Foreign Direct Investment Analysis

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


from google.colab import files
uploaded = files.upload()
```

```
Choose Files No file chosen  Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving FDI data.csv to FDI data.csv
```

```
df = pd.read_csv('/content/FDI data.csv')
```

Data Exploration

```
df.head()
```

	Sector	2000-01	2001-02	2002-03	2003-04	2004-05	2005-06	2006-07	2007-08	2008-09	2009-10	2010-11	2011-12	2012-13	2013-14	201
0	METALLURGICAL INDUSTRIES	22.69	14.14	36.61	8.11	200.38	149.13	169.94	1175.75	959.94	419.88	1098.14	1786.14	1466.23	567.63	359
1	MINING	1.32	6.52	10.06	23.48	9.92	7.40	6.62	444.36	34.16	174.40	79.51	142.65	57.89	12.73	684
2	POWER	89.42	757.44	59.11	27.09	43.37	72.69	157.15	988.68	907.66	1271.79	1271.77	1652.38	535.68	1066.08	707
	NON-															

```
df.tail()
```

	Sector	2000-01	2001-02	2002-03	2003-04	2004-05	2005-06	2006-07	2007-08	2008-09	2009-10	2010-11	2011-12	2012-13	20
58	PRINTING OF BOOKS (INCLUDING LITHO PRINTING IN...	0.00	0.00	6.30	0.00	0.06	9.90	20.04	35.54	31.61	70.51	36.63	47.39	14.34	113
59	COIR	0.00	0.00	0.00	0.00	0.47	0.59	0.04	0.01	0.00	0.25	0.10	0.55	0.15	0
60	CONSTRUCTION (INFRASTRUCTURE) ACTIVITIES	0.00	0.00	0.00	0.00	0.00	0.93	64.06	182.92	172.70	324.56	675.07	386.28	283.89	485
	CONSTRUCTION														

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 63 entries, 0 to 62
Data columns (total 18 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   Sector   63 non-null     object
 1   2000-01  63 non-null     float64
 2   2001-02  63 non-null     float64
 3   2002-03  63 non-null     float64
 4   2003-04  63 non-null     float64
 5   2004-05  63 non-null     float64
 6   2005-06  63 non-null     float64
 7   2006-07  63 non-null     float64
 8   2007-08  63 non-null     float64
 9   2008-09  63 non-null     float64
 10  2009-10  63 non-null     float64
 11  2010-11  63 non-null     float64
 12  2011-12  63 non-null     float64
 13  2012-13  63 non-null     float64
```

```
 14  2013-14  63 non-null     float64
 15  2014-15  63 non-null     float64
 16  2015-16  63 non-null     float64
 17  2016-17  63 non-null     float64
dtypes: float64(17), object(1)
memory usage: 9.0+ KB
```

```
df.describe()
```

	2000-01	2001-02	2002-03	2003-04	2004-05	2005-06	2006-07	2007-08	2008-09	2009-10	2
count	63.000000	63.000000	63.000000	63.000000	63.000000	63.000000	63.000000	63.000000	63.000000	63.000000	63
mean	37.757302	63.931587	42.925714	34.727778	51.090317	87.932540	198.281905	390.085714	498.348571	410.069524	339
std	112.227860	157.878737	86.606439	67.653735	101.934873	206.436967	686.783115	1026.249935	1134.649040	926.814626	627
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0
25%	0.000000	0.000000	0.200000	0.215000	0.715000	1.230000	4.160000	9.950000	11.950000	7.880000	8
50%	4.030000	5.070000	11.010000	6.370000	9.090000	22.620000	25.820000	58.820000	84.880000	69.740000	58
75%	23.510000	44.830000	36.555000	38.660000	43.205000	63.855000	108.325000	279.270000	383.320000	341.595000	304
max	832.070000	873.230000	419.960000	368.320000	527.900000	1359.970000	4713.780000	6986.170000	6183.490000	5466.130000	3296

```
df.columns
```

```
Index(['Sector', '2000-01', '2001-02', '2002-03', '2003-04', '2004-05',
       '2005-06', '2006-07', '2007-08', '2008-09', '2009-10', '2010-11',
       '2011-12', '2012-13', '2013-14', '2014-15', '2015-16', '2016-17'],
      dtype='object')
```

```
missing_values = df.isnull().sum()
print("Missing Values\n", missing_values)
```

```
Missing Values
 Sector     0
2000-01    0
2001-02    0
2002-03    0
2003-04    0
2004-05    0
2005-06    0
2006-07    0
2007-08    0
2008-09    0
2009-10    0
2010-11    0
2011-12    0
2012-13    0
2013-14    0
2014-15    0
2015-16    0
2016-17    0
dtype: int64
```

```
duplicate_values = df.duplicated().sum()
print('Number of duplicates', duplicate_values)
```

```
Number of duplicates 0
```

KPI's

Total FDI Inflow (Year-Wise)

```
# Transpose the data to get years as rows for easier aggregation
total_fdi_yearwise = df.set_index('Sector').sum()
total_fdi_yearwise
```

	0
2000-01	2378.71
2001-02	4027.69
2002-03	2704.32
2003-04	2187.85
2004-05	3218.69
2005-06	5539.75
2006-07	12491.76
2007-08	24575.40
2008-09	31395.96
2009-10	25834.38
2010-11	21383.07
2011-12	35120.78
2012-13	22423.59
2013-14	24299.32
2014-15	30930.47
2015-16	40000.99
2016-17	43478.26

dtype: float64

Sector-wise FDI Inflow

```
# Sector-wise FDI Inflow
total_fdi_sectorwise = df.set_index('Sector').sum(axis=1)
total_fdi_sectorwise
```

	0
Sector	
METALLURGICAL INDUSTRIES	10330.54
MINING	2271.83
POWER	11589.13
NON-CONVENTIONAL ENERGY	5181.49
COAL PRODUCTION	27.74
...	...
PRINTING OF BOOKS (INCLUDING LITHO PRINTING INDUSTRY)	634.66
COIR	4.06
CONSTRUCTION (INFRASTRUCTURE) ACTIVITIES	9817.47
CONSTRUCTION DEVELOPMENT: Townships, housing, built-up infrastructure and construction-development projects	24293.09
MISCELLANEOUS INDUSTRIES	10043.45

63 rows × 1 columns

dtype: float64

Compound Annual Growth Rate (CAGR)

```
def calculate_cagr(fdi_start, fdi_end, years):
    return (fdi_end / fdi_start) ** (1 / years) - 1

# Example for total FDI across all sectors (CAGR from 2000-01 to 2016-17)
fdi_start = total_fdi_yearwise['2000-01']
fdi_end = total_fdi_yearwise['2016-17']
years = 2017 - 2000

cagr_fdi = calculate_cagr(fdi_start, fdi_end, years)
cagr_fdi
```

```
0.18640021060790324
```

Top Contributing Sectors

```
# Top Contributing Sectors (e.g., top 5 sectors)
top_sectors = total_fdi_sectorwise.sort_values(ascending=False).head(5)
top_sectors
```

Sector	0
SERVICES SECTOR (Fin.,Banking,Insurance,Non Fin/Business,Outsourcing,R&D,Courier,Tech. Testing and Analysis, Other)	59476.49
COMPUTER SOFTWARE & HARDWARE	24669.49
CONSTRUCTION DEVELOPMENT: Townships, housing, built-up infrastructure and construction-development projects	24293.09
TELECOMMUNICATIONS	23946.01
AUTOMOBILE INDUSTRY	16673.92

FDI Contribution Percentage (Sector-wise)

```
# Total FDI for all sectors over the entire period
total_fdi_all = total_fdi_sectorwise.sum()

# Sector-wise Contribution Percentage
sector_percentage = (total_fdi_sectorwise / total_fdi_all) * 100
sector_percentage
```

Sector	0
METALLURGICAL INDUSTRIES	3.111693
MINING	0.684305
POWER	3.490797
NON-CONVENTIONAL ENERGY	1.560732
COAL PRODUCTION	0.008356
...	...
PRINTING OF BOOKS (INCLUDING LITHO PRINTING INDUSTRY)	0.191168
COIR	0.001223
CONSTRUCTION (INFRASTRUCTURE) ACTIVITIES	2.957150
CONSTRUCTION DEVELOPMENT: Townships, housing, built-up infrastructure and construction-development projects	7.317394
MISCELLANEOUS INDUSTRIES	3.025218

63 rows × 1 columns

Year-over-Year (YoY) Growth Rate

```
# Calculate YoY growth for each sector
df.set_index(df.columns[0], inplace=True)
df = df.T  # Transpose so years are rows

# Year-over-Year Growth Rate
yoy_growth = df.pct_change() * 100
# Optionally, reset index if you need the years as a column again
yoy_growth.reset_index(inplace=True)
yoy_growth
```

22.69	2000-01	14.14	36.61	8.11	200.38	149.13	169.94	1175.75	959.94	41
0	1.32	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
1	89.42	1.151718e+04	487.574553	1.537479e+01	337.197581	882.297297	2273.867069	1.224953e+02	2.557084e+03	6.292374
2	0.00	-1.000000e+02	-97.124006	-8.471761e+01	-97.071709	-98.142798	-98.447343	-9.405065e+01	-8.613137e+01	-5.105167
3	0.00	NaN	-100.000000	-9.903382e+01	-100.000000	577.037037	-46.721311	-7.606256e+01	-9.982523e+01	-1.000000
4	9.35	inf	inf	2.015000e+05	inf	32.275711	6646.923077	9.878977e+03	1.586682e+05	
...	...	...	...	...	...	...	...	...	...	
57	0.00	-1.000000e+02	inf	-1.000000e+02	500.000000	592.307692	223.225806	8.764520e+01	-1.475189e+01	7.651534
58	0.00	NaN	-100.000000	NaN	683.333333	-94.040404	-99.800399	-9.997186e+01	-1.000000e+02	-9.964544
59	0.00	NaN	NaN	NaN	-100.000000	57.627119	160050.000000	1.829100e+06	inf	1.297240
60	24.33	inf	inf	inf	inf	24492.473118	2074.445832	2.025153e+03	2.596879e+03	1.584166
61	832.07	3.277681e+02	505.983380	4.005952e+02	-19.880310	-27.961174	-78.113357	-8.640661e+01	-6.672686e+01	-7.900599

62 rows × 17 columns

Sector-wise Average FDI per Year

```
# Sector-wise Average FDI per Year
df.set_index(df.columns[0], inplace=True)
fdi_data = df.apply(pd.to_numeric, errors='coerce')
average_fdi_sectorwise = df.mean(axis=1)
average_fdi_sectorwise
```

	0
36.61	
10.06	160.995000
59.11	763.082857
1.70	369.985000
0.00	1.981429
56.78	469.925714
...	...
6.30	44.882857
0.00	0.290000
0.00	701.247857
36.10	1727.207857
218.76	626.517857

62 rows × 1 columns

Volatility of FDI (Standard Deviation)

```
# Volatility of FDI (Standard Deviation)
sector_volatility = df.std(axis=1)
sector_volatility
```

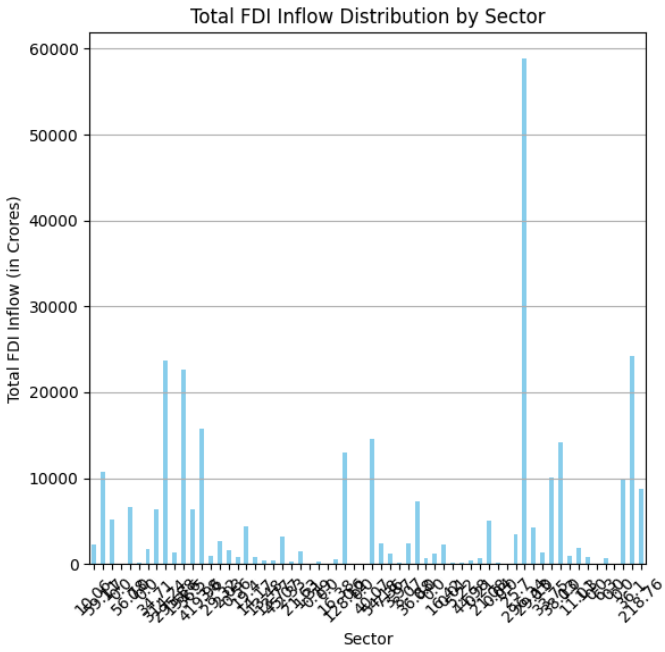
	0
36.61	
10.06	221.827041
59.11	523.591507
1.70	365.131948
0.00	4.272657
56.78	607.424548
...	...
6.30	38.677425
0.00	0.385507
0.00	1201.777983
36.10	1828.387280
218.76	474.493065

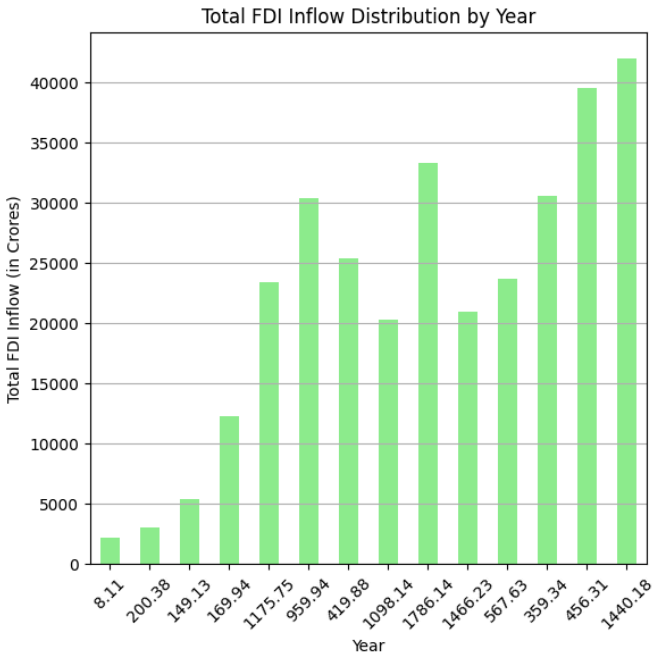
62 rows × 1 columns

Data Analyis and Visualisation

FDI Inflow Distribution both Sector-wise and Year-wise

```
# Set 'Sector' as the index
df.set_index(df.columns[0], inplace=True)

# Convert data to numeric (in case there are any non-numeric values)
df = fdi_data.apply(pd.to_numeric, errors='coerce')

# Calculate Total FDI by Sector
total_fdi_by_sector = fdi_data.sum(axis=1)

# Calculate Total FDI by Year
total_fdi_by_year = fdi_data.sum(axis=0)

# Plot Sector-wise FDI Distribution
plt.figure(figsize=(12, 6))

# Bar plot for Sector-wise FDI
plt.subplot(1, 2, 1)
total_fdi_by_sector.plot(kind='bar', color='skyblue')
plt.title('Total FDI Inflow Distribution by Sector')
plt.xlabel('Sector')
plt.ylabel('Total FDI Inflow (in Crores)')
plt.xticks(rotation=45)
plt.grid(axis='y')

# Plot Year-wise FDI Distribution
plt.subplot(1, 2, 2)
total_fdi_by_year.plot(kind='bar', color='lightgreen')
plt.title('Total FDI Inflow Distribution by Year')
plt.xlabel('Year')
plt.ylabel('Total FDI Inflow (in Crores)')
plt.xticks(rotation=45)
plt.grid(axis='y')

# Show the plots
plt.tight_layout()
plt.show()
```

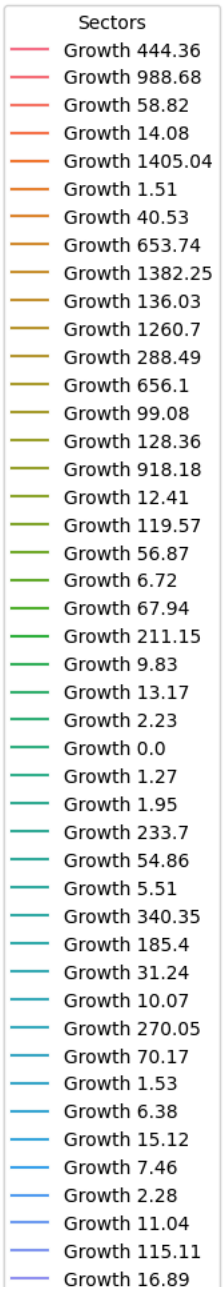




YOY Growth Rate

```
# Step 1: Set 'Sector' as the index
df.set_index(df.columns[0], inplace=True)

# Step 2: Convert data to numeric (in case there are any non-numeric values)
df = df.apply(pd.to_numeric, errors='coerce')

# Step 3: Transpose for easier calculations
df_transposed = df.T

# Step 4: Calculate Year-over-Year (YoY) Growth Rate
yoy_growth = df_transposed.pct_change() * 100
yoy_growth.reset_index(inplace=True)

# Rename columns for clarity
yoy_growth.columns = ['Year'] + [f'Growth {sector}' for sector in df.index]

# Step 5: Melt the DataFrame for easier plotting
yoy_growth_melted = yoy_growth.melt(id_vars=['Year'], var_name='Sector', value_name='Growth Rate')

# Step 6: Calculate Average FDI per Sector
average_fdi_sectorwise = fdi_data.mean(axis=1)

# Step 7: Calculate Total FDI by Sector
total_fdi_by_sector = fdi_data.sum(axis=1)

# Step 8: Calculate Total FDI by Year
total_fdi_by_year = fdi_data.sum(axis=0)

# Visualization of KPIs
plt.figure(figsize=(18, 12))

# Plot 1: YoY Growth Rate
plt.subplot(3, 2, 1)

# Use melted data for plotting

sns.lineplot(data=yoy_growth_melted, x='Year', y='Growth Rate', hue='Sector', dashes=False)

plt.title('Year-over-Year Growth Rate of FDI')
plt.xlabel('Year')
plt.ylabel('Growth Rate (%)')
plt.xticks(rotation=45)
plt.legend(title='Sectors')
plt.tight_layout()
plt.show()
```

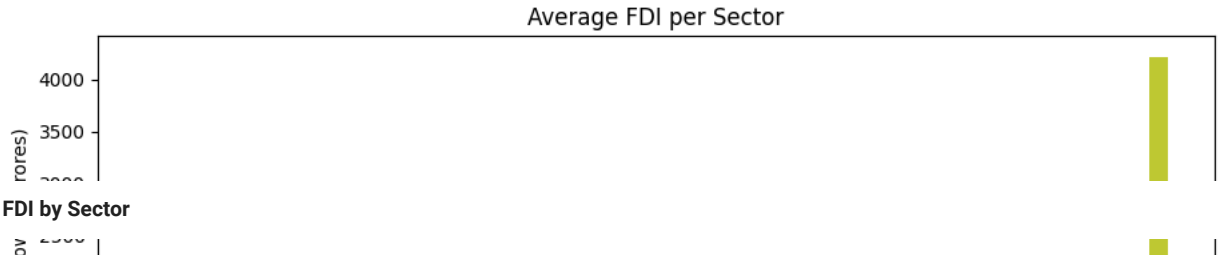
```
<ipython-input-54-75ef5fb2a06c>:44: UserWarning: Tight layout not applied. tight_layout cannot make axes height small enough to a
  plt.tight_layout()
```



Sector-wise Average FDI

```
plt.figure(figsize=(18, 12))
plt.subplot(3, 2, 2)
sns.barplot(x=average_fdi_sectorwise.index, y=average_fdi_sectorwise.values, palette='viridis')
plt.title('Average FDI per Sector')
plt.xlabel('Sector')
plt.ylabel('Average FDI Inflow (in Crores)')
plt.tight_layout()
plt.show()
```

