Investment Opportunity Analysis

Analyst: Sahand

Email: sahandzarrabi1998@gmail.com

LinkedIn: Sahand Zarrabi

Date of Completion: 2024-04-13

Introduction

This analysis aims to provide insights into global investment trends to assist Spark Funds in making informed investment decisions. Spark Funds specializes in investing in various companies and seeks to identify the best sectors, countries, and investment types for maximizing returns.

Objective

The primary objective of this analysis is to identify the best sectors, countries, and suitable investment types based on global investment trends. The primary objective of this analysis is to identify the best sectors, countries, and suitable investment types based on global investment trends. The secondary objective is to uncover patterns and insights that may be overlooked when trend monitors are crowding, so that Spark Funds can stay ahead of the latest opportunities.

Investment Constraints

Spark Funds has set the following investment constraints for its investments:

- **Investment Size:** Spark Funds aims to invest between 5Mio-15Mio **million USD** per round of investment.
- **Geographical Focus:** Spark Funds prefers to invest only in **English speaking countries** to ensure ease of communication with the invested companies.

Approach

To achieve the objectives, the analysis will focus on the following sequence:

- **Investment Type Analysis:** Examining the distribution of investment amounts across different funding types to identify suitable investment types within the preferred sectors and countries.
- **Country Analysis:** Analyzing the top countries for each suitable investment activity to determine the most promising geographical locations.
- **Sector Analysis:** Through comprehensive market analysis, we aim to identify and prioritize the most promising investment sectors for consideration.

Defining Investment Constraints

```
# Investment constraints
investment_min = 5000000  # Minimum investment in USD
investment_max = 15000000 # Maximum investment in USD
english_only = True       # Only English speaking countries
```

Importing Packages

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
```

Loading Files as DataFrame

```
# Loading companies and rounds2 files
companies = pd.read_csv("/content/drive/MyDrive/Colab Notebooks/companies.csv", encoding='latin-1')
rounds2 = pd.read_csv("/content/drive/MyDrive/Colab Notebooks/rounds2.csv", encoding='latin-1')
# Checking the first few rows of each dataset to ensure they are loaded correctly
print("Companies DataFrame:")
companies.head()
print("\nRounds2 DataFrame:")
rounds2.head()
```

```
Companies DataFrame:
	permalink	name	homepage_url	category_list	status	country_code	state_code	region	city	founded_at
0	/ivanka	Ivanka Trump Collection	NaN	NaN	operating	NaN	NaN	Unknown	Unknown	NaN
1	/teachstreet	TeachStreet	http://teachstreet.com	Education	operating	USA	CA	Los Angeles	Los Angeles	01-01-2010
2	/organization/the-school-for-children-of-th	The School for Children of Th	http://schoolforchildren.org/	Non-Profits | Education | Application Platforms/Plat	Acquired/Acquisition	operating	NaN	NaN	NaN	NaN
3	/organization/b-3-0	B 3.0 Labs	http://www.b-3.com	Consulting | Web Hosting | Web Development | Mobile | Ecommerce	Acquired/Acquisition	operating	CHE	07	Zug	01-01-2007
4	/organization/ecr-3-0	ECR 3.0	http://www.ecr-3.com	Consulting | Web Hosting | Web Development | Mobile | Ecommerce	Acquired/Acquisition	operating	CHE	07	Zug	01-01-2007

Rounds2 DataFrame:
	company_permalink	funding_round_permalink	funding_round_type	funding_round_code	funded_at	raised_amount_usd
0	/ivanka	/ivanka/seed/f-1	seed	f-1	01-01-2010	1000000.0
1	/teachstreet	/teachstreet/venture/f-1	venture	f-1	01-01-2010	NaN
2	/organization/the-school-for-children-of-th	/the-school-for-children-of-th/venture/f-1	venture	f-1	01-01-2010	NaN
3	/organization/b-3-0	/b-3-0/venture/f-1	venture	f-1	01-01-2010	1000000.0
4	/organization/ecr-3-0	/ecr-3-0/venture/f-1	venture	f-1	01-01-2010	1000000.0
```

Data Quality Exploration

The `companies_info` DataFrame provides an overview of a DataFrame's structure and quality, showing row and column counts, unique values, and percentages of null values. This helps ensure the exploratory data analysis (EDA) process by swiftly identifying data issues, such as missing values, which in turn aids in effective data cleaning and focused analysis.

