Adverse Drug Effect Analysis

In this assignment, we have to analyse FAERS database to find out the adverse effects of the usual drugs such as gapapentin or lyrica to Tramal. Our objectives are: Find 10 adverse effects of Tramal, Compare to medicine brand lyrica and finally look for other comparison metrics.

In [18]:
```
#Obtaining the main dataset
import pandas as pd
data1=pd.read_csv("C:\\Users\\sujoydutta\\Downloads\\druganalysis.csv")

data1.head()
```

Out[18]:

	primaryid	manufacturer	age	sex	weight	Date	occp_cod	country	side_effects
0	100046573	PFIZER	71.0	F	81.63	03-11-20	LW	US	Type 2 diabetes mellitus
1	100046962	NOVARTIS	23.0	M	NaN	21-10-20	HP	PL	Type 2 diabetes mellitus
2	100048793	PFIZER	51.0	F	NaN	02-11-20	LW	US	Abnormal behaviour
3	100051383	PFIZER	50.0	F	83.00	20-10-20	LW	US	Abnormal behaviour
4	100075524	PFIZER	38.0	F	90.70	21-10-20	LW	US	Abnormal behaviour

In [19]:
```
#describing the data
data1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 436148 entries, 0 to 436147
Data columns (total 9 columns):
 #   Column         Non-Null Count   Dtype  
---  ------         --------------   -----  
 0   primaryid      436148 non-null  int64  
 1   manufacturer   436148 non-null  object 
 2   age            230001 non-null  float64
 3   sex            358171 non-null  object 
 4   weight         83112 non-null   float64
 5   Date           436148 non-null  object 
 6   occp_cod       427479 non-null  object 
 7   country        435461 non-null  object 
 8   side_effects   436148 non-null  object 
dtypes: float64(2), int64(1), object(6)
memory usage: 29.9+ MB
```

In [20]:
```
#obtaining the secondary dataset

data2=pd.read_csv("C:\\Users\\sujoydutta\\Downloads\\drug20q4.csv")
data2.head()
```

Out[20]:

	primaryid	caseid	drug_seq	role_cod	drugname
0	100046573	10004657	1	PS	LIPITOR
1	100046573	10004657	2	C	TOPROL XL
2	100046962	10004696	1	PS	QUETIAPINE.
3	100046962	10004696	2	I	CITALOPRAM
4	100046962	10004696	3	I	CITALOPRAM

```
In [22]: #seeing dimensions of data
         data1.shape

Out[22]: (436148, 9)
```

```
In [23]: #seeing dimensions of data
         data2.shape

Out[23]: (1048575, 5)
```

```
In [24]: #changing the columns to right data type

         data1['primaryid'] = data1['primaryid'].astype(str)
         data2['primaryid'] = data2['primaryid'].astype(str)
```

```
In [25]: # Merging the datasets on 'primaryid'
         data = pd.merge(data1, data2, on='primaryid')
         data.head()
```

Out[25]:

	primaryid	manufacturer	age	sex	weight	Date	occp_cod	country	side_effects	caseid	drug_seq	role_
0	100046573	PFIZER	71.0	F	81.63	03-11-20	LW	US	Type 2 diabetes mellitus	10004657	1	
1	100046573	PFIZER	71.0	F	81.63	03-11-20	LW	US	Type 2 diabetes mellitus	10004657	2	
2	100046962	NOVARTIS	23.0	M	NaN	21-10-20	HP	PL	Type 2 diabetes mellitus	10004696	1	
3	100046962	NOVARTIS	23.0	M	NaN	21-10-20	HP	PL	Type 2 diabetes mellitus	10004696	2	
4	100046962	NOVARTIS	23.0	M	NaN	21-10-20	HP	PL	Type 2 diabetes mellitus	10004696	3	

```
In [35]: #converting date column to valid format
         valid_rows = ~data['Date'].str.contains('--')

         data['Date'] = pd.to_datetime(data['Date'], errors='coerce')
```

```
C:\Users\sujoydutta\AppData\Local\Temp\ipykernel_5168\654779872.py:4: UserWarning: Could
not infer format, so each element will be parsed individually, falling back to `dateutil
`. To ensure parsing is consistent and as-expected, please specify a format.
  data['Date'] = pd.to_datetime(data['Date'], errors='coerce')
```

```
In [36]: #seeing the unique dates
         data['Date'].unique()

Out[36]: <DatetimeArray>
['2020-03-11 00:00:00', '2020-10-21 00:00:00', '2020-02-11 00:00:00',
 '2020-10-20 00:00:00', '2020-11-24 00:00:00', '2020-05-10 00:00:00',
 '2020-06-10 00:00:00', '2020-11-18 00:00:00', '2020-10-24 00:00:00',
 '2020-10-27 00:00:00',
 ...
 '2020-05-27 00:00:00', '2020-07-27 00:00:00', '2016-04-15 00:00:00',
 '2015-12-21 00:00:00', '2020-09-14 00:00:00', '2020-08-31 00:00:00',
 '2020-08-17 00:00:00', '2020-03-08 00:00:00', '2020-07-24 00:00:00',
 '2020-07-22 00:00:00']
Length: 168, dtype: datetime64[ns]
```

In [37]:
```
# Replacing NaT values with interpolated values
data['Date'] = data['Date'].interpolate(method='linear')
print(data)
```

```
         primaryid         manufacturer   age  sex  weight       Date  \
0        100046573               PFIZER  71.0    F   81.63 2020-03-11
1        100046573               PFIZER  71.0    F   81.63 2020-03-11
2        100046962             NOVARTIS  23.0    M     NaN 2020-10-21
3        100046962             NOVARTIS  23.0    M     NaN 2020-10-21
4        100046962             NOVARTIS  23.0    M     NaN 2020-10-21
...            ...                  ...   ...  ..     ...        ...
1048570  184428301  BRISTOL MYERS SQUIBB  54.0    M   82.00 2020-10-29
1048571  184428311             NOVARTIS  82.0    M     NaN 2020-10-29
1048572  184428321             NOVARTIS   NaN    M     NaN 2020-10-29
1048573  184428331               TAKEDA   NaN    M     NaN 2020-10-29
1048574  184428331               TAKEDA   NaN    M     NaN 2020-10-29

        occp_cod country               side_effects    caseid  drug_seq  \
0             LW      US  Type 2 diabetes mellitus  10004657         1
1             LW      US  Type 2 diabetes mellitus  10004657         2
2             HP      PL  Type 2 diabetes mellitus  10004696         1
3             HP      PL  Type 2 diabetes mellitus  10004696         2
4             HP      PL  Type 2 diabetes mellitus  10004696         3
...          ...     ...                       ...       ...       ...
1048570       CN      US     Foreign body in throat  18442830         1
1048571       CN      GB     Foreign body in throat  18442831         1
1048572       CN      US     Foreign body in throat  18442832         1
1048573       MD      US     Foreign body in throat  18442833         1
1048574       MD      US     Foreign body in throat  18442833         2

        role_cod     drugname
0             PS      LIPITOR
1              C    TOPROL XL
2             PS  QUETIAPINE.
3              I   CITALOPRAM
4              I   CITALOPRAM
...          ...          ...
1048570       PS       OPDIVO
1048571       PS     ENTRESTO
1048572       PS     ENTRESTO
1048573       PS   Idursulfase
1048574       SS   Idursulfase

[1048575 rows x 13 columns]
```

In [38]:
```
#examining the new dataset
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1048575 entries, 0 to 1048574
Data columns (total 13 columns):
 #   Column         Non-Null Count    Dtype
---  ------         --------------    -----
 0   primaryid      1048575 non-null  object
 1   manufacturer   1048575 non-null  object
 2   age            674478 non-null   float64
 3   sex            897982 non-null   object
 4   weight         361821 non-null   float64
 5   Date           1048575 non-null  datetime64[ns]
 6   occp_cod       1003905 non-null  object
 7   country        1048485 non-null  object
 8   side_effects   1048575 non-null  object
 9   caseid         1048575 non-null  int64
 10  drug_seq       1048575 non-null  int64
 11  role_cod       1048575 non-null  object
 12  drugname       1048519 non-null  object
dtypes: datetime64[ns](1), float64(2), int64(2), object(8)
memory usage: 104.0+ MB
```

In [39]:
```
#removing whitespaces
data.columns = data.columns.str.strip()
data.columns
```

Out[39]:
```
Index(['primaryid', 'manufacturer', 'age', 'sex', 'weight', 'Date', 'occp_cod',
       'country', 'side_effects', 'caseid', 'drug_seq', 'role_cod',
       'drugname'],
      dtype='object')
```

In [40]:
```
#removing unnecessary columns
data=data.drop(['caseid','drug_seq','role_cod'],axis=1)
data.head()
```

Out[40]:

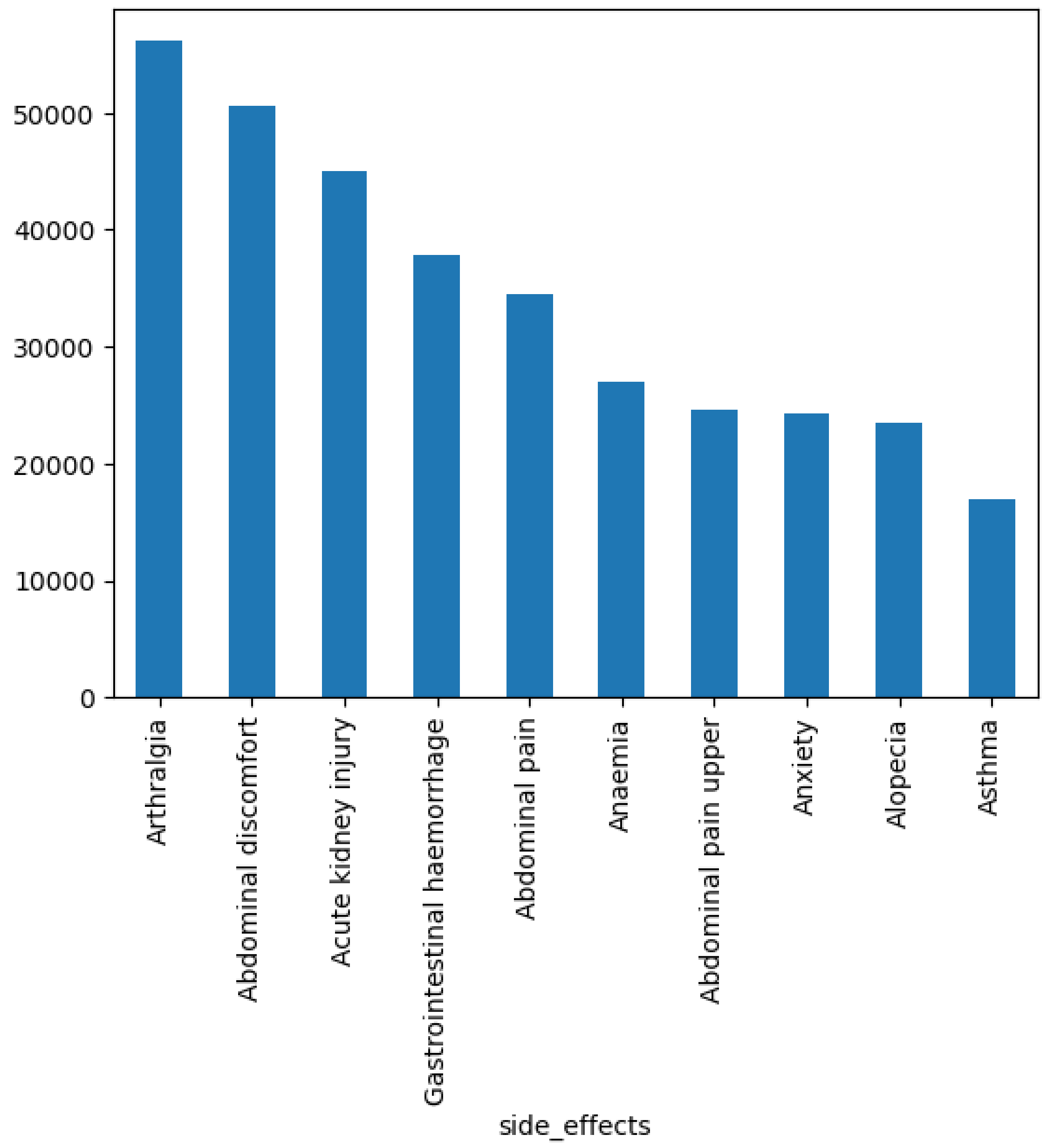
	primaryid	manufacturer	age	sex	weight	Date	occp_cod	country	side_effects	drugname
0	100046573	PFIZER	71.0	F	81.63	2020-03-11	LW	US	Type 2 diabetes mellitus	LIPITOR
1	100046573	PFIZER	71.0	F	81.63	2020-03-11	LW	US	Type 2 diabetes mellitus	TOPROL XL
2	100046962	NOVARTIS	23.0	M	NaN	2020-10-21	HP	PL	Type 2 diabetes mellitus	QUETIAPINE.
3	100046962	NOVARTIS	23.0	M	NaN	2020-10-21	HP	PL	Type 2 diabetes mellitus	CITALOPRAM
4	100046962	NOVARTIS	23.0	M	NaN	2020-10-21	HP	PL	Type 2 diabetes mellitus	CITALOPRAM

In [41]:
```
# Using median for imputation for numeric columns
data['age'] = data['age'].fillna(data['age'].median())
data['weight'] = data['weight'].fillna(data['weight'].median())
```

In [44]:
```
# Using mode because of categorical columns
data['sex'] = data['sex'].fillna(data['sex'].mode()[0])
data['occp_cod'] = data['occp_cod'].fillna(data['occp_cod'].mode()[0])
data['country'] = data['country'].fillna(data['country'].mode()[0])
data['drugname'] = data['drugname'].fillna(data['drugname'].mode()[0])
```

In [45]:
```
#top 10 most common side effects
commonsideeffects=data.groupby('side_effects')['primaryid'].count().sort_values(ascendin
commonsideeffects.plot(kind='bar')
```

Out[45]:
```
<Axes: xlabel='side_effects'>
```



In [48]:
```
#  Filtering for Tramal
tramal_data = data[data['drugname'].str.contains('TRAMAL', case=False, na=False)]
tramal_data.head()
```

Out[48]:

	primaryid	manufacturer	age	sex	weight	Date	occp_cod	country	side_effects	drugname
3963	1072495210	INCYTE	61.0	F	74.84	2020-10-13	MD	JP	Asthenia	TRAMAL
10019	113145689	EISAI	64.0	F	41.00	2020-12-11	MD	JP	Arthralgia	TRAMAL OD
24435	122109424	CLINIGEN	41.0	M	62.50	2020-10-26	MD	JP	Abdominal pain upper	TRAMAL
53510	131858184	GILEAD	54.0	M	74.84	2020-11-12	MD	PL	Haemorrhage	TRAMAL
76630	1417151610	TAKEDA	49.0	M	74.84	2020-10-12	MD	JP	Coma	Tramal

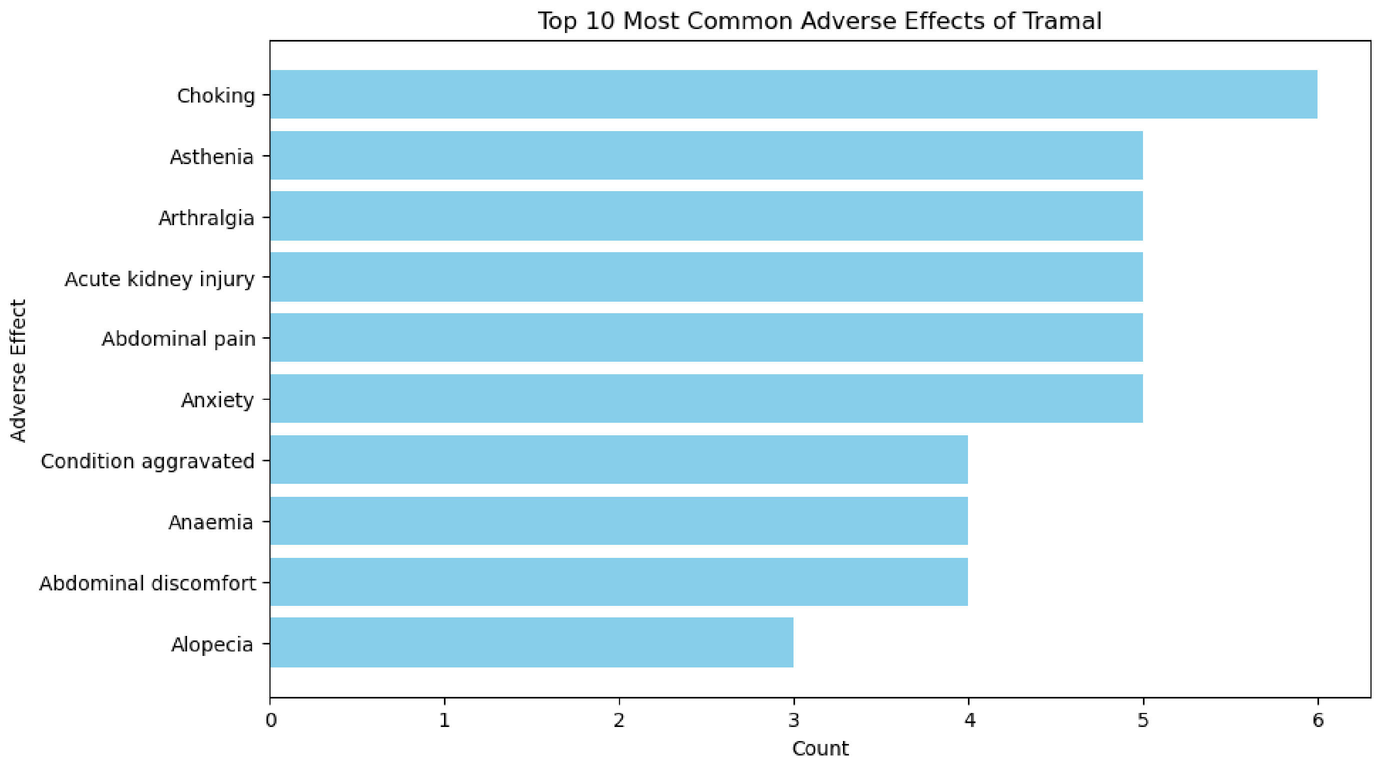
In [49]:
```
# Count the frequency of each adverse effect
adverse_effects_count = tramal_data['side_effects'].value_counts().reset_index()
adverse_effects_count.columns = ['side_effect', 'count']
```

In [50]:
```
# Getting the top 10 most common adverse effects
top_10_adverse_effects = adverse_effects_count.head(10)
top_10_adverse_effects
```

Out[50]:

	side_effect	count
0	Choking	6
1	Asthenia	5
2	Arthralgia	5
3	Acute kidney injury	5
4	Abdominal pain	5
5	Anxiety	5
6	Condition aggravated	4
7	Anaemia	4
8	Abdominal discomfort	4
9	Alopecia	3

In [53]:
```
# Plotting the top 10 adverse effects
import matplotlib.pyplot as plt
plt.figure(figsize=(10, 6))
plt.barh(top_10_adverse_effects['side_effect'], top_10_adverse_effects['count'], color='
plt.xlabel('Count')
plt.ylabel('Adverse Effect')
plt.title('Top 10 Most Common Adverse Effects of Tramal')
plt.gca().invert_yaxis()
plt.show()
```

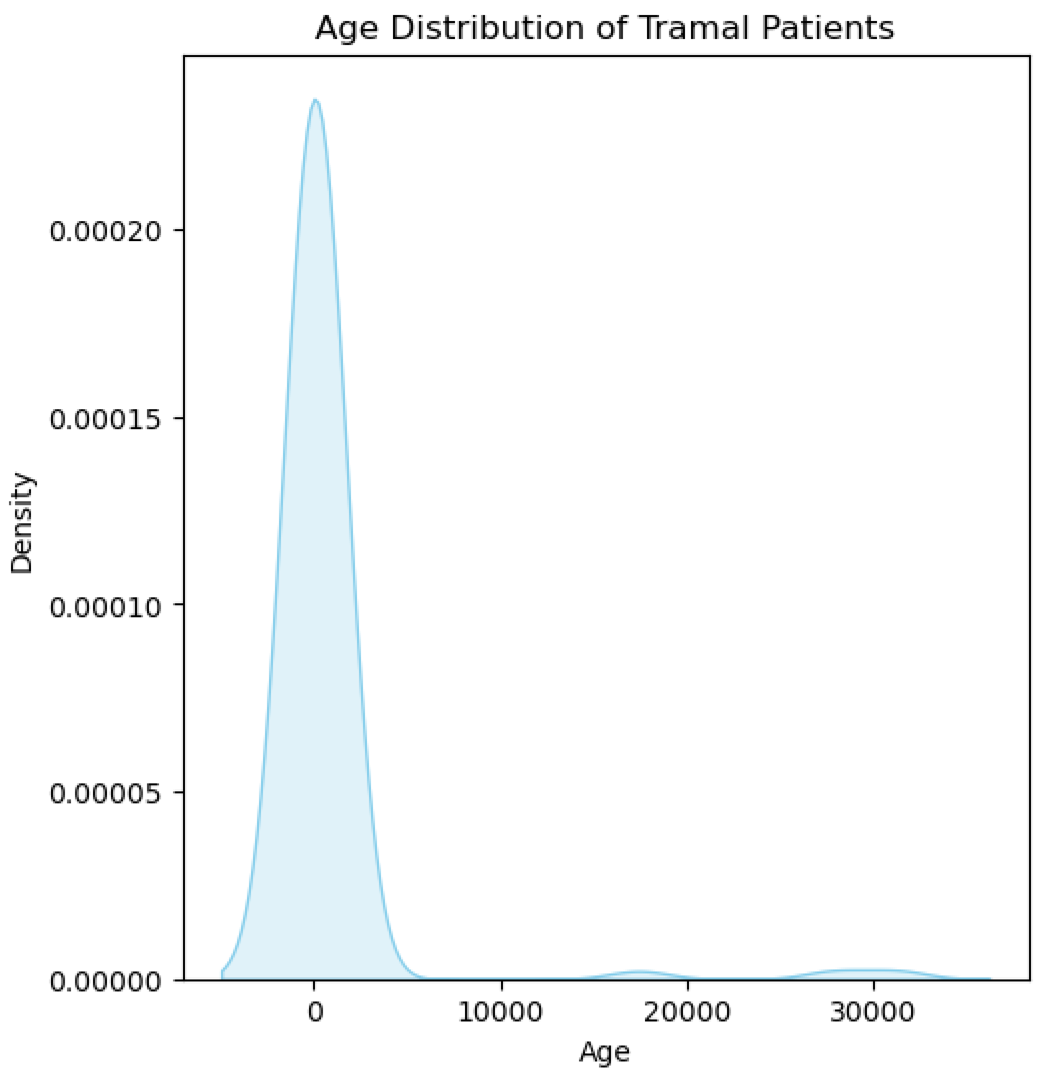


In [79]:
```
# Plotting the age distribution using KDE plot
import seaborn as sns
plt.figure(figsize=(12, 6))

plt.subplot(1, 2, 1)
sns.kdeplot(tramal_data['age'].dropna(), shade=True, color='skyblue')
```

```
plt.xlabel('Age')
plt.ylabel('Density')
plt.title('Age Distribution of Tramal Patients')
```

```
C:\Users\sujoydutta\AppData\Local\Temp\ipykernel_5168\1810214485.py:6: FutureWarning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  sns.kdeplot(tramal_data['age'].dropna(), shade=True, color='skyblue')
```

Out[79]:
```
Text(0.5, 1.0, 'Age Distribution of Tramal Patients')
```



In [54]:
```
#  Filtering for lyrica
lyrica_data = data[data['drugname'].str.contains('lyrica', case=False, na=False)]
lyrica_data.head()
```

Out[54]:

	primaryid	manufacturer	age	sex	weight	Date	occp_cod	country	side_effects	drugname
353	1002745712	PFIZER	51.0	F	73.00	2020-12-14	MD	US	Type 2 diabetes mellitus	LYRICA
354	1002745712	PFIZER	51.0	F	73.00	2020-12-14	MD	US	Type 2 diabetes mellitus	LYRICA
355	1002745712	PFIZER	51.0	F	73.00	2020-12-14	MD	US	Type 2 diabetes mellitus	LYRICA
858	101532733	PFIZER	59.0	F	62.60	2020-10-16	MD	US	Anaemia	LYRICA
906	1015551611	PFIZER	56.0	F	67.12	2020-	MD	US	Anaemia	LYRICA

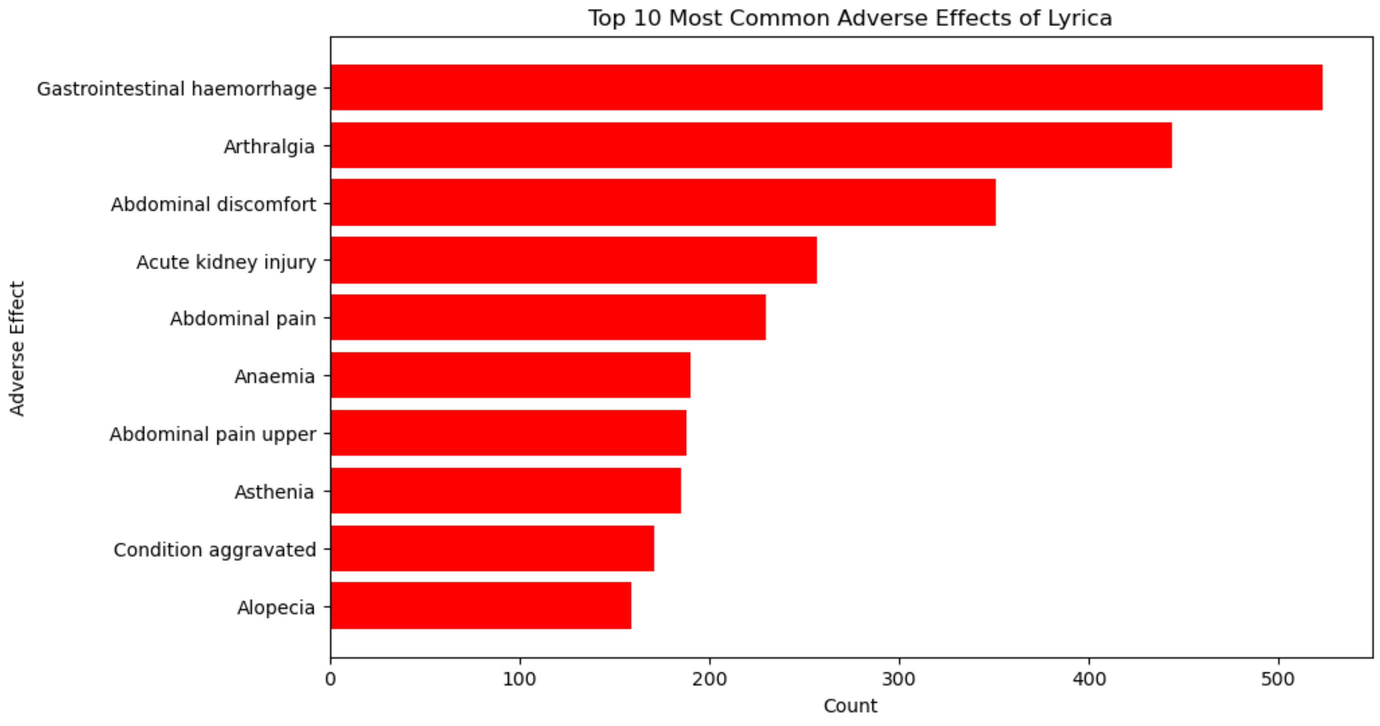
In [55]:
```
# Count the frequency of each adverse effect
adverse_effects_count = lyrica_data['side_effects'].value_counts().reset_index()
adverse_effects_count.columns = ['side_effect', 'count']
```

In [56]:
```
# Getting the top 10 most common adverse effects
top_10_adverse_effects = adverse_effects_count.head(10)
top_10_adverse_effects
```

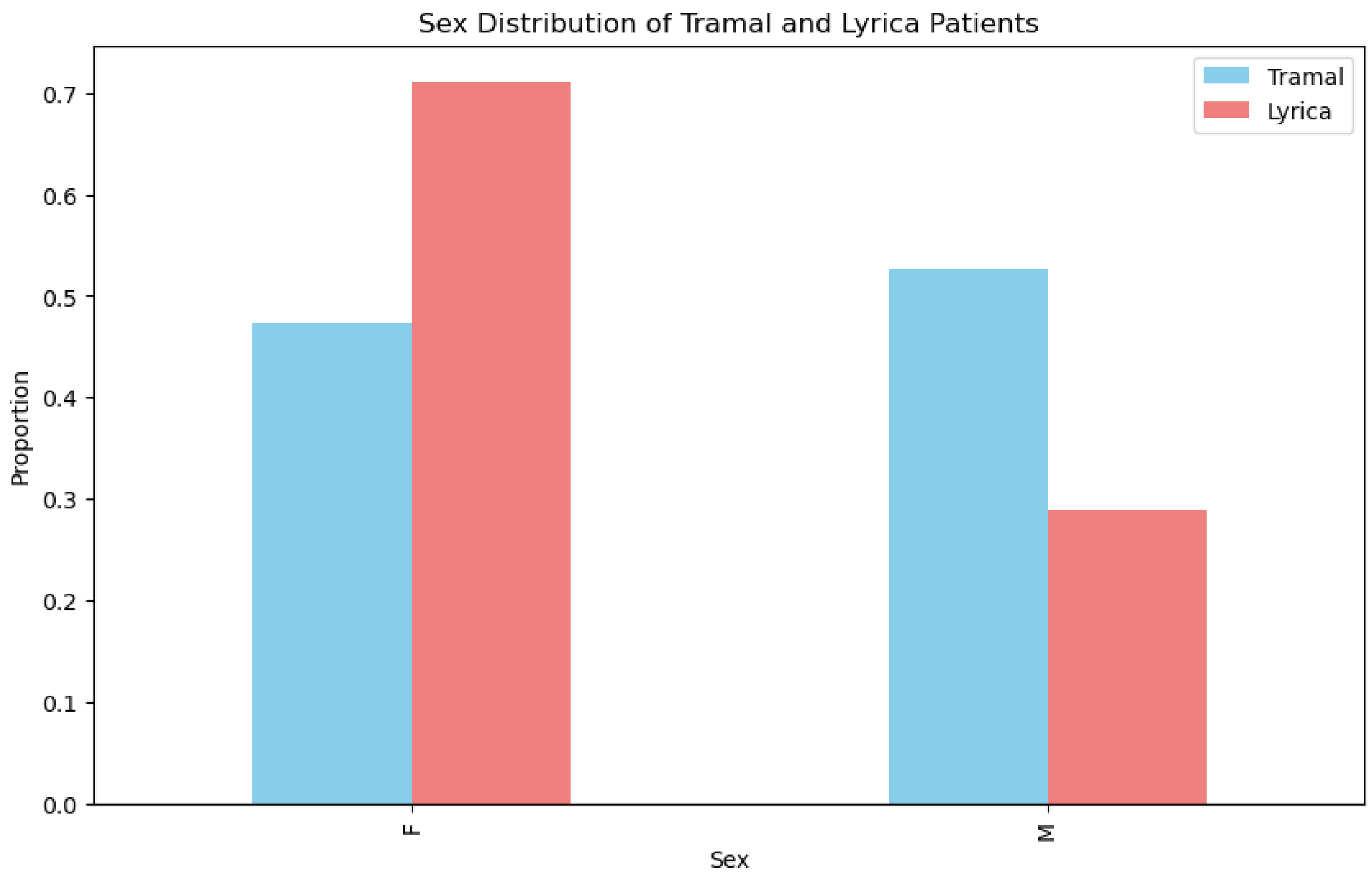
Out[56]:

	side_effect	count
0	Gastrointestinal haemorrhage	523
1	Arthralgia	444
2	Abdominal discomfort	351
3	Acute kidney injury	257
4	Abdominal pain	230
5	Anaemia	190
6	Abdominal pain upper	188
7	Asthenia	185
8	Condition aggravated	171
9	Alopecia	159

In [58]:
```
# Plotting the top 10 adverse effects

plt.figure(figsize=(10, 6))
plt.barh(top_10_adverse_effects['side_effect'], top_10_adverse_effects['count'], color='
plt.xlabel('Count')
plt.ylabel('Adverse Effect')
plt.title('Top 10 Most Common Adverse Effects of Lyrica')
plt.gca().invert_yaxis()
plt.show()
```



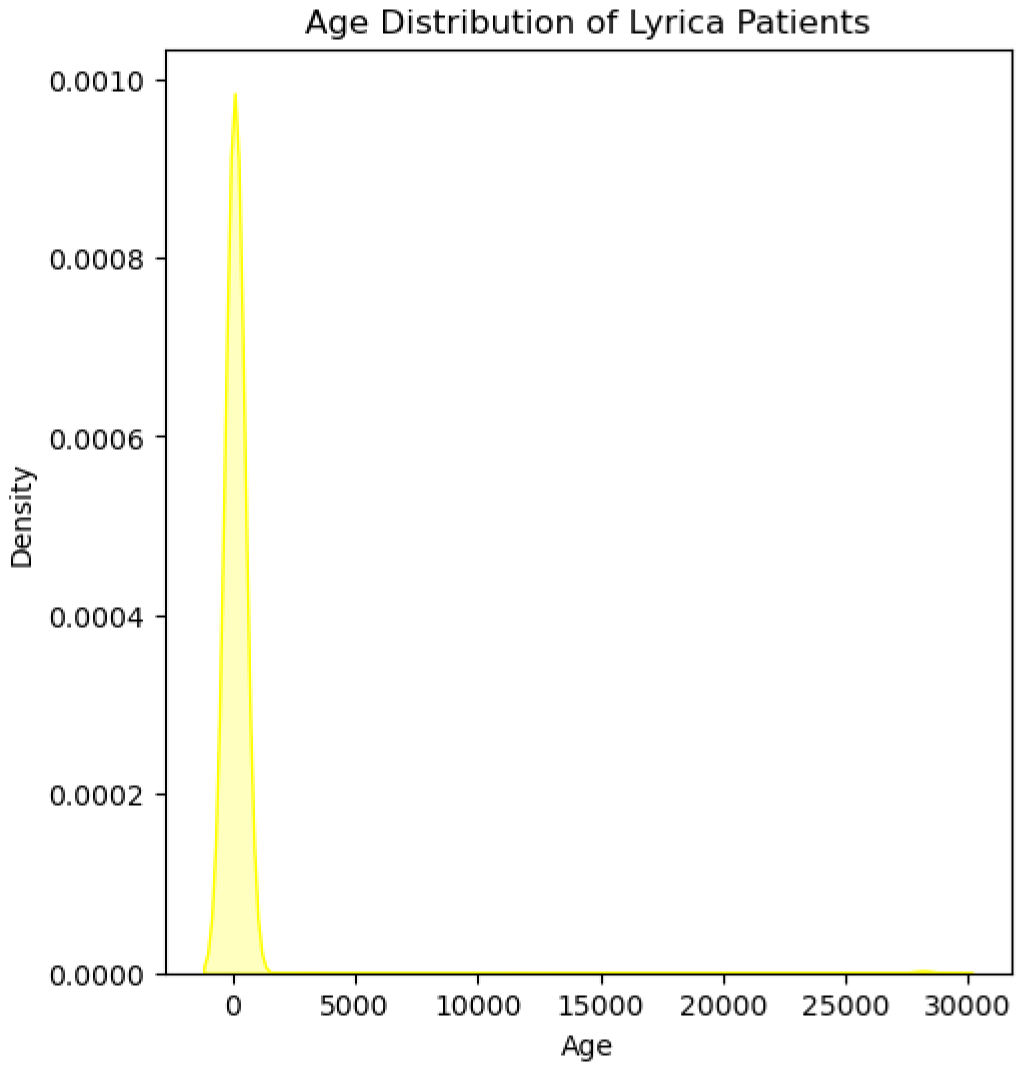
In [81]:
```
# Comparing sex distribution of both drugs
sex_counts_tramal = tramal_data['sex'].value_counts(normalize=True)
sex_counts_lyrica = lyrica_data['sex'].value_counts(normalize=True)

sex_df = pd.DataFrame({
    'Tramal': sex_counts_tramal,
    'Lyrica': sex_counts_lyrica
}).reset_index()

sex_df.columns = ['sex', 'Tramal', 'Lyrica']

sex_df.plot(x='sex', y=['Tramal', 'Lyrica'], kind='bar', figsize=(10, 6), color=['skyblu
plt.xlabel('Sex')
plt.ylabel('Proportion')
plt.title('Gender Distribution of Tramal and Lyrica Patients')
plt.show()
```



In [80]:
```
# Plotting the age distribution using KDE plot

plt.figure(figsize=(12, 6))

plt.subplot(1, 2, 1)
sns.kdeplot(lyrica_data['age'].dropna(), shade=True, color='yellow')
plt.xlabel('Age')
plt.ylabel('Density')
plt.title('Age Distribution of Lyrica Patients')
```

```
C:\Users\sujoydutta\AppData\Local\Temp\ipykernel_5168\4230241145.py:6: FutureWarning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  sns.kdeplot(lyrica_data['age'].dropna(), shade=True, color='yellow')
```

Out[80]:
```
Text(0.5, 1.0, 'Age Distribution of Lyrica Patients')
```

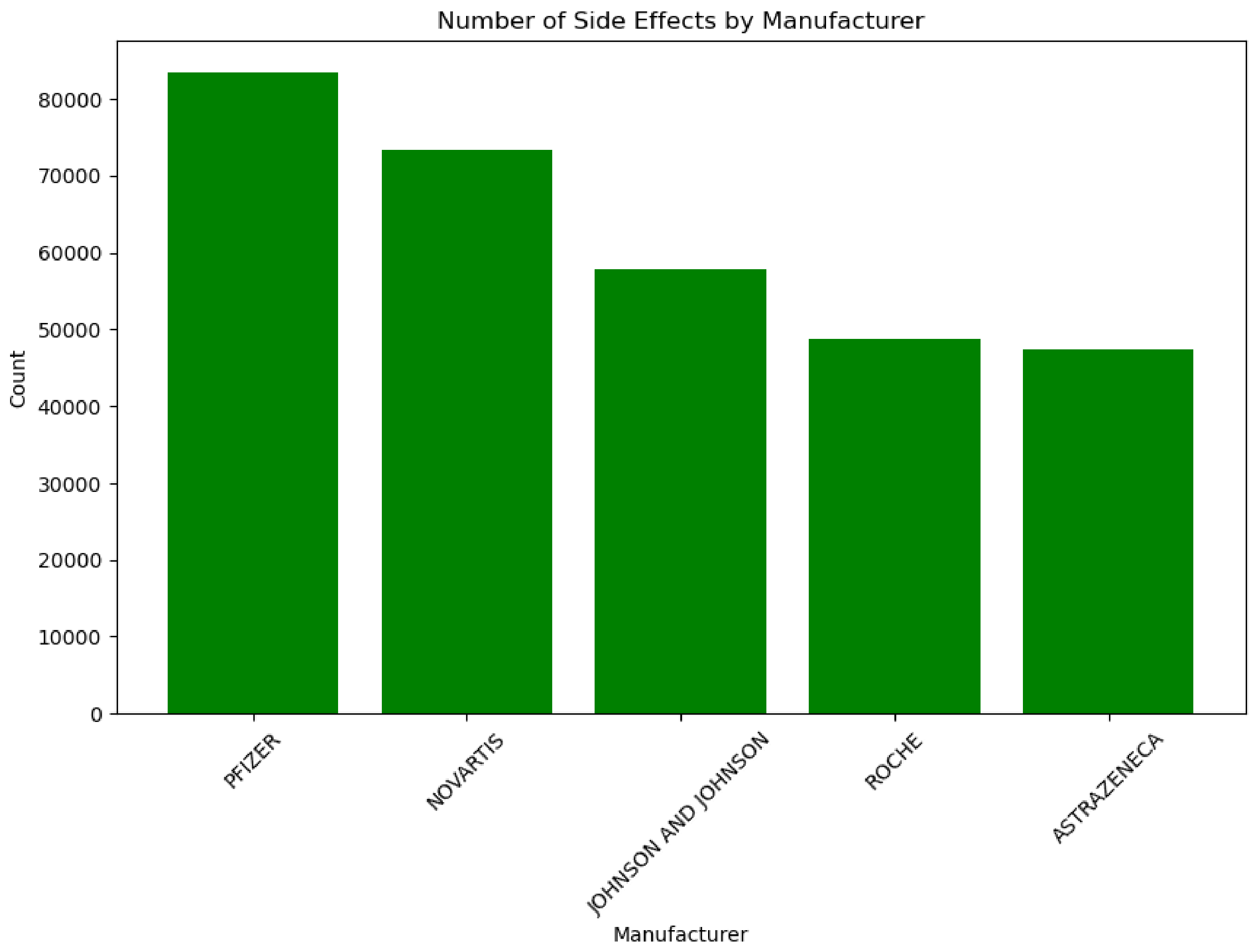


In [62]:
```
# Counting side effects by manufacturer
manufacturer_counts = data['manufacturer'].value_counts().reset_index()
manufacturer_counts.columns = ['manufacturer', 'count']
top5manufacturers=manufacturer_counts.head(5)
top5manufacturers
```

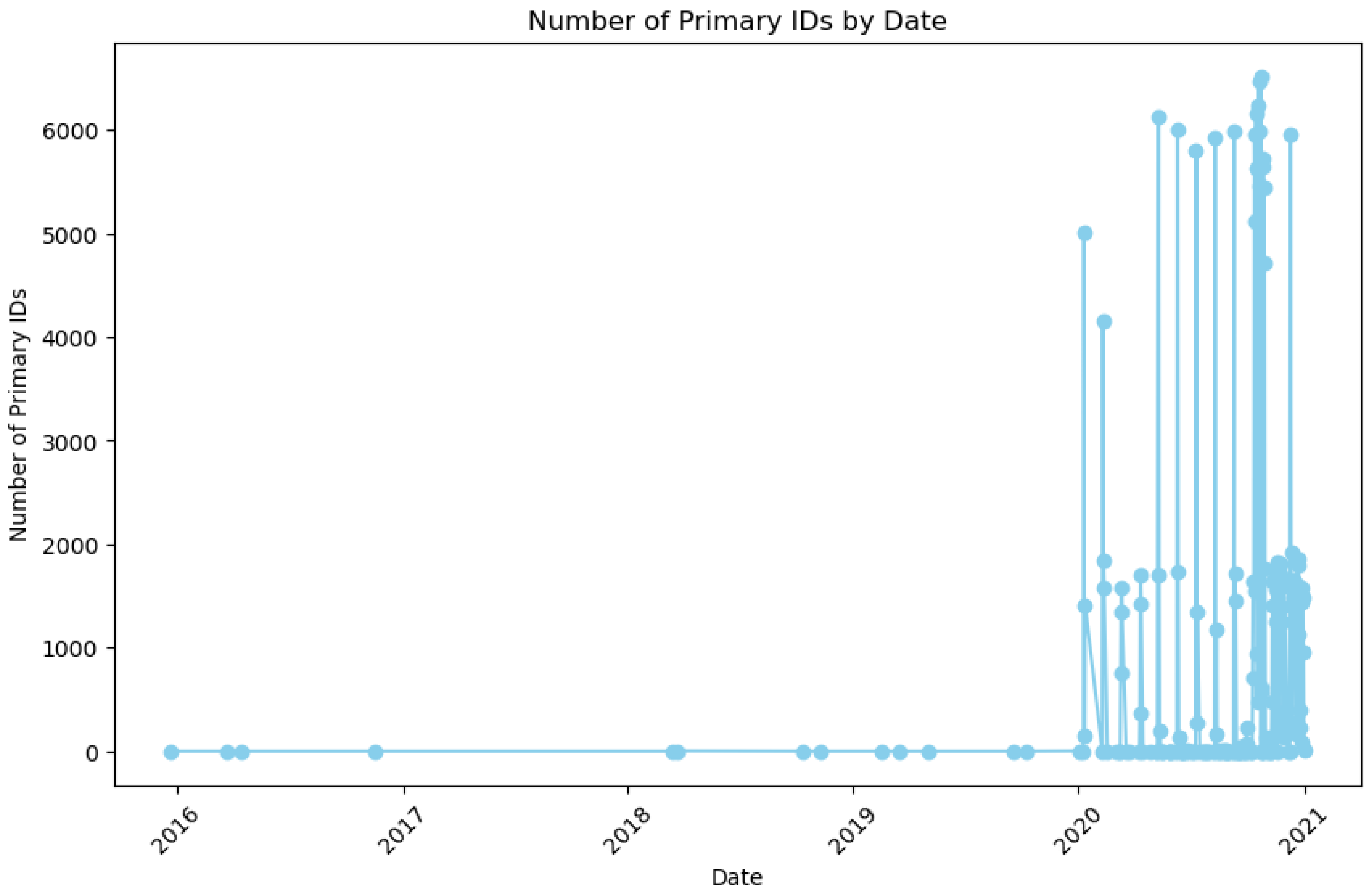
Out[62]:

	manufacturer	count
0	PFIZER	83392
1	NOVARTIS	73450
2	JOHNSON AND JOHNSON	57758
3	ROCHE	48708
4	ASTRAZENECA	47326

In [64]:
```
# Plot side effects by manufacturer
plt.figure(figsize=(10, 6))
plt.bar(top5manufacturers['manufacturer'], top5manufacturers['count'], color='green')
plt.xlabel('Manufacturer')
plt.ylabel('Count')
plt.title('Top five manufacturers who have the most side effects')
plt.xticks(rotation=45)
plt.show()
```

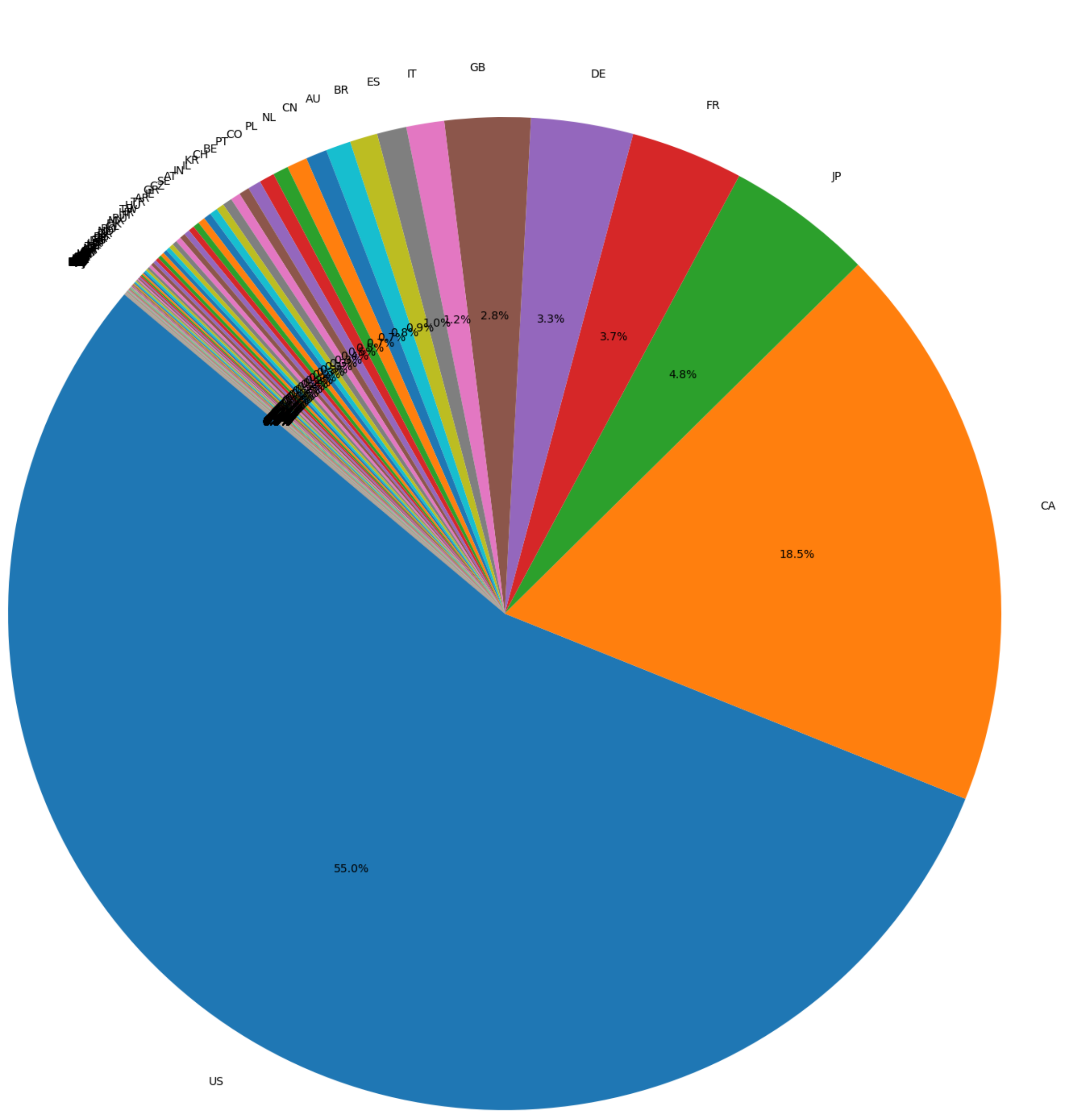


In [66]:
```
# Plotting the number of primaryid by date
date_counts = data.groupby('Date')['primaryid'].nunique().reset_index()
date_counts.columns = ['Date', 'count']
plt.figure(figsize=(10, 6))
plt.plot(date_counts['Date'], date_counts['count'], marker='o', linestyle='-', color='sk
plt.xlabel('Date')
plt.ylabel('Number of Primary IDs')
plt.title('Number of Primary IDs by Date')
plt.xticks(rotation=45)
plt.show()
```



In [68]:
```
# Plotting share of side effects by country
country_counts = data['country'].value_counts(normalize=True).reset_index()
country_counts.columns = ['country', 'share']
plt.figure(figsize=(20, 20))
plt.pie(country_counts['share'], labels=country_counts['country'], autopct='%1.1f%%', st
plt.title('Share of Side Effects by Country')
plt.show()
```

Share of Side Effects by Country



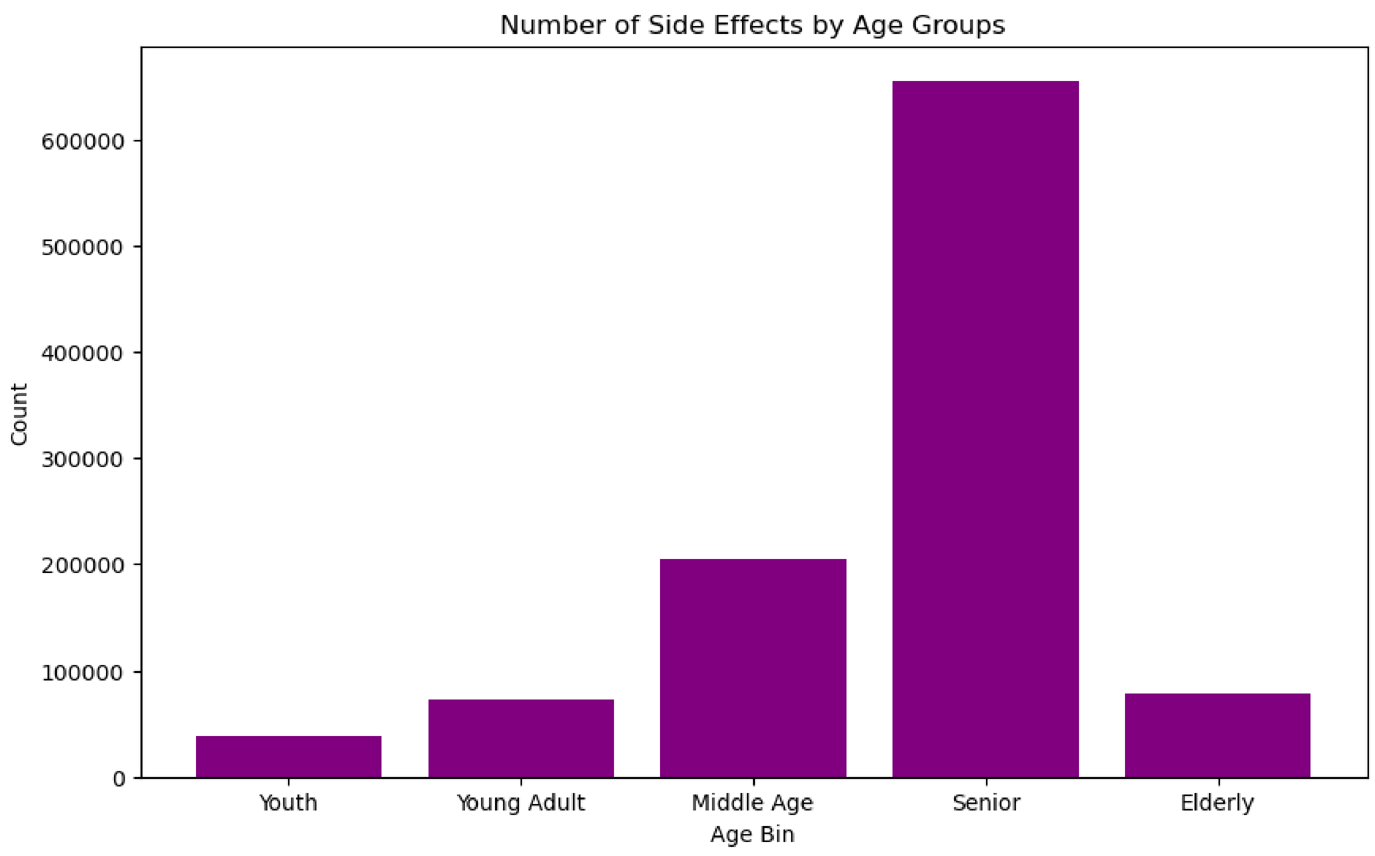
In [71]:
```
# Creating different labels for age
age_min = data['age'].min()
age_max = data['age'].max()
age_bins = [age_min, 20, 40, 60, 80, age_max]
age_labels = ['Youth', 'Young Adult', 'Middle Age', 'Senior', 'Elderly']
data['age_bin'] = pd.cut(data['age'], bins=age_bins, labels=age_labels, right=False)

age_bin_counts = data['age_bin'].value_counts().sort_index().reset_index()
age_bin_counts.columns = ['age_bin', 'count']
age_bin_counts
```

Out[71]:

	age_bin	count
0	Youth	37796
1	Young Adult	72569
2	Middle Age	204914
3	Senior	654883

```
4    Elderly    78412
```

In [74]:
```
# Plotting number ofside effects by age bin
plt.figure(figsize=(10, 6))
plt.bar(age_bin_counts['age_bin'], age_bin_counts['count'], color='purple')
plt.xlabel('Age Bin')
plt.ylabel('Count')
plt.title('Number of Side Effects by Age Groups')
plt.show()
```


Number of Side Effects by Age Groups
Count
Youth
Young Adult
Middle Age
Senior
Elderly
Age Bin