Project Title	**COVID-19 Clinical Trials EDA Pandas**
Tools	Python, ML, SQL, Excel
Domain	Data Analyst & Data scientist
Project Difficulties level	intermediate

Dataset : Dataset is available in the given link. You can download it at your convenience.

Click here to download data set

About Dataset

Dataset Description

ClinicalTrials.gov is a database of privately and publicly funded clinical studies conducted around the world. It is maintained by the National Institute of Health. All data is publicly available and the site provides a direct download feature which makes it super easy to use relevant data for analysis.

This dataset consists of clinical trials related to COVID 19 studies presented on the site.

The dataset consists of XML files where each XML file corresponds to one study. The filename is the NCT number which is a unique identifier of a study in the ClinicalTrials repository. Additionally, a CSV file has also been provided, which might not have as much information as contained in the XML file, but does give sufficient information.

Please refer to this notebook for details on the dataset :
https://www.kaggle.com/parulpandey/eda-on-covid-19-clinical-trials

Acknowledgements

ClinicalTrials.gov is a resource provided by the U.S. National Library of Medicine.

IMPORTANT:

Listing a study does not mean it has been evaluated by the U.S. Federal Government. Read our disclaimer for details.
Before participating in a study, talk to your health care provider and learn about the risks and potential benefits.

NOTE :

1. this project is only for your guidance, not exactly the same you have to create. Here I am trying to show the way or idea of what steps you can follow and how your projects look. Some projects are very advanced (because it will be made with the help of flask, nlp, advance ai, advance DL and some advanced things) which you can not understand .

2. You can make or analyze your project with yourself, with your idea, make it more creative from where we can get some information and understand about

our business. make sure what overall things you have created all things you understand very well.

Example: You can get the basic idea how you can create a project from here

Here's a step-by-step guide for performing Exploratory Data Analysis (EDA) on a COVID-19 Clinical Trials dataset using Pandas, tailored for beginners.

Project Title:

Exploratory Data Analysis of COVID-19 Clinical Trials

1. Objective

The objective is to explore the dataset to gain insights into the characteristics of COVID-19 clinical trials, such as their status, phases, study designs, and demographics.

2. Importing Libraries and Loading Data

First, you'll need to import the necessary libraries and load your dataset.

```
import pandas as pd

# Load the dataset
df = pd.read_csv('covid_clinical_trials.csv')  # Replace with
your dataset's path
```

3. Initial Data Exploration

Start by exploring the basic structure and content of the dataset.

```
# View the first few rows of the dataset
print(df.head())

# Check the columns and data types
print(df.info())

# Summary statistics for numerical columns
print(df.describe())

# Summary statistics for categorical columns
print(df.describe(include='object'))
```

4. Handling Missing Data

Check for missing values and decide how to handle them.

```
# Check for missing values
print(df.isnull().sum())

# Drop columns with a high percentage of missing values or fill
them
df = df.drop(columns=['Acronym', 'Study Documents'])  # Example
of dropping columns
df['Results First Posted'].fillna('Unknown', inplace=True)  #
```

```
Example of filling missing data
```

5. Univariate Analysis

Analyze each column individually to understand the distribution and key characteristics.

- **Status Distribution**: Analyze the status of clinical trials (e.g., Completed, Ongoing).

```
print(df['Status'].value_counts())
df['Status'].value_counts().plot(kind='bar', title='Status of Clinical Trials')
```

- **Phase Distribution**: Understand the distribution of trial phases.

```
print(df['Phases'].value_counts())
df['Phases'].value_counts().plot(kind='bar',
title='Distribution of Phases')
```

- **Age Group Analysis**: Analyze the distribution of age groups.

```
print(df['Age'].value_counts())
df['Age'].value_counts().plot(kind='bar', title='Age Group Distribution')
```

6. Bivariate Analysis

Explore relationships between different variables.

- **Status vs. Phases**: Explore how trial phases are distributed across different statuses.

```
status_phase = pd.crosstab(df['Status'], df['Phases'])
print(status_phase)
status_phase.plot(kind='bar', stacked=True, title='Status vs.
Phases')
```

- **Conditions vs. Outcome Measures**: Understand the common outcome measures for different conditions.

```
conditions_outcomes = df.groupby('Conditions')['Outcome
Measures'].apply(lambda x: ', '.join(x)).reset_index()
print(conditions_outcomes)
```

7. Time Series Analysis

Analyze the trends over time, such as the number of trials started over the months.

```
# Convert date columns to datetime
df['Start Date'] = pd.to_datetime(df['Start Date'],
errors='coerce')
df['Primary Completion Date'] = pd.to_datetime(df['Primary
Completion Date'], errors='coerce')
```

```
# Plot the number of trials started over time
df['Start
Date'].dt.to_period('M').value_counts().sort_index().plot(kind=
'line', title='Trials Started Over Time')
```

8. Conclusion

Summarize the findings from your EDA. For example:

- The majority of trials are in the "Completed" phase.
- Most trials target adult populations.
- There's a steady increase in the number of trials over time.

9. Saving Results

You can save the processed data or specific analysis results for further use.

```
# Save the cleaned data
df.to_csv('cleaned_covid_clinical_trials.csv', index=False)
```

10. Output and Visuals

After running the code, you should observe:

- Bar charts showing the distribution of trial statuses, phases, and age groups.
- A time series plot illustrating the trend of trials over time.

This project will provide a solid foundation in EDA using Pandas, with practical insights into the clinical trials landscape for COVID-19.

Example: You can get the basic idea how you can create a project from here

Sample code with output

Import Required Libraries¶

In [1]:

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

Load The DataSet

In [2]:

```
df =
pd.read_csv('../input/covid19-clinical-trials-dataset/COVID
clinical trials.csv' , index_col = 0)
```

Exploratory Data Analysis

In [3]:

```
# print the first 5 rows in the dataset
df.head(n = 5)
```

Out[3]:

	NCTNumber	Title	Acronym	Status	StudyResults	Conditions	Interventions	Outcome Measures	Sponsor/Collaborators	Gender	...	OtherIDs	StartDate	PrimaryCompletionDate	CompletionDate	FirstPosted	ResultsFirstPosted	LastUpdatePosted	Locations	StudyDocuments	URL
Ran																					

k																				
1	NCT04785898	Diagnostic Performance of the IDNow™	COVID-IDNow	Active, not recruiting	NoResultsAvailable	Covid 19	Diagnostic Test: IDNow™ COVID-19	Evaluate the diagnostic performance of the ID ...	Groupe Hospitalier Paris Saint Joseph	All	. .	COVID-IDNow	November 9, 2020	December 22, 2020	April 30, 2021	March 8, 2021	NaN	March 8, 2021	Groupe Hospitalier Paris Saint-Joseph	NaN	https://ClinicalTrials.gov/show/NCT04785898