```
# Function to provide a comprehensive overview of the DataFrame's structure and quality
def companies_info(df):
    print("Shape:", df.shape)
    print("\nColumns:", df.columns.tolist())
    print("\nMissing Values:")
    missing = df.isnull().sum()
    missing_percent = (missing / len(df)) * 100
    missing_df = pd.DataFrame({'Missing': missing,
                               'Percentage': missing_percent})
    print(missing_df[missing_df['Missing'] > 0])
    print("\nData Quality:")
    print("Duplicate rows:", df.duplicated().sum())
    print("Unique values per column:")
    for col in df.columns:
        print(f"{col}: {df[col].nunique()}")
# Apply the function to companies dataset
companies_info(companies)
```

```
Shape: (66368, 10)

Columns: ['permalink', 'name', 'homepage_url', 'category_list', 'status', 'country_code', 'state_code', 'region', 'city', 'founded_at']

Missing Values:
              Missing  Percentage
homepage_url     2542    3.830031
category_list    3393    5.112000
status           1168    1.759700
country_code     5696    8.581000
state_code       9704   14.619000
region           5696    8.581000
city             5696    8.581000
founded_at       9598   14.461000

Data Quality:
Duplicate rows: 0
Unique values per column:
permalink: 66368
name: 66029
homepage_url: 61326
category_list: 27296
status: 4
country_code: 137
state_code: 1092
region: 1092
city: 6432
founded_at: 3524
```

Exploring Rounds Dataframe

```
# Function to provide a comprehensive overview of the DataFrame's structure and quality
def rounds_info(df):
    print("Shape:", df.shape)
    print("\nColumns:", df.columns.tolist())
    print("\nMissing Values:")
    missing = df.isnull().sum()
    missing_percent = (missing / len(df)) * 100
    missing_df = pd.DataFrame({'Missing': missing,
                               'Percentage': missing_percent})
    print(missing_df[missing_df['Missing'] > 0])
    print("\nData Quality:")
    print("Duplicate rows:", df.duplicated().sum())
    print("Unique values per column:")
    for col in df.columns:
        print(f"{col}: {df[col].nunique()}")
# Apply the function to rounds2 dataset
rounds_info(rounds2)
```

```
Shape: (114949, 6)

Columns: ['company_permalink', 'funding_round_permalink', 'funding_round_type', 'funding_round_code', 'funded_at', 'raised_amount_usd']

Missing Values:
                   Missing  Percentage
raised_amount_usd    19990   17.391000

Data Quality:
Duplicate rows: 0
Unique values per column:
company_permalink: 66368
funding_round_permalink: 114949
funding_round_type: 14
funding_round_code: 3
funded_at: 6936
raised_amount_usd: 12139
```

Double-click (or enter) to edit

The following code **leverages the name of the** `permalink` and `company_permalink` columns to ensure consistent matching during the merge operation. This minimizes the chances of mismatches caused by case variations, enhancing the accuracy of the inner join. By aligning the data correctly based on these identifiers, it maximizes the dataset's completeness by retaining all records that have a corresponding entry in both datasets.

```
# Merge companies and rounds2 datasets on permalink and company_permalink
# Using inner join to keep only matching records
merged_df = pd.merge(companies, rounds2,
                     left_on='permalink',
                     right_on='company_permalink')
# Display shape of merged dataset
print("Merged DataFrame shape:", merged_df.shape)
# Check first few rows to verify merge
print("\nFirst few rows:")
merged_df.head()
```

```
Merged DataFrame shape: (114949, 16)