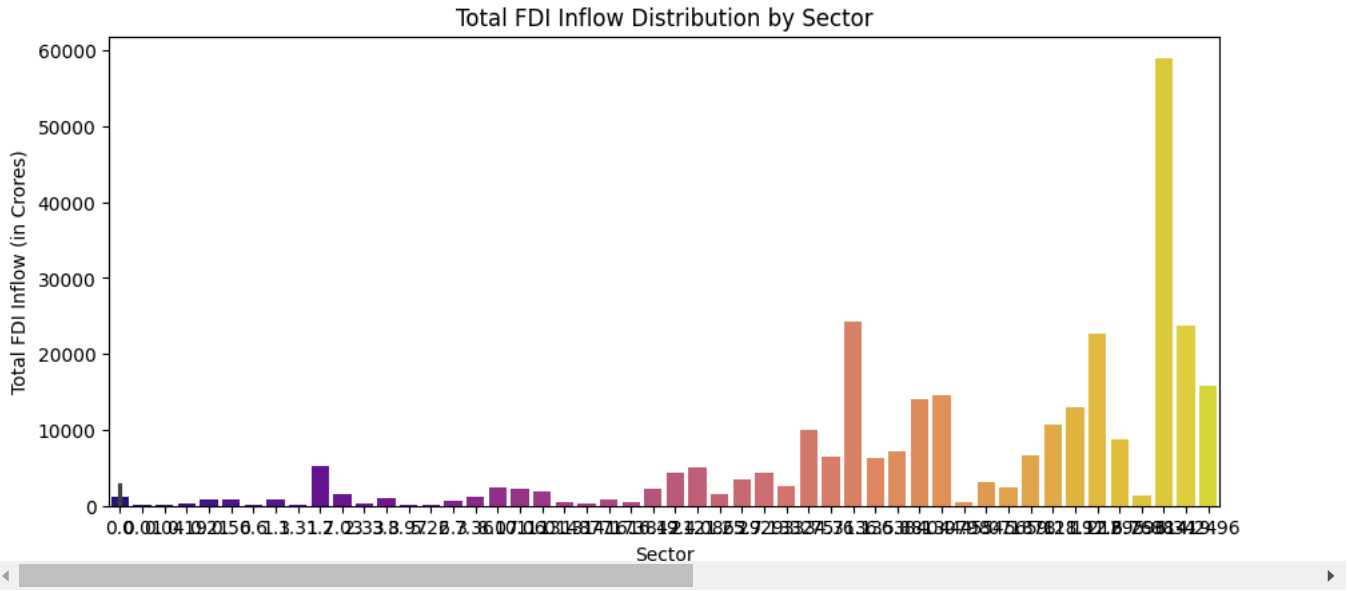
```
<ipython-input-51-ca8fef5be86f>:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le

  sns.barplot(x=average_fdi_sectorwise.index, y=average_fdi_sectorwise.values, palette='viridis')
```



Total FDI by Sector

```
plt.figure(figsize=(18, 12))
plt.subplot(3, 2, 3)
sns.barplot(x=total_fdi_by_sector.index, y=total_fdi_by_sector.values, palette='plasma')
plt.title('Total FDI Inflow Distribution by Sector')
plt.xlabel('Sector')
plt.ylabel('Total FDI Inflow (in Crores)')
plt.tight_layout()
plt.show()
```

```
<ipython-input-52-923dd591038a>:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le

  sns.barplot(x=total_fdi_by_sector.index, y=total_fdi_by_sector.values, palette='plasma')
```



Total FDI by Year

```
plt.figure(figsize=(18, 12))
plt.subplot(3, 2, 4)
sns.barplot(x=total_fdi_by_year.index, y=total_fdi_by_year.values, palette='magma')
plt.title('Total FDI Inflow Distribution by Year')
plt.xlabel('Year')
plt.ylabel('Total FDI Inflow (in Crores)')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

```
<ipython-input-53-b7bf51451a45>:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `le
```