250k_medicine_usage_analysis

September 18, 2024

[1]:
```python
import pandas as pd

# Load the dataset
file_path = (r"C:\Users\Admin\OneDrive\Desktop\Unified Mentor Projects\250k␣
↪Medicines Usage, Side Effects and Substitutes.csv")
df = pd.read_csv(file_path)
```

```
C:\Users\Admin\AppData\Local\Temp\ipykernel_17020\3410007629.py:5: DtypeWarning:
Columns (42,43,44,45,46,47,48) have mixed types. Specify dtype option on import
or set low_memory=False.
  df = pd.read_csv(file_path)
```

[ ]:
```python

```

[3]:
```python
df.head()
```

[3]:
```
   id                     name                      substitute0  \
0   1  augmentin 625 duo tablet  Penciclav 500 mg/125 mg Tablet
1   2       azithral 500 tablet          Zithrocare 500mg Tablet
2   3          ascoril ls syrup                  Solvin LS Syrup
3   4       allegra 120mg tablet                     Lcfex Tablet
4   5             avil 25 tablet                Eralet 25mg Tablet

               substitute1             substitute2             substitute3  \
0  Moxikind-CV 625 Tablet  Moxiforce-CV 625 Tablet     Fightox 625 Tablet
1         Azax 500 Tablet          Zady 500 Tablet  Cazithro 500mg Tablet
2       Ambrodil-LX Syrup        Zerotuss XP Syrup         Capex LS Syrup
3      Etofex 120mg Tablet     Nexofex 120mg Tablet    Fexise 120mg Tablet
4                     NaN                      NaN                    NaN

                substitute4 sideEffect0        sideEffect1     sideEffect2  …  \
0  Novamox CV 625mg Tablet    Vomiting             Nausea        Diarrhea  …
1    Trulimax 500mg Tablet    Vomiting             Nausea  Abdominal pain  …
2           Broxum LS Syrup      Nausea           Vomiting        Diarrhea  …
3      Histafree 120 Tablet    Headache        Drowsiness       Dizziness  …
4                       NaN  Sleepiness  Dryness in mouth             NaN  …

  sideEffect41                                                        use0  \
0          NaN                  Treatment of Bacterial infections
1          NaN                  Treatment of Bacterial infections
2          NaN                      Treatment of Cough with mucus
3          NaN  Treatment of Sneezing and runny nose due to al…
4          NaN                   Treatment of Allergic conditions

                              use1 use2 use3 use4  \
0                              NaN  NaN  NaN  NaN
1                              NaN  NaN  NaN  NaN
2                              NaN  NaN  NaN  NaN
3  Treatment of Allergic conditions  NaN  NaN  NaN
4                              NaN  NaN  NaN  NaN

              Chemical Class Habit Forming Therapeutic Class  \
0                        NaN            No   ANTI INFECTIVES
1                 Macrolides            No   ANTI INFECTIVES
2                        NaN            No       RESPIRATORY
3  Diphenylmethane Derivative           No       RESPIRATORY
4      Pyridines Derivatives            No       RESPIRATORY

                              Action Class
0                                      NaN
1                               Macrolides
2                                      NaN
3  H1 Antihistaminics (second Generation)
4   H1 Antihistaminics (First Generation)

[5 rows x 58 columns]
```

[5]:
```python
# Display basic info about the dataset
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 248218 entries, 0 to 248217
Data columns (total 58 columns):
 #   Column              Non-Null Count   Dtype
---  ------              --------------   -----
 0   id                  248218 non-null  int64
 1   name                248218 non-null  object
 2   substitute0         238621 non-null  object
 3   substitute1         233867 non-null  object
 4   substitute2         230233 non-null  object
 5   substitute3         226856 non-null  object
 6   substitute4         223962 non-null  object
 7   sideEffect0         248218 non-null  object
 8   sideEffect1         238416 non-null  object
 9   sideEffect2         229500 non-null  object
 10  sideEffect3         207638 non-null  object
 11  sideEffect4        163560 non-null  object
 12  sideEffect5        131258 non-null  object
 13  sideEffect6        91857 non-null   object
 14  sideEffect7        67750 non-null   object
 15  sideEffect8        48506 non-null   object
 16  sideEffect9        37708 non-null   object
 17  sideEffect10       27274 non-null   object
 18  sideEffect11       20331 non-null   object
 19  sideEffect12       16282 non-null   object
 20  sideEffect13       14727 non-null   object
 21  sideEffect14       10419 non-null   object
 22  sideEffect15       7681 non-null    object
 23  sideEffect16       6009 non-null    object
 24  sideEffect17       5382 non-null    object
 25  sideEffect18       4515 non-null    object
 26  sideEffect19       3946 non-null    object
 27  sideEffect20       3223 non-null    object
 28  sideEffect21       3125 non-null    object
 29  sideEffect22       3048 non-null    object
 30  sideEffect23       2905 non-null    object
 31  sideEffect24       2723 non-null    object
 32  sideEffect25       1503 non-null    object
 33  sideEffect26       1503 non-null    object
 34  sideEffect27       1494 non-null    object
 35  sideEffect28       1494 non-null    object
 36  sideEffect29       1438 non-null    object
 37  sideEffect30       1329 non-null    object
 38  sideEffect31       1329 non-null    object
 39  sideEffect32       1328 non-null    object
 40  sideEffect33       1169 non-null    object
 41  sideEffect34       1166 non-null    object
 42  sideEffect35       2 non-null       object
 43  sideEffect36       2 non-null       object
 44  sideEffect37       2 non-null       object
 45  sideEffect38       2 non-null       object
 46  sideEffect39       2 non-null       object
 47  sideEffect40       2 non-null       object
 48  sideEffect41       2 non-null       object
 49  use0               248218 non-null  object
 50  use1               73365 non-null   object
 51  use2               28307 non-null   object
 52  use3               7379 non-null    object
 53  use4               4971 non-null    object
 54  Chemical Class     137791 non-null  object
 55  Habit Forming      248218 non-null  object
 56  Therapeutic Class  248149 non-null  object
 57  Action Class       138036 non-null  object
dtypes: int64(1), object(57)
```

```
memory usage: 109.8+ MB
```

[7]:
```
df.describe()
```

[7]:
```
                  id
count  248218.000000
mean   124109.500000
std     71654.508896
min         1.000000
25%     62055.250000
50%    124109.500000
75%    186163.750000
max    248218.000000
```

[9]:
```
print(df.isnull().sum())
```

```
id                   0
name                 0
substitute0       9597
substitute1      14351
substitute2      17985
substitute3      21362
substitute4      24256
sideEffect0          0
sideEffect1       9802
sideEffect2      18718
sideEffect3      40580
sideEffect4      84658
sideEffect5     116960
sideEffect6     156361
sideEffect7     180468
sideEffect8     199712
sideEffect9     210510
sideEffect10    220944
sideEffect11    227887
sideEffect12    231936
sideEffect13    233491
sideEffect14    237799
sideEffect15    240537
sideEffect16    242209
sideEffect17    242836
sideEffect18    243703
sideEffect19    244272
sideEffect20    244995
sideEffect21    245093
sideEffect22    245170
sideEffect23    245313
sideEffect24    245495
```

```
sideEffect25         246715
sideEffect26         246715
sideEffect27         246724
sideEffect28         246724
sideEffect29         246780
sideEffect30         246889
sideEffect31         246889
sideEffect32         246890
sideEffect33         247049
sideEffect34         247052
sideEffect35         248216
sideEffect36         248216
sideEffect37         248216
sideEffect38         248216
sideEffect39         248216
sideEffect40         248216
sideEffect41         248216
use0                      0
use1                 174853
use2                 219911
use3                 240839
use4                 243247
Chemical Class       110427
Habit Forming             0
Therapeutic Class        69
Action Class         110182
dtype: int64
```

[11]:
```
# Fill missing values for substitutes and side effects
substitute_cols = [f'substitute{i}' for i in range(5)]
side_effect_cols = [f'sideEffect{i}' for i in range(42)]

df[substitute_cols] = df[substitute_cols].fillna('No substitute available')
df[side_effect_cols] = df[side_effect_cols].fillna('No known side effects')

# Fill missing values in 'Habit Forming' column
df['Habit Forming'] = df['Habit Forming'].fillna('NO')
```

[13]:
```
# Fill missing usage columns with 'Not specified'
usage_cols = [f'use{i}' for i in range(5)]
df[usage_cols] = df[usage_cols].fillna('Not specified')
```

[15]:
```
# Handle missing values in 'Chemical Class', 'Therapeutic Class', and 'Action␣
 ↪Class' with 'Unknown'
df['Chemical Class'] = df['Chemical Class'].fillna('Unknown')
df['Therapeutic Class'] = df['Therapeutic Class'].fillna('Unknown')
df['Action Class'] = df['Action Class'].fillna('Unknown')
```

```
# Verify that there are no more missing values in these columns
missing_values_summary = df[['Chemical Class', 'Therapeutic Class', 'Action␣
↪Class']].isnull().sum()
print(missing_values_summary)

# Check if missing values are handled properly
missing_summary = df.isnull().sum()
missing_summary
```