		COVID-19...					Screening Test											eph, Paris,...		
2	NCT04595136	Study to Evaluate the Effic	COVID-19	Not yet recruiting	No Results Available	SARS-CoV-2 Infection	Drug: Drug COVID19-	Change on viral load results from baseline aft..	United Medical Specialties	All	..	COVID19-001-USR	November 2, 2020	December 15, 2020	January 29, 2021	October 20, 2020	NaN	October 20, 2020	Clinical, Barranquill	NaN	https://ClinicalTrials.gov/show/NCT04595136

		acyofCOVID19-0001...			le		0001-USR|Drug:normalsaline	.											a,Atlantico,Colombia		
3	NCT0	Lung	TAC-	Recr	NoRe	covid19	Other:	A qualitative	University of	All	..	TAC-	May7	June	June	May2	NaN	Nove	Ospe	NaN	https://ClinicalTrials.gov/show/NCT0

	4395482	CTScanAnalysisofSARS-CoV2InducedLungCTscananalysisinCOVID-19patient	COVID19	uiting	sultsAvailable		Lung CT scan analysis of parenchymal lung damage in COVID-19 patient	analysis of parenchymal lung damage in COVID-19 patient	Milano Bicocca			COVID19	,	2020	15, 2021	15, 2021	0, 2020		mber 9, 2020	dalePapaGiovanniXXIII, Bergamo, Italy| P		

		un...					s												...		
4	NCT04416061	The Role of a Private Hospital in Hon	COVID-19	Active, not recruiting	No Results Available	COVID	Diagnostic Test: COVID19 Diagno	Proportion of asymptomatic subjects|Proportion...	Hong Kong Sanatorium & Hospital	All	...	RC-2020-08	May 25, 2020	July 31, 2020	August 31, 2020	June 4, 2020	NaN	June 4, 2020	Hong Kong Sanatorium & Hospit	NaN	https://ClinicalTrials.gov/show/NCT04416061

		gKongAm...					stic Test												al, HongKong, Ho...		
5	NCT04395924	Maternal-foetal Tra	TMF-COVID-1	Recruiting	NoResultsAv	Maternal Fetal Infection Transmis	Diagnostic Test:	COVID-19 by positive PCR incor	Centre Hospitalier Régional d'Orléans|	Female	. .	CHRO-2020-1	May 5, 2020	May 2021	May 2021	May 20, 2020	NaN	June 4, 2020	CHROrléans, Orl	NaN	https://ClinicalTrials.gov/show/NCT04395924

		nsmissionofSARS-Cov-2	9		ailable	sion|COVID-19...	DiagnosisofSARS-Cov2byRT-...	dblood and/o...	Centred...			0							éans, France		

5 rows × 26 columns

In [4]:

```
# Shape of the DataSet
```

```
df.shape
```

```
Out[4]:

(5783, 26)

In [5]:
# Columns in the dataset
df.columns
```

```
Out[5]:
Index(['NCT Number', 'Title', 'Acronym', 'Status', 'Study
Results',
       'Conditions', 'Interventions', 'Outcome Measures',
       'Sponsor/Collaborators', 'Gender', 'Age', 'Phases',
'Enrollment',
       'Funded Bys', 'Study Type', 'Study Designs', 'Other
IDs', 'Start Date',
       'Primary Completion Date', 'Completion Date', 'First
Posted',
       'Results First Posted', 'Last Update Posted',
'Locations',
       'Study Documents', 'URL'],
```

```
      dtype='object')
```

In [6]:

```
# Categorical Features
df.select_dtypes(include = 'object').columns
```

Out[6]:

```
Index(['NCT Number', 'Title', 'Acronym', 'Status', 'Study
Results',
       'Conditions', 'Interventions', 'Outcome Measures',
       'Sponsor/Collaborators', 'Gender', 'Age', 'Phases',
'Funded Bys',
       'Study Type', 'Study Designs', 'Other IDs', 'Start
Date',
       'Primary Completion Date', 'Completion Date', 'First
Posted',
       'Results First Posted', 'Last Update Posted',
'Locations',
       'Study Documents', 'URL'],

      dtype='object')
```

In [7]:

```
# Neumrical Features
df.select_dtypes(exclude = 'object').columns
```

Out[7]:

```
Index(['Enrollment'], dtype='object')
```

In [8]:

```
# Detecting (Percentage) Missing Data
missing_data = df.isnull().mean() * 100
missing_data
```

Out[8]:

```
NCT Number                 0.000000
Title                      0.000000
Acronym                   57.115684
Status                     0.000000
Study Results              0.000000
Conditions                 0.000000
Interventions             15.320768
Outcome Measures           0.605222
Sponsor/Collaborators      0.000000
Gender                     0.172921
Age                        0.000000
Phases                    42.555767
Enrollment                 0.587930
```

