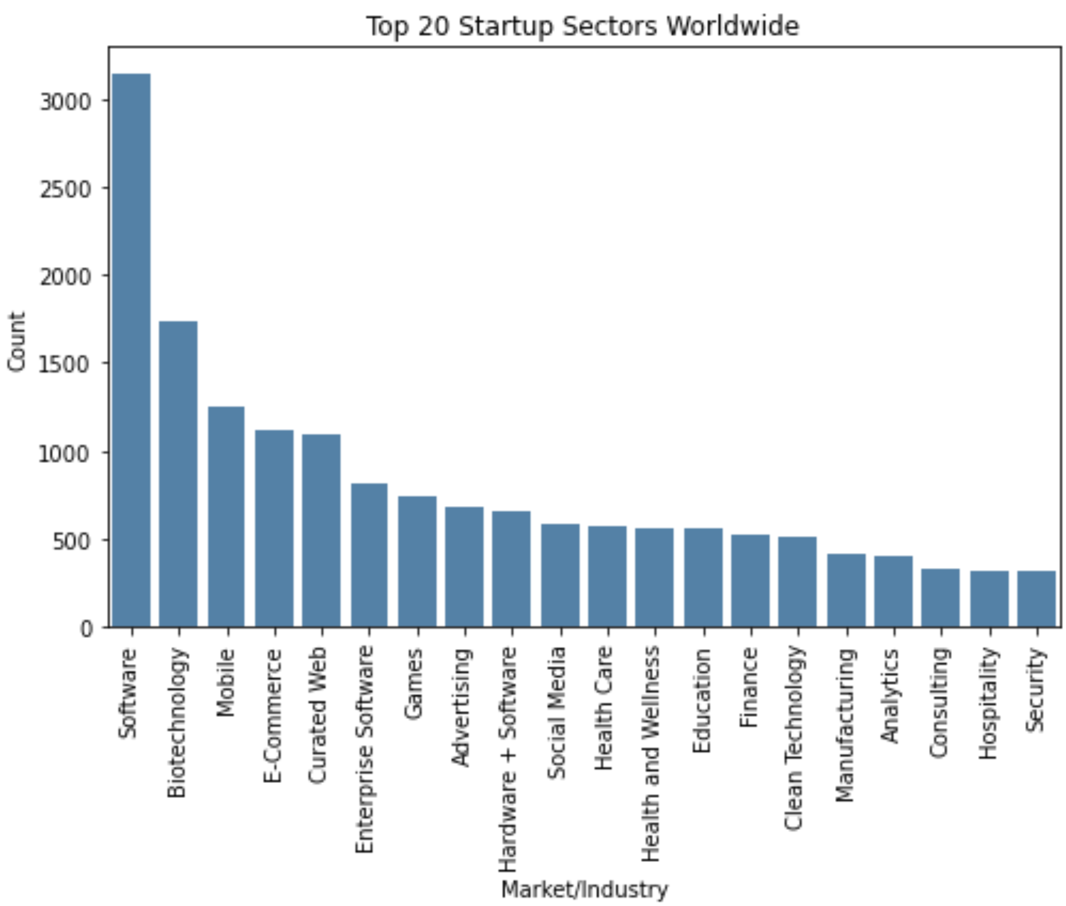


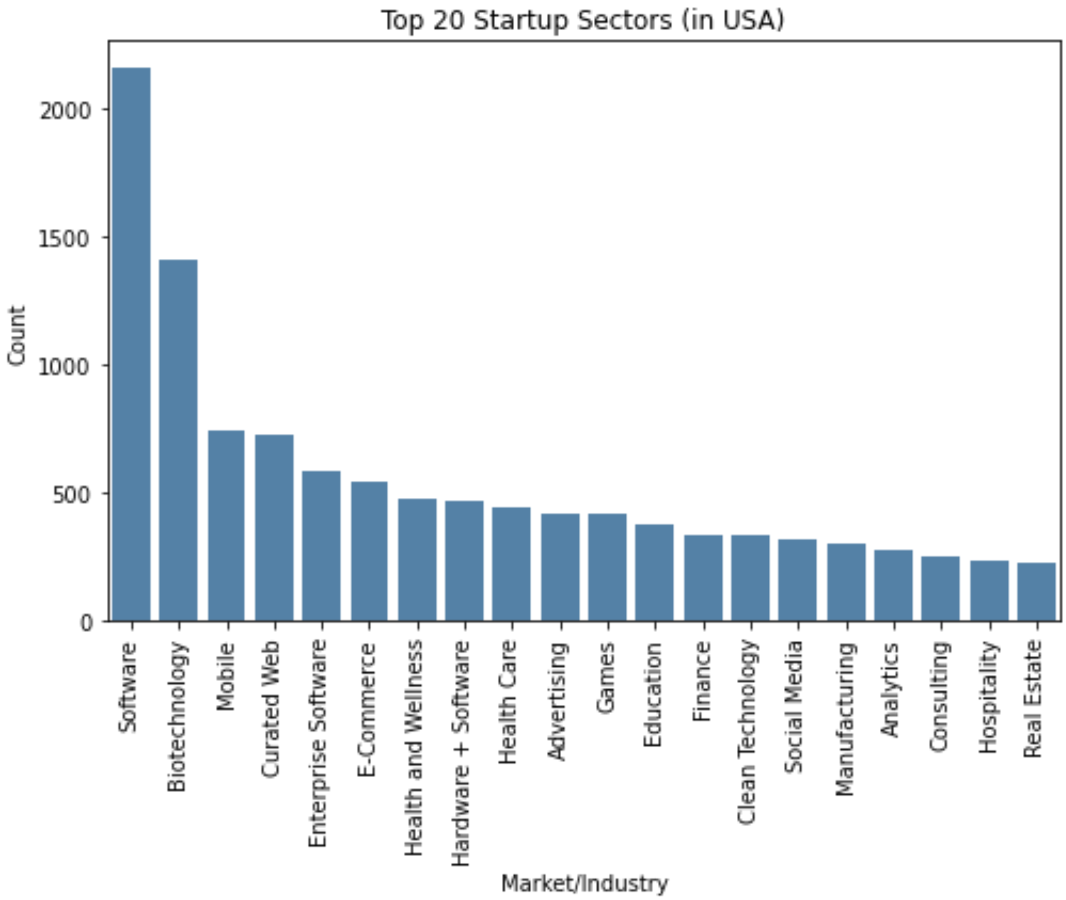
Explanation:

The pie chart depicts startup distribution across countries. The United States dominates with 65.2%, followed by the United Kingdom (6.01%), Canada (3.31%), Germany (2.47%), France (2.07%), India (1.81%), and Israel (1.55%).

The United States leads in startups due to factors like capital availability, a talented workforce, and favorable regulations. Other countries like the United Kingdom, Canada, Germany, France, India, and Israel also have significant startup presence. Success factors for startups may vary across countries but often include capital availability, a skilled workforce, and supportive regulations.

In [17]:
```
def top_20_startup_industry(df, country_code=None):
    plt.figure(figsize=(8, 5))

    if country_code:
        filtered_df = df[df['country_code'] == country_code]
        top_20_industries = filtered_df['market'].value_counts().nlargest(20
        df_top_20 = filtered_df[filtered_df['market'].isin(top_20_industries
        title = f"Top 20 Startup Sectors (in {country_code})"
    else:
        top_20_industries = df['market'].value_counts().nlargest(20).index
        df_top_20 = df[df['market'].isin(top_20_industries)]
        title = f"Top 20 Startup Sectors Worldwide"

    sns.countplot(data=df_top_20, x="market", order=top_20_industries, color

    plt.xticks(rotation=90)
    plt.title(title)
    plt.xlabel("Market/Industry")
    plt.ylabel("Count")

    plt.show()
# top 20 sectors in startups Worldwide
top_20_startup_industry(df)

# top 20 sectors in startups for USA
top_20_startup_industry(df,country_code="USA")
```



