

Drugs Side Effect And Medical Condition

Step 1: Imports

```
import pandas as pd
```

```
import numpy as np
```

```
import seaborn as sns
```

```
import matplotlib.pyplot as plt
```

```
import re
```

```
from sklearn.preprocessing import LabelEncoder, StandardScaler
```

```
import warnings
```

```
warnings.filterwarnings('ignore')
```

try:

Step 2: Load the dataset

```
print("Loading dataset...")
```

```
df = pd.read_csv("drugs_side_effects_drugs_com.csv")
```

```
print(f"Dataset loaded successfully. Shape: {df.shape}")
```

```
print("\nColumns in the dataset:")
```

```
print(df.columns.tolist())
```

Step 3: Basic Cleaning

```
print("\nCleaning data...")
```

Check if columns exist before cleaning

```
required_columns = ['alcohol', 'side_effects', 'related_drugs', 'generic_name',
```

```
                    'drug_classes', 'rx_otc', 'pregnancy_category', 'rating',
```

```
                    'no_of_reviews', 'medical_condition', 'csa']
```

```
missing_columns = [col for col in required_columns if col not in df.columns]
```

```

if missing_columns:
    raise ValueError(f"Missing required columns: {missing_columns}")

# Clean the data
df['alcohol'] = df['alcohol'].replace({np.nan: 0, 'X': 1})
df['side_effects'] = df['side_effects'].fillna('Unknown')
df['related_drugs'] = df['related_drugs'].fillna('Unknown')
df['generic_name'] = df['generic_name'].fillna('Unknown')
df['drug_classes'] = df['drug_classes'].fillna('Unknown')
df['rx_otc'] = df['rx_otc'].fillna('Unknown')
df['pregnancy_category'] = df['pregnancy_category'].fillna('Unknown')
df['rating'] = pd.to_numeric(df['rating'], errors='coerce').fillna(0)
df['no_of_reviews'] = pd.to_numeric(df['no_of_reviews'], errors='coerce').fillna(0)

# Step 4: Label Encoding
print("\nPerforming label encoding...")
cols_to_encode = ['generic_name', 'medical_condition', 'side_effects',
                  'drug_classes', 'rx_otc', 'pregnancy_category', 'csa']

encoder = LabelEncoder()
for col in cols_to_encode:
    if col in df.columns:
        df[col] = encoder.fit_transform(df[col].astype(str))

# Step 5: Normalize numeric columns
print("\nNormalizing numeric columns...")
scale_cols = ['generic_name', 'medical_condition', 'no_of_reviews', 'side_effects',
              'rating', 'csa', 'pregnancy_category', 'rx_otc', 'alcohol']

```

```

scaler = StandardScaler()

df_scaled = pd.DataFrame(scaler.fit_transform(df[scale_cols]), columns=scale_cols)

# Step 6: EDA - Side Effects Frequency
print("\nAnalyzing side effects...")

def extract_side_effects(text):
    if pd.isna(text) or text == 'Unknown':
        return []
    return [s.strip().lower() for s in re.split(r'[;,]', str(text))]

side_effects_series = df['side_effects'].astype(str).apply(extract_side_effects).explode()
side_effect_counts = side_effects_series.value_counts().head(10)

plt.figure(figsize=(12,6))
side_effect_counts.plot(kind='bar', color='salmon')
plt.title("Top 10 Side Effects")
plt.xlabel("Side Effect")
plt.ylabel("Frequency")
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()

# Step 7: Medical Condition Frequency
print("\nAnalyzing medical conditions...")

plt.figure(figsize=(12,6))
df['medical_condition'].value_counts().head(10).plot(kind='bar', color='skyblue')
plt.title("Top 10 Medical Conditions")

```

```
plt.xlabel("Condition")
plt.ylabel("Frequency")
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

Step 8: Rating Distribution

```
print("\nAnalyzing rating distribution...")
plt.figure(figsize=(10,6))
sns.histplot(df['rating'], bins=20, kde=True)
plt.title("Distribution of Drug Ratings")
plt.xlabel("Rating")
plt.ylabel("Count")
plt.tight_layout()
plt.show()
```

Step 9: Save cleaned dataset