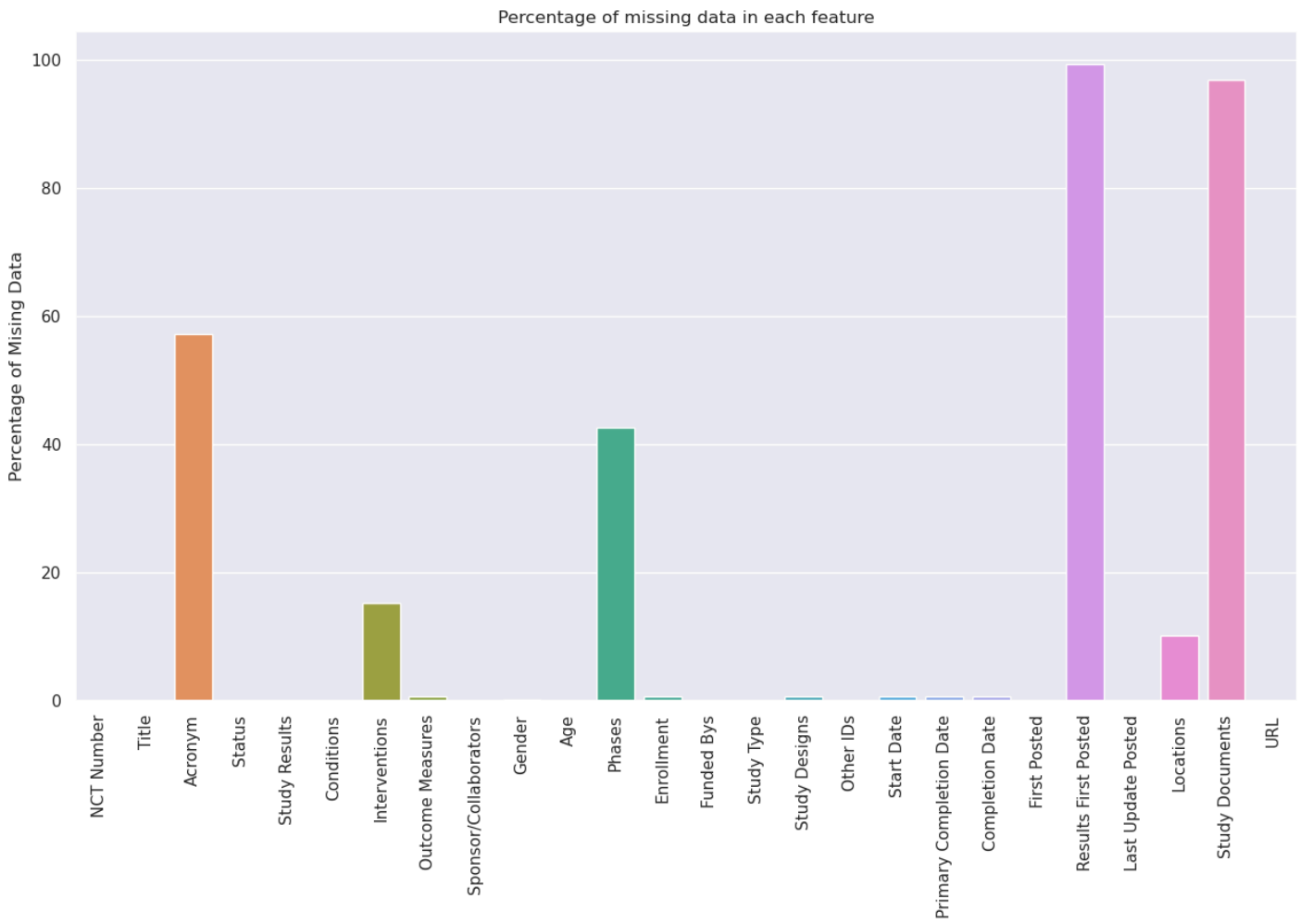
```
Funded Bys                      0.000000
Study Type                      0.000000
Study Designs                   0.605222
Other IDs                       0.017292
Start Date                      0.587930
Primary Completion Date         0.622514
Completion Date                 0.622514
First Posted                    0.000000
Results First Posted           99.377486
Last Update Posted              0.000000
Locations                      10.115857
Study Documents                96.852845
URL                             0.000000

dtype: float64
```

In [9]:

```
# Visualize data without calculating
def visualize_data(data , caption = '' , ylabel = 'Percentage
of Mising Data'):

    # set figure size
    sns.set(rc={'figure.figsize':(15,8.27)})
    # make ticks vertical
    plt.xticks(rotation=90)
```

```
    # set title to the image and plot it or the highest 40
    fig = sns.barplot(x = data.keys()[:min(40 ,
len(data))].tolist() , y = data.values[: min(40 ,
len(data))].tolist()) \
    .set_title(caption)

    # set labels
    plt.ylabel(ylabel)

    plt.show()
```

```
In [10]:
visualize_data(missing_data , 'Percentage of missing data in
each feature')
```



As shown the percentae of missing data in **Results First Posted** is **99.3%** and **Study Documents** is **96.8%**, so it's impossible to impute them without destoying our dataset.

In [11]:

```
# Drop Study Documents and Results First Posted
df.drop(['Results First Posted' , 'Study Documents'] , inplace = True , axis = 1 )
```

In [12]:

```
# Columns in the dataset after dropping Study Documents and
Results First Posted
df.columns
```

```
Out[12]:
Index(['NCT Number', 'Title', 'Acronym', 'Status', 'Study
Results',
       'Conditions', 'Interventions', 'Outcome Measures',
       'Sponsor/Collaborators', 'Gender', 'Age', 'Phases',
'Enrollment',
       'Funded Bys', 'Study Type', 'Study Designs', 'Other
IDs', 'Start Date',
       'Primary Completion Date', 'Completion Date', 'First
Posted',
       'Last Update Posted', 'Locations', 'URL'],

      dtype='object')
```

```
In [13]:
# Drop Duplicate Rows
print(f"Shape before dropping duplicates data {df.shape}")
df.drop_duplicates(inplace = True)
print(f"Shape after dropping duplicates data {df.shape}")
```

```
Shape before dropping duplicates data (5783, 24)
Shape after dropping duplicates data (5783, 24)
```

There is no duplicate rows in the dataset.

In [14]:

```
# Drop rows that have less than 10 non-null values
print(f"Shape before dropping Null rows {df.shape}")
df.dropna(how = 'any' , axis = 0 , thresh = 10 , inplace =
True)
print(f"Shape after dropping Null rows {df.shape}")
```

```
Shape before dropping Null rows (5783, 24)
Shape after dropping Null rows (5783, 24)
```

There is no rows with less than 10 non-null values

In [15]:

```
df.isnull().mean() * 100
```

Out[15]:

```
NCT Number                    0.000000
Title                         0.000000
```

```
Acronym                    57.115684
Status                      0.000000
Study Results               0.000000
Conditions                  0.000000
Interventions              15.320768
Outcome Measures            0.605222
Sponsor/Collaborators       0.000000
Gender                      0.172921
Age                         0.000000
Phases                     42.555767
Enrollment                  0.587930
Funded Bys                  0.000000
Study Type                  0.000000
Study Designs               0.605222
Other IDs                   0.017292
Start Date                  0.587930
Primary Completion Date     0.622514
Completion Date             0.622514
First Posted                0.000000
Last Update Posted          0.000000
Locations                  10.115857
URL                         0.000000

dtype: float64

In [16]:
```

```
# We can extract a new feature form The Location which is the
country where the study hold
countries = [ str(df.Locations.iloc[i]).split(',')[-1] for i in
range(df.shape[0])]
df['Country'] = countries
```

In [17]:

```
df.columns
```

Out[17]:

```
Index(['NCT Number', 'Title', 'Acronym', 'Status', 'Study
Results',
       'Conditions', 'Interventions', 'Outcome Measures',
       'Sponsor/Collaborators', 'Gender', 'Age', 'Phases',
'Enrollment',
       'Funded Bys', 'Study Type', 'Study Designs', 'Other
IDs', 'Start Date',
       'Primary Completion Date', 'Completion Date', 'First
Posted',
       'Last Update Posted', 'Locations', 'URL', 'Country'],

      dtype='object')
```

In [18]:

```
df.Country.value_counts()[:35]
```

```
Out[18]:
United States           1267
 France                  647
nan                      585
 United Kingdom          306
 Italy                   235
 Spain                   234
 Turkey                  219
 Canada                  202
 Egypt                   192
 China                   171
 Brazil                  137
 Germany                 128
 Belgium                  91
 Mexico                   88
 Switzerland              76
 Russian Federation       69
 Sweden                   57
 Denmark                  56
 Israel                   56
 India                    55
 Pakistan                 53
```

```
 Argentina                47
 Netherlands              46
 Norway                   38
 Hong Kong                36
 Colombia                 33
 Republic of              31
 Austria                  29
 Poland                   29
 Singapore                29
 Saudi Arabia             27
 Australia                26
 Greece                   26
 Islamic Republic of      23
 South Africa             22

Name: Country, dtype: int64
```

Now We need to clasify the missing data to one of these categories

```
1) Missing Completely At Random (MCAR)
```

```
2) Missing At Random (MAR)
```

```
3) Not Missing At Random (NMAR)
```

```
In [19]:
# Lets's start with Acronym
```

```
print(f"Number of unique values is {df.Acronym.nunique()} \n")
df.Acronym.value_counts()
```

```
Number of unique values is 2338
```

```
Out[19]:
COVID-19        47
PROTECT          7
CORONA           6
RECOVER          5
SCOPE            5
                ..
ASD              1
VICO             1
LICORNE          1
LOSVID           1
MindMyMindFU     1

Name: Acronym, Length: 2338, dtype: int64

In [20]:
# Find the realtion between null values in Acronym and Countries
```

```
(df.Acronym.isnull().groupby(df.Country).mean().sort_values(asc
ending = False) * 100)[:60]
```

```
Out[20]:
Country
 Iraq                          100.000000
 Belarus                       100.000000
 Rwanda                        100.000000
 South Sudan                   100.000000
 Cambodia                      100.000000
 Bulgaria                      100.000000
 Cyprus                        100.000000
 Bosnia and Herzegovina        100.000000
 Guinea-Bissau                 100.000000
 Dominican Republic            100.000000
 Ecuador                       100.000000
 North Macedonia               100.000000
 Bahrain                       100.000000
 Azerbaijan                    100.000000
 Uruguay                       100.000000
 Uzbekistan                    100.000000
 Kyrgyzstan                    100.000000
 Cape Verde                    100.000000
 Republic of                    96.774194
```

Taiwan	93.750000
Singapore	93.103448
Japan	88.888889
Kuwait	87.500000
China	87.134503
Turkey	86.757991
Ukraine	85.714286
Malaysia	84.615385
Egypt	83.854167
Hungary	83.333333
Hong Kong	80.555556
Bangladesh	80.000000
India	80.000000
Kazakhstan	80.000000
Saudi Arabia	77.777778
Puerto Rico	76.470588
Israel	75.000000
Zimbabwe	75.000000
Jordan	72.727273
Poland	72.413793
Indonesia	71.428571
United States	69.376480
Romania	69.230769
Kenya	66.666667
Nepal	66.666667

```
 New Zealand                66.666667
 Ethiopia                   66.666667
 Slovakia                   66.666667
 Thailand                   66.666667
 Lebanon                    66.666667
nan                         66.324786
 Islamic Republic of        65.217391
 Russian Federation         65.217391
 Chile                      64.705882
 Austria                    62.068966
 Pakistan                   60.377358
 Brazil                     59.124088
 Mexico                     57.954545
 Sweden                     57.894737
 Argentina                  57.446809
 Canada                     55.940594

Name: Acronym, dtype: float64
```

- After inspecting the relation between the missing values in Acronym and Country we can conclude that there is a sort of relation between these two features, so we can say that Data is Missing At Random (MAR).
- So we can Impute by Missing Category.

```
In [21]:
```

```
# impute by a missing Indicator
df.Acronym = df.Acronym.fillna("Missing Acronym")
```

In [22]:

```
# Detecting (Percentage) Missing Data
df.isnull().mean() * 100
```

Out[22]:

```
NCT Number                 0.000000
Title                      0.000000
Acronym                    0.000000
Status                     0.000000
Study Results              0.000000
Conditions                 0.000000
Interventions             15.320768
Outcome Measures           0.605222
Sponsor/Collaborators      0.000000
Gender                     0.172921
Age                        0.000000
Phases                    42.555767
Enrollment                 0.587930
Funded Bys                 0.000000
Study Type                 0.000000
Study Designs              0.605222
Other IDs                  0.017292
```

```
Start Date                    0.587930
Primary Completion Date       0.622514
Completion Date               0.622514
First Posted                  0.000000
Last Update Posted            0.000000
Locations                    10.115857
URL                           0.000000
Country                       0.000000

dtype: float64
```

We can do the same for other categorical features such as Interventions , Phases , Locations and other categorical features

In [23]:

```
# Impute Interventions , Phases , Locations by Missing Category


categorical_features = df.select_dtypes(include =
object).columns


features =
categorical_features[df[categorical_features].isnull().mean() >
0]


for feature in features:
    df[feature] = df[feature].fillna(f"Missing {feature}")
```

In [24]:

```
# Detecting (Percentage) Missing Data
df.isnull().mean() * 100
```

Out[24]:

NCT Number	0.00000
Title	0.00000
Acronym	0.00000
Status	0.00000
Study Results	0.00000
Conditions	0.00000
Interventions	0.00000
Outcome Measures	0.00000
Sponsor/Collaborators	0.00000
Gender	0.00000
Age	0.00000
Phases	0.00000
Enrollment	0.58793
Funded Bys	0.00000
Study Type	0.00000
Study Designs	0.00000
Other IDs	0.00000

```
Start Date                    0.00000
Primary Completion Date       0.00000
Completion Date               0.00000
First Posted                  0.00000
Last Update Posted            0.00000
Locations                     0.00000
URL                           0.00000
Country                       0.00000

dtype: float64
```

Now the Time to handle The missing data for the Enrollment

In [25]:

```
# Check the skewness
df.Enrollment.skew()
```

Out[25]:

```
34.06593382031148
```

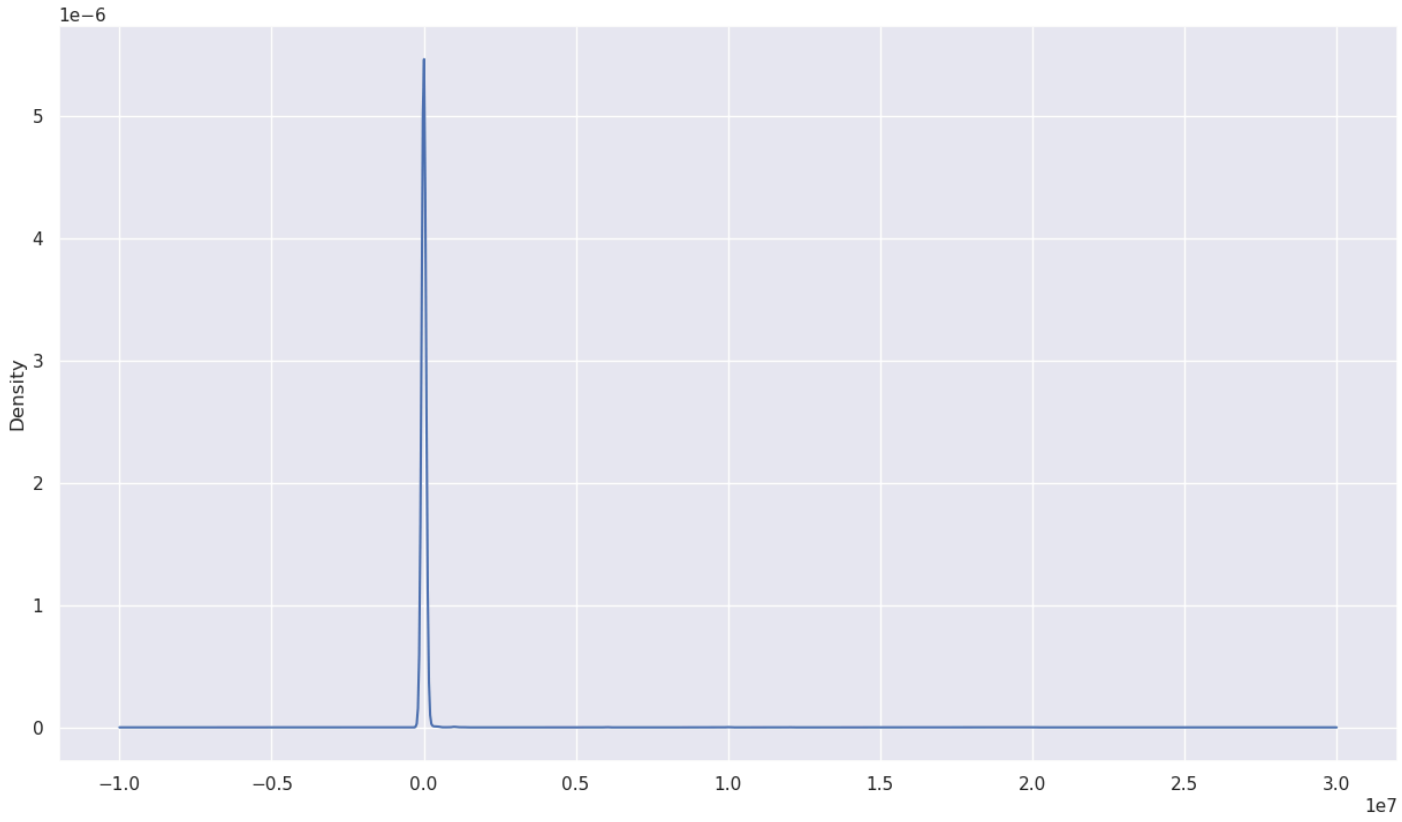
The value of Skewness is 34 which means that we This feature isn't normally distributed

In [26]:

```
# Plotting the distribution of the enrollment
df.Enrollment.plot(kind = 'kde')
```

Out[26]:

```
<AxesSubplot:ylabel='Density'>
```



So We will impute by the median

In [27]:

```
# Some Statstical Valuse for the Enrollment Column


min_Value = df.Enrollment.min()
max_Value = df.Enrollment.max()
mean_Value = df.Enrollment.mean()
```

```
median_Value = df.Enrollment.median()
std_Value = df.Enrollment.std()

print(f"the min value is {min_Value} \n \
The max value is {max_Value} \n \
The mean is {mean_Value} \n \
The Median is {median_Value} \n \
Standard Devation is {std_Value}")
```

```
the min value is 0.0
 The max value is 20000000.0
 The mean is 18319.48860671421
 The Median is 170.0
 Standard Devation is 404543.7287841079
```

In [28]:

```
# Using Median to impute Missing Values
df.Enrollment = df.Enrollment.fillna(median_Value)
```

In [29]:

```
# Detecting (Percentage) Missing Data
df.isnull().mean() * 100
```

Out[29]:

NCT Number	0.0
Title	0.0
Acronym	0.0
Status	0.0
Study Results	0.0
Conditions	0.0
Interventions	0.0
Outcome Measures	0.0
Sponsor/Collaborators	0.0
Gender	0.0
Age	0.0
Phases	0.0
Enrollment	0.0
Funded Bys	0.0
Study Type	0.0
Study Designs	0.0
Other IDs	0.0
Start Date	0.0
Primary Completion Date	0.0
Completion Date	0.0
First Posted	0.0
Last Update Posted	0.0

```
Locations                       0.0
URL                             0.0
Country                         0.0

dtype: float64
```

In [30]:

```
df.head()
```

Out[30]:

	NCTNumber	Title	Acronym	Status	StudyResults	Conditions	Interventions	OutcomeMeasures	Sponsor/Collaborators	Gender	...	StudyDesigns	OtherIDs	StartDate	PrimaryCompletionD	CompletionDate	FirstPosted	LastUpdatePoste	Locations	URL	Country

														ate			d			
Rank																				
1	NCT04785898	Diagnostic Performance	COVID-IDNow	Active, not recruit	No Results Available	Covid 19	Diagnostic Test: IDNow™	Evaluate the diagnostic performance of the ID ...	Groupe Hospitalier Paris Saint Joseph	All	...	Allocation: N/A| Intervention Model: Single Gro...	COVID-IDNow	November 9, 2020	December 22, 2020	April 30, 2021	March 8, 2021	March 8, 2021	Groupe Hospitalier Pa	https://ClinicalTrials.gov/show/NCT04785898	France

		of the IDNow™ COVID-19...		ing	e		COVID-19 Screening Test														
2	NCT04	Study	COVID	Notye	NoRes	SARS-CoV-2 Inf	Drug:	Change on vir	United Medical	All	...	Allocation: Random	COVID	Novem	Decem	Janua	Octob	Octob	Cimed	https://ClinicalTrials.gov/show/NCT0459513	Coln

	595136	toEvaluatetheEfficacyofCOVID19-0	-19	trecruiting	ultsAvailable	ection	DrugCOVID19-00011-USR|Drug:nor	aloadresultsfrombaselineaft..	Specialties			ized|Intervention Model: Par..	19-0011-USR	ber2,2020	ber15,2020	ry29,2021	er200,2020	er200,2020	ical,Barranquilla,Atlantico,Colombi	6	bia

		001...					malsaline											a		
3	NCT04395482	Lung CT Scan Analysis of SA	TAC-COVID19	Recruiting	No Results Available	covid19	Other: Lung CT scan analysi	A qualitative analysis of parenchymal lung dam...	University of Milano Bicocca	All	...	Observational Model: Cohort| Time Perspective: ...	TAC-COVID19	May 7, 2020	June 15, 2021	June 15, 2021	May 20, 2020	November 9, 2020	Osped ale Papa Giovanni	https://ClinicalTrials.gov/show/NCT04395482	San Marino

		RS-CoV2 Induced Lun...					sin COVID-19 patients											XXIII, Bergamo, Italy | P...		
4	NCT04411	The Role o	COVID-1	Active,	Not Resul	COVID	Diagnostic	Proportion of asymp	Hong Kong Sanatoriu	All	...	Observational Model: Coh	RC-2020	May 25, 2	July 31,	August 3	June 4, 2	June 4, 2	Hong Kon	https://ClinicalTrials.gov/show/NCT04416061	Hong Kon

	6061	faPrivateHospitalinHongKongAm...	9	notrecruiting	tsAvailable		Test:COVID19DiagnosticTest	tomaticsubjects|Proportion...	m&Hospital			ort|TimePerspective: ...	-08	020	2020	1,2020	020	020	gSanatorium&Hospital,HongKong,Ho		g

																		...		
5	NCT04395924	Maternal-foetal Transmission of	TMF-COVID-19	Recruiting	No Results Available	Maternal Fetal Infection Transmission|COVID-19...	Diagnostic Test: Diagnosis of SARS	COVID-19 by positive PCR in cord blood and / o...	Centre Hospitalier Régional d'Orléans| Centred...	Female	...	Observational Model: Cohort| Time Perspective: ...	CHRO-2020-10	May 5, 2020	May 2021	May 2021	May 20, 2020	June 4, 2020	CHR Orléans, Orléans, France	https://ClinicalTrials.gov/show/NCT04395924	France

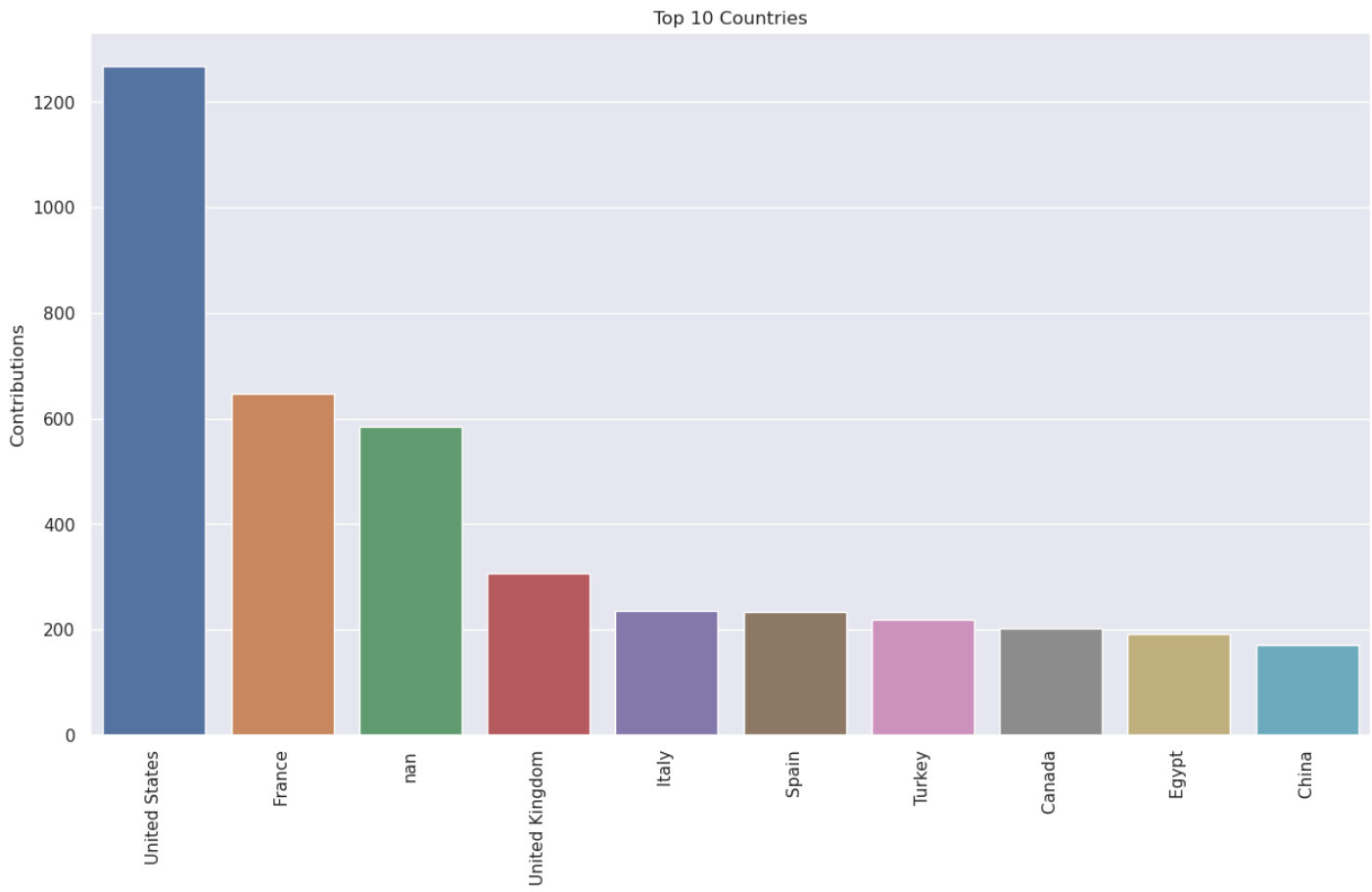
		SARS-Cov-2					-Cov2byRT-..														

