Startups Analysisin 2018

In [1]:
```python
# Loading dataset
import pandas as pd
df_2018 = pd.read_excel('C:/Users/Snehal/Downloads/2018_data.xlsx')
df_2018.head()
```

Out[1]:

	Sr. No.	Date	Startup Name	Industry/Vertical	Sub-Vertical	City	Investor Name	Investment Type	Amount(in USD)
0	1.0	2018-09-01 00:00:00	Netmeds	Consumer Internet	Online Pharmacy Chain	Chennai	Sistema Asia Fund, Sistema JSFC and Tanncam In...	Private Equity	35,000,000
1	2.0	2018-09-03 00:00:00	Udaan	B2B Platform	Logistics and Shipping	Bengaluru	DST Global and Lightspeed Venture Partners' gl...	Private Equity	225,000,000
2	3.0	2018-09-03 00:00:00	Daily hunt	Consumer Internet	News and ebooks Mobile App	Bengaluru	Falcon Edge	Private Equity	63,90,000
3	4.0	2018-09-04 00:00:00	3HCare	Healthcare	Healthcare Service Provider	Delhi	NaN	Seed / Angel Funding	1,000,000
4	5.0	2018-09-04 00:00:00	HappyGoEasy	Consumer Internet	Online Travel Agecy	Gurugram	Korea Investment Partners (KIP), Samsung and C...	Private Equity	NaN

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

def renaming_columns(df_2018):
    df_2018.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_2018)
```

In [3]:
```python
# Extracting required columns
df_2018 = df_2018[['Date','Startup_Name','Industry','Sub_Industry','Location','Investors','Investment_Type','Am
```

In [4]:
```python
# Converting date column to datetime to extract Year & month
def date_opertion(df_2018):
    df_2018['Date'] = pd.to_datetime(df_2018['Date'], format="%d-%m%Y")
    df_2018['Month'] = df_2018['Date'].dt.strftime('%B')
    df_2018['Year'] = df_2018['Date'].dt.year
date_opertion(df_2018)
```

In [5]:
```python
# 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_2018)
```

```
No Duplicate Rows.
```

In [6]:
```python
# 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_2018)
```

In [7]:
```python
# Editing Location column