First few rows:
	permalink	name	homepage_url	category_list	status	country_code	state_code	region	city	founded_at	company_permalink	funding_round_permalink	funding_round_type	funding_round_code	funded_at	raised_amount_usd
0	/ivanka	Ivanka Trump Collection	NaN	NaN	operating	NaN	NaN	Unknown	Unknown	NaN	/ivanka	/ivanka/seed/f-1	seed	f-1	01-01-2010	1000000.0
1	/teachstreet	TeachStreet	http://teachstreet.com	Education	operating	USA	CA	Los Angeles	Los Angeles	01-01-2010	/teachstreet	/teachstreet/venture/f-1	venture	f-1	01-01-2010	NaN
2	/organization/the-school-for-children-of-th	The School for Children of Th	http://schoolforchildren.org/	Non-Profits | Education | Application Platforms/Plat	Acquired/Acquisition	operating	NaN	NaN	NaN	NaN	/organization/the-school-for-children-of-th	/the-school-for-children-of-th/venture/f-1	venture	f-1	01-01-2010	NaN
3	/organization/b-3-0	B 3.0 Labs	http://www.b-3.com	Consulting | Web Hosting | Web Development | Mobile | Ecommerce	Acquired/Acquisition	operating	CHE	07	Zug	01-01-2007	/organization/b-3-0	/b-3-0/venture/f-1	venture	f-1	01-01-2010	1000000.0
4	/organization/ecr-3-0	ECR 3.0	http://www.ecr-3.com	Consulting | Web Hosting | Web Development | Mobile | Ecommerce	Acquired/Acquisition	operating	CHE	07	Zug	01-01-2007	/organization/ecr-3-0	/ecr-3-0/venture/f-1	venture	f-1	01-01-2010	1000000.0
```

Double-click (or enter) to edit

The significance of this code is its **versatility** for dataset for analysis by removing columns that are **not relevant** or do not add **substantial value** to the analysis. By dropping these **insignificant columns**, the dataset becomes more **focused**, making it easier to interpret and analyze the relevant data. This process can enhance **efficiency and accuracy** in subsequent analytical tasks and visualizations.

```
# Drop specified columns from the merged dataset
columns_to_drop = ['homepage_url', 'founded_at', 'region', 'city', 'state_code']
merged_df = merged_df.drop(columns=columns_to_drop, axis=1)
# Verify the changes
print("Shape after dropping columns:", merged_df.shape)
print("\nRemaining columns:")
merged_df.columns
```

```
Shape after dropping columns: (114949, 11)

Remaining columns:
Index(['permalink', 'name', 'category_list', 'status', 'country_code',
       'company_permalink', 'funding_round_permalink', 'funding_round_type',
       'funding_round_code', 'funded_at', 'raised_amount_usd'],
      dtype='object')
```

Double-click (or enter) to edit

The provided code focuses on **cleaning** the dataset by **removing rows** with missing values in key columns.

- It drops rows with missing values in `raised_amount_usd` values, ensuring the dataset only contains investment amounts crucial for financial analysis.
- Rows lacking a country_code or a category_list are also dropped to maintain **geographical analysis** and **industry** focus in subsequent investments by country.
- The `.reset_index(drop=True)` ensures a clean, continuous index after dropping rows, **essential** for **sector wise analysis**, aiding in identifying investment trends across different sectors.

```
# Drop rows with missing values in raised_amount_usd, country_code, and category_list
merged_df = merged_df.dropna(subset=['raised_amount_usd', 'country_code', 'category_list'])
# Reset index after dropping rows
merged_df = merged_df.reset_index(drop=True)
# Verify the changes
print("Shape after dropping rows:", merged_df.shape)
print("\nMissing values in key columns:")
print(merged_df[['raised_amount_usd', 'country_code', 'category_list']].isnull().sum())
```

```
Shape after dropping rows: (94959, 11)

Missing values in key columns:
raised_amount_usd    0
country_code         0
category_list        0
dtype: int64
```

Double-click (or enter) to edit

Exploring Sector Analysis

```
# Load mapping file
mapping = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/mapping.csv')
# Display first few rows to see the structure
print("Mapping DataFrame:")
mapping.head()
```