5 rows × 25 columns

Data Visualizations

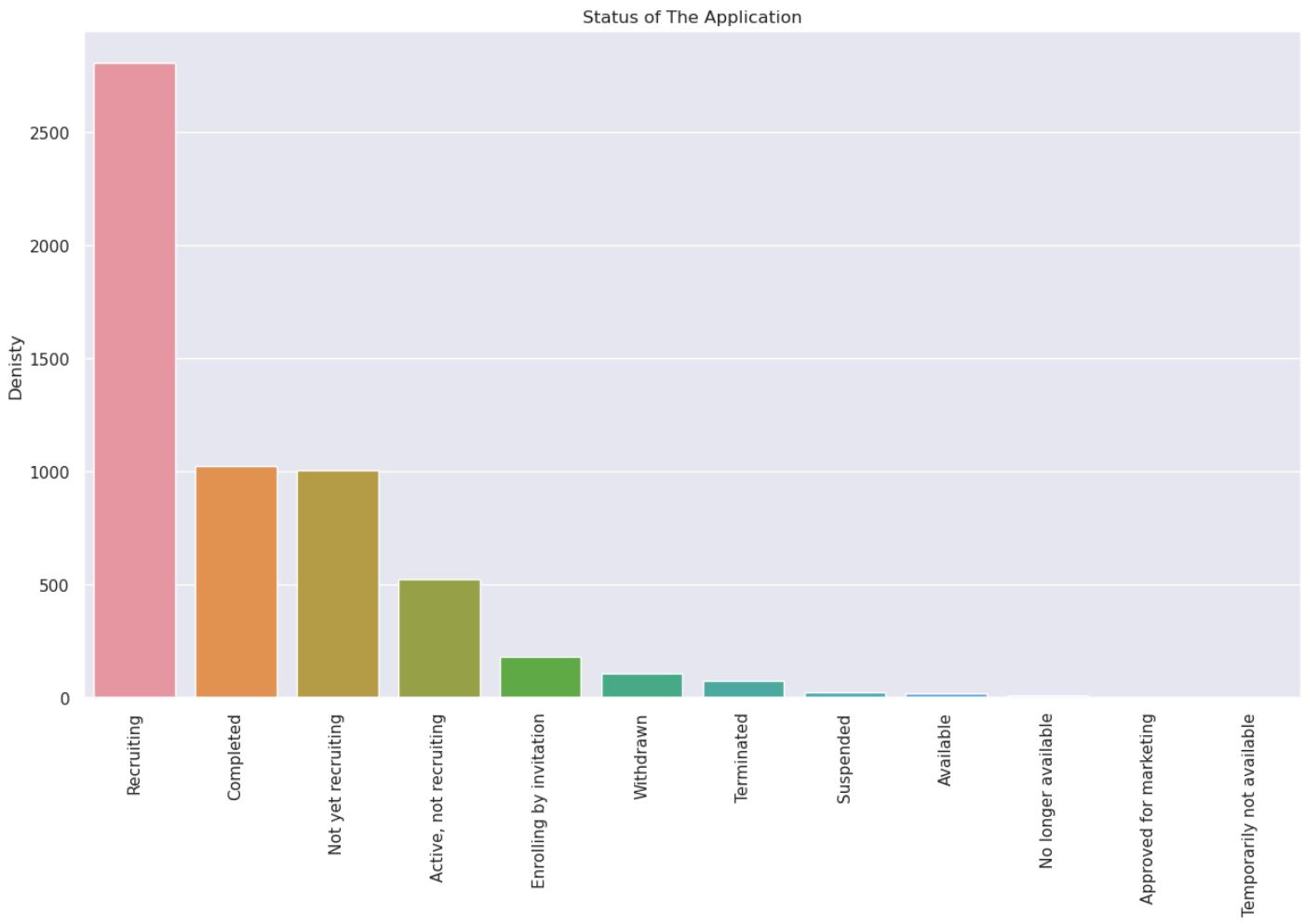
In [31]:

```
# Get Countires with highest Contributiuons
top_10_Countires = df.Country.value_counts()[:10]
visualize_data(top_10_Countires , caption = 'Top 10 Countries'
, ylabel = 'Contributions')
```



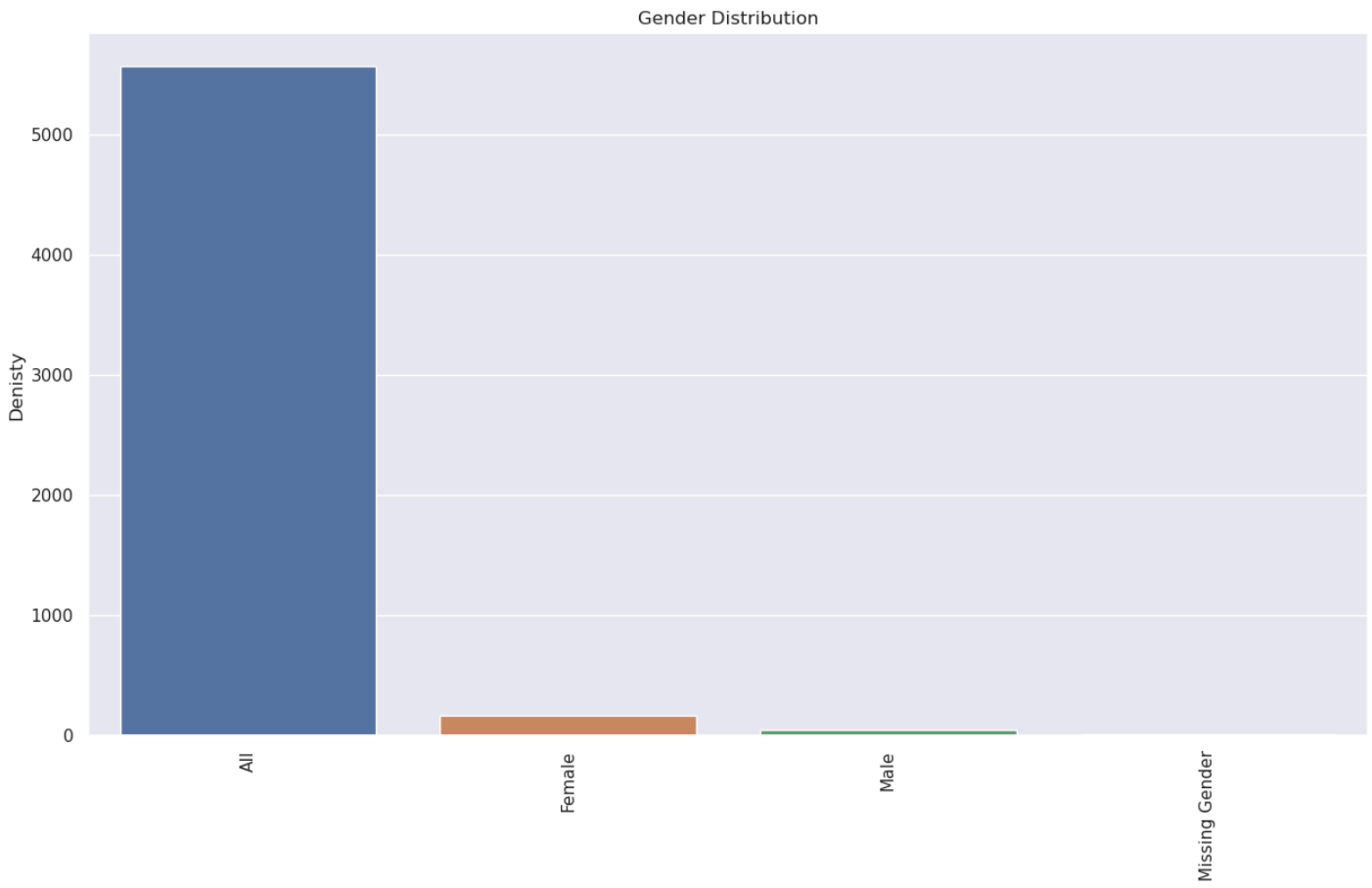
In [32]:

```
# Status of the Application
status = df.Status.value_counts()

visualize_data(status , caption = 'Status of The Application' ,
ylabel = 'Denisty')
```



