Startups Analysis in 2019

In [1]:
```
import pandas as pd
df_2019 = pd.read_excel('C:/Users/Snehal/Downloads/2019_data.xlsx')
df_2019.head()
```

Out[1]:

	Sr. No.	Date	Startup Name	Industry/Vertical	Sub-Vertical	City	Investor Name	Investment Type	Amount(in USD)
0	1.0	2019-09-05	FPL Technologies	FinTech	Financial Services	Pune	Matrix Partners India, Sequoia India	Maiden Round	4,500,000
1	2.0	2019-09-04	Cashflo	FinTech	Invoice discounting platform and SME lending m...	Mumbai	SAIF Partners	Series A	3,300,000
2	3.0	2019-09-04	Digital F5	Advertising, Marketing	Digital marketing firm	Mumbai	TIW Private Equity	Private Equity Round	6,000,000
3	4.0	2019-09-04	3rdFlix	SaaS	Education Technology	Hyderabad	Exfinity Venture Partners	pre-series A	5,000,000
4	5.0	2019-09-04	75F	IoT	Building automation system	Burnsville	Breakthrough Energy Ventures	Series A	18,000,000

In [2]:
```
# Renaming columns for our convinience

def renaming_columns(df_2019):
    df_2019.rename(columns={
    'Startup Name': 'Startup_Name',
    'City': 'Location',
    'Investor Name': 'Investors',
    'Investment Type': 'Investment_Type',
    'Amount(in USD)': 'Amount($)',
    'Sub-Vertical': 'Sub_Industry',
    'Industry/Vertical':'Industry'
     }, inplace=True)
renaming_columns(df_2019)
```

In [3]:
```
# Extracting required columns
df_2019 = df_2019[['Date','Startup_Name','Industry','Sub_Industry','Location','Investors','Investment_Type','Am
```

In [4]:
```
# Converting date column to datetime to extract Year & Month
def date_opertion(df_2019):
    df_2019['Date'] = pd.to_datetime(df_2019['Date'], format="%d-%m%Y")
    df_2019['Month'] = df_2019['Date'].dt.strftime('%B')
    df_2019['Year'] = df_2019['Date'].dt.year
date_opertion(df_2019)
```

In [5]:
```
# Dealing with duplicate rows
def duplicate_rows(data):
    duplicate_rows = data[data.duplicated()]
    if len(duplicate_rows) > 0:
        data = data.drop_duplicates()
        print('Droped',len(duplicate_rows),'Duplicate Rows.')
    else:
        print('No Duplicate Rows.')
duplicate_rows(df_2019)
```

```
No Duplicate Rows.
```

In [6]:
```
# Dealing with Amount column data type
def amount_column(data):
    data['Amount($)'] = data['Amount($)'].fillna(0)
    data['Amount($)'] = data['Amount($)'].astype(str)
    data['Amount($)'] = data['Amount($)'].str.replace(',', '')
    data['Amount($)'] = pd.to_numeric(data['Amount($)'], errors='coerce')
    data['Amount($)'] = data['Amount($)'].fillna(0).astype(int)
amount_column(df_2019)
```

In [7]:
```
# Editing Industry column
values_to_replace = {'AI' : 'Artificial Intelligence',
                     'Customer Service' : 'Customer Service Platform',
                     'Ecommerce' : 'E-Commerce',
                     'E-commerce' : 'E-Commerce',
                     'EdTech' : 'Ed-Tech',
                     'Education' : 'Ed-Tech','Edtech': 'Ed-Tech','FinTech' : 'Fin-Tech', 'Fintech':'Fin-Tech',
                     'Health Care' : 'Healthcare','Health and wellness' : 'Healthcare','Health and Wellness':'He
                     'Saas':'SaaS','Tech':'Technology','Transport':'Transportation',}
def replace_values(df):
    df['Industry'] = df['Industry'].replace(values_to_replace)
replace_values(df_2019)
```

In [8]:
```
df_2019.head()
```

Out[8]:

	Date	Startup_Name	Industry	Sub_Industry	Location	Investors	Investment_Type	Amount($)	Month	Year
0	2019-09-05	FPL Technologies	Fin-Tech	Financial Services	Pune	Matrix Partners India, Sequoia India	Maiden Round	4500000	September	2019.0
1	2019-09-04	Cashflo	Fin-Tech	Invoice discounting platform and SME lending m...	Mumbai	SAIF Partners	Series A	3300000	September	2019.0
2	2019-09-04	Digital F5	Advertising, Marketing	Digital marketing firm	Mumbai	TIW Private Equity	Private Equity Round	6000000	September	2019.0
3	2019-09-04	3rdFlix	SaaS	Education Technology	Hyderabad	Exfinity Venture Partners	pre-series A	5000000	September	2019.0
4	2019-09-04	75F	IoT	Building automation system	Burnsville	Breakthrough Energy Ventures	Series A	18000000	September	2019.0