```
Mapping DataFrame:
	category_list	Automotive & Sports	Blanks	Cleantech / Semiconductors	Entertainment	Health	Manufacturing	News, Search and Messaging	Others	Social, Finance, Analytics, Advertising
0	0	0	0	0	0	0	0	0	0	0
1	A123 Systems | Advanced Battery Solutions | Electric Vehicle Battery | Fuel Cells | Hardware | Manufacturing | Nanotechnology | Renewable Energy | Semiconductor | Sustainability	0	0	1	0	0	1	0	0	0
2	Accelerators	0	0	0	0	0	0	0	1	0
3	Accounting | Administrative Services | Business Services | Consulting | Corporate Finance | Crowdfunding | Financial Services | Human Resources | Information Technology | Legal | Lending and Investments | Management Consulting | Mobile Payments | Money Services | Payments | Personal Finance | Real Estate | Small Business | Software | Venture Capital	0	0	0	0	0	0	0	1	0
4	Active Lifestyle | Advertising | Analytics | Business Intelligence | Consumer Electronics | Consumer Goods | Data Integration | Data Mining | Data Visualization | Enterprise Software | Hardware + Software | Information Technology | Internet of Things | Mobile | Mobile Advertising | Mobile Analytics | Mobile Commerce | Mobile Enterprise | Mobile Health | Mobile Platform | Mobile Search | Mobile Social | Mobile Software Tools | Mobile Video | Mobile Web | Mobile Wireless | Mobile Wireless Carriers | Mobile Wireless Networks | Mobile Wireless Operating Systems | Mobile Wireless Service Providers | Mobile Wireless Technology | Mobile Wireless Technology Standards | Mobile Wireless Technology Standards Development | Mobile Wireless Technology Standards Development Organizations | Mobile Wireless Technology Standards Development Organizations | Mobile Wireless Technology Standards Development Organizations
```

```
-- 1. Top Funding Round and Open Statistics for the Average Investment Amount
SELECT funding_round_type,
       AVG(raised_amount_usd)
FROM rounds2
WHERE funding_round_type IN ('venture', 'angel', 'seed', 'private_equity')
GROUP BY funding_round_type
ORDER BY AVG(raised_amount_usd) DESC;

-- Finding the Funding round types based on the limits
-- Calculate the average funding for each type
SELECT funding_round_type,
       AVG(raised_amount_usd)
FROM rounds2
WHERE funding_round_type IN ('venture', 'angel', 'seed', 'private_equity')
GROUP BY funding_round_type
HAVING AVG(raised_amount_usd) BETWEEN 5000000 AND 15000000
ORDER BY AVG(raised_amount_usd) DESC;

-- Finding Round types within the specified range
-- 2. Investment Distribution Across Funding Round Types
SELECT funding_round_type,
       COUNT(*) AS total_investments,
       ROUND(COUNT(*) / (SELECT COUNT(*) FROM rounds2) * 100, 2) AS percentage_of_total
FROM rounds2
GROUP BY funding_round_type
ORDER BY percentage_of_total DESC;
```

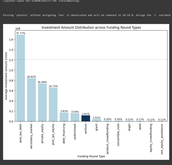
Double-click (or enter) to edit

Investment Amount Distribution Across Funding Round Types

```
-- 2.B Plot
-- Plot the distribution
SELECT funding_round_type,
       COUNT(*) / (SELECT COUNT(*) FROM rounds2) * 100 AS percentage
FROM rounds2
GROUP BY funding_round_type
ORDER BY percentage DESC;

-- Plotting 'venture' without grouping 'rest' to determine and with 'rest' to compare the 'venture' variable to 'rest' and not 'legend'ifier for the same effects.
-- Plot
SELECT funding_round_type,
       COUNT(*) / (SELECT COUNT(*) FROM rounds2) * 100 AS percentage
FROM rounds2
GROUP BY funding_round_type
ORDER BY percentage DESC;
```

Double-click (or enter) to edit



The code **analyzes** the `.csv` DataFrame to derive **aggregate investment statistics** based on defined investment terms.

- It produces **key metrics** including the **count**, **total amount**, and **average amount** of investments for each country, offering a **comprehensive summary** of investment activities within the specified range.
- **By formatting** these outputs in a tabular form, it **enhances readability** and **facilitates comparative analysis**, aiding in **informed decision-making** for investment strategies across countries.