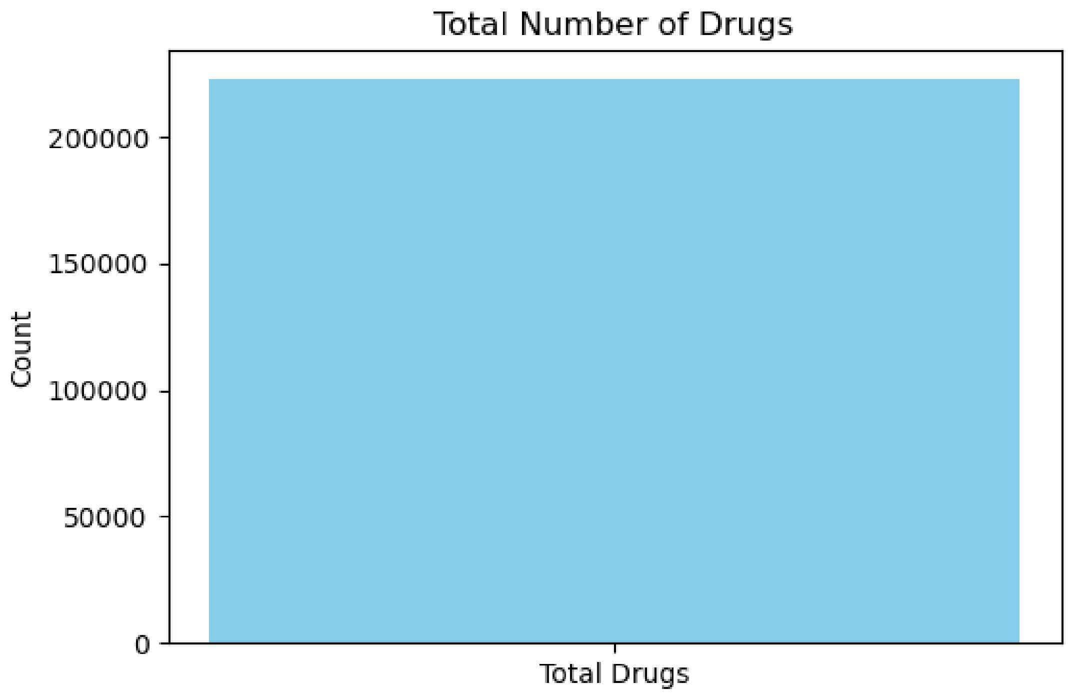
```
Chemical Class       0
Therapeutic Class    0
Action Class         0
dtype: int64
```

[15]:
```
id                   0
name                 0
substitute0          0
substitute1          0
substitute2          0
substitute3          0
substitute4          0
sideEffect0          0
sideEffect1          0
sideEffect2          0
sideEffect3          0
sideEffect4          0
sideEffect5          0
sideEffect6          0
sideEffect7          0
sideEffect8          0
sideEffect9          0
sideEffect10         0
sideEffect11         0
sideEffect12         0
sideEffect13         0
sideEffect14         0
sideEffect15         0
sideEffect16         0
sideEffect17         0
sideEffect18         0
sideEffect19         0
sideEffect20         0
sideEffect21         0
sideEffect22         0
sideEffect23         0
sideEffect24         0
```

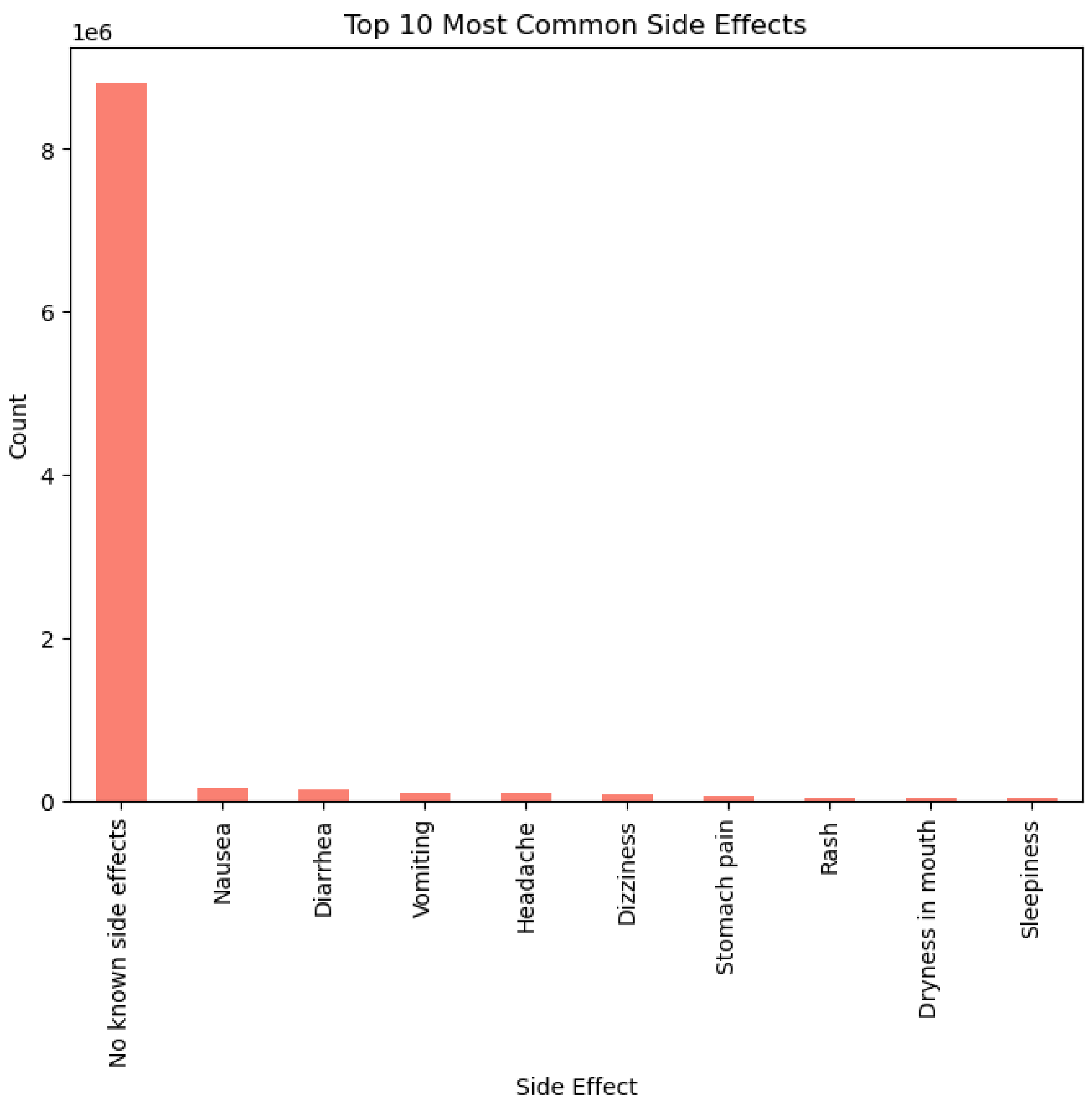
```
sideEffect25        0
sideEffect26        0
sideEffect27        0
sideEffect28        0
sideEffect29        0
sideEffect30        0
sideEffect31        0
sideEffect32        0
sideEffect33        0
sideEffect34        0
sideEffect35        0
sideEffect36        0
sideEffect37        0
sideEffect38        0
sideEffect39        0
sideEffect40        0
sideEffect41        0
use0                0
use1                0
use2                0
use3                0
use4                0
Chemical Class      0
Habit Forming       0
Therapeutic Class   0
Action Class        0
dtype: int64
```

[19]:
```
# Total number of unique drugs
total_drugs = df['name'].nunique()
print(f"Total number of drugs: {total_drugs}")

import matplotlib.pyplot as plt

plt.figure(figsize=(6,4))
plt.bar(['Total Drugs'], [total_drugs], color='skyblue')
plt.title('Total Number of Drugs')
plt.ylabel('Count')
plt.show()
```