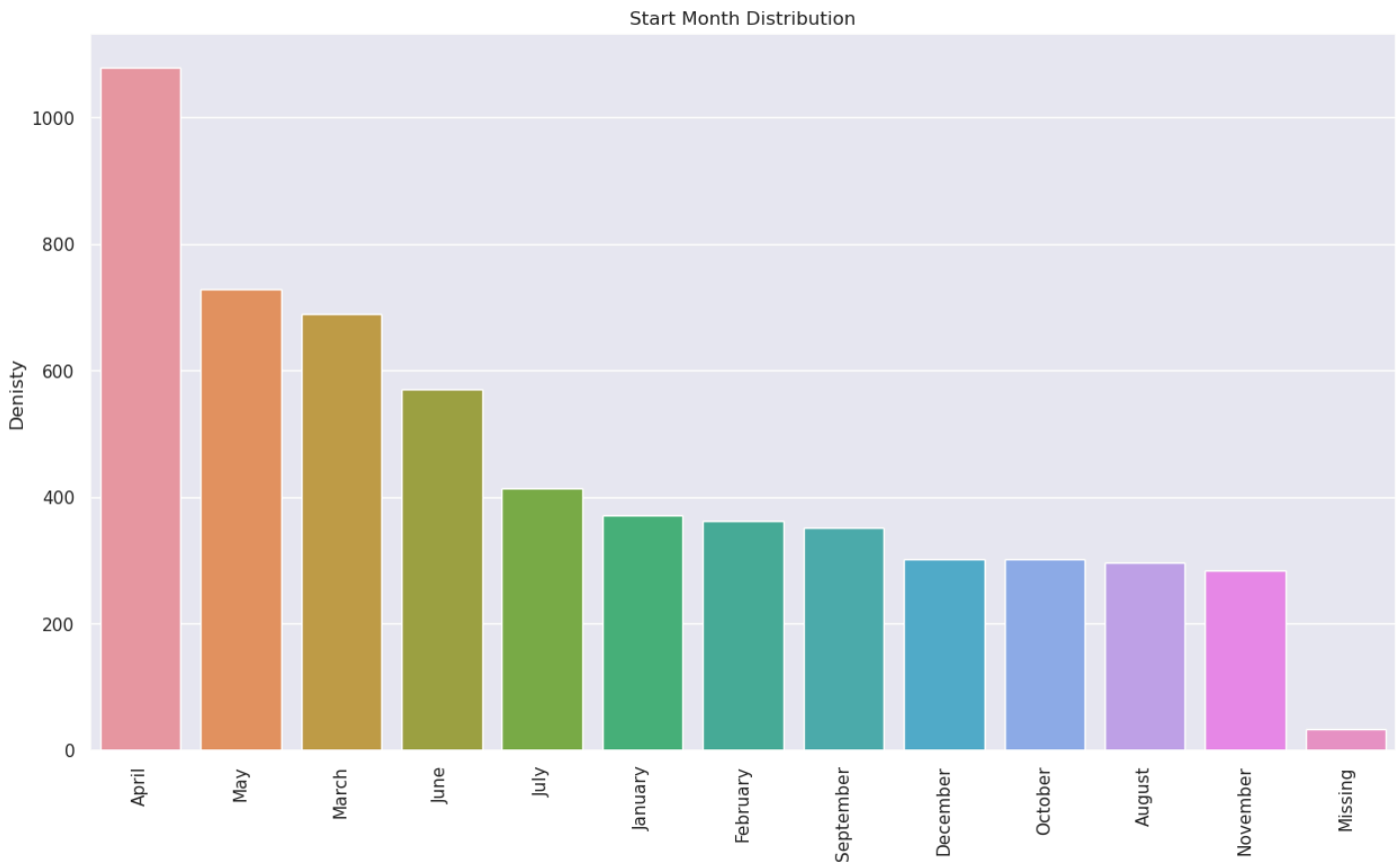
In [33]:

```
# Gender Visualiztions
gender = df.Gender.value_counts()
visualize_data(gender , caption = 'Gender Distribution' ,
ylabel = 'Denisty')
```



In [34]:

```
# Which month has the highest start
start_month = pd.Series([ str(df['Start Date'].iloc[i]).split(' ')[0] for i in range (df.shape[0])])

start_month_Distribution = start_month.value_counts()

visualize_data(start_month_Distribution , caption = 'Start Month Distribution' , ylabel = 'Denisty')
```



In [35]:

```
print(f"The shape of data frame is {df.shape}")
print(f"Nunique in NCT Number is {df['NCT Number'].nunique()}")
print(f"Nunique in URL is {df.URL.nunique()}")
```

```
The shape of data frame is (5783, 25)
Nunique in NCT Number is 5783
Nunique in URL is 5783
```

So If We are going to apply a (Machine Learning) ML model we can drop NCT

Number and URL because there is an index already which is Rank. To reduce the number of categorical Features, Specially because they will need to be doecoded inorder to be used in a ML Model.

Reference link