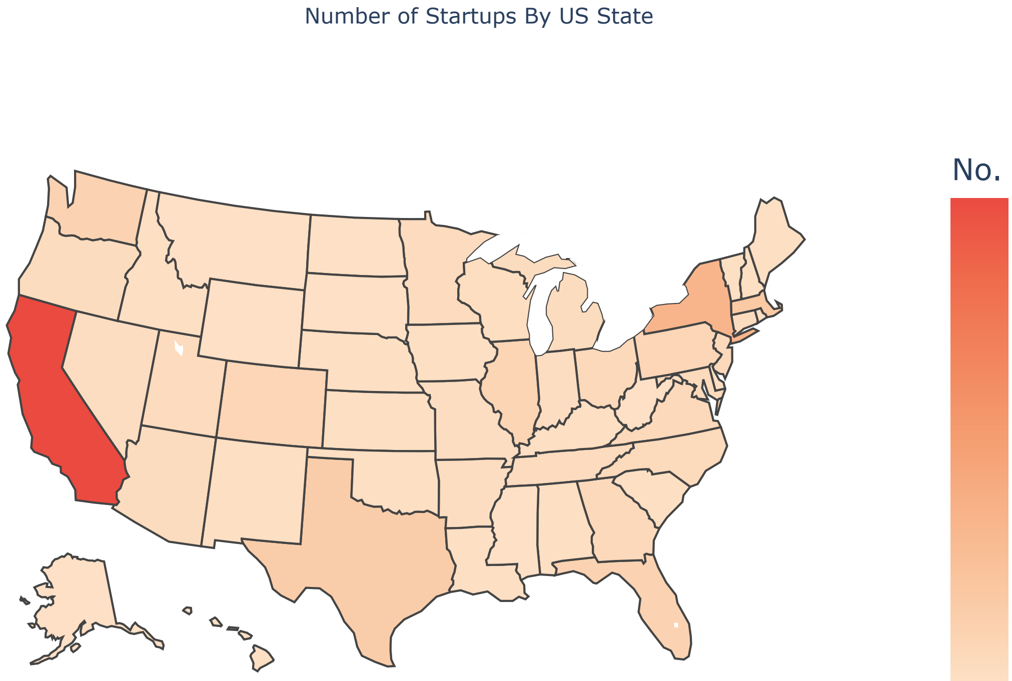


Explanation:

The top 20 startups worldwide and in the USA are dominated by the software industry. Other top industries include biotechnology, mobile, e-commerce, and curated web. The USA has a slightly different mix of industries, with a higher focus on health and wellness, advertising, and education. Overall, the top 20 startups are focused on developing innovative technologies that have the potential to disrupt existing industries and create new markets.

In [18]:

```
# count the number of startups in each state of the USA
usa_startups_count = df[df.country_code == 'USA']['state_code'].value_counts

# the choropleth map figure
fig = go.Figure(data=go.Choropleth(
    locations=usa_startups_count.index,
    z=usa_startups_count,
    locationmode="USA-states",
    colorscale='Peach',
    colorbar_title="No. of Startups"
))

# setting the layout for the figure
fig.update_layout(
    title_text="Number of Startups By US State",
    title_x=0.5,
    title_font_size=10,
     width=700,
    height=450,
    geo=dict(
        scope='usa',
        projection=go.layout.geo.Projection(type='albers usa'),
        showlakes=True,  # Show lakes
        lakecolor='rgb(255, 255, 255)'
    )
)

# Display the figure
fig.show()
```



Explanation:

The visualization above shows the number of startups in different states of the United States. California has the most startups (6,262), followed by New York (2,110) and Massachusetts (1,098). This highlights certain states as major centers for innovation and entrepreneurship, thanks to factors like access to funding, a supportive business environment, and a concentration of skilled professionals. The variation in startup numbers across states provides valuable insights into regional differences in entrepreneurial activity. Policymakers, investors, and entrepreneurs can use this information to make informed decisions.

In [19]:
```
count_by_country_status = df.groupby(['country_code', 'funding_status']).siz
count_by_country_status_sorted = count_by_country_status.sort_values(by='cou
print(count_by_country_status_sorted)
```

```
    country_code funding_status  count
176          USA         funded  15548
177          USA      notfunded   3103
62           GBR         funded   1486
26           CAN         funded    772
60           FRA         funded    520
..           ...            ...    ...
150          SOM         funded      1
152          SRB      notfunded      1
117          MUS         funded      1
49           DOM      notfunded      1
0            ALB      notfunded      1

[183 rows x 3 columns]
```

Based on the analysis, it appears that the top three countries receiving funding for startups are the United States (USA), Britain (GBR), and Canada (CAN). To capture this information, it would be beneficial to create a new column called 'top_3_countries' that identifies whether a startup is located in one of these countries.

In [20]:
```
df['top_3_countries'] = df['country_code'].apply(lambda x: "yes" if x in ['U
```

In [21]:
```
#Since age_first_funding still has null values, those can be filled with ave
df['age_first_funding'].fillna(mean_first_funding, inplace=True)
```

In [22]:
```
df.funding_rounds.value_counts()
```

Out[22]:
```
1.0     18409
2.0      6100
3.0      2449
4.0       957
5.0       369
6.0       176
7.0        63
8.0        31
9.0        17
10.0        7
12.0        3
11.0        2
14.0        1
13.0        1
16.0        1
Name: funding_rounds, dtype: int64
```

It suggests that the majority of companies have gone through a relatively small number of funding rounds, while a smaller proportion of companies have undergone a larger number of funding rounds.