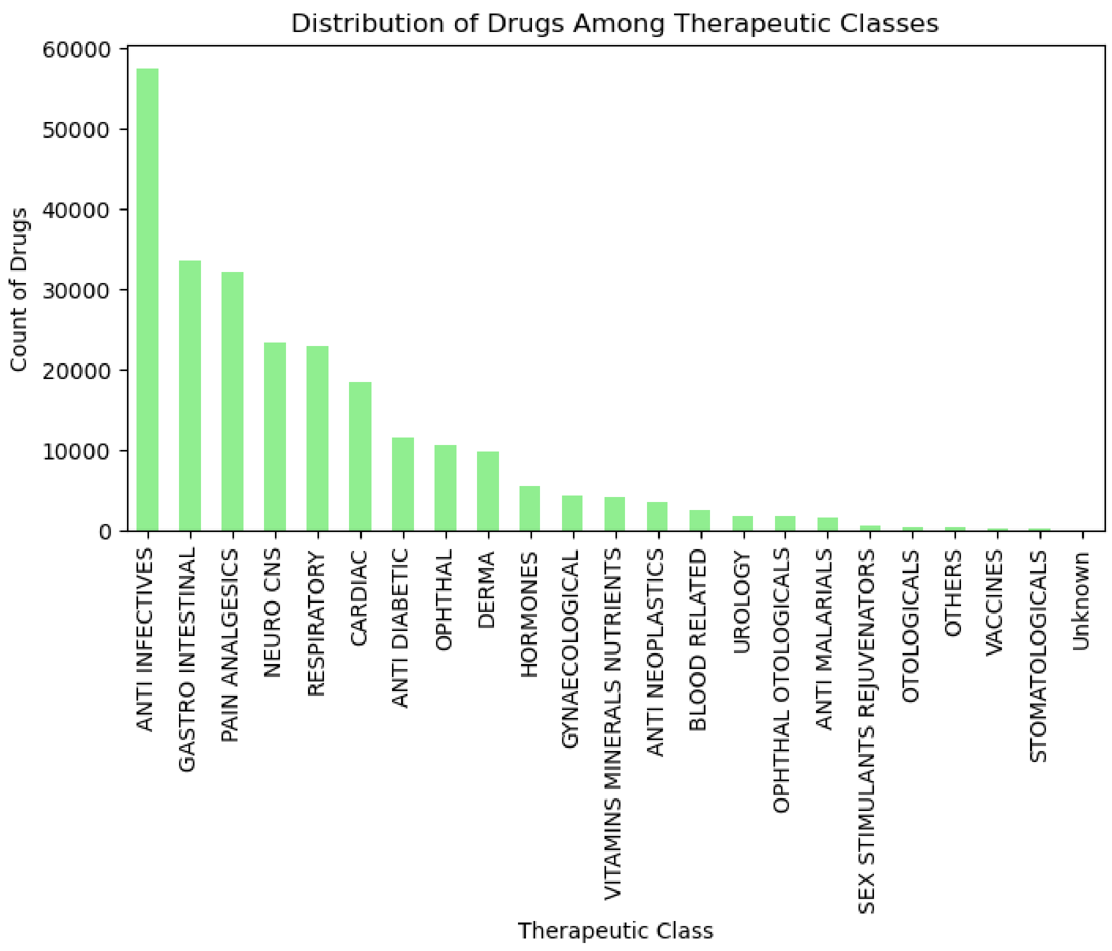
```
Total number of drugs: 222825
```



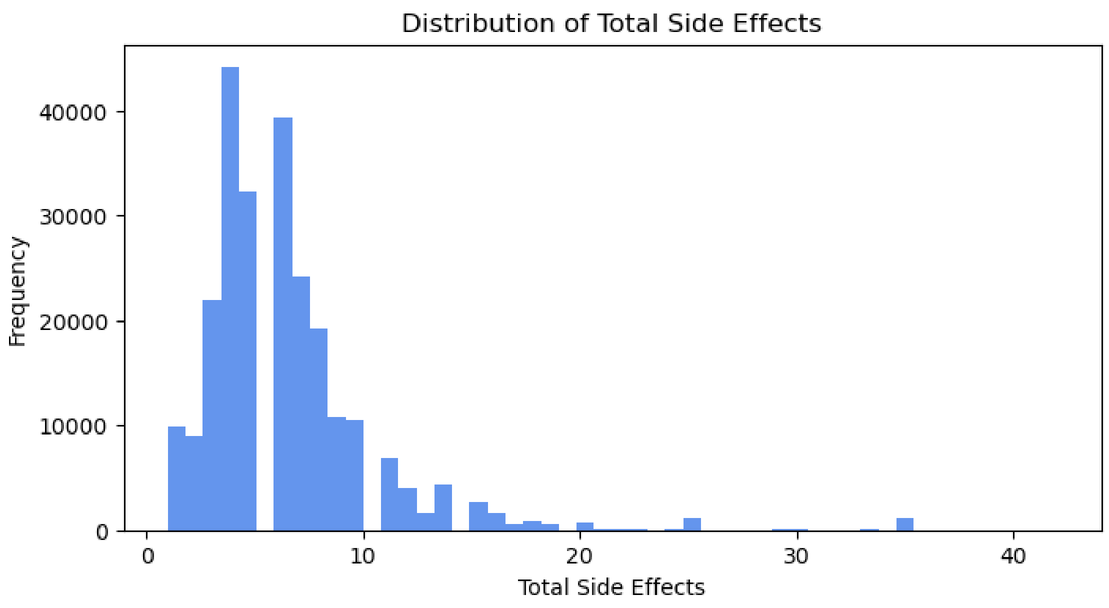
```
[21]: # Combining all side effect columns
      side_effect_columns = [col for col in df.columns if 'sideEffect' in col]
      side_effects = pd.Series(df[side_effect_columns].values.ravel()).dropna()

      # Getting the top 10 most common side effects
      common_side_effects = side_effects.value_counts().head(10)

      # Plotting the most common side effects
      plt.figure(figsize=(8,6))
      common_side_effects.plot(kind='bar', color='salmon')
      plt.title('Top 10 Most Common Side Effects')
      plt.xlabel('Side Effect')
      plt.ylabel('Count')
      plt.xticks()
      plt.show()
```



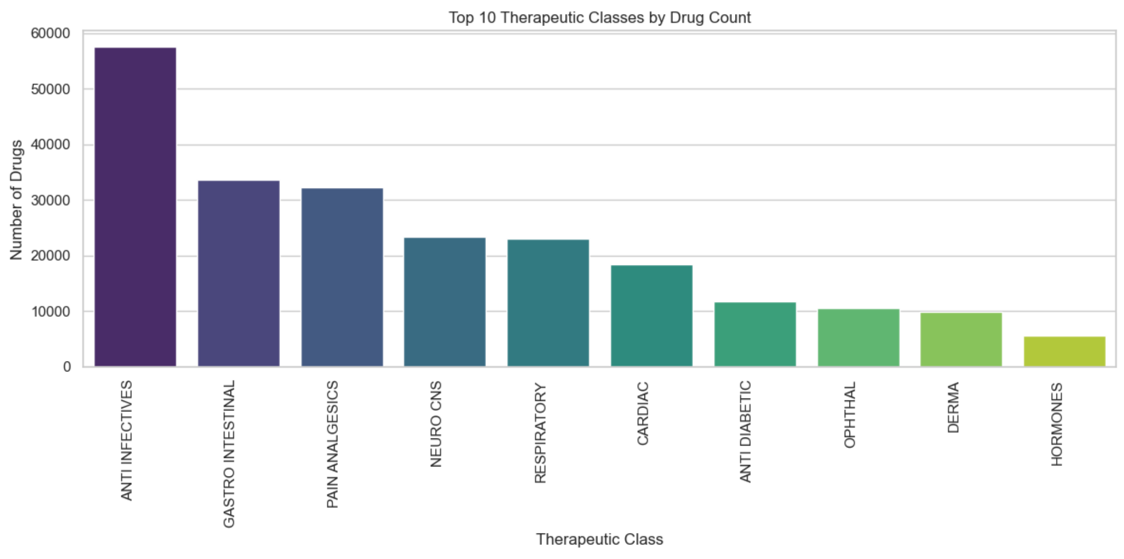
[29]:
```python
# Distribution of drugs among therapeutic classes
# Distribution of drugs among therapeutic classes
therapeutic_class_distribution = df['Therapeutic Class'].value_counts()

# Plotting the distribution
plt.figure(figsize=(8,4))
therapeutic_class_distribution.plot(kind='bar', color='lightgreen')
plt.title('Distribution of Drugs Among Therapeutic Classes')
plt.xlabel('Therapeutic Class')
plt.ylabel('Count of Drugs')
plt.xticks(rotation=90)
plt.show()
```



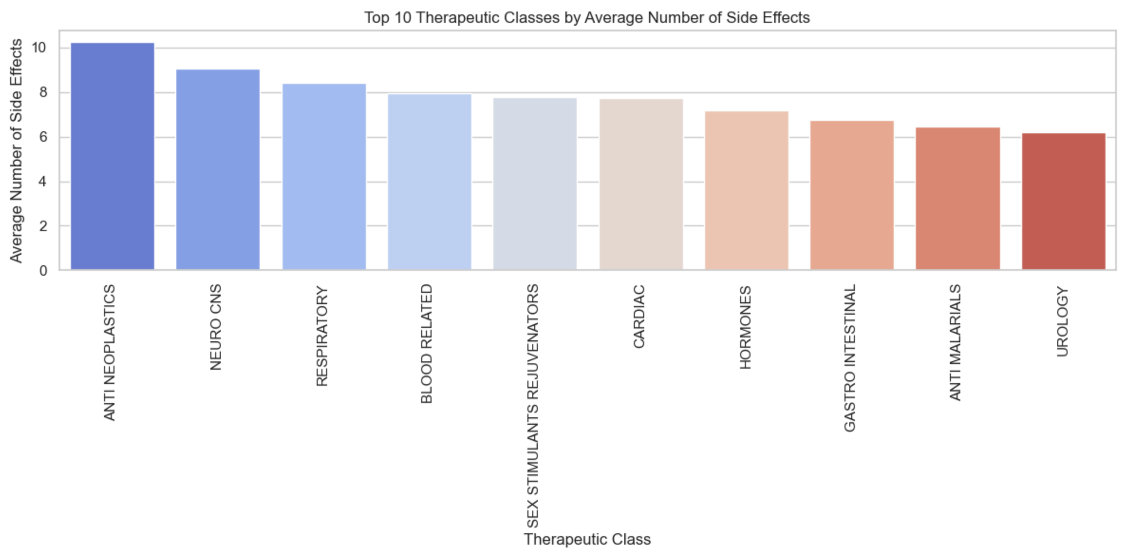
[31]:
```
# List of side effect columns
side_effect_cols = [col for col in df.columns if 'sideEffect' in col]

# Creating the 'side_effect_count' column by counting non-'No known side␣
↪effects' entries
df['side_effect_count'] = df[side_effect_cols].apply(lambda row: row[row != 'No␣
↪known side effects'].count(), axis=1)


# Distribution of total side effects
plt.figure(figsize=(8,4))
plt.hist(df['side_effect_count'], bins=50, color='cornflowerblue')
plt.title('Distribution of Total Side Effects')
plt.xlabel('Total Side Effects')
plt.ylabel('Frequency')
plt.show()
```