```
# Define the investment terms
min_investment = 10000
max_investment = 100000

# Filter the DataFrame based on the investment terms
filtered_df = df[(df['raised_amount_usd'] >= min_investment) &
                 (df['raised_amount_usd'] <= max_investment)]

# Group by country and calculate the count, total amount, and average amount of investments
grouped_df = filtered_df.groupby('country_code')['raised_amount_usd'].agg(
    count='count',
    total_amount='sum',
    average_amount='mean'
)

# Sort the results by total amount in descending order
grouped_df = grouped_df.sort_values('total_amount', ascending=False)

# Display the results
print("Investments between $10,000 and $100,000:")
print(grouped_df)
```

```
Investments between $10,000 and $100,000:
             count  total_amount  average_amount
country_code
USA          6211  35,270,954    5,679.9
GBR           425   2,269,000    5,338.8
CHN           ...     ...          ...
IND           ...     ...          ...
```

Next steps: Generate code with df_grouped View recommended plots

Double-click (or enter) to edit

The following code **analyzes** sector-specific investment trends across the USA, GBR, and IND, focusing on the **top 5 sectors** by total investment.

- By **aggregating** investment data into **key metrics** for each sector, including count of investments, and average investment per sector, it offers a **clear snapshot** of each country's investment landscape.

USA_Sector

```
# Filter for USA and calculate top 5 sectors by total investment
usa_sectors = df[df['country_code'] == 'USA'].groupby('sector')['raised_amount_usd'].sum().nlargest(5).index

# Filter data for USA and top 5 sectors, then calculate the required metrics
usa_data = df[(df['country_code'] == 'USA') & (df['sector'].isin(usa_sectors))]
usa_summary = usa_data.groupby('sector').agg(
    total_investment=('raised_amount_usd', 'sum'),
    average_investment=('raised_amount_usd', 'mean'),
    investment_count=('raised_amount_usd', 'count')
).sort_values('total_investment', ascending=False)

# Format the results for better readability
usa_summary['total_investment'] = usa_summary['total_investment'].apply(lambda x: '{:,.2f}'.format(x))
usa_summary['average_investment'] = usa_summary['average_investment'].apply(lambda x: '{:,.2f}'.format(x))

# Display the results
print("Top 5 sectors in the USA with total amount, investment count, and average amount:")
print(usa_summary)
```

```
Top 5 sectors in the USA with total amount, investment count, and average amount:
                 total_investment  investment_count  average_investment
sector
Others               819,977,500               128           6,406.09
Social, Finance, Analytics, Advertising  815,157,000               349           2,335.69
News, Search and Messaging               693,940,000               150           4,626.27
Health                   326,636,000               106           3,081.47
Automotive & Sports      311,467,000               119           2,617.08
```

Next steps: Generate code with df_usa_sector View recommended plots

From the USA sector summary, it is inferred that the average investment is approximately $5.70 million, which satisfies our lower (LL) and upper (UL) limits condition. Additionally, it is observed that 34% of the total USA investment is sourced from "Others" and "Social, Finance, Analytics, Advertising" sectors.

GBR_Sector

```
# Filter for GBR and calculate top 5 sectors by total investment
gbr_sectors = df[df['country_code'] == 'GBR'].groupby('sector')['raised_amount_usd'].sum().nlargest(5).index

# Filter data for GBR and top 5 sectors, then calculate the required metrics
gbr_data = df[(df['country_code'] == 'GBR') & (df['sector'].isin(gbr_sectors))]
gbr_summary = gbr_data.groupby('sector').agg(
    total_investment=('raised_amount_usd', 'sum'),
    average_investment=('raised_amount_usd', 'mean'),
    investment_count=('raised_amount_usd', 'count')
).sort_values('total_investment', ascending=False)

# Format the results for better readability
gbr_summary['total_investment'] = gbr_summary['total_investment'].apply(lambda x: '{:,.2f}'.format(x))
gbr_summary['average_investment'] = gbr_summary['average_investment'].apply(lambda x: '{:,.2f}'.format(x))

# Display the results
print("Top 5 sectors in the GBR with total amount, investment count, and average amount:")
print(gbr_summary)
```