```
print("\nSaving cleaned dataset...")
df.to_csv("cleaned_drug_dataset.csv", index=False)
print("Analysis complete! Cleaned dataset saved as 'cleaned_drug_dataset.csv'")
```

except FileNotFoundError:

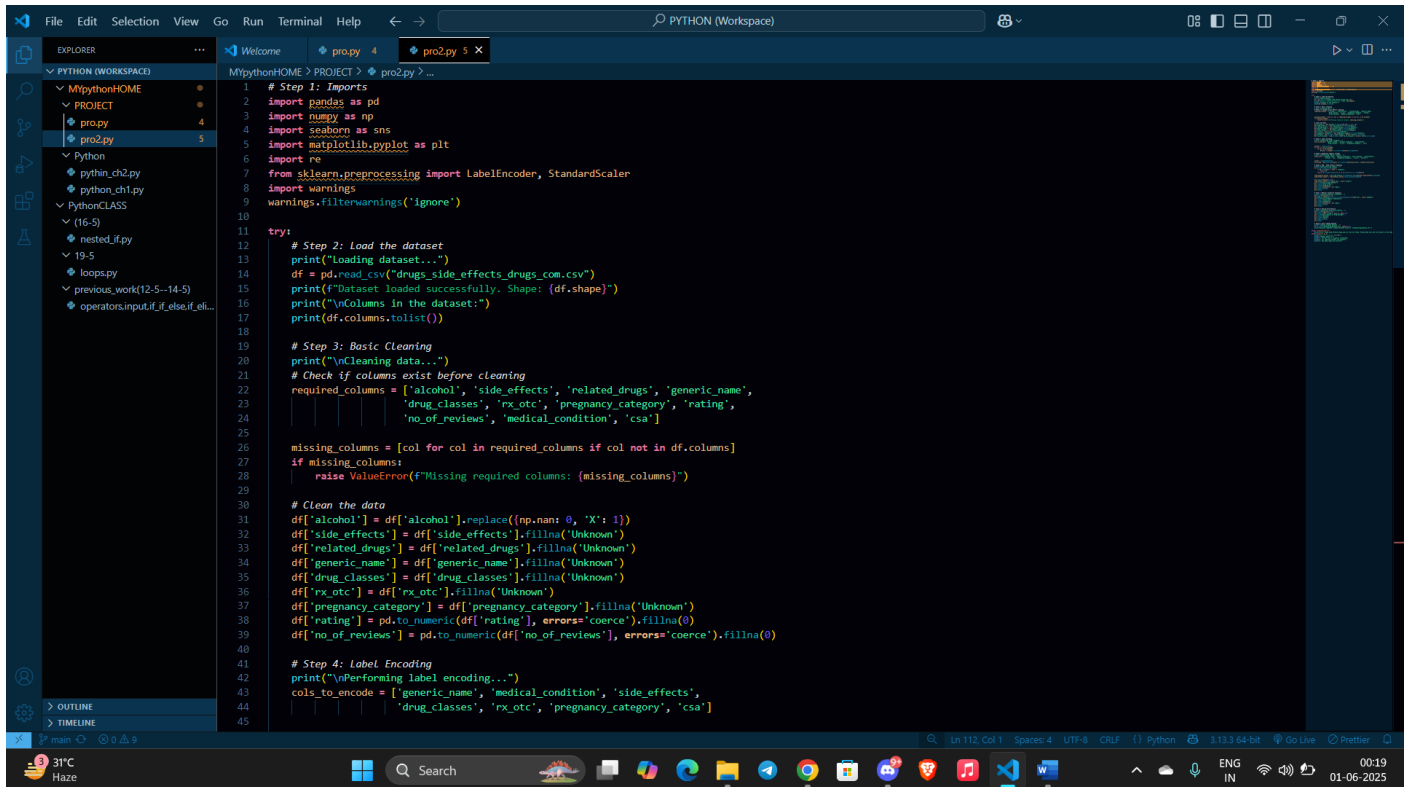
```
    print("Error: drugs_side_effects_drugs_com.csv file not found. Please make sure the file
exists in the same directory as this script.")
```

except Exception as e:

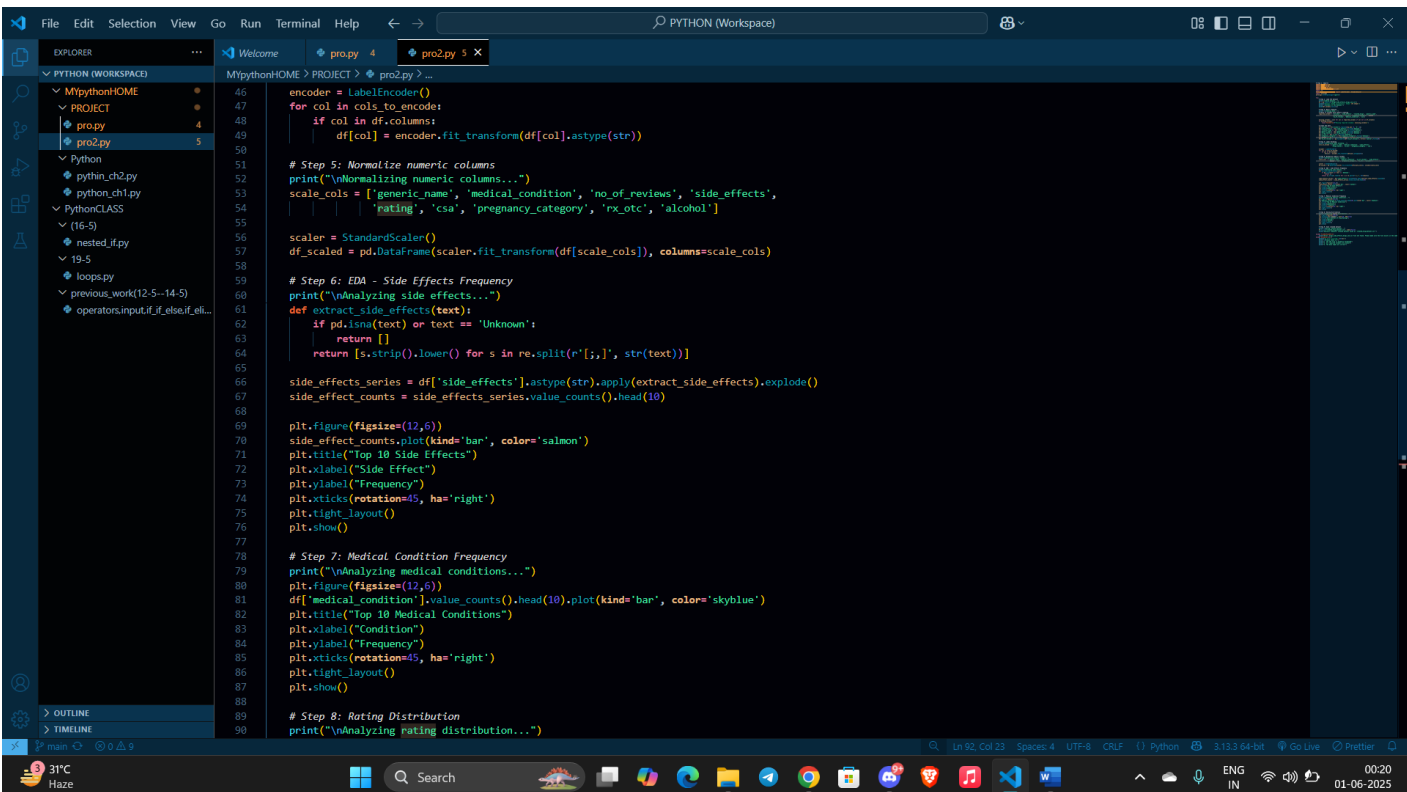
```
    print(f"An error occurred: {str(e)}")
    print("\nPlease check if:")
    print("1. The CSV file is properly formatted")
```