def replace_values(df):
    value_to_replace = {'Ahmedabad': 'Ahemadabad', 'Ahemdabad' : 'Ahemadabad', 'Bengaluru' : 'Bangalore','Kolka
                       'Bhubneswar' : 'Bhubaneswar'}
    df['Location'] = df['Location'].replace(value_to_replace)
replace_values(df_2018)
```

In [8]:
```python
# Editing Industry column
values_to_replace = {'Ecommerce' : 'E-Commerce',
                    'E-commerce' : 'E-Commerce',
                    'E-Commerce' : 'E-Commerce',
                    'Ecommerce' : 'E-Commerce',
                    'B2B Platform' : 'B2B',
                    'Consumer internet' : 'Consumer Internet',
                    'Ed-tech' : 'Ed-Tech',
                    'Fiinance' : 'Finance',
                    'Food Tech' : 'Food-Tech','Food and Beverages' : 'Food & Beverages','Food and Beverage':'Fo
                    'Services':'Services Platform','Finance':'Financial Tech'}
def replace_values(df):
    df['Industry'] = df['Industry'].replace(values_to_replace)
replace_values(df_2018)
```

In [9]:
```
df_2018.head()
```

Out[9]:

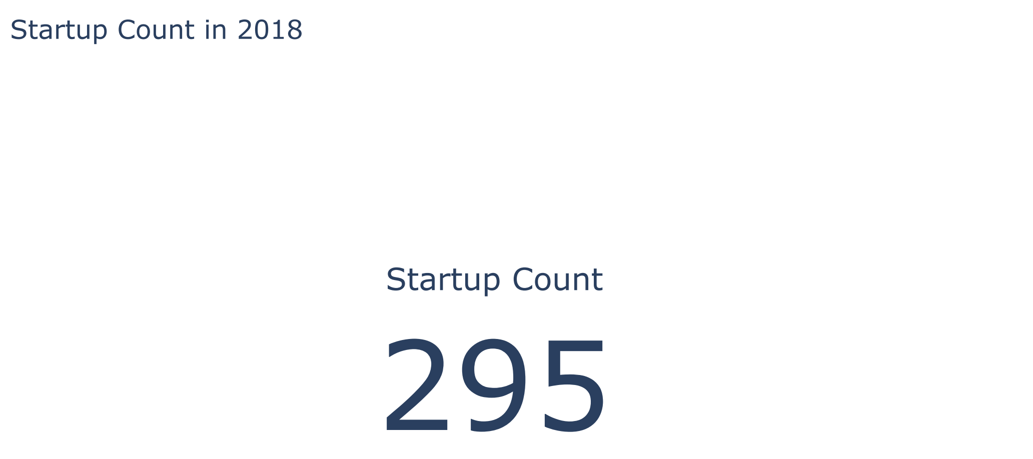
	Date	Startup_Name	Industry	Sub_Industry	Location	Investors	Investment_Type	Amount($)	Month	Year
0	2018-09-01	Netmeds	Consumer Internet	Online Pharmacy Chain	Chennai	Sistema Asia Fund, Sistema JSFC and Tanncam In...	Private Equity	35000000	September	2018.0
1	2018-09-03	Udaan	B2B	Logistics and Shipping	Bangalore	DST Global and Lightspeed Venture Partners' gl...	Private Equity	225000000	September	2018.0
2	2018-09-03	Daily hunt	Consumer Internet	News and ebooks Mobile App	Bangalore	Falcon Edge	Private Equity	6390000	September	2018.0
3	2018-09-04	3HCare	Healthcare	Healthcare Service Provider	Delhi	NaN	Seed / Angel Funding	1000000	September	2018.0
4	2018-09-04	HappyGoEasy	Consumer Internet	Online Travel Agecy	Gurugram	Korea Investment Partners (KIP), Samsung and C...	Private Equity	0	September	2018.0

Summary of the year 2018

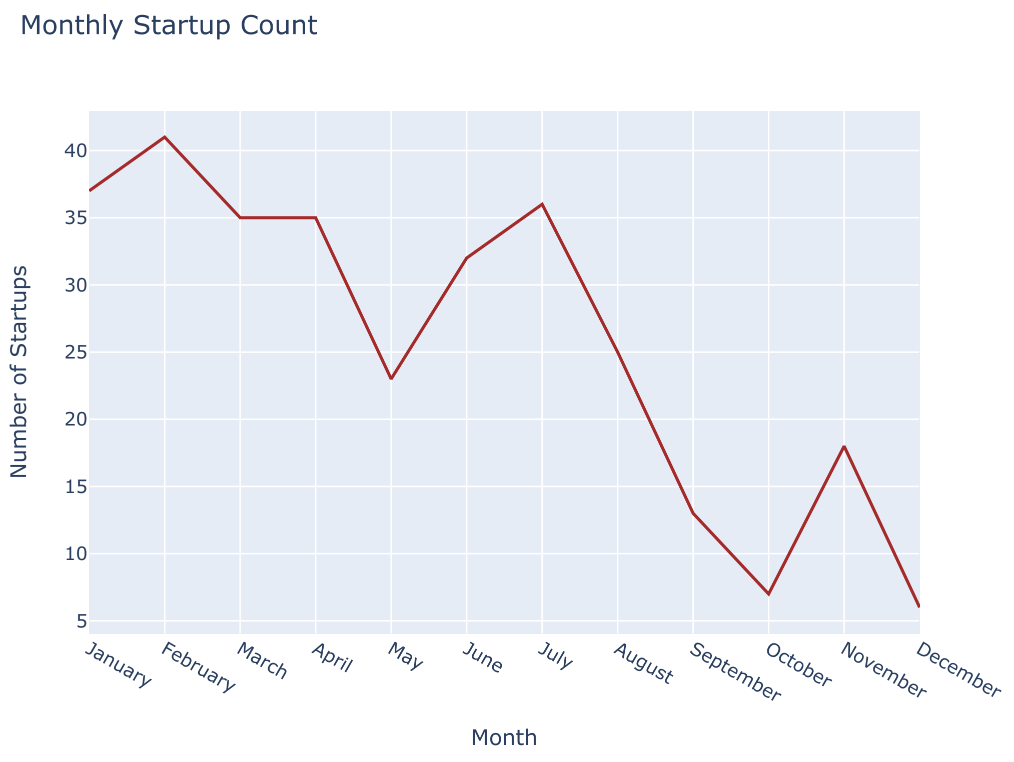
- Shape = (309, 10)
- Unique Industry = 38
- Unique Sub_Industry = 268
- Unique Location = 29
- Unique Investment_Type = 23

Graphs

In [10]:
```
# Total Startup count in 2018
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_2018))

fig.update_layout(title_text="Startup Count in 2018")
fig.show()
```



```
In [11]: # Month wise startups
         import plotly.express as px

         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_2018)
```



```
In [12]: # 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_2018)
```

In [13]:

```
# 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_2018)
```

In [14]:
```python
# 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_2018)
```

In [19]:
```python
import pandas as pd
import ipywidgets as widgets
from IPython.display import display, HTML
import plotly.graph_objects as go


def create_table(selected_industry):
    if selected_industry == 'All':
        display(HTML("<p>Select an industry to view the table.</p>"))
    else:
        filtered_df = df_2018[df_2018['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)

def update_subindustry_options(change):
    pass

industry_dropdown = widgets.Dropdown(options=['All'] + sorted(df_2018['Industry'].dropna().unique()), value='Al
industry_dropdown.observe(update_subindustry_options, names='value')
```

```
widgets.interactive(create_table, selected_industry=industry_dropdown)
```

Out[19]:

```
interactive(children=(Dropdown(description='Industry:', options=('All', 'Agriculture', 'Automation', 'Automobi…
```

In [16]:

```python
# 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_2018)
```

In [17]:

```python
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_2018[df_2018['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_2018['Location'].dropna().unique()), value='Al
widgets.interactive(create_table, selected_location=location_dropdown)
```

Out[17]:

```
interactive(children=(Dropdown(description='Location:', options=('All', 'Ahemadabad', 'Bangalore', 'Bhubaneswa…
```

In [ ]:

```
```