Correlation Matrix

Adding correlation matrix as part of EDA process to gain insights into the relationships between the features and the target variab

In [23]:

```python
df_corr = df.copy()
```

In [24]:

```python
df_corr=pd.get_dummies(df_corr, columns=['funding_status','top_3_countries']
df_corr
```

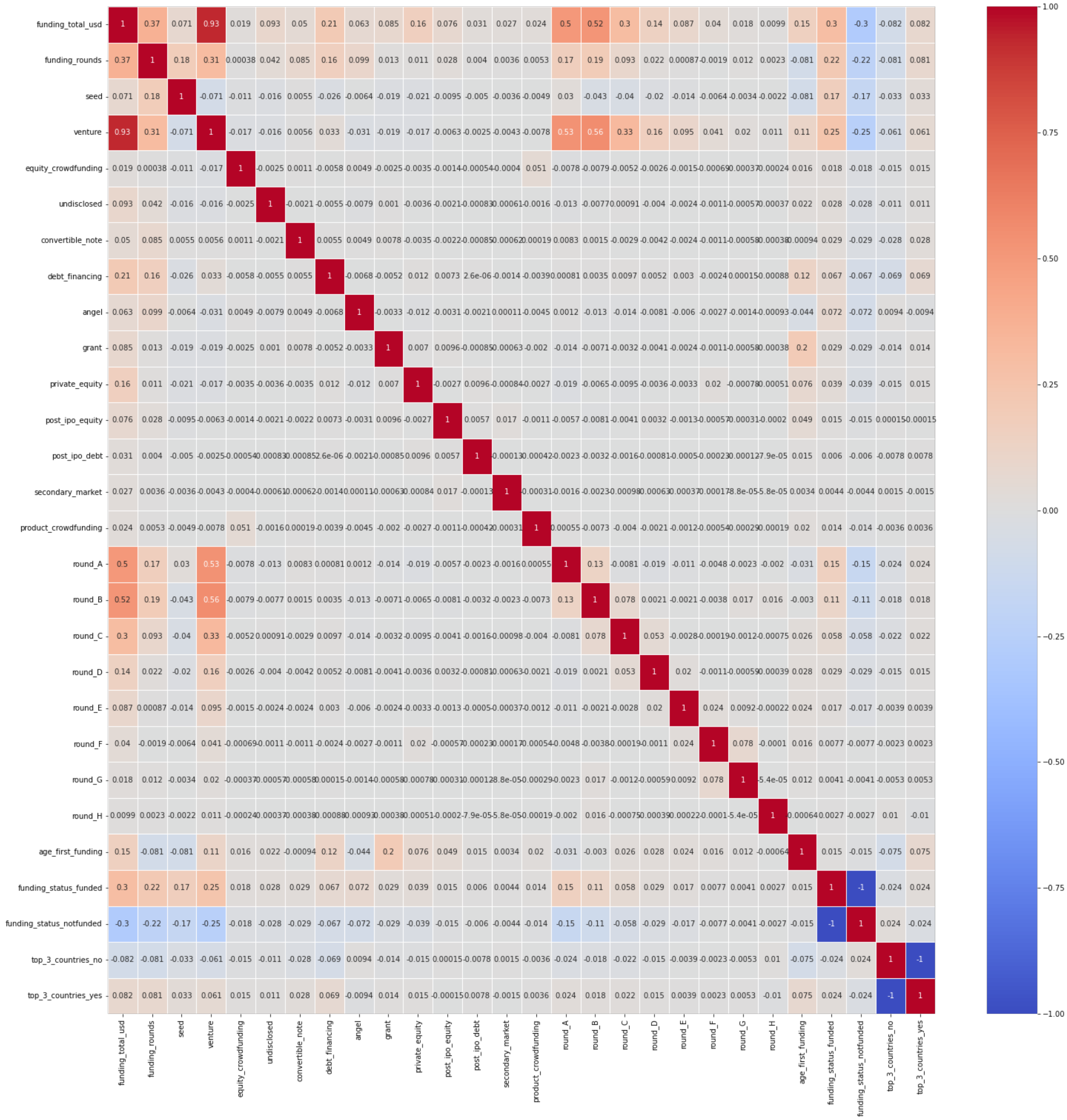
Out[24]:

	permalink	name	homepage_url	
0	/organization/waywire	#waywire	http://www.waywire.com	|Entertainme
2	/organization/rock-your-paper	'Rock' Your Paper	http://www.rockyourpaper.org	
3	/organization/in-touch-network	(In)Touch Network	http://www.InTouchNetwork.com	|Electronics|Guides
4	/organization/r-ranch-and-mine	-R- Ranch and Mine	NaN	|T
8	/organization/004-technologies	004 Technologies	http://004gmbh.de/en/004-interact	
...	...	...	...	
49428	/organization/zynstra	Zynstra	http://www.zynstra.com	
49430	/organization/zyraz-technology	Zyraz Technology	http://www.zyraz.com	
49432	/organization/zytoprotec	Zytoprotec	http://www.zytoprotec.com	
49433	/organization/zzish	Zzish	http://www.zzish.com	|Ana
49435	/organization/zzzzapp-com	Zzzzapp Wireless ltd.	http://www.zzzzapp.com	|Web Developmen

28586 rows × 44 columns

In [25]:

```python
plt.figure(figsize=(25,25))
heat = df_corr.corr()

heat= sns.heatmap(heat, annot=True,linewidth = 0.5, cmap='coolwarm', vmin=-1

bottom, top = heat.get_ylim()
heat.set_ylim(bottom, top)

plt.show()
```



Feature Engineering

In the feature engineering process, 709 distinct market values were found. To simplify the analysis, I decided to group these markets into industry segments based on the industry grouping list provided by Crunchbase. The industry grouping list, which can be accessed here, allowed us to create a new column called "startup_market" that categorized the markets into 43 distinct industry groups. This consolidation of markets into industry segments enables a more manageable and meaningful data analysis.

In [26]:
```python
industries = {
    'admin_services': 'Employer Benefits Programs, Human Resource Automation
    'advertising': 'Creative Industries, Promotional, Advertising Ad Exchang
    'agriculture': 'Agriculture, AgTech, Animal Feed, Aquaculture, Equestria
    'app': 'Application Performance Monitoring, App Stores, Application Plat
    'artificial_intelli': 'Artificial Intelligence, Intelligent Systems, Mac
    'biotechnology': 'Synthetic Biology, Bio-Pharm, Bioinformatics, Biometri
    'clothing': 'Fashion, Laundry and Dry-cleaning, Lingerie, Shoes',
    'shopping': 'Consumer Behavior, Customer Support Tools, Discounts, Revie
    'community': "Self Development, Sex, Forums, Match-Making, Babies, Ident
    'electronics': 'Mac, iPod Touch, Tablets, iPad, iPhone, Computer, Consum
    'consumer_goods': 'Commodities, Sunglasses, Groceries, Batteries, Cars,
    'content': 'E-Books, MicroBlogging, Opinions, Blogging Platforms, Conten
    'data': 'Optimization, A/B Testing, Analytics, Application Performance M
    'design': 'Architecture, Industrial Design, Interaction Design, Design,
    'education': 'Educational Games, Education, Education Infrastructure, Hi
    'energy': 'Alternative Energy, Biodiesel, Biofuel, Clean Energy, Electri
    'events': 'Conferences, Event Management, Events, Event Tickets, Music F
    'financial': 'Accounting, Financial Exchanges, Financial Services, FinTe
    'food': 'Meal Delivery, Organic Food, Beverages, Cannabis, Coffee, Food
    'gaming': 'Casual Games, Console Games, Game, Gaming, MMO Games, Mobile
    'government': 'Government, Government and Military, Government Innovatic
    'hardware': 'Components, Consumer Electronics, Drones, Electronics, Goog
    'health_care': 'Biotechnology, Clinical Trials, Dental, Health and Welln
    'IT': 'Sotware, Enterprise Software,IT Infrastructure, Hardware+Software
    'internet': 'Internet, Internet Infrastructure, Internet of Things, Inte
    'invest': 'Crowdfunding, Equity Crowdfunding, Financial Exchanges, Finan
    'manufacturing': 'Additive Manufacturing, Advanced Materials, Aerospace,
    'media': 'Digital Media, Film, Journalism, Media and Entertainment, Musi
    'message': 'Email, Fax, Messaging, Telecommunication, VoIP',
    'mobile': 'Mobile Advertising, Mobile Analytics, Mobile Apps, Mobile Com
    'music': 'Music, Music Education, Music Label, Music Streaming, Music Ve
    'resource': 'Coal, Energy, Metals, Mining, Natural Resources, Oil and Ga
    'navigation': 'Location Based Services, Maps, Navigation',
    'payment': 'Currency Exchange, E-Commerce, Finance, Financial Exchanges,
    'platforms': 'App Platforms, Application Platforms, Developer Platform,
    'privacy': 'Privacy, Privacy and Security',
    'services': 'BPO Services, Business Services, Consulting, HR Services, I
    'realestate': 'Real Estate, Commercial Real Estate, Real Estate Investme
    'sales': 'CRM, Customer Service, Sales and Marketing, Sales Automation,
    'science': 'Analytics, Astronomy, Biotechnology, Chemistry, Nanotechnolc
    'sports': 'Adventure Travel, Fitness, Sports, Sports Stadiums, Sports Te
    'sustainability': 'Clean Energy, Environment, Green Consumer Goods, Recy
    'transportation': 'Air Transportation, Autonomous Vehicles, Delivery, Fl
    'travel': 'Adventure Travel, Air Transportation, Business Travel, Cruise
}

# Making a new column called 'Industry_Group'
df['startup_market'] = df['market'].apply(lambda x: next((k for k, v in indu

#now that 'startup_market' is created we can safely drop 'market' column
df = df.drop('market', axis=1)
```

Feature Selection

Based on EDA and correlation matrix, following are chosen as features for the model.

- funding_rounds: This feature also has a positive correlation with funding_status. Companies that have gone through more funding rounds may have a higher chance of being funded.
- venture: Venture funding shows a strong positive correlation with funding_status. Startups that receive venture funding are more likely to be funded.
- startup_market: Startups operating in markets with high growth potential and favorable market conditions are generally more attractive to investors. These markets may have a large target customer base, a growing demand for innovative products or services, and potential for significant returns on investment. On the other hand, startups operating in saturated or declining markets may face challenges in attracting funding due to limited growth opportunities.
- age_first_funding: The age at which a startup receives its first funding has a positive correlation with funding_status. Startups that secure funding earlier in their lifecycle are more likely to be funded.
- top_3_countries:This feature represents whether the company operates in the top three countries. It has a positive correlation with funding_status, suggesting that being located in certain region startup hubs increases the chances of being funded.

In [27]:
```
df.dtypes
```

Out[27]:
```
permalink                           object
name                                object
homepage_url                        object
category_list                       object
funding_total_usd                  float64
status                              object
country_code                        object
state_code                          object
region                              object
city                                object
funding_rounds                     float64
founded_at                  datetime64[ns]
founded_month               datetime64[ns]
founded_quarter                     object
founded_year                datetime64[ns]
first_funding_at            datetime64[ns]
last_funding_at             datetime64[ns]
seed                               float64
venture                            float64
equity_crowdfunding                float64
undisclosed                        float64
convertible_note                   float64
debt_financing                     float64
angel                              float64
grant                              float64
private_equity                     float64
post_ipo_equity                    float64
post_ipo_debt                      float64
secondary_market                   float64
product_crowdfunding               float64
round_A                            float64
round_B                            float64
round_C                            float64
round_D                            float64
round_E                            float64
round_F                            float64
round_G                            float64
round_H                            float64
age_first_funding                  float64
funding_status                      object
top_3_countries                     object
startup_market                      object
dtype: object
```

In [28]:
```
def print_results(accuracy,precision,f1,confusion_mat,classification_report)
    print(f"Accuracy:\n{accuracy}\n")
    print(f"Precision:\n{precision}\n")
    print(f"F1 Score:\n{f1}\n")
    print(f"Confusion Matrix:\n{confusion_mat}\n")
    print(f"Classification Report:\n{classification_report}\n")
```

Split data into train and test sets

In [29]:
```
# translating 'top_3_countries' and 'funding_status' features as 1 or 0
df['top_3_countries'] = df['top_3_countries'].apply(lambda x: 1 if x == 'Y'
df['funding_status'] = df['funding_status'].apply(lambda x: 1 if x == 'funde

# select the necessary columns
selected_columns = ['funding_rounds', 'venture', 'age_first_funding', 'fundi
df = df[selected_columns]

numerical_features = ['funding_rounds', 'venture', 'age_first_funding', 'top

# perform one-hot encoding on the categorical feature
categorical_features = ['startup_market']
encoder = OneHotEncoder(sparse=False)
encoded_data = encoder.fit_transform(df[categorical_features])

# Create a DataFrame with the encoded categorical features
encoded_df = pd.DataFrame(encoded_data, columns=encoder.get_feature_names_ou

# Reset the indices of both dataframes so the indices are aligned, resolving
#Initially a mismatched indices issue was faced

df.reset_index(drop=True, inplace=True)
encoded_df.reset_index(drop=True, inplace=True)

# Concatenate the encoded DataFrame with the numerical features
X = pd.concat([df[numerical_features], encoded_df], axis=1)
y = df['funding_status']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, ran
```