```
print("2. All required columns are present")
```

```
print("3. The data types are correct")
```



```
1 # Step 1: Imports
2 import pandas as pd
3 import numpy as np
4 import seaborn as sns
5 import matplotlib.pyplot as plt
6 import re
7 from sklearn.preprocessing import LabelEncoder, StandardScaler
8 import warnings
9 warnings.filterwarnings('ignore')
10
11 try:
12     # Step 2: Load the dataset
13     print("Loading dataset...")
14     df = pd.read_csv("drugs_side_effects_drugs.com.csv")
15     print(f"Dataset loaded successfully. Shape: {df.shape}")
16     print(f"Columns in the dataset:")
17     print(df.columns.tolist())
18
19     # Step 3: Basic Cleaning
20     print("Cleaning data...")
21     # Check if columns exist before cleaning
22     required_columns = ['alcohol', 'side_effects', 'related_drugs', 'generic_name',
23                        'drug_classes', 'rx_otc', 'pregnancy_category', 'rating',
24                        'no_of_reviews', 'medical_condition', 'csa']
25
26     missing_columns = [col for col in required_columns if col not in df.columns]
27     if missing_columns:
28         raise ValueError(f"Missing required columns: {missing_columns}")
29
30     # Clean the data
31     df['alcohol'] = df['alcohol'].replace((np.nan, 0, 'X': 1))
32     df['side_effects'] = df['side_effects'].fillna('Unknown')
33     df['related_drugs'] = df['related_drugs'].fillna('Unknown')
34     df['generic_name'] = df['generic_name'].fillna('Unknown')
35     df['drug_classes'] = df['drug_classes'].fillna('Unknown')
36     df['rx_otc'] = df['rx_otc'].fillna('Unknown')
37     df['pregnancy_category'] = df['pregnancy_category'].fillna('Unknown')
38     df['rating'] = pd.to_numeric(df['rating'], errors='coerce').fillna(0)
39     df['no_of_reviews'] = pd.to_numeric(df['no_of_reviews'], errors='coerce').fillna(0)
40
41     # Step 4: Label Encoding
42     print("Performing label encoding...")
43     cols_to_encode = ['generic_name', 'medical_condition', 'side_effects',
44                      'drug_classes', 'rx_otc', 'pregnancy_category', 'csa']
45
```



```
46 encoder = LabelEncoder()
47 for col in cols_to_encode:
48     if col in df.columns:
49         df[col] = encoder.fit_transform(df[col].astype(str))
50
51 # Step 5: Normalize numeric columns
52 print("Normalizing numeric columns...")
53 scale_cols = ['generic_name', 'medical_condition', 'no_of_reviews', 'side_effects',
54              'rating', 'csa', 'pregnancy_category', 'rx_otc', 'alcohol']
55
56