Summary of the year 2019

- Shape = (114, 10)
- Unique Industry = 41
- Unique Sub_Industry = 103
- Unique Location = 34
- Unique Investment_Type = 36

Graphs

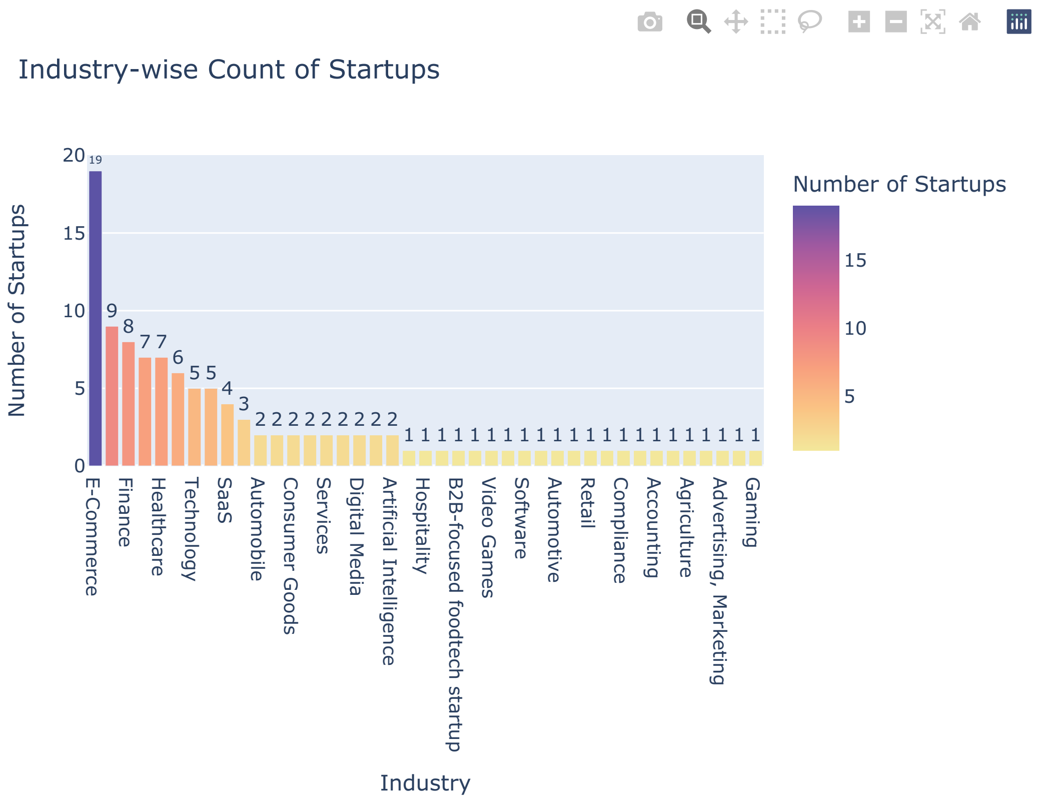
In [9]:
```
# Total Startup count in 2019
import plotly.graph_objects as go
from plotly.subplots import make_subplots
import pandas as pd

def startup_count(data):
    startup_count = data['Startup_Name'].nunique()
    return go.Indicator(
        mode="number",
        value=startup_count,
        title="Startup Count")

fig = make_subplots(rows=1, cols=1)

fig.add_trace(startup_count(df_2019))

fig.update_layout(title_text="Startup Count in 2019")
fig.show()
```

In [10]:
```
import pandas as pd
import ipywidgets as widgets
from IPython.display import display, HTML
import plotly.graph_objects as go
```

```python
def update_subindustry_options(change):
    selected_industry = change.new

def create_table(selected_industry):
    if selected_industry == 'All':
        display(HTML("<p>Select an industry to view the table.</p>"))
    else:
        filtered_df = df_2019[df_2019['Industry'] == selected_industry]

        if filtered_df.empty:
            display(HTML("<p>No data available for the selected criteria.</p>"))
        else:
            trace = go.Table(
                header=dict(values=["Startup_Name", "Industry", "Sub_Industry", "Investors", "Investment_Type",
                               fill=dict(color='#abb8e7'),
                               align=['left', 'center']),
                cells=dict(values=[filtered_df['Startup_Name'],
                                   filtered_df['Industry'],
                                   filtered_df['Sub_Industry'],
                                   filtered_df['Investors'],
                                   filtered_df['Investment_Type'],
                                   filtered_df['Location'],
                                   filtered_df['Amount($)'],
                                   filtered_df['Month']],
                           fill=dict(color=['white', 'lightgray']),
                           align=['left', 'center'])
            )

            layout = dict(width=1000, height=400)
            fig = go.Figure(data=[trace], layout=layout)
            fig.update_layout(margin=dict(l=0, r=0, t=0, b=0))
            display(fig)

industry_dropdown = widgets.Dropdown(options=['All'] + sorted(df_2019['Industry'].dropna().unique()), value='Al
industry_dropdown.observe(update_subindustry_options, names='value')
widgets.interactive(create_table, selected_industry=industry_dropdown)
```