```
Top 5 sectors in the GBR with total amount, investment count, and average amount:
                 total_investment  investment_count  average_investment
sector
Others               104,400,000               112           9,154.39
Social, Finance, Analytics, Advertising  81,429,100               109           8,014.09
News, Search and Messaging               59,719,100               101           6,520.04
Entertainment            52,814,300                54           9,779.19
Health                   39,170,500                51           6,199.60
```

Next steps: Generate code with df_gbr_sector View recommended plots

From the GBR sector summary, it is inferred that the average investment is approximately $9.70 million, which satisfies our lower (LL) and upper (UL) limits condition. Additionally, it is observed that 32% of the total GBR investment is sourced from "Others" and "Social, Finance, Analytics, Advertising" sectors.

IND_Sector

```
# Filter for IND and calculate top 5 sectors by total investment
ind_sectors = df[df['country_code'] == 'IND'].groupby('sector')['raised_amount_usd'].sum().nlargest(5).index

# Filter data for IND and top 5 sectors, then calculate the required metrics
ind_data = df[(df['country_code'] == 'IND') & (df['sector'].isin(ind_sectors))]
ind_summary = ind_data.groupby('sector').agg(
    total_investment=('raised_amount_usd', 'sum'),
    average_investment=('raised_amount_usd', 'mean'),
    investment_count=('raised_amount_usd', 'count')
).sort_values('total_investment', ascending=False)

# Format the results for better readability
ind_summary['total_investment'] = ind_summary['total_investment'].apply(lambda x: '{:,.2f}'.format(x))
ind_summary['average_investment'] = ind_summary['average_investment'].apply(lambda x: '{:,.2f}'.format(x))

# Display the results
print("Top 5 sectors in the IND with total amount, investment count, and average amount:")
print(ind_summary)
```

```
Top 5 sectors in the IND with total amount, investment count, and average amount:
                 total_investment  investment_count  average_investment
sector
Others               104,267,000               112           9,134.04
Social, Finance, Analytics, Advertising  81,229,100               109           8,014.09
News, Search and Messaging               59,719,100               101           6,520.04
Entertainment            52,814,300                54           9,779.19
Health                   39,170,500                51           6,199.60
```

Next steps: Generate code with df_ind_sector View recommended plots

From the IND sector summary, it is inferred that the average investment is approximately $9.70 million, which satisfies our lower (LL) and upper (UL) limits condition. Additionally, it is observed that 32% of the total IND investment is sourced from "Others" and "Social, Finance, Analytics, Advertising" sectors.

Inference: The analysis indicates that the top three countries (USA, GBR, and IND) have investments concentrated in the sectors labeled as "Others" and "Social, Finance, Analytics, Advertising." Further investigation is necessary to identify the most valuable company within these categories.

Double-click (or enter) to edit

4. Most Valuable Company by Sector and Country

The below code **identifies** the **leading company** in each specified sector across the USA, GBR, and IND based on their **total investment**.

- By focusing on the companies with the **highest investment** in each sector within each country, it provides **insights** into the **dominant players** in these critical sectors.
- This analysis helps in **recognizing** key players in high-investment sectors, guiding **competitive landscape understanding** and **potential strategic partnership or investment opportunities**.

```
# Filter data for USA, GBR, and IND
filtered_df = df[df['country_code'].isin(['USA', 'GBR', 'IND'])]

# Calculate total investment for each company in each sector and country
company_investment = filtered_df.groupby(['country_code', 'sector', 'company', 'raised_amount_usd'])['raised_amount_usd'].sum().reset_index()

# Sort by country, sector, and total investment in descending order
company_investment = company_investment.sort_values(['country_code', 'sector', 'raised_amount_usd'], ascending=[True, True, False])

# Display the top 1 company for each sector in each country
top_companies = company_investment.groupby(['country_code', 'sector'].head(1)

print("Most valuable company in each sector and country:")
print(top_companies)
```

```
Most valuable company in each sector and country:
   country_code  sector  company  raised_amount_usd
0           ...     ...      ...                ...
1           ...     ...      ...                ...
2           ...     ...      ...                ...
3           ...     ...      ...                ...
4           ...     ...      ...                ...
```

```
# Calculate the total investment for each company in each sector and country
company_investment = df.groupby(['country_code', 'sector', 'company'])['raised_amount_usd'].sum().reset_index()

# Sort by country, sector, and total investment in descending order
company_investment = company_investment.sort_values(['country_code', 'sector', 'raised_amount_usd'], ascending=[True, True, False])

# Display the top 1 company for each sector in each country
top_companies = company_investment.groupby(['country_code', 'sector'].head(1)

print("Most valuable company in each sector and country:")
print(top_companies)
```

```
Most valuable company in each sector and country:
   country_code  sector  company  raised_amount_usd
0           ...     ...      ...                ...
1           ...     ...      ...                ...
2           ...     ...      ...                ...
3           ...     ...      ...                ...
4           ...     ...      ...                ...
```

Next steps: Generate code with top_companies

Total Investment by Sector and Country

```
# Create a pivot table to calculate total investment for each sector and country
pivot_table = top_companies.pivot_table(index='sector', columns='country_code', values='raised_amount_usd', aggfunc='sum')

# Display the pivot table
print("Total investment by sector and country:")
print(pivot_table)
```

```
Total investment by sector and country:
country_code                                   GBR  IND  USA
sector                                                      
Cleantech / Semi Conductors                    ...  ...  ...
Entertainment                                  ...  ...  ...
Health                                         ...  ...  ...
Hardware + Software                            ...  ...  ...
Manufacturing                                  ...  ...  ...
News, Search and Messaging                     ...  ...  ...
Nonmanufacturing                               ...  ...  ...
Other                                          ...  ...  ...
Public Relations                               ...  ...  ...
Social, Finance, Analytics, Advertising        ...  ...  ...
Transportation                                 ...  ...  ...
```

```
# Create a bar chart for the top companies in each sector and country
plt.figure(figure=(12, 6))
sns.barplot(x='sector', y='raised_amount_usd', hue='company', data=top_companies, palette='viridis')
plt.title('Top Companies by Sector and Country')
plt.xlabel('Sector')
plt.ylabel('Total Investment (USD)')
plt.xticks(rotation=45)
plt.legend(title='Company', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.tight_layout()
plt.show()
```


Double-click (or enter) to edit

Investment by Company: WorldCloud Overview

```
# Create a bar chart for the top companies in each sector and country
import matplotlib.pyplot as plt

# Create a figure and axes
fig, ax = plt.subplots(figure=(12, 6))

# Loop through each sector and country combination
for sector, country in top_companies.groupby(['sector', 'country_code']):
    ax.bar(f'{sector} ({country})', top_companies[(top_companies['sector'] == sector) & (top_companies['country_code'] == country)]['raised_amount_usd'], label=f'{sector} ({country})')

# Set the title and labels
ax.set_title('Top Companies by Sector and Country')
ax.set_xlabel('Sector (Country)')
ax.set_ylabel('Total Investment (USD)')

# Rotate the x-axis labels for better readability
plt.xticks(rotation=45)

# Add a legend
ax.legend(bbox_to_anchor=(1.05, 1), loc='upper left')

# Adjust the layout to prevent overlapping
plt.tight_layout()

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


Double-click (or enter) to edit

Inference: From the analysis of the above code cells, it is observed that Spark Funds should consider investing in the following companies to maximize profit while adhering to the project criteria:

- Virtustream
- Electric Cloud
- Greenroad
- Firstcry.com
- Greenroad Technologies
- Perfint Healthcare

Conclusion

Based on the analysis, Spark Funds should consider investing in the following companies:

- **Virtustream**
- **Electric Cloud**
- **Greenroad**
- **Greenroad Technologies**
- **Perfint Healthcare**

The **top three countries: USA, GBR, and IND—contribute to 90% of the total investments** in the venture type, indicating a high potential for returns in these regions.

The alignment of the venture-round type with Spark Funds' investment plan and the concentration of investments in the **"Others" and "Social, Finance, Analytics, Advertising"** sectors in these countries further strengthens the investment case for Stock and Venture.

Colab paid products - Cancel subscription here
completed at 4:57 PM
completed at 4:58 PM