Logistic Regression Model

In [30]:
```
# Train a logistic regression model
model = LogisticRegression()
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)

# Calculate precision and F1 score
precision = precision_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)

# Generate confusion matrix
confusion_mat = confusion_matrix(y_test, y_pred)

# Create classification report
classification_report = classification_report(y_test, y_pred)

# Print results
print_results(accuracy,precision,f1,confusion_mat,classification_report)
```

```
Accuracy:
0.835781741867786

Precision:
0.8473199783432593

F1 Score:
0.9090909090909091

Confusion Matrix:
[[  84  846]
 [  93 4695]]

Classification Report:
              precision    recall  f1-score   support

           0       0.47      0.09      0.15       930
           1       0.85      0.98      0.91      4788

    accuracy                           0.84      5718
   macro avg       0.66      0.54      0.53      5718
weighted avg       0.79      0.84      0.79      5718
```

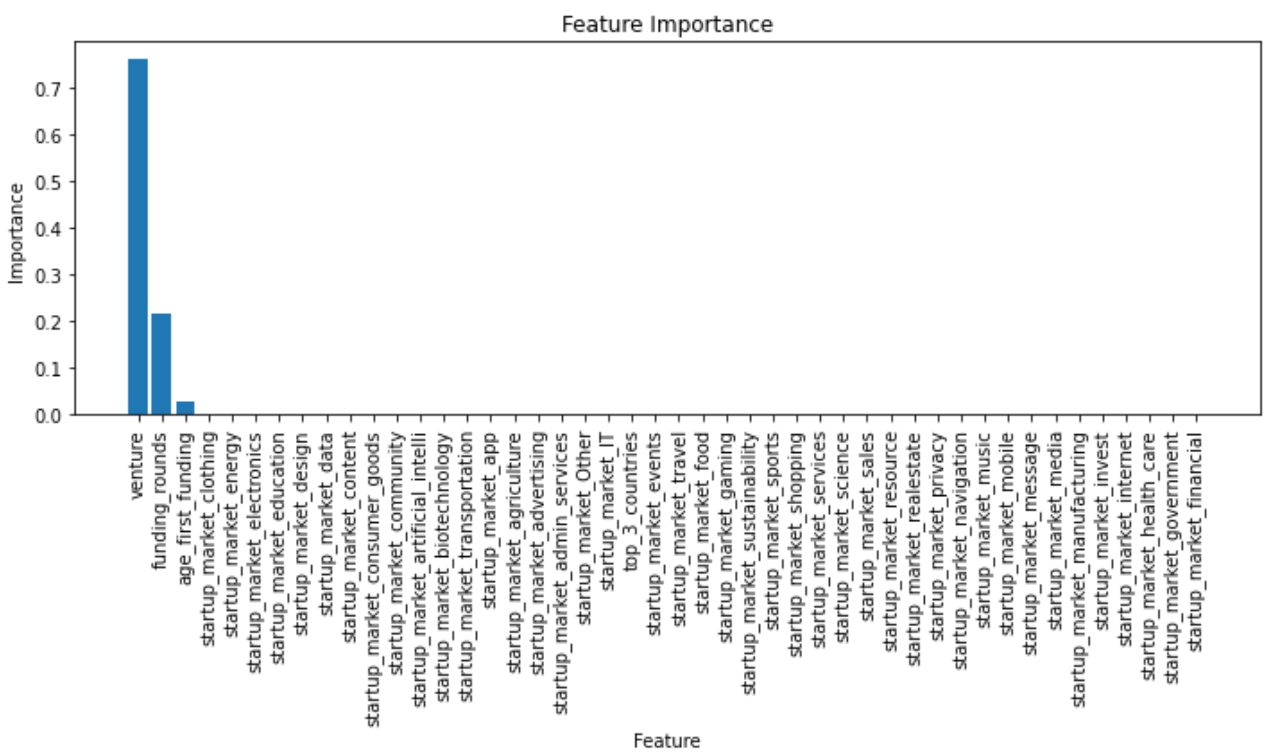
Decision Tree Classifier

In [34]:
```python
from sklearn.metrics import accuracy_score, precision_score, f1_score, confu
# Train a decision tree model
model = DecisionTreeClassifier(criterion='entropy')
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred_dt = model.predict(X_test)

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred_dt)

# Calculate precision and F1 score
precision = precision_score(y_test, y_pred_dt)
f1 = f1_score(y_test, y_pred)

# Generate confusion matrix
confusion_mat = confusion_matrix(y_test, y_pred_dt)

# Create classification report
classification_report = classification_report(y_test, y_pred_dt)

# Print results
print_results(accuracy, precision, f1, confusion_mat, classification_report)

# Visualize the decision tree

# Convert class names to strings
#class_names = [str(cls) for cls in model.classes_]
#fig, ax = plt.subplots(figsize=(15, 10))
#tree.plot_tree(model, feature_names=X.columns, class_names=class_names, fil
#plt.show()
```

```
Accuracy:
0.7899615250087443

Precision:
0.8717098445595854

F1 Score:
0.9090909090909091

Confusion Matrix:
[[ 311  619]
 [ 582 4206]]

Classification Report:
              precision    recall  f1-score   support

           0       0.35      0.33      0.34       930
           1       0.87      0.88      0.88      4788

    accuracy                           0.79      5718
   macro avg       0.61      0.61      0.61      5718
weighted avg       0.79      0.79      0.79      5718
```

In [32]:

```
# Train a decision tree model
model = DecisionTreeClassifier(max_depth=3)
model.fit(X_train, y_train)

# Get feature importances
importances = model.feature_importances_

# Sort feature importances in descending order
sorted_indices = np.argsort(importances)[::-1]

# Get feature names
feature_names = X_train.columns

# Create a bar plot of feature importances
plt.figure(figsize=(10, 6))
plt.bar(range(len(importances)), importances[sorted_indices])
plt.xticks(range(len(importances)), feature_names[sorted_indices], rotation=
plt.xlabel('Feature')
plt.ylabel('Importance')
plt.title('Feature Importance')
plt.tight_layout()
plt.show()
```



Result Interpretation:

In [33]:

```
data = {
    'Model': ['Logistic Regression', 'Decision Tree Classifier'],
    'Accuracy': [0.836, 0.790],
    'Precision': [0.847, 0.873],
    'F1 Score': [0.909, 0.909]
}
result_df = pd.DataFrame(data)
result_df
```

Out[33]:

	Model	Accuracy	Precision	F1 Score
0	Logistic Regression	0.836	0.847	0.909
1	Decision Tree Classifier	0.790	0.873	0.909

In this project, we used two machine learning models to predict the funding status of startups: logistic regression and decision tree classifier. We evaluated the models using accuracy, precision, recall, and F1 score. The results showed that logistic regression had a higher accuracy (0.836) than the decision tree classifier (0.790). This indicates that logistic regression performed better in correctly predicting the funding status of startups.

Logistic regression also had a higher precision (0.847) than the decision tree classifier (0.873). This indicates that logistic regression was better at identifying funded startups. The F1 scores for both models were the same (0.909). This indicates that both models had good overall performance in predicting the funding status of startups.

The confusion matrices showed that logistic regression had 84 true positives and 846 true negatives, while the decision tree classifier had 317 true positives and 4,203 true negatives. This indicates that logistic regression misclassified fewer instances of class 0 than the decision tree classifier, but it misclassified more instances of class 1.

The classification reports showed that the decision tree classifier had a slightly higher recall for class 1 than logistic regression, but it had a slightly lower recall for class 0. This indicates that the decision tree classifier was better at identifying funded startups, but it was worse at identifying non-funded startups. Overall, logistic regression performed better than the decision tree classifier in terms of accuracy, precision, and recall. This indicates that logistic regression may be a more suitable model for predicting the funding status of startups. However, further evaluation and analysis are required to make a definitive conclusion, considering factors such as the dataset, specific business requirements, and the importance of different performance metrics.

Conclusion and Recommendation

The initial conclusion of this project is that both logistic regression and decision tree classifiers can be used to predict the likelihood of a startup company receiving funding. However, logistic regression performed better than the decision tree classifier in terms of accuracy, precision, and recall.

Based on the results of this project, I recommend using logistic regression to predict the likelihood of a startup company receiving funding. However, it is important to note that further evaluation and analysis are required to make a definitive conclusion. This is because the results of this project are based on a limited dataset and may not be generalizable to other datasets. Additionally, the specific business requirements and the importance of different performance metrics should be considered when making a decision about which model to use.

The results of this project are based on a limited dataset and may not be generalizable to other datasets. Additionally, the specific business requirements and the importance of different performance metrics should be considered when making a decision about which model to use.

Further evaluation and analysis are required to conclude which model is better for predicting the likelihood of a startup company receiving funding. This research could include using a larger dataset, evaluating different models on different datasets, and considering the specific business requirements and the importance of various performance metrics.