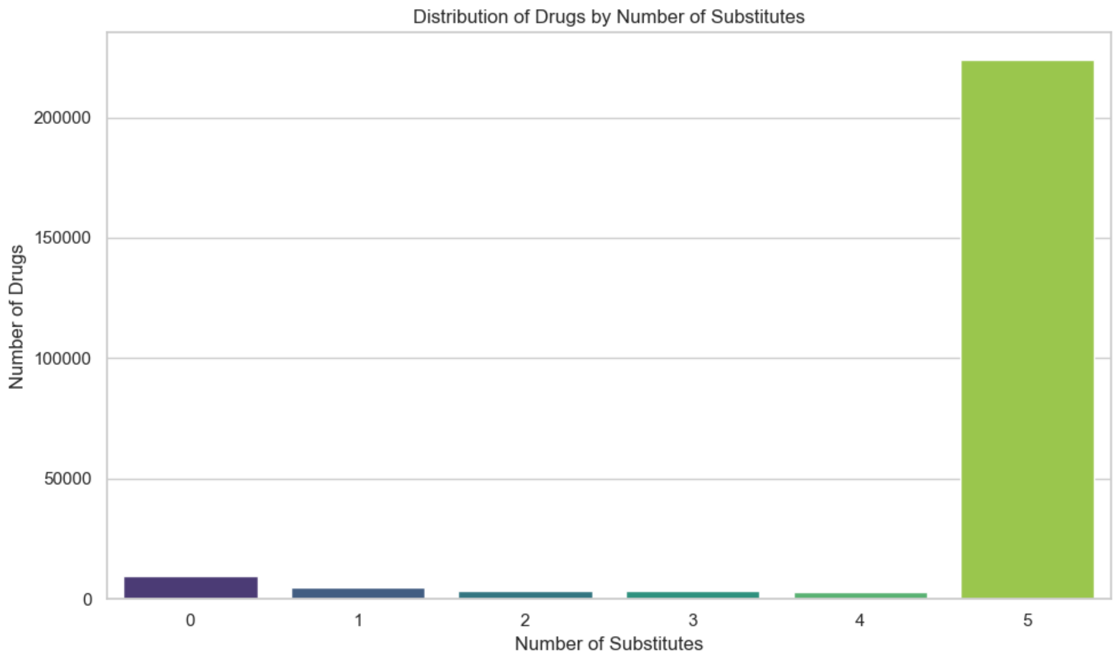
[41]:
```python
import matplotlib.pyplot as plt
import seaborn as sns

# Set plot styles for better visuals
sns.set(style="whitegrid")
plt.figure(figsize=(12, 6))

# Step 1: Distribution of drugs by Therapeutic Class
therapeutic_class_counts = df['Therapeutic Class'].value_counts().head(10)  #␣
 ↪Top 10 therapeutic classes
sns.barplot(x=therapeutic_class_counts.index, y=therapeutic_class_counts.
 ↪values, palette="viridis")
plt.title('Top 10 Therapeutic Classes by Drug Count')
plt.ylabel('Number of Drugs')
plt.xlabel('Therapeutic Class')
plt.xticks(rotation=90, ha='right')
plt.tight_layout()
plt.show()
```



[43]:
```
# Analyzing Average Side Effects Per Therapeutic Class
# Count the number of non-empty side effects for each drug
df['side_effect_count'] = df[side_effect_cols].apply(lambda row: row[row != 'No␣
↪known side effects'].count(), axis=1)

# Calculate the average side effects per therapeutic class
avg_side_effects_per_class = df.groupby('Therapeutic␣
↪Class')['side_effect_count'].mean().sort_values(ascending=False)

# Plot the result
plt.figure(figsize=(12, 6))
sns.barplot(x=avg_side_effects_per_class.index[:10],␣
↪y=avg_side_effects_per_class.values[:10], palette="coolwarm")
plt.title('Top 10 Therapeutic Classes by Average Number of Side Effects')
plt.ylabel('Average Number of Side Effects')
plt.xlabel('Therapeutic Class')
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```



[47]:
```
# Analyzing the Availability of Substitutes
# Count the number of available substitutes for each drug
df['substitute_count'] = df[substitute_cols].apply(lambda row: row[row != 'No␣
↪substitute available'].count(), axis=1)

# Analyze how many drugs have substitutes
substitute_availability = df['substitute_count'].value_counts()

# Plot
plt.figure(figsize=(10, 6))
sns.barplot(x=substitute_availability.index, y=substitute_availability.values,␣
↪palette="viridis")
plt.title('Distribution of Drugs by Number of Substitutes')
plt.ylabel('Number of Drugs')
plt.xlabel('Number of Substitutes')
plt.tight_layout()
plt.show()
```



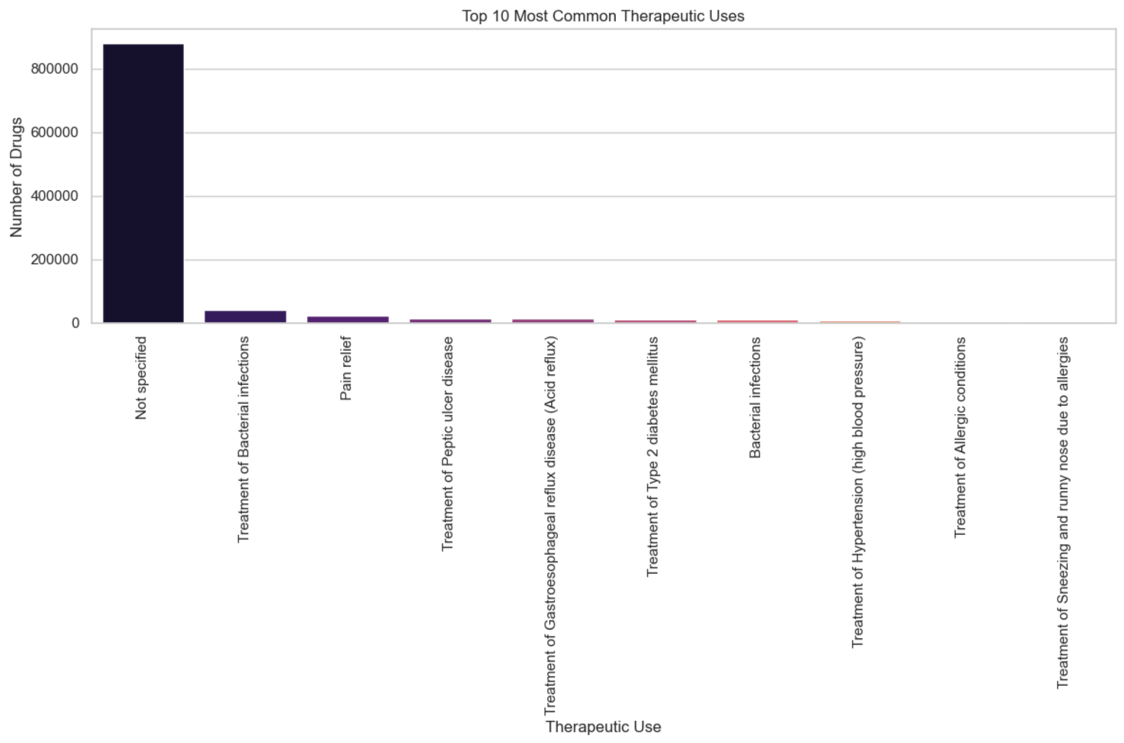
```
[ ]: # Check unique values in the "Habit Forming" column
     print(df['Habit Forming'].unique())
```

```
[ ]: # Clean the 'Habit Forming' column by replacing unexpected values
     df['Habit Forming'] = df['Habit Forming'].str.upper()  # Make all values␣
      ↪uppercase
     df['Habit Forming'] = df['Habit Forming'].fillna('NO')  # Fill NaN values with␣
      ↪'NO'

     # If there are any other specific variations (e.g. "Y", "N"), handle them
     df['Habit Forming'] = df['Habit Forming'].replace({'Y': 'YES', 'N': 'NO'})
```

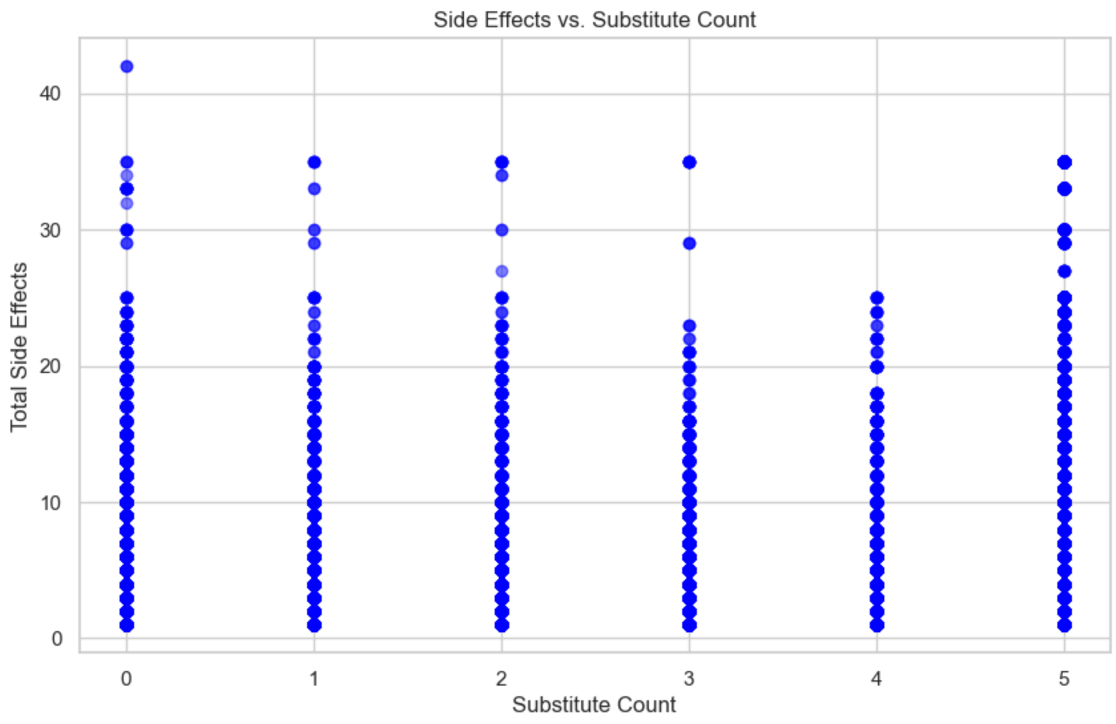
```
[55]: # Count of drugs with substitutes
      drugs_with_substitutes = df[df['substitute_count'] > 0].shape[0]

      # Count of habit-forming drugs
      habit_forming_drugs = df[df['Habit Forming'] == 'YES'].shape[0]

      print(f"Drugs with substitutes: {drugs_with_substitutes}")
      print(f"Habit-forming drugs: {habit_forming_drugs}")
```

```
Drugs with substitutes: 238621
Habit-forming drugs: 6003
```

[70]:
```python
#Exploring Usage Patterns
# Combine all use columns into a single column for analysis
all_uses = pd.concat([df[f'use{i}'] for i in range(5)]).value_counts().head(10)

# Plot the top 10 most common therapeutic uses
plt.figure(figsize=(12, 8))
sns.barplot(x=all_uses.index, y=all_uses.values, palette="magma")
plt.title('Top 10 Most Common Therapeutic Uses')
plt.ylabel('Number of Drugs')
plt.xlabel('Therapeutic Use')
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```



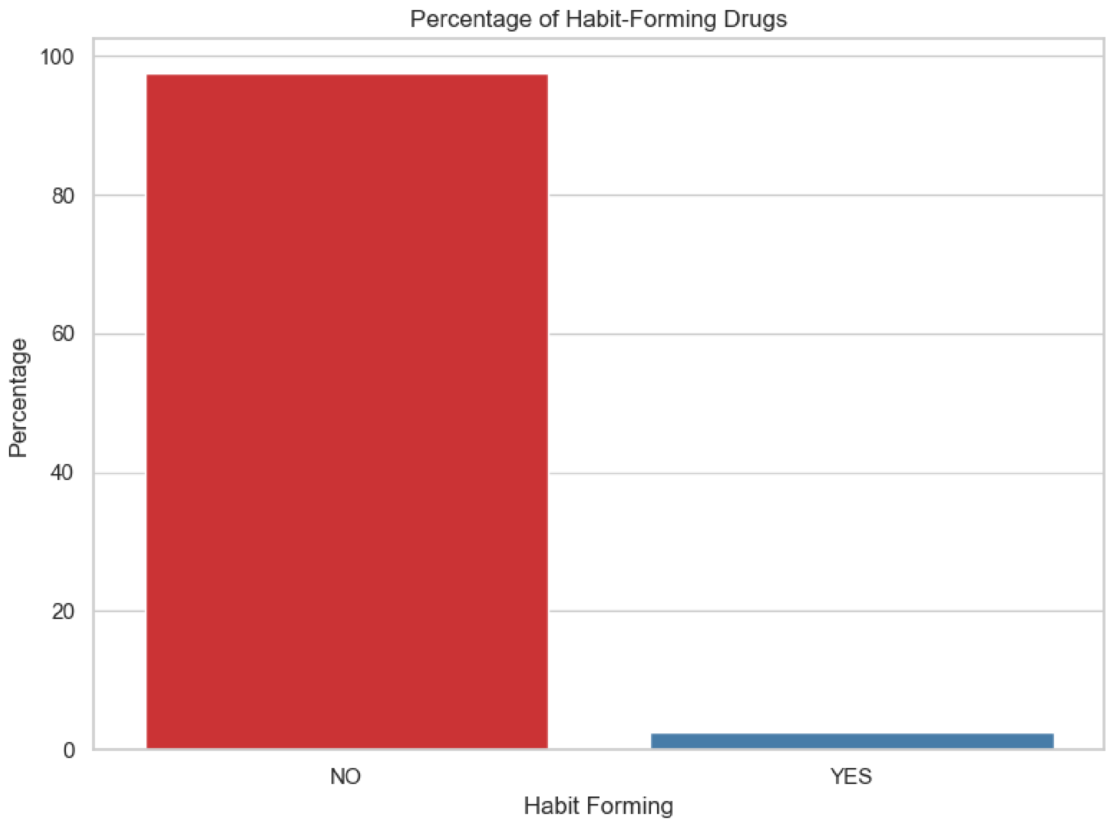
[72]:
```python
# Relationship between side effect count and substitutes
plt.figure(figsize=(10,6))
plt.scatter(df['substitute_count'], df['side_effect_count'], alpha=0.5,␣
 ↪color='blue')
plt.title('Side Effects vs. Substitute Count')
plt.xlabel('Substitute Count')
plt.ylabel('Total Side Effects')
plt.show()
```



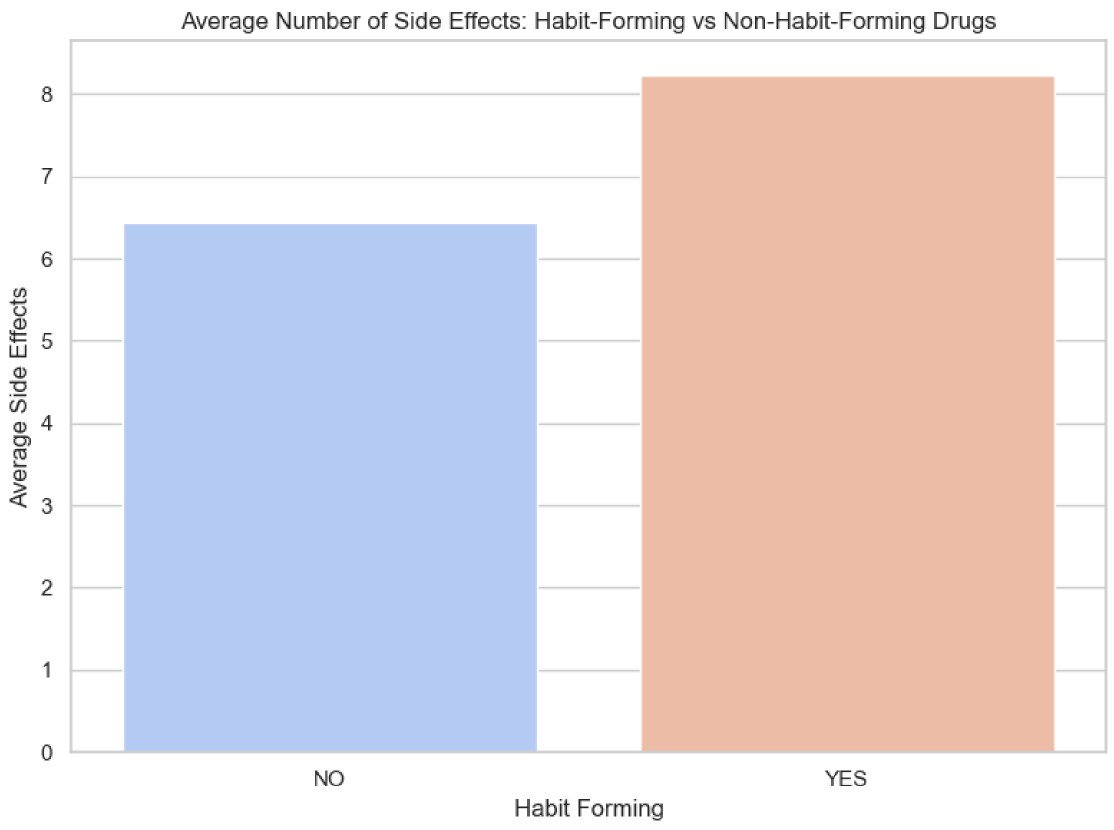
[74]:
```
# Habit-forming drugs with and without substitutes
habit_with_substitutes = df[(df['Habit Forming'] == 'YES') &␣
 ↪(df['substitute_count'] > 0)].shape[0]
habit_without_substitutes = df[(df['Habit Forming'] == 'YES') &␣
 ↪(df['substitute_count'] == 0)].shape[0]

print(f"Habit-forming drugs with substitutes: {habit_with_substitutes}")
print(f"Habit-forming drugs without substitutes: {habit_without_substitutes}")
```

```
Habit-forming drugs with substitutes: 5756
Habit-forming drugs without substitutes: 247
```

[78]:
```
# Habit-Forming Drug Analysis
# Calculate the percentage of habit-forming drugs
habit_forming_percentage = df['Habit Forming'].value_counts(normalize=True) *␣
 ↪100

# Plot the result
plt.figure(figsize=(8, 6))
sns.barplot(x=habit_forming_percentage.index, y=habit_forming_percentage.
 ↪values, palette="Set1")
plt.title('Percentage of Habit-Forming Drugs')
plt.ylabel('Percentage')
plt.xlabel('Habit Forming')
plt.tight_layout()
plt.show()
```

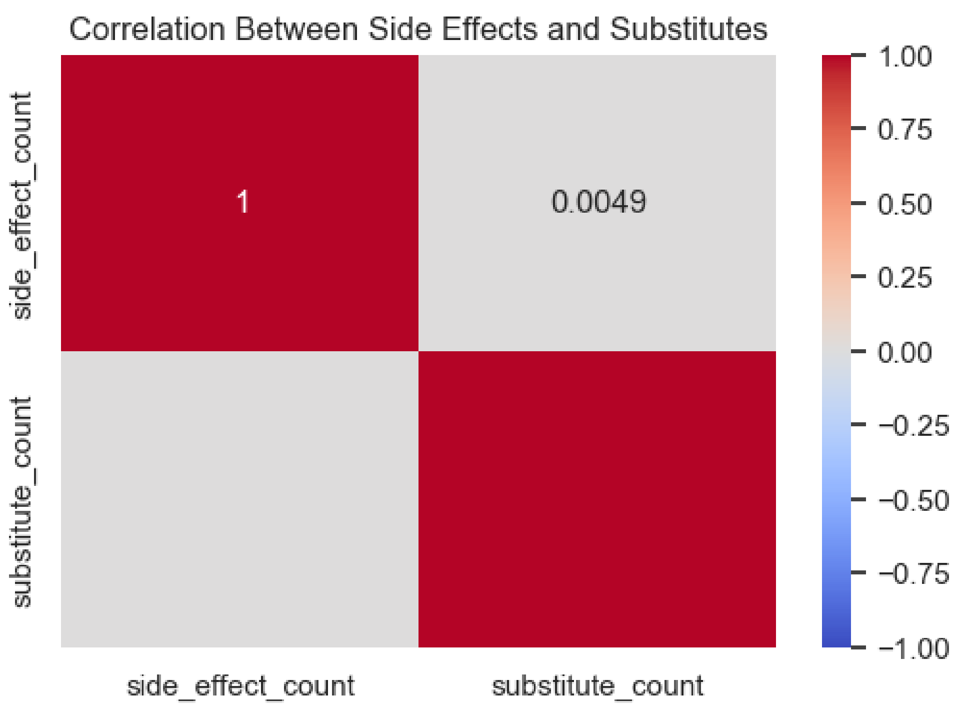


[80]:
```
# Compare average side effects for habit-forming vs non-habit-forming drugs
side_effects_habit = df.groupby('Habit Forming')['side_effect_count'].mean()

# Plot
plt.figure(figsize=(8, 6))
sns.barplot(x=side_effects_habit.index, y=side_effects_habit.values,␣
 ↪palette="coolwarm")
plt.title('Average Number of Side Effects: Habit-Forming vs Non-Habit-Forming␣
 ↪Drugs')
plt.ylabel('Average Side Effects')
plt.xlabel('Habit Forming')
plt.tight_layout()
plt.show()
```

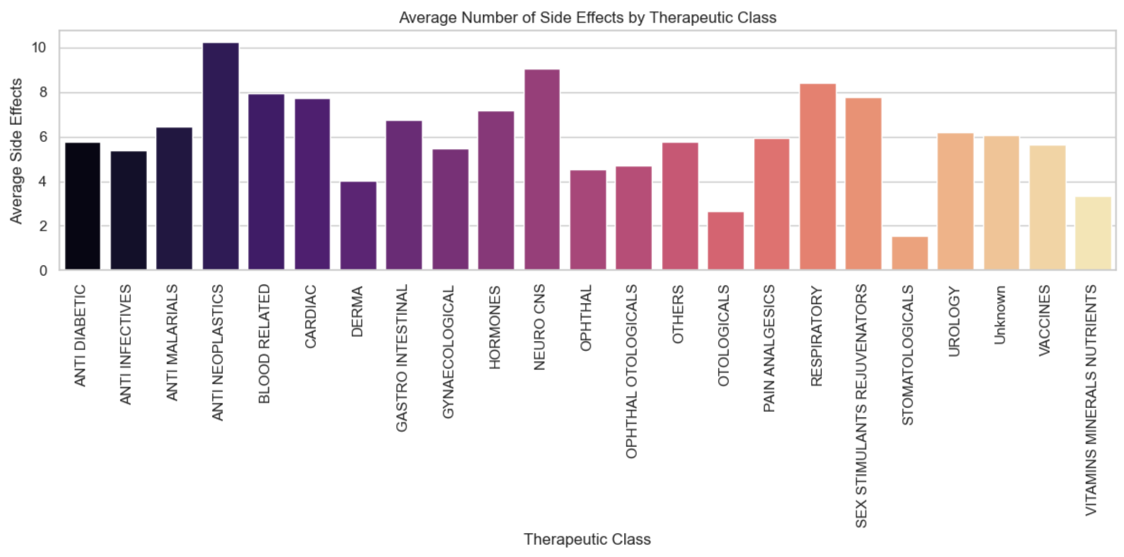


[82]:
```
# Calculate correlation between numerical columns
correlation_matrix = df[['side_effect_count', 'substitute_count']].corr()

# Display the correlation matrix
print(correlation_matrix)

# Plot the correlation heatmap
plt.figure(figsize=(6, 4))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', vmin=-1, vmax=1)
plt.title('Correlation Between Side Effects and Substitutes')
plt.show()
```

```
                   side_effect_count  substitute_count
side_effect_count           1.000000          0.004866
substitute_count            0.004866          1.000000
```



```
[86]: # Group by Therapeutic Class and calculate the average side effects
      avg_side_effects_by_class = df.groupby('Therapeutic␣
       ↪Class')['side_effect_count'].mean()

      # Plot the results
      plt.figure(figsize=(12, 6))
      sns.barplot(x=avg_side_effects_by_class.index, y=avg_side_effects_by_class.
       ↪values, palette="magma")
      plt.title('Average Number of Side Effects by Therapeutic Class')
      plt.xticks(rotation=90)
      plt.ylabel('Average Side Effects')
      plt.tight_layout()
      plt.show()
```



[88]:
```
# Simplify Therapeutic Class by grouping rare classes into 'Other'
threshold = 100  # Adjust based on your dataset
therapeutic_class_counts = df['Therapeutic Class'].value_counts()

# Create a new column 'Therapeutic Class Simplified'
df['Therapeutic Class Simplified'] = df['Therapeutic Class'].apply(
    lambda x: x if therapeutic_class_counts[x] >= threshold else 'Other'
)

# Verify the column exists and has values
print(df['Therapeutic Class Simplified'].head())
```

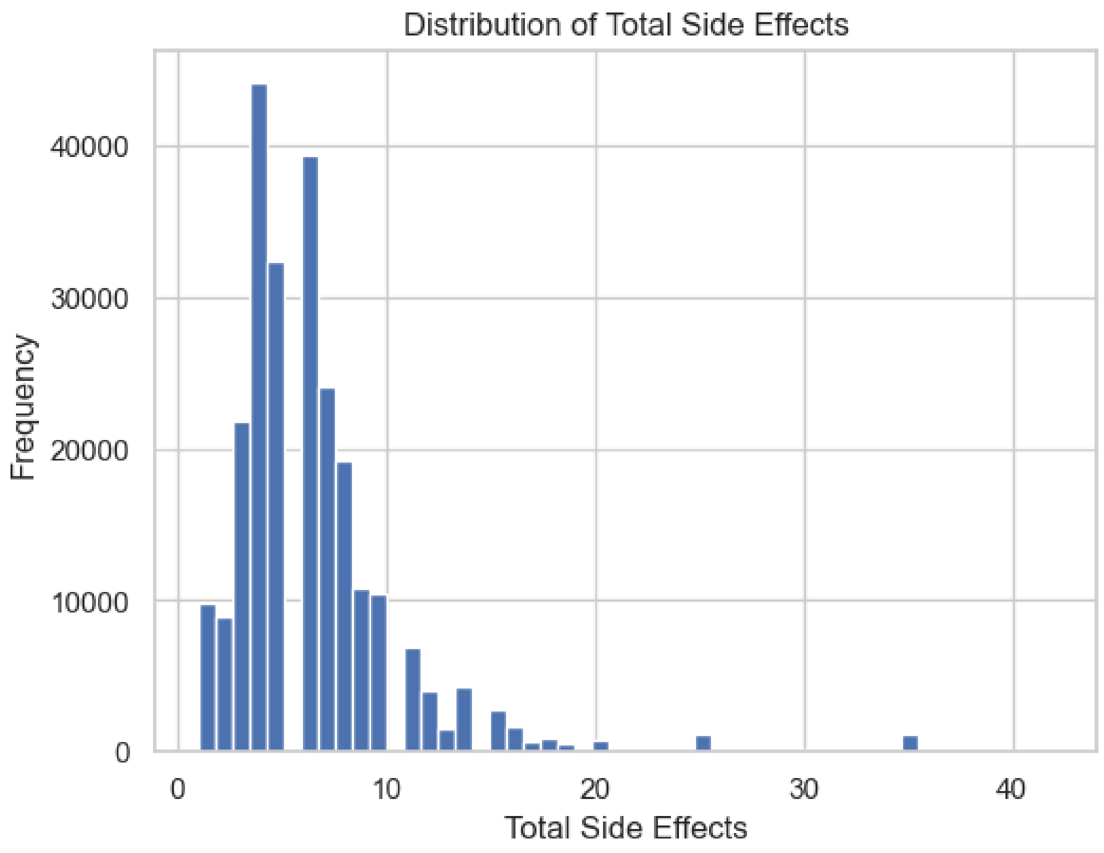
```
0    ANTI INFECTIVES
1    ANTI INFECTIVES
2        RESPIRATORY
3        RESPIRATORY
4        RESPIRATORY
Name: Therapeutic Class Simplified, dtype: object
```

[92]:
```
# Create a new column 'total_side_effects' by summing up the side effect columns
df['total_side_effects'] = df.filter(like='side_effect').sum(axis=1)

# Check the distribution of the target variable
print(df['total_side_effects'].describe())
```

```
count    248218.000000
mean          6.485299
std           4.199711
min           1.000000
25%           4.000000
```

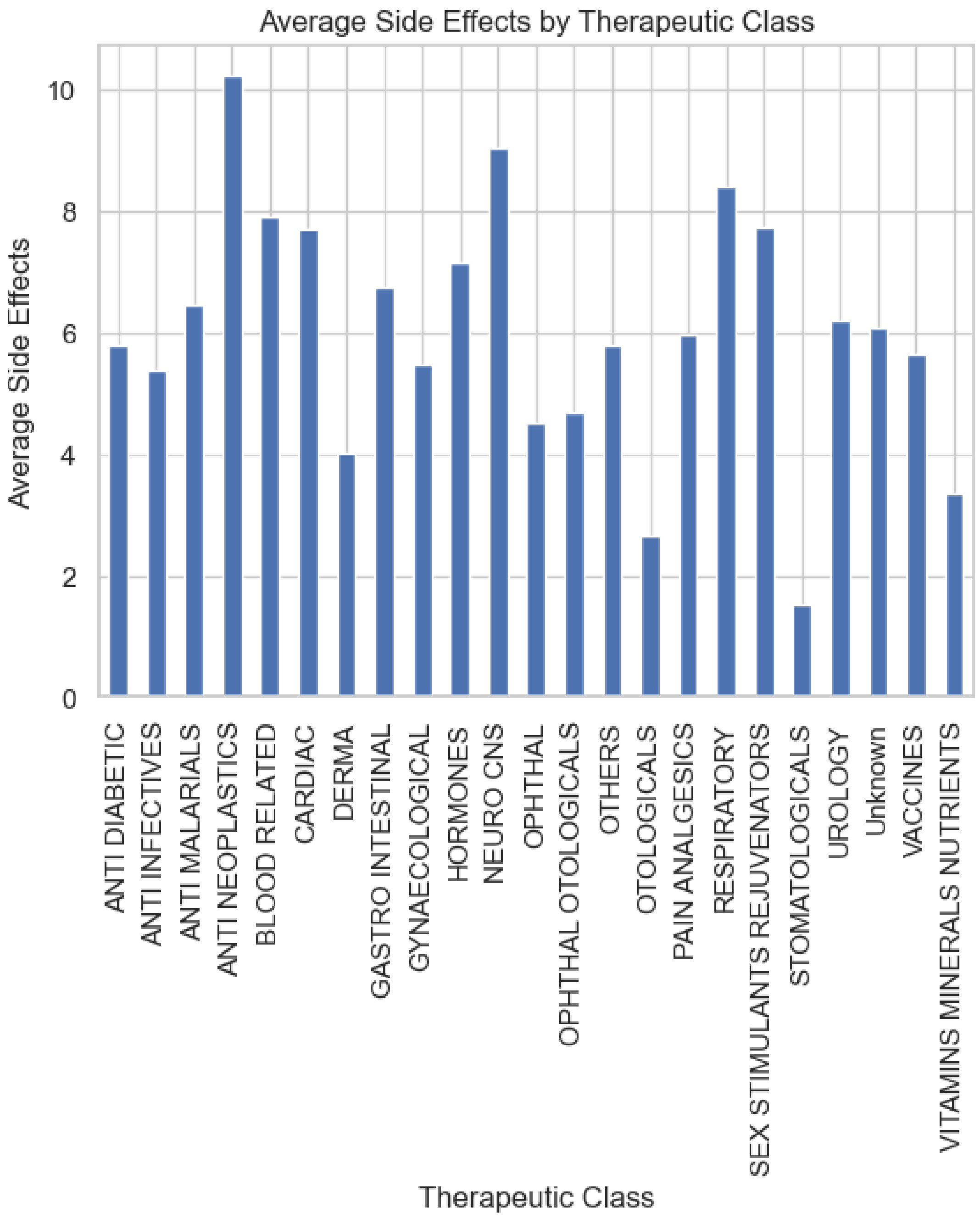
```
50%              6.000000
75%              8.000000
max             42.000000
Name: total_side_effects, dtype: float64
```

[94]:
```
import matplotlib.pyplot as plt

# Plot the distribution of the target variable
plt.hist(df['total_side_effects'], bins=50)
plt.xlabel('Total Side Effects')
plt.ylabel('Frequency')
plt.title('Distribution of Total Side Effects')
plt.show()
```

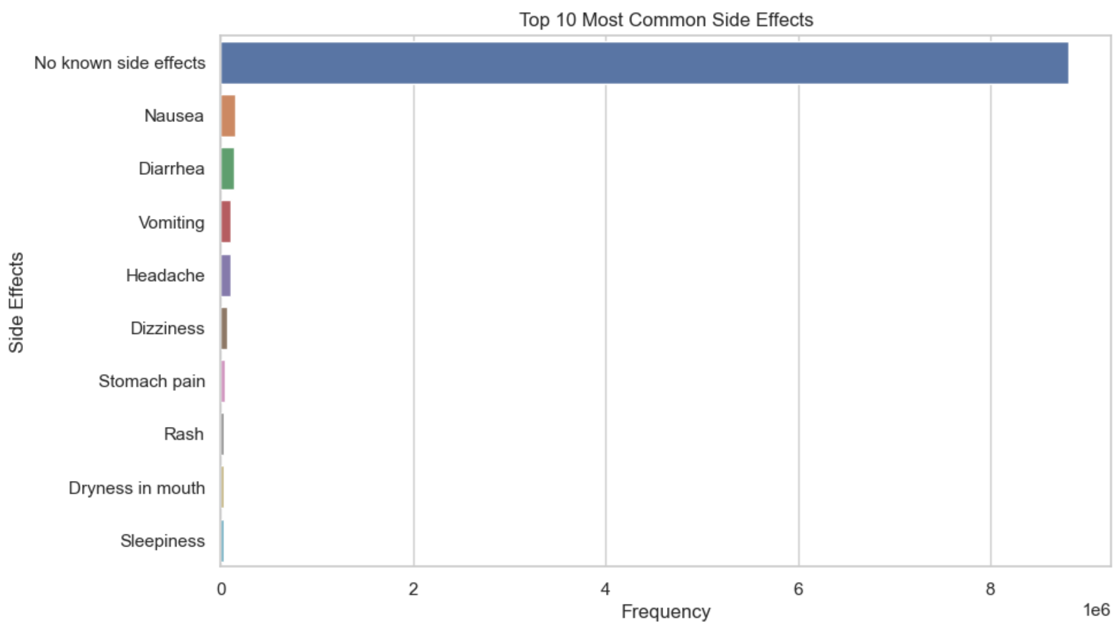


[96]:
```
# Grouping data by therapeutic class and calculating mean side effects
side_effect_by_therapeutic = df.groupby('Therapeutic␣
 ↪Class')['total_side_effects'].mean()

# Visualize using a bar plot
side_effect_by_therapeutic.plot(kind='bar')
```

```
plt.title('Average Side Effects by Therapeutic Class')
plt.xlabel('Therapeutic Class')
plt.ylabel('Average Side Effects')
plt.show()
```

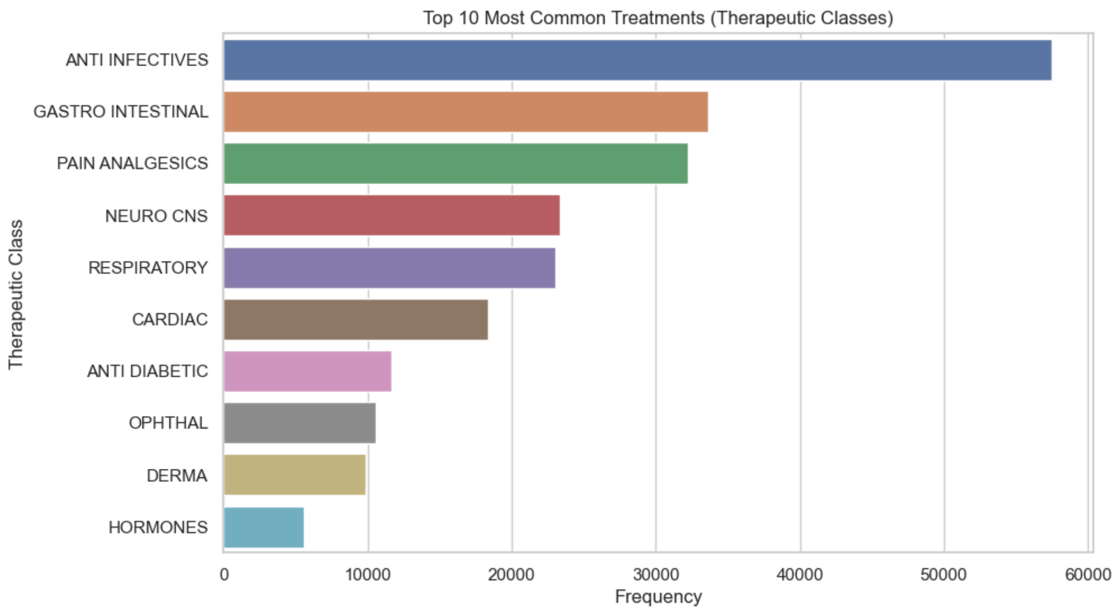


[98]:
```
# Popular Side Effects
# Melting side effects columns into a long format

import matplotlib.pyplot as plt
import seaborn as sns
side_effect_columns = [f"sideEffect{i}" for i in range(42)] # Adjust based on␣
 ↪side effect count
melted_side_effects = df.melt(value_vars=side_effect_columns,␣
 ↪value_name='side_effect', var_name='side_effect_col')
side_effect_counts = melted_side_effects['side_effect'].value_counts()

# Plot the most common side effects
plt.figure(figsize=(10,6))
sns.barplot(x=side_effect_counts[:10].values, y=side_effect_counts[:10].index)
plt.title('Top 10 Most Common Side Effects')
plt.xlabel('Frequency')
plt.ylabel('Side Effects')
plt.show()
```



[102]:
```
# Popular Treatment
# Count the occurrences of each therapeutic class
therapeutic_class_counts = df['Therapeutic Class'].value_counts()

# Plot popular therapeutic classes
plt.figure(figsize=(10,6))
```

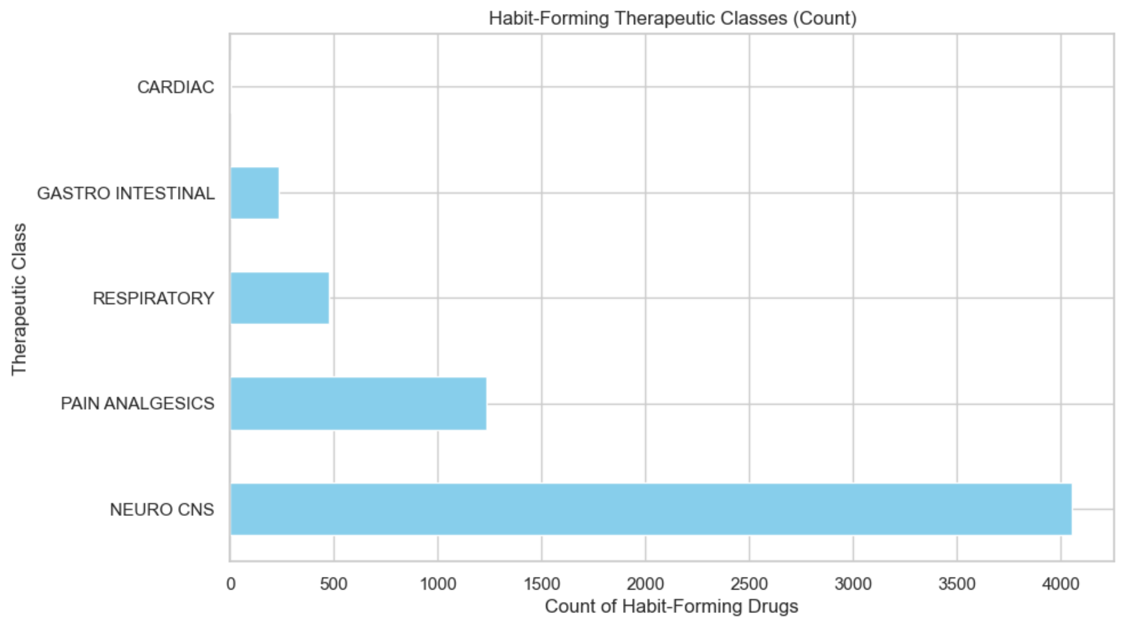
```
sns.barplot(x=therapeutic_class_counts[:10].values, y=therapeutic_class_counts[:
↪10].index)
plt.title('Top 10 Most Common Treatments (Therapeutic Classes)')
plt.xlabel('Frequency')
plt.ylabel('Therapeutic Class')
plt.show()
```



[104]:
```
# Cross-checking the count of habit-forming drugs per therapeutic class
habit_forming_classes = df[df['Habit Forming'] == 'YES']['Therapeutic Class'].
↪value_counts()
print(habit_forming_classes)

import matplotlib.pyplot as plt

# Plot for Habit-Forming drugs
plt.figure(figsize=(10, 6))
habit_forming_classes.plot(kind='barh', color='skyblue')
plt.title('Habit-Forming Therapeutic Classes (Count)')
plt.xlabel('Count of Habit-Forming Drugs')
plt.ylabel('Therapeutic Class')
plt.show()
```

```
Therapeutic Class
NEURO CNS            4054
PAIN ANALGESICS      1233
RESPIRATORY           474
GASTRO INTESTINAL     236
```

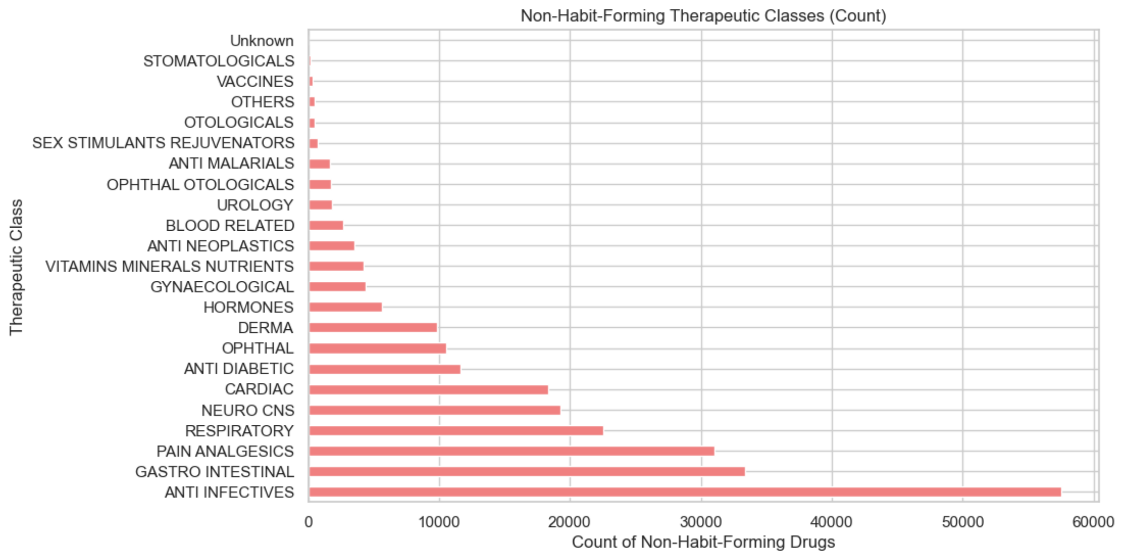
```
CARDIAC                    6
Name: count, dtype: int64
```



[106]:
```
habit_forming_classes = df[df['Habit Forming'] == 'NO']['Therapeutic Class'].
  ↪value_counts()
print(habit_forming_classes)
# Plot for Non-Habit-Forming drugs
plt.figure(figsize=(10, 6))
non_habit_forming_classes = df[df['Habit Forming'] == 'NO']['Therapeutic␣
  ↪Class'].value_counts()
non_habit_forming_classes.plot(kind='barh', color='lightcoral')
plt.title('Non-Habit-Forming Therapeutic Classes (Count)')
plt.xlabel('Count of Non-Habit-Forming Drugs')
plt.ylabel('Therapeutic Class')
plt.show()
```

```
Therapeutic Class
ANTI INFECTIVES              57503
GASTRO INTESTINAL            33394
PAIN ANALGESICS              31034
RESPIRATORY                  22578
NEURO CNS                    19265
CARDIAC                      18375
ANTI DIABETIC                11679
OPHTHAL                      10573
DERMA                         9883
HORMONES                      5629
```

```
GYNAECOLOGICAL                 4406
VITAMINS MINERALS NUTRIENTS    4216
ANTI NEOPLASTICS               3513
BLOOD RELATED                  2659
UROLOGY                        1844
OPHTHAL OTOLOGICALS            1725
ANTI MALARIALS                 1679
SEX STIMULANTS REJUVENATORS     723
OTOLOGICALS                     485
OTHERS                          481
VACCINES                        329
STOMATOLOGICALS                 173
Unknown                          69
Name: count, dtype: int64
```



[110]:
```python
# Count of drugs with substitutes
drugs_with_substitutes = df[df['substitute_count'] > 0].shape[0]
drugs_without_substitutes = df[df['substitute_count'] < 0].shape[0]


# Count of habit-forming drugs
habit_forming_drugs = df[df['Habit Forming'] == 'YES'].shape[0]
non_habit_forming_drugs = df[df['Habit Forming'] == 'NO'].shape[0]

print(f"Drugs with substitutes: {drugs_with_substitutes}")
print(f"Drugs without substitutes: {drugs_without_substitutes}")
print(f"Habit-forming drugs: {habit_forming_drugs}")
print(f"Non Habit-forming drugs: {non_habit_forming_drugs}")
```

```
Drugs with substitutes: 238621
Drugs without substitutes: 0
Habit-forming drugs: 6003
Non Habit-forming drugs: 242215
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```