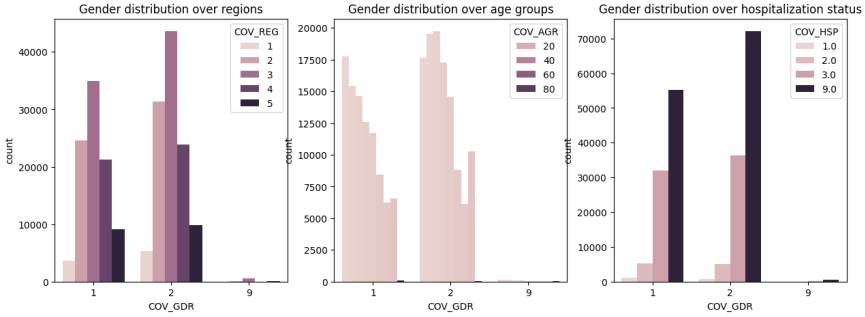
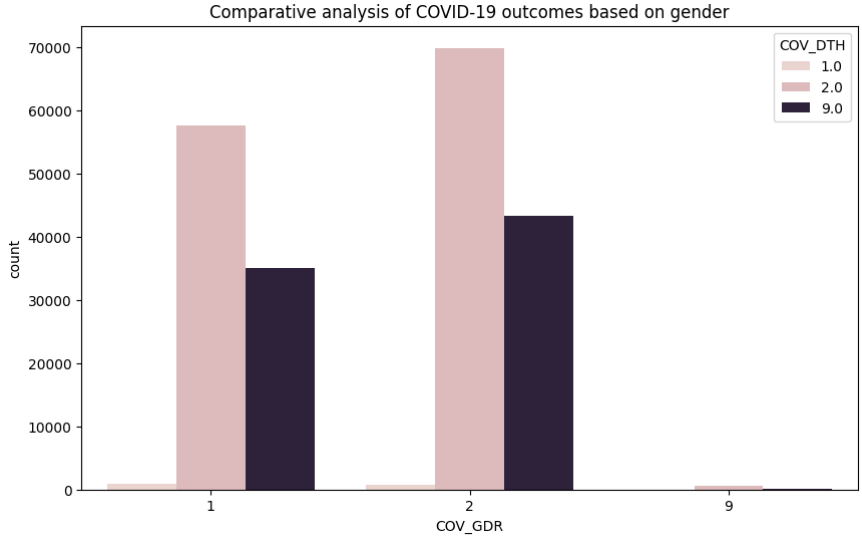
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns



df = pd.read_csv('/COVID19-eng.csv')

# Analysis based on Gender (COV_GDR)
def gender_analysis():
    # Count and percentage distribution of cases by gender
    gender_distribution = df['COV_GDR'].value_counts()
    gender_percentage = df['COV_GDR'].value_counts(normalize=True) * 100

    print("Count and percentage distribution of cases by gender:")
    print(gender_distribution)
    print("\nPercentage distribution:")
    print(gender_percentage)

    # Gender distribution over different regions, age groups, and hospitalization statuses
    plt.figure(figsize=(15, 5))
    plt.subplot(1, 3, 1)
    sns.countplot(data=df, x='COV_GDR', hue='COV_REG')
    plt.title('Gender distribution over regions')

    plt.subplot(1, 3, 2)
    sns.countplot(data=df, x='COV_GDR', hue='COV_AGR')
    plt.title('Gender distribution over age groups')

    plt.subplot(1, 3, 3)
    sns.countplot(data=df, x='COV_GDR', hue='COV_HSP')
    plt.title('Gender distribution over hospitalization status')

    plt.show()

    # Comparative analysis of COVID-19 outcomes based on gender
    plt.figure(figsize=(10, 6))
    sns.countplot(data=df, x='COV_GDR', hue='COV_DTH')
    plt.title('Comparative analysis of COVID-19 outcomes based on gender')
    plt.show()

# Analysis based on Hospitalization Status (COV_HSP)
def hospitalization_analysis():
    # Count and percentage distribution of cases based on hospitalization status
    hospitalization_distribution = df['COV_HSP'].value_counts()
    hospitalization_percentage = df['COV_HSP'].value_counts(normalize=True) * 100

    print("Count and percentage distribution of cases based on hospitalization status:")
    print(hospitalization_distribution)
    print("\nPercentage distribution:")
    print(hospitalization_percentage)

    # Hospitalization status distribution across different genders, age groups, and regions
    plt.figure(figsize=(15, 5))
    plt.subplot(1, 3, 1)
    sns.countplot(data=df, x='COV_HSP', hue='COV_GDR')
    plt.title('Hospitalization status distribution across genders')

    plt.subplot(1, 3, 2)
    sns.countplot(data=df, x='COV_HSP', hue='COV_AGR')
    plt.title('Hospitalization status distribution over age groups')

    plt.subplot(1, 3, 3)
    sns.countplot(data=df, x='COV_HSP', hue='COV_REG')
    plt.title('Hospitalization status distribution across regions')

    plt.show()

    # Analyze the relationship between hospitalization status and clinical outcomes
    plt.figure(figsize=(10, 6))
    sns.countplot(data=df, x='COV_HSP', hue='COV_DTH')
    plt.title('Relationship between hospitalization status and clinical outcomes')
    plt.show()

# Analysis based on Age Group (COV_AGR)
def age_group_analysis():
    # Count and percentage distribution of cases across different age groups
    age_group_distribution = df['COV_AGR'].value_counts()
    age_group_percentage = df['COV_AGR'].value_counts(normalize=True) * 100
```

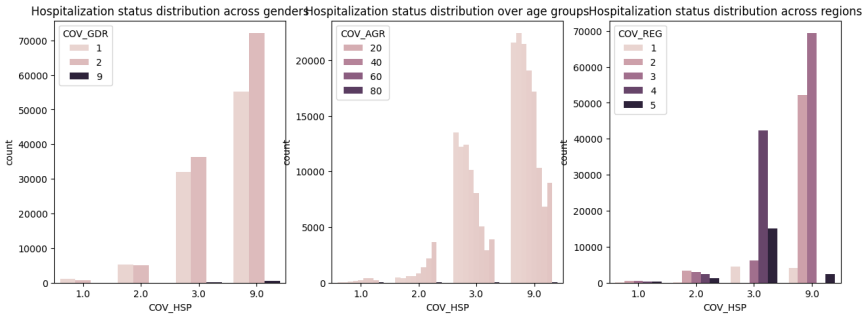
```
    print("Count and percentage distribution of cases across different age groups:")
    print(age_group_distribution)
    print("\nPercentage distribution:")
    print(age_group_percentage)

    # Age group distribution over regions, gender, and hospitalization status
    plt.figure(figsize=(15, 5))
    plt.subplot(1, 3, 1)
    sns.countplot(data=df, x='COV_AGR', hue='COV_REG')
    plt.title('Age group distribution over regions')

    plt.subplot(1, 3, 2)
    sns.countplot(data=df, x='COV_AGR', hue='COV_GDR')
    plt.title('Age group distribution over gender')

    plt.subplot(1, 3, 3)
    sns.countplot(data=df, x='COV_AGR', hue='COV_HSP')
    plt.title('Age group distribution over hospitalization status')

    plt.show()

    # Analyze the correlation between age groups and clinical outcomes
    plt.figure(figsize=(10, 6))
    sns.countplot(data=df, x='COV_AGR', hue='COV_DTH')
    plt.title('Correlation between age groups and clinical outcomes')
    plt.show()

# Analysis based on Region (COV_REG)
def region_analysis():
    # Count and percentage distribution of cases across different regions
    region_distribution = df['COV_REG'].value_counts()
    region_percentage = df['COV_REG'].value_counts(normalize=True) * 100

    print("Count and percentage distribution of cases across different regions:")
    print(region_distribution)
    print("\nPercentage distribution:")
    print(region_percentage)

    # Region-wise distribution of gender, age groups, and hospitalization status
    plt.figure(figsize=(15, 5))
    plt.subplot(1, 3, 1)
    sns.countplot(data=df, x='COV_REG', hue='COV_GDR')
    plt.title('Region-wise distribution of gender')

    plt.subplot(1, 3, 2)
    sns.countplot(data=df, x='COV_REG', hue='COV_AGR')
    plt.title('Region-wise distribution of age groups')

    plt.subplot(1, 3, 3)
    sns.countplot(data=df, x='COV_REG', hue='COV_HSP')
    plt.title('Region-wise distribution of hospitalization status')

    plt.show()

    # Comparative analysis of COVID-19 outcomes across different regions
    plt.figure(figsize=(10, 6))
    sns.countplot(data=df, x='COV_REG', hue='COV_DTH')
    plt.title('Comparative analysis of COVID-19 outcomes across regions')
    plt.show()

# Analysis based on Episode Week (COV_EW and COV_EWG)
def episode_week_analysis():
    # Analyze the distribution of cases over different episode weeks and episode week groups
    plt.figure(figsize=(15, 5))
    plt.subplot(1, 2, 1)
    sns.countplot(data=df, x='COV_EW')
    plt.title('Distribution of cases over different episode weeks')

    plt.subplot(1, 2, 2)
    sns.countplot(data=df, x='COV_EWG')
    plt.title('Distribution of cases over episode week groups')

    plt.show()


gender_analysis()
hospitalization_analysis()
age_group_analysis()
region_analysis()
episode_week_analysis()
```

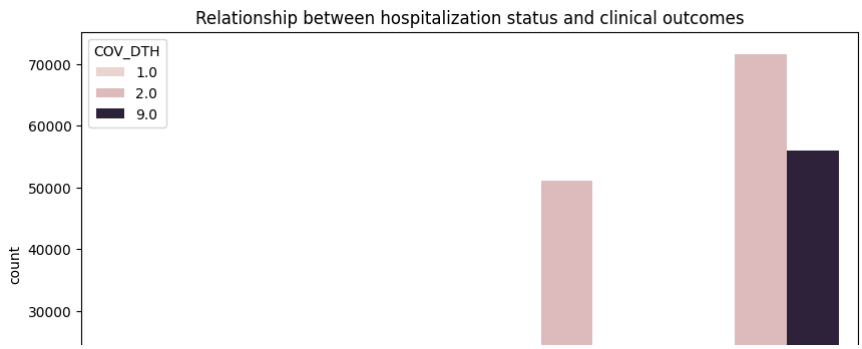
```
Count and percentage distribution of cases by gender:
2    113990
1     93404
9       694
Name: COV_GDR, dtype: int64

Percentage distribution:
2    54.779709
1    44.886779
9     0.333513
Name: COV_GDR, dtype: float64
```

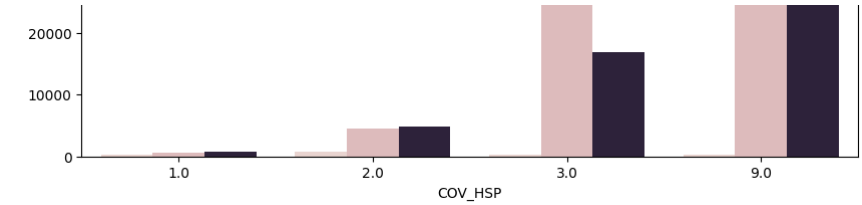




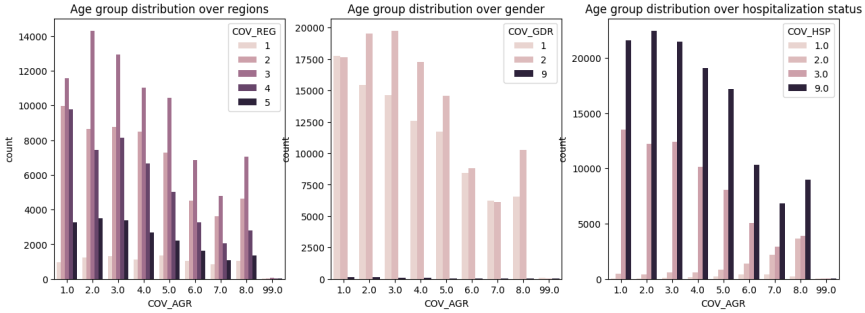

```
Count and percentage distribution of cases based on hospitalization status:
9.0    127936
3.0     68262
2.0     10180
1.0      1709
Name: COV_HSP, dtype: int64

Percentage distribution:
9.0    61.481976
3.0    32.804548
2.0     4.892185
1.0     0.821291
Name: COV_HSP, dtype: float64
```

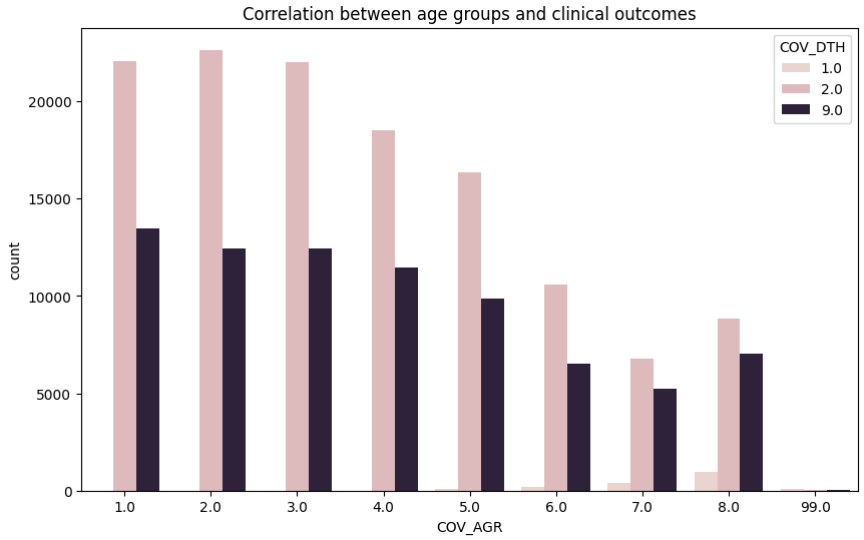




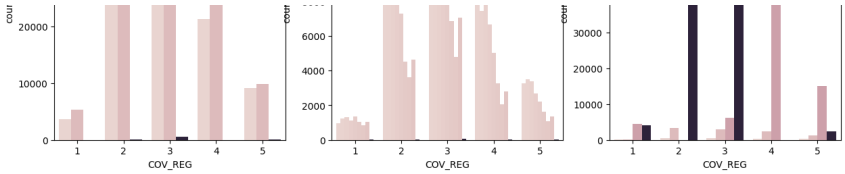


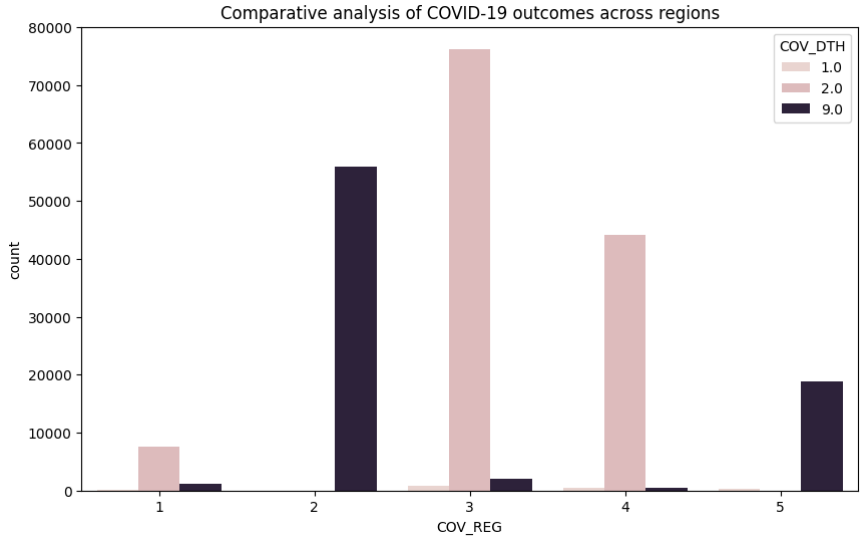

```
Count and percentage distribution of cases across different age groups:
1.0     35577
2.0     35102
3.0     34489
4.0     29974
5.0     26335
6.0     17278
8.0     16814
7.0     12379
99.0      139
Name: COV_AGR, dtype: int64

Percentage distribution:
1.0     17.097176
2.0     16.868906
3.0     16.574317
4.0     14.404552
5.0     12.655764
6.0      8.303258
8.0      8.080274
7.0      5.948954
99.0     0.066799
Name: COV_AGR, dtype: float64
```

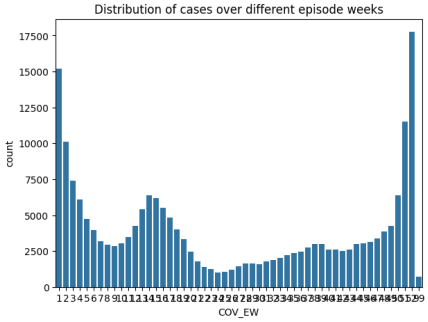


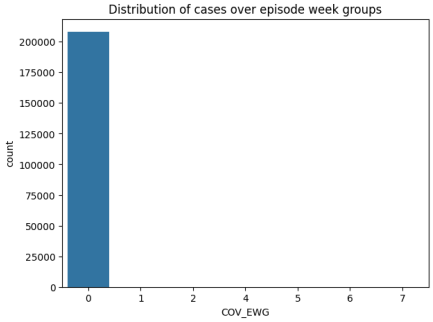



```
Count and percentage distribution of cases across different regions:
3    79004
2    55944
4    45129
5    19101
1     8910
Name: COV_REG, dtype: int64

Percentage distribution:
3    37.966630
2    26.884780
4    21.687459
5     9.179290
1     4.281842
Name: COV_REG, dtype: float64
```


count
COV_REG
count
COV_REG
count
COV_REG

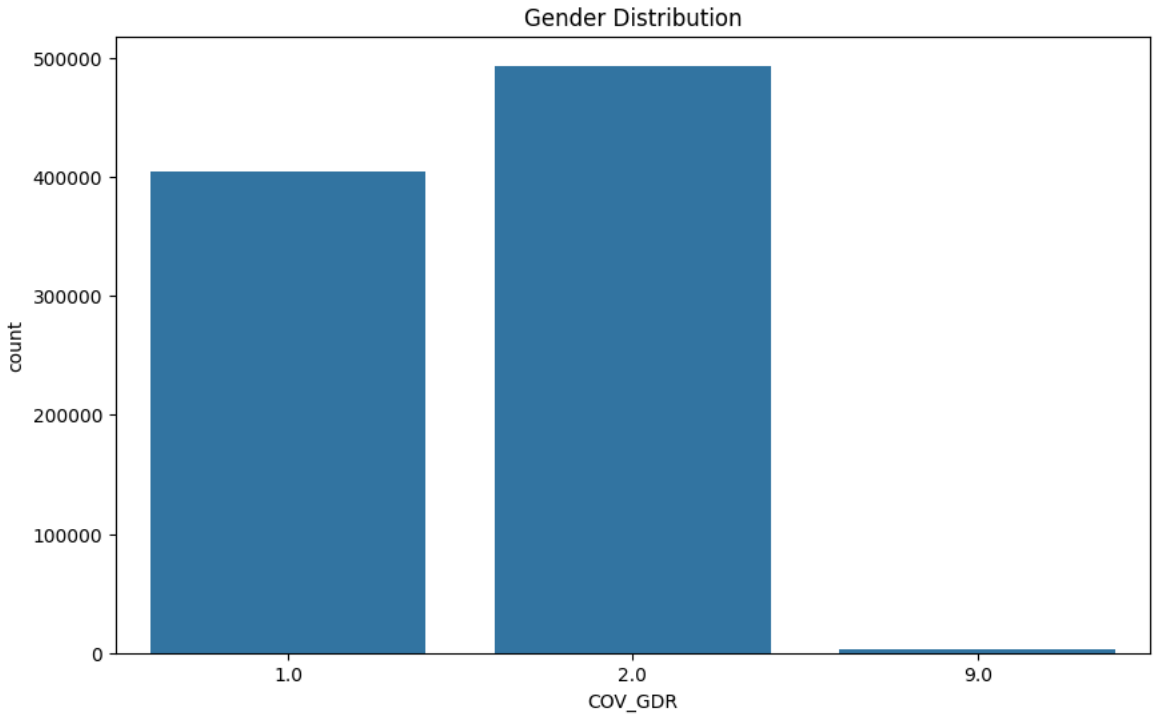

Comparative analysis of COVID-19 outcomes across regions
COV_DTH
1.0
2.0
9.0
count
COV_REG

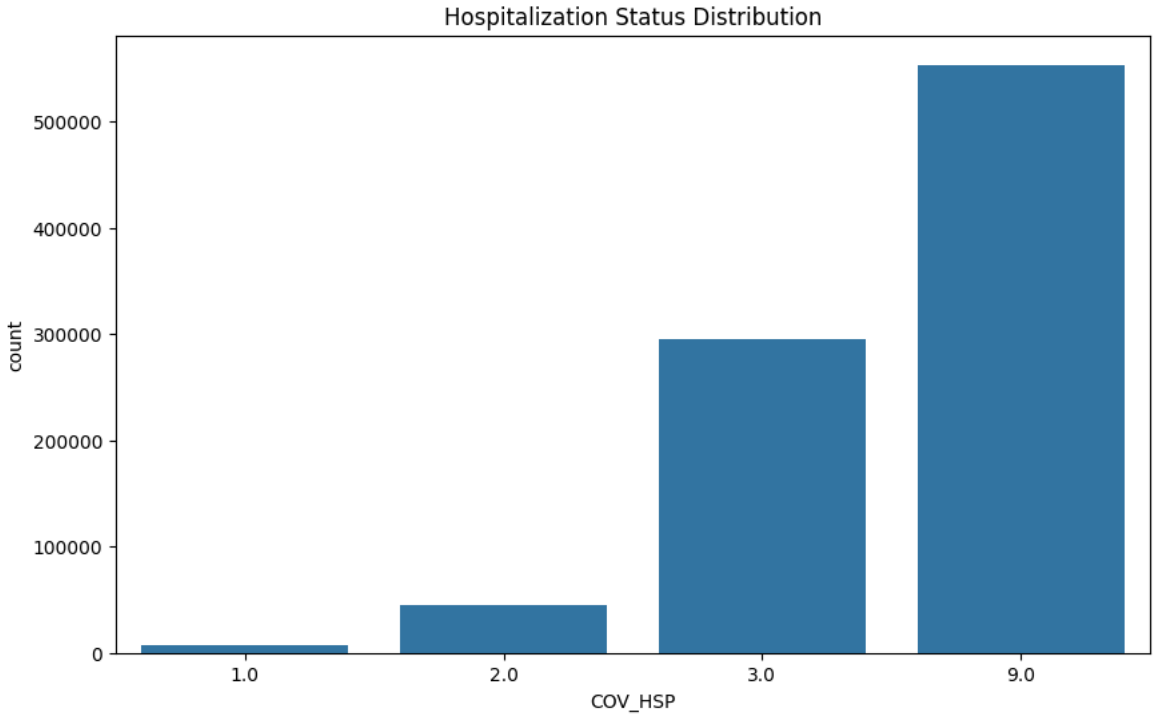

Distribution of cases over different episode weeks
count
COV_EW

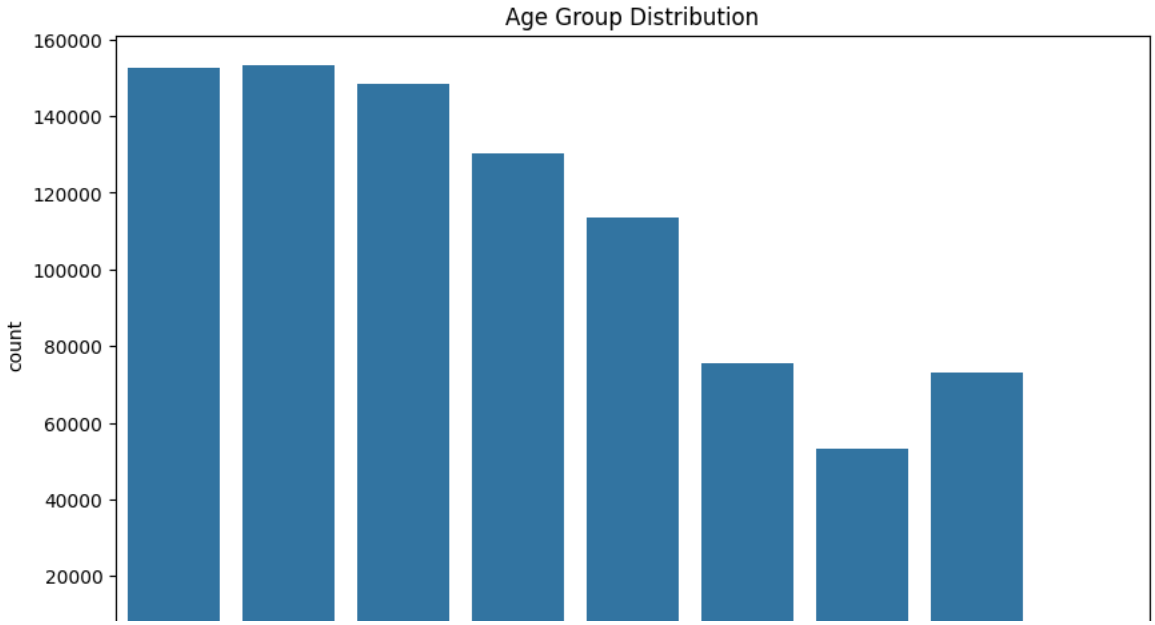

Distribution of cases over episode week groups
count
COV_EWG

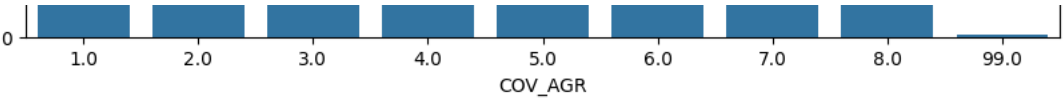
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns



df = pd.read_csv('/COVID19-eng.csv')

# Univariate Analysis
def univariate_analysis():
    # Gender distribution
    plt.figure(figsize=(10, 6))
    sns.countplot(data=df, x='COV_GDR')
    plt.title('Gender Distribution')
    plt.show()

    # Hospitalization status distribution
    plt.figure(figsize=(10, 6))
    sns.countplot(data=df, x='COV_HSP')
    plt.title('Hospitalization Status Distribution')
    plt.show()

    # Age group distribution
    plt.figure(figsize=(10, 6))
    sns.countplot(data=df, x='COV_AGR')
    plt.title('Age Group Distribution')
    plt.show()

    # Region-wise distribution
    plt.figure(figsize=(12, 6))
    sns.countplot(data=df, x='COV_REG')
```

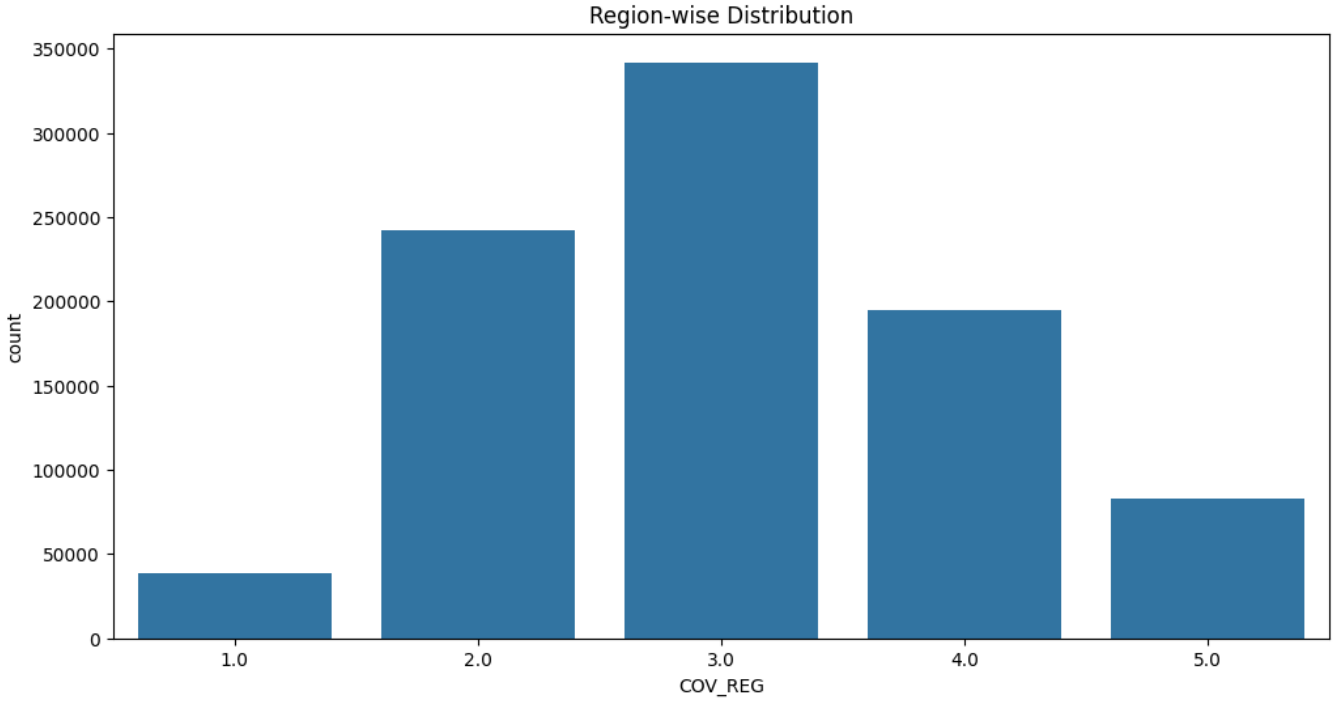
```
        plt.title('Region-wise Distribution')
        plt.show()

        # Episode week distribution
        plt.figure(figsize=(12, 6))
        sns.countplot(data=df, x='COV_EW')
        plt.title('Episode Week Distribution')
        plt.show()

        # Clinical outcome distribution
        plt.figure(figsize=(10, 6))
        sns.countplot(data=df, x='COV_DTH')
        plt.title('Clinical Outcome Distribution')
        plt.show()

    # Bivariate Analysis
    def bivariate_analysis():
        # Gender distribution over different regions
        plt.figure(figsize=(15, 6))
        sns.countplot(data=df, x='COV_REG', hue='COV_GDR')
        plt.title('Gender Distribution over Regions')
        plt.show()

        # Hospitalization status distribution across different genders
        plt.figure(figsize=(12, 6))
        sns.countplot(data=df, x='COV_GDR', hue='COV_HSP')
        plt.title('Hospitalization Status Distribution across Genders')
        plt.show()

        # Age group distribution over different regions
        plt.figure(figsize=(15, 6))
        sns.countplot(data=df, x='COV_REG', hue='COV_AGR')
        plt.title('Age Group Distribution over Regions')
        plt.show()

        # Comparative analysis of COVID-19 outcomes based on gender
        plt.figure(figsize=(10, 6))
        sns.countplot(data=df, x='COV_GDR', hue='COV_DTH')
        plt.title('Comparative analysis of COVID-19 outcomes based on Gender')
        plt.show()



    # Call the analysis functions
    univariate_analysis()
    bivariate_analysis()
```

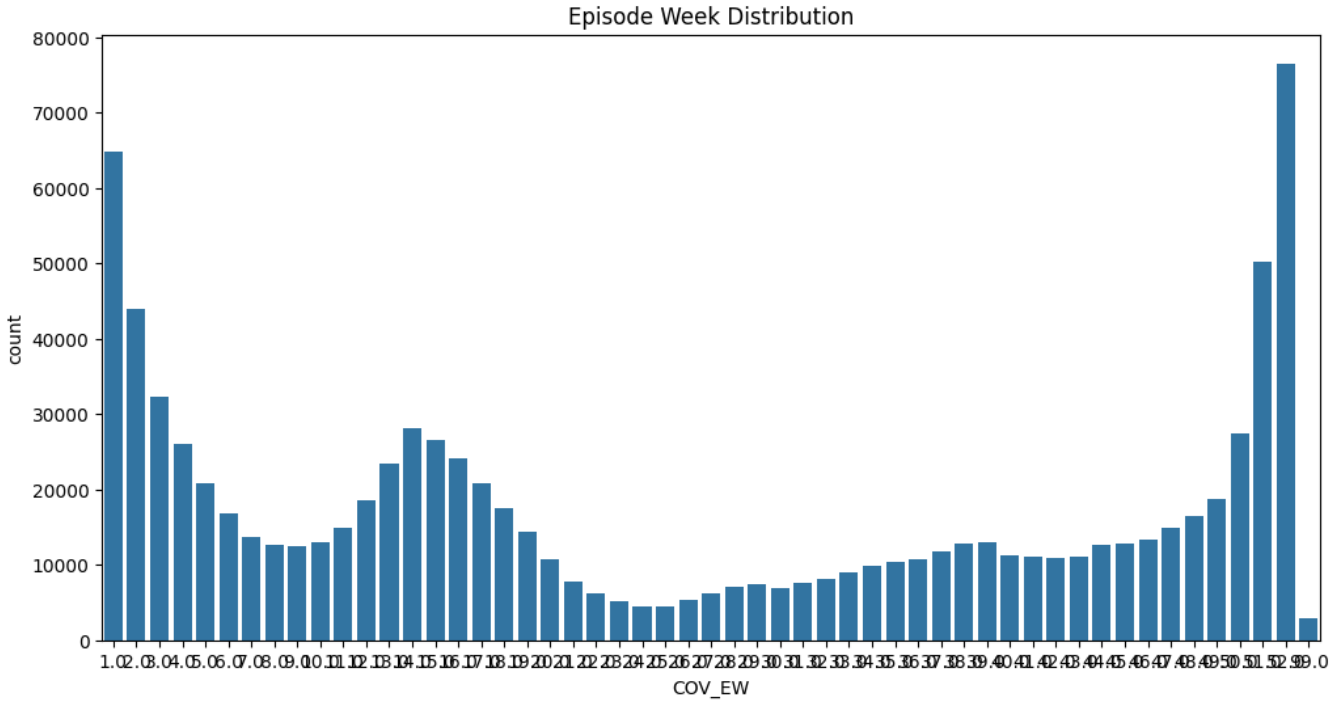

Gender Distribution
count
500000
400000
300000
200000
100000
0
1.0
2.0
9.0
COV_GDR

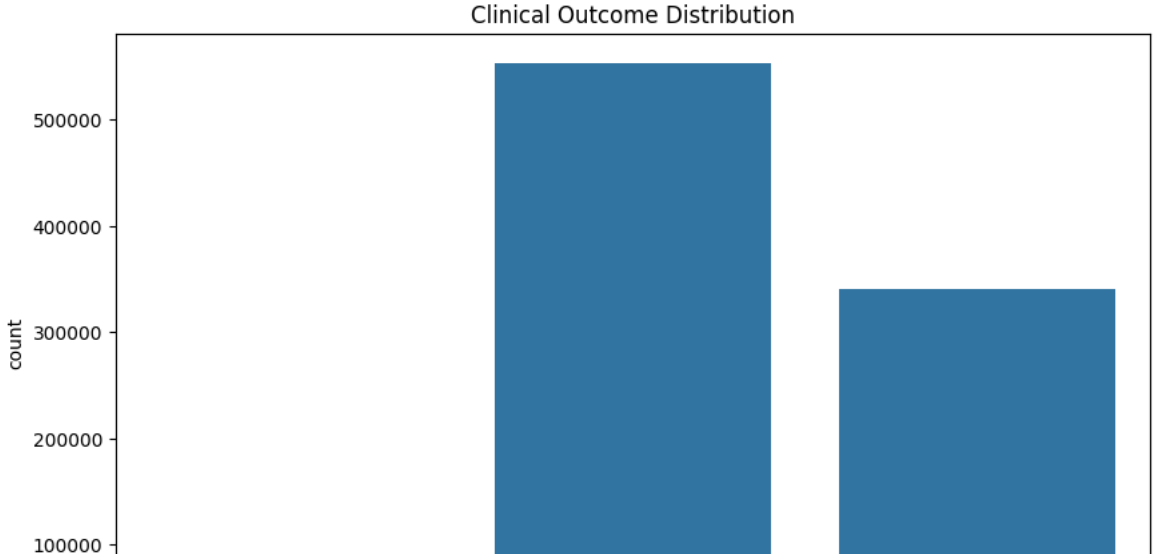

Hospitalization Status Distribution
count
500000
400000
300000
200000
100000
0
1.0
2.0
3.0
9.0
COV_HSP

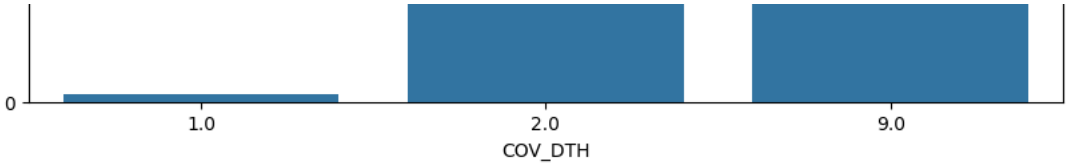

Age Group Distribution
count
160000
140000
120000
100000
80000
60000
40000
20000

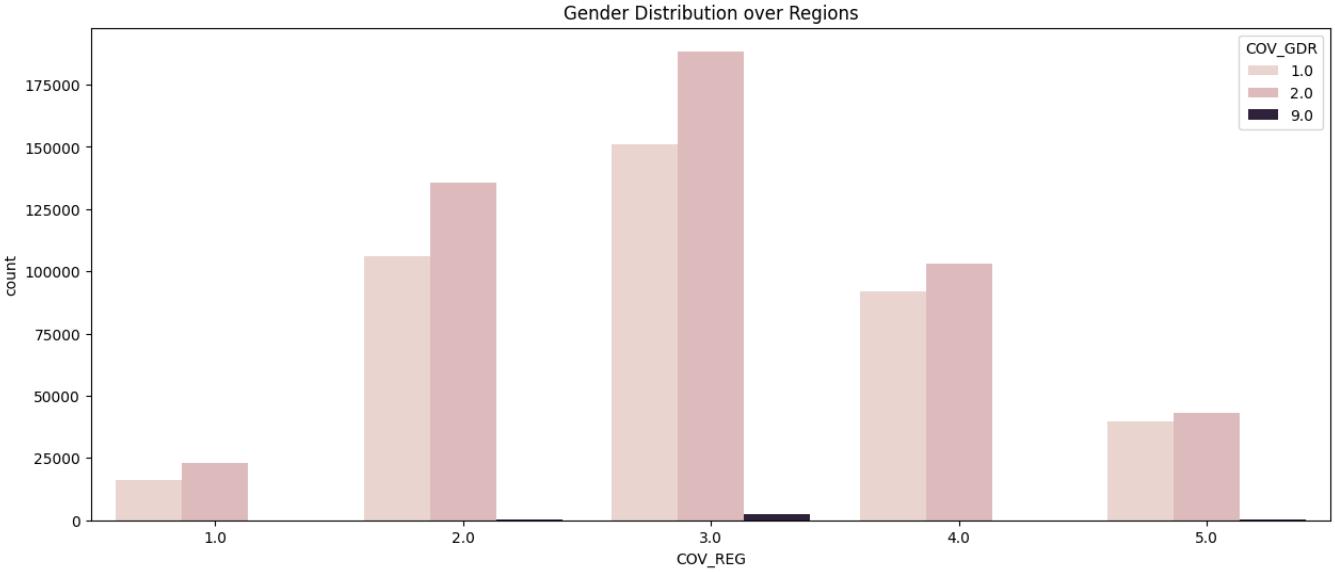

0
1.0
2.0
3.0
4.0
5.0
6.0
7.0
8.0
99.0
COV_AGR

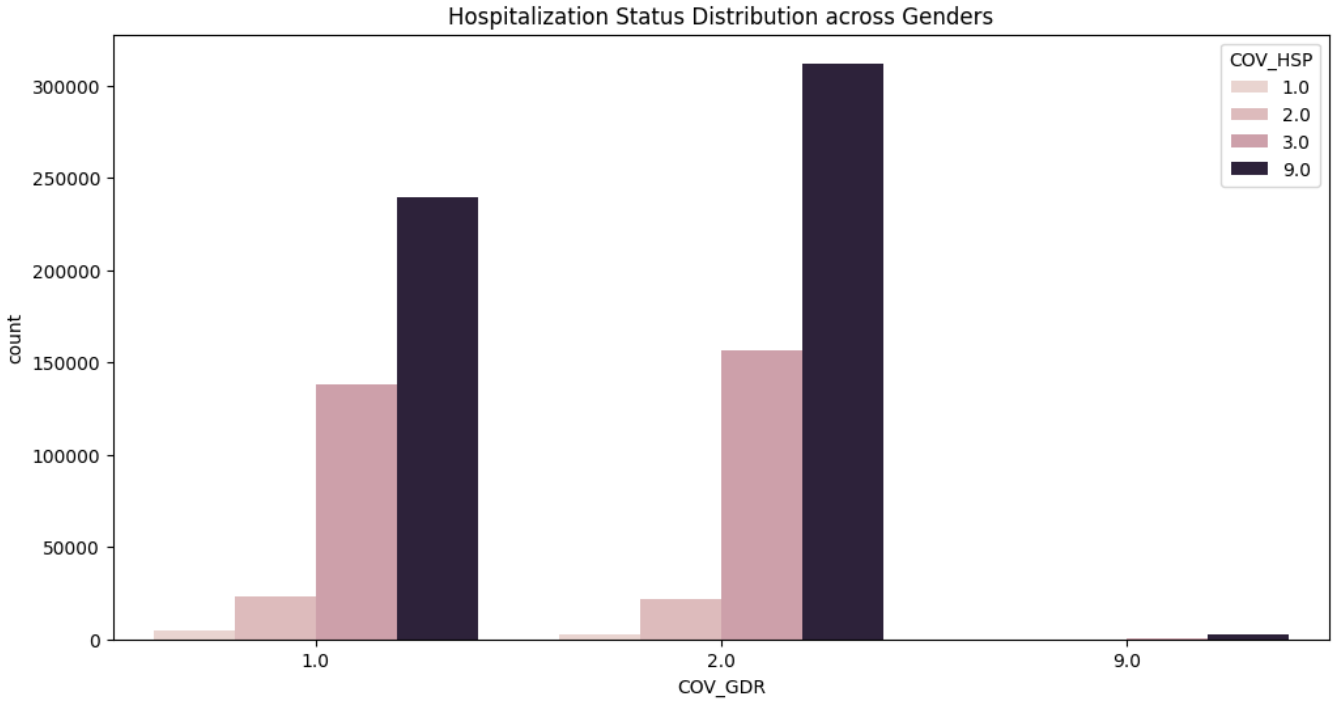

Region-wise Distribution
count
350000
300000
250000
200000
150000
100000
50000
0
1.0
2.0
3.0
4.0
5.0
COV_REG

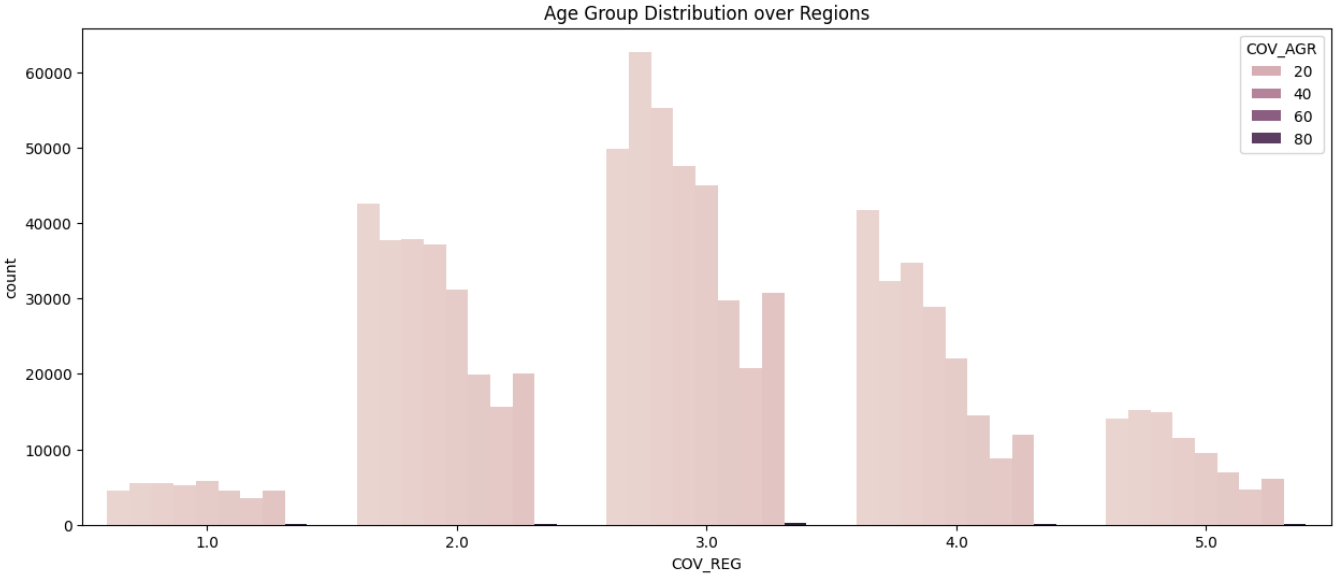

Episode Week Distribution
count
80000
70000
60000
50000
40000
30000
20000
10000
0
COV_EW


Clinical Outcome Distribution
count
500000
400000
300000
200000
100000

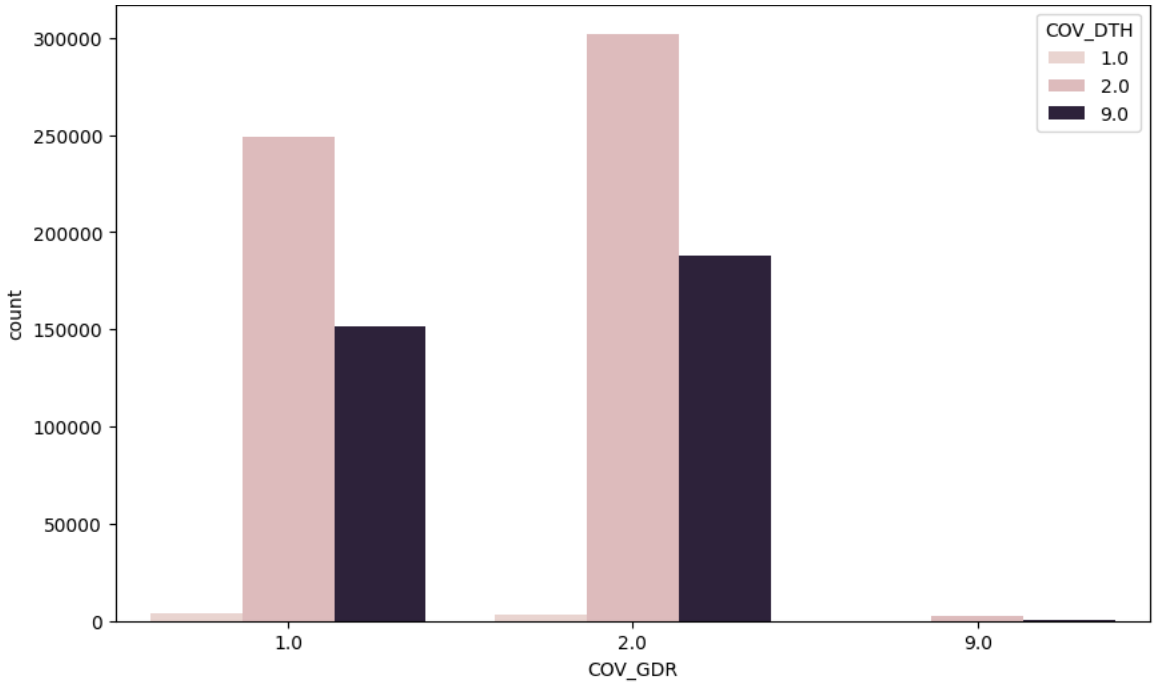








Comparative analysis of COVID-19 outcomes based on Gender

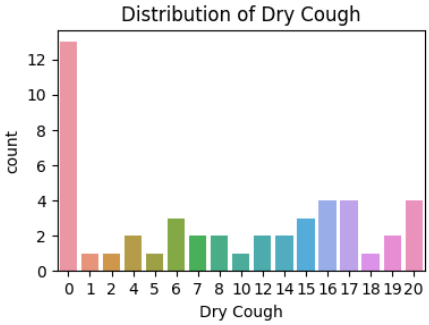


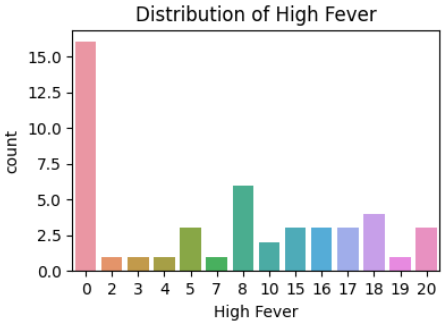

```
pip install numpy
!pip install pandas
!pip install seaborn
!pip install  matplotlib
!pip install seaborn
!pip install scikit-learn
```

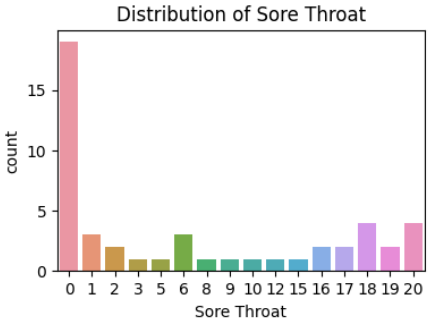
```
  File "<ipython-input-2-480414a0d787>", line 1
    pip install numpy
        ^
SyntaxError: invalid syntax
```

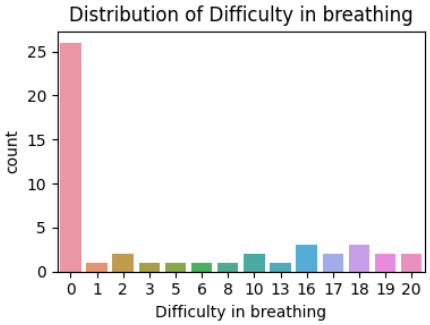
SEARCH STACK OVERFLOW

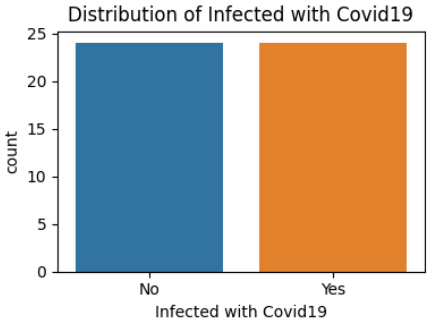
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_excel('/content/covid19-symptoms-dataset.xlsx')

# Assuming the columns in your dataset are named as follows
columns_of_interest = ["Dry Cough", "High Fever", "Sore Throat", "Difficulty in breathing", "Infected with Covid19"]

# Count and percentage distribution of each symptom
plt.figure(figsize=(12, 6))
for column in columns_of_interest[:-1]:
    plt.subplot(2, 3, columns_of_interest.index(column) + 1)
    sns.countplot(data=df, x=column)
    plt.title(f'Distribution of {column}')

# Visualize the distribution of the target variable "Infected with Covid19"
plt.subplot(2, 3, 6)
sns.countplot(data=df, x="Infected with Covid19")
plt.title('Distribution of Infected with Covid19')

plt.tight_layout()
plt.show()
```



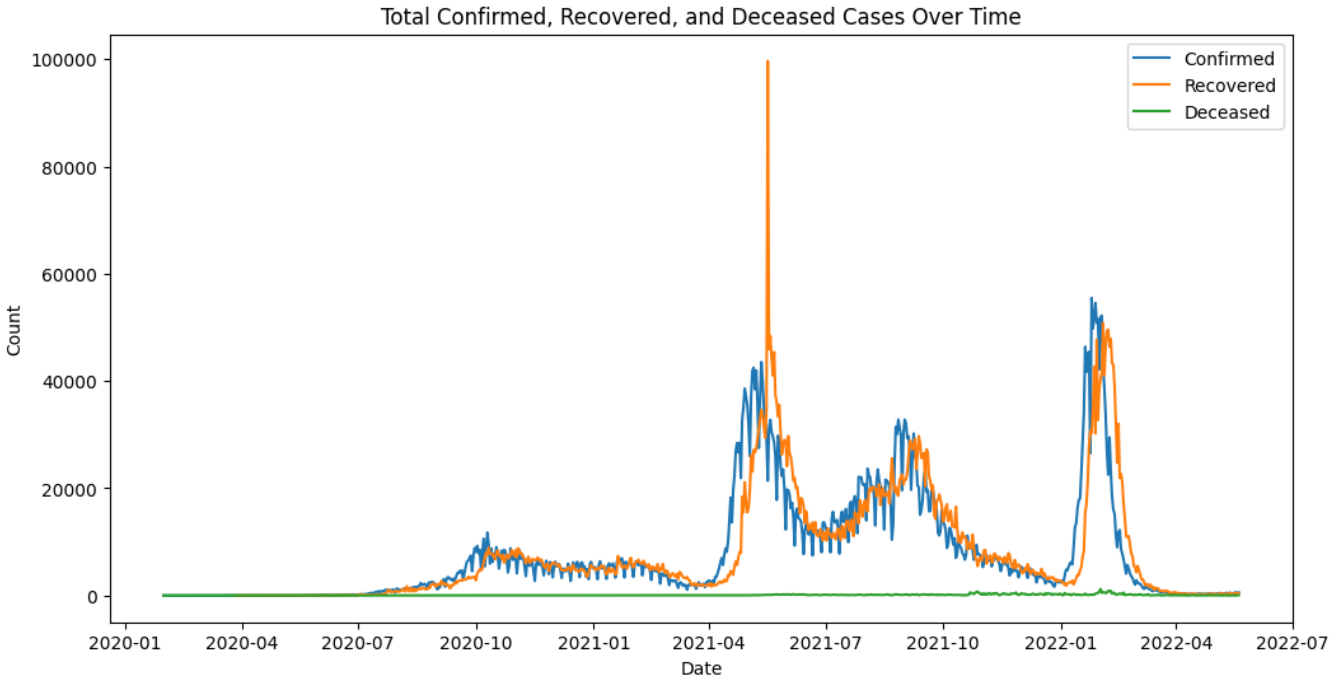









```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns


df = pd.read_csv('/content/covid_data_kerala.csv')


date_column = 'Date'
confirmed_column = 'Confirmed'
recovered_column = 'Recovered'
deceased_column = 'Deceased'

# Convert the 'Date' column to datetime type
df[date_column] = pd.to_datetime(df[date_column])

# Plot the total confirmed, recovered, and deceased cases over time
plt.figure(figsize=(12, 6))
sns.lineplot(x=date_column, y=confirmed_column, data=df, label='Confirmed')
sns.lineplot(x=date_column, y=recovered_column, data=df, label='Recovered')
sns.lineplot(x=date_column, y=deceased_column, data=df, label='Deceased')

plt.title('Total Confirmed, Recovered, and Deceased Cases Over Time')
plt.xlabel('Date')
plt.ylabel('Count')
plt.legend()
plt.show()
```


Total Confirmed, Recovered, and Deceased Cases Over Time
Confirmed
Recovered
Deceased
Count
100000
80000
60000
40000
20000
0
2020-01
2020-04
2020-07
2020-10
2021-01
2021-04
2021-07
2021-10
2022-01
2022-04
2022-07
Date

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

df = pd.read_csv('/content/Latest Covid-19 India Status.csv')

state_column = 'State/UTs'
total_cases_column = 'Total Cases'
active_column = 'Active'
discharged_column = 'Discharged'
deaths_column = 'Deaths'
active_ratio_column = 'Active Ratio'
discharge_ratio_column = 'Discharge Ratio'
death_ratio_column = 'Death Ratio'
population_column = 'Population'

# Absolute Frequency
absolute_counts = df[[state_column, total_cases_column, active_column, discharged_column, deaths_column,
                      active_ratio_column, discharge_ratio_column, death_ratio_column, population_column]]

# Display the absolute counts
print("Absolute Frequency:")
print(absolute_counts)

# Relative Frequency (percentage)
relative_percentages = df[[state_column, active_ratio_column, discharge_ratio_column, death_ratio_column, population_column]]

# Convert ratios to percentages
relative_percentages[active_ratio_column] *= 100
relative_percentages[discharge_ratio_column] *= 100
relative_percentages[death_ratio_column] *= 100

# Display the relative percentages
print("\nRelative Frequency (Percentage):")
print(relative_percentages)

# Visualize the data using bar plots
plt.figure(figsize=(12, 8))

# Bar plot for Total Cases, Active, Discharged, and Deaths
plt.subplot(2, 2, 1)
sns.barplot(x=state_column, y=total_cases_column, data=df)
plt.title('Total Cases')

plt.subplot(2, 2, 2)
sns.barplot(x=state_column, y=active_column, data=df)
plt.title('Active Cases')

plt.subplot(2, 2, 3)
sns.barplot(x=state_column, y=discharged_column, data=df)
plt.title('Discharged Cases')

plt.subplot(2, 2, 4)
sns.barplot(x=state_column, y=deaths_column, data=df)
plt.title('Deaths')

plt.tight_layout()
plt.show()
```

Absolute Frequency:

	State/UTs	Total Cases	Active	Discharged	Deaths	Active Ratio	Discharge Ratio	Death Ratio	Population
0	Andaman and Nicobar	10766	0	10637	129	0.00	98.80	1.20	100896618
1	Andhra Pradesh	2340676	0	2325943	14733	0.00	99.37	0.63	128500364
2	Arunachal Pradesh	67049	0	66753	296	0.00	99.56	0.44	658019
3	Assam	746159	5	738119	8035	0.00	98.92	1.08	290492
4	Bihar	855267	1	842952	12314	0.00	98.56	1.44	40100376
5	Chandigarh	100693	0	99508	1185	0.00	98.82	1.18	30501026
6	Chhattisgarh	1187695	0	1173505	14190	0.00	98.81	1.19	28900667
7	Dadra and Nagar Haveli and Daman and Diu	11592	0	11588	4	0.00	99.97	0.03	231502578
8	Delhi	2040910	14	2014230	26666	0.00	98.69	1.31	773997
9	Goa	263346	3	259329	4014	0.00	98.47	1.52	3772103
10	Gujarat	1291383	5	1280299	11079	0.00	99.14	0.86	70400153
11	Haryana	1078903	27	1068121	10755	0.00	99.00	1.00	7503010
12	Himachal Pradesh	322905	4	318660	4241	0.00	98.69	1.31	3436948
13	Jammu and Kashmir	482023	0	477231	4792	0.00	99.01	0.99	66001
14	Jharkhand	443826	1	438491	5334	0.00	98.80	1.20	124904071
15	Karnataka	4088769	12	4048399	40358	0.00	99.01	0.99	1711947
16	Kerala	6907241	18	6835181	72042	0.00	98.96	1.04	91702478
17	Ladakh	29602	0	29371	231	0.00	99.22	0.78	4184959
18	Lakshadweep	11415	0	11363	52	0.00	99.54	0.46	11700099
19	Madhya Pradesh	1056351	0	1045565	10786	0.00	98.98	1.02	14999397
20	Maharashtra	8171048	214	8022276	148558	0.00	98.18	1.82	399001
21	Manipur	140034	0	137885	2149	0.00	98.47	1.53	47099270
22	Meghalaya	96983	3	95352	1628	0.00	98.32	1.68	79502477
23	Mizoram	239560	1	238825	734	0.00	99.69	0.31	1308967
24	Nagaland	36033	0	35251	782	0.00	97.83	2.17	38157311
25	Odisha	1348409	59	1339135	9215	0.00	99.31	0.68	19301096
26	Puducherry	177547	0	175566	1981	0.00	98.88	1.12	2073074
27	Punjab	793644	1233	773073	19338	0.16	97.41	2.44	34698876
28	Rajasthan	1326465	2	1316727	9736	0.00	99.27	0.73	1521992
29	Sikkim	44927	4	44422	501	0.01	98.88	1.12	83697770
30	Tamil Nadu	3610655	5	3572569	38081	0.00	98.95	1.05	35998752
31	Telengana	844432	8	840313	4111	0.00	99.51	0.49	69599762
32	Tripura	108493	1	107550	942	0.00	99.13	0.87	1646050
33	Uttar Pradesh	2145431	57	2121662	23712	0.00	98.89	1.11	1158040
34	Uttarakhand	452571	0	444803	7768	0.00	98.28	1.72	85002417
35	West Bengal	2126282	135	2104592	21555	0.01	98.98	1.01	32199722

Relative Frequency (Percentage):

	State/UTs	Active Ratio	Discharge Ratio
0	Andaman and Nicobar	0.0	9880.0
1	Andhra Pradesh	0.0	9937.0
2	Arunachal Pradesh	0.0	9956.0
3	Assam	0.0	9892.0
4	Bihar	0.0	9856.0
5	Chandigarh	0.0	9882.0
6	Chhattisgarh	0.0	9881.0
7	Dadra and Nagar Haveli and Daman and Diu	0.0	9997.0
8	Delhi	0.0	9869.0
9	Goa	0.0	9847.0
10	Gujarat	0.0	9914.0