Out[10]: interactive(children=(Dropdown(description='Industry:', options=('All', 'Accounting', 'Advertising, Marketing'…

In [11]:
```python
# Industry wise startup count
import plotly.express as px

def Industry_wise_startup(df):
    industry_count = df['Industry'].value_counts().reset_index()
    industry_count.columns = ['Industry', 'Startup Count']
    fig = px.bar(industry_count, x='Industry', y='Startup Count',
             title='Industry-wise Count of Startups',
             labels={'Industry': 'Industry', 'Startup Count': 'Number of Startups'},color = 'Startup Count',tex
                color_continuous_scale='sunset')
    fig.update_traces(textposition='outside')
    fig.show()
Industry_wise_startup(df_2019)
```



In [12]:
```python
# Top 10 Sub-Industries by Startups Count
```

In [12]:
```
# Top 10 Sub-Industries by Startups count

def sub_industry_top_10(data):
    top_sub_industries = data['Sub_Industry'].value_counts().nlargest(10)
    top_sub_df = top_sub_industries.reset_index()
    top_sub_df.columns = ['Sub_Industry', 'Count']
    fig = px.bar(top_sub_df, y='Sub_Industry', x='Count', title='Top 10 Sub-Industries by Startups Count', orie
                 color_continuous_scale='sunset')
    fig.update_yaxes(categoryorder='total ascending')
    fig.show()

sub_industry_top_10(df_2019)
```

In [13]:
```
# Top 10 Locations startup count wise
def top_10_location(data):
    location_counts = data.groupby('Location').size().reset_index(name='Count')
    top_10_locations = location_counts.sort_values(by='Count', ascending=False).head(10)
    fig = px.bar(top_10_locations, x='Location', y='Count', color='Count',color_continuous_scale='sunset',
             title='Top 10 Locations by Startups Count')
    fig.update_layout(xaxis_title='Location', yaxis_title='Count')
    fig.show()
top_10_location(df_2019)
```

In [14]:
```
# Top 10 Invetors
def top_investors(data):
    top_investors = data['Investors'].value_counts().nlargest(10)
    top_investors_df = top_investors.reset_index()
    top_investors_df.columns = ['Investor', 'Investment Count']
    fig = px.bar(top_investors_df, x='Investor', y='Investment Count',
                 title='Top 10 Investors by Investment Count',
                 labels={'Investor': 'Investor Name', 'Investment Count': 'Number of Investments'},
                 color='Investment Count',color_continuous_scale='sunset')
    fig.show()

top_investors(df_2019)
```

In [15]:
```
# Month wise startups
def monthly_startup_count(data):
    month_order = ['January', 'February', 'March', 'April', 'May', 'June', 'July', 'August', 'September', 'Octo
    data['Month'] = pd.Categorical(data['Month'], categories=month_order, ordered=True)
    monthly_count = data.groupby('Month')['Startup_Name'].nunique().reset_index()
    fig = px.line(monthly_count, x='Month', y='Startup_Name', title='Monthly Startup Count', labels={'Startup_N
    fig.update_layout(xaxis=dict(title='Month'))
```

```
    fig.update_traces(line=dict(color='brown'))
    fig.show()

monthly_startup_count(df_2019)
```

In [16]:
```
import pandas as pd
import ipywidgets as widgets
from IPython.display import display, HTML


# Location wise Startups
def create_table(selected_location):
    if selected_location == 'All':
        display(HTML("<p>Select an option from the dropdown to view the table.</p>"))
    else:
        filtered_df = df_2019[df_2019['Location'] == selected_location]

        if filtered_df.empty:
            display(HTML("<p>No data available for the selected location.</p>"))
        else:
            trace = go.Table(
                header=dict(values=["Startup Name", "Sub Industry", "Investor", "Investment Type", "Amount($)",
                            fill=dict(color='lightblue'),
                            align=['left', 'center']),
                cells=dict(values=[filtered_df['Startup_Name'],
                                   filtered_df['Sub_Industry'],
                                   filtered_df['Investors'],
                                   filtered_df['Investment_Type'],
                                   filtered_df['Amount($)'],
                                   filtered_df['Month']],
                           fill=dict(color=['white', 'lightgray']),
                           align=['left', 'center'])
            )

            layout = dict(width=1000, height=800)
            fig = go.Figure(data=[trace], layout=layout)
            display(fig)

location_dropdown = widgets.Dropdown(options=['All'] + sorted(df_2019['Location'].dropna().unique()), value='Al
widgets.interactive(create_table, selected_location=location_dropdown)
```

Out[16]:
```
interactive(children=(Dropdown(description='Location:', options=('All', 'Amritsar', 'Andheri', 'Bengaluru', 'B…
```

In [ ]:

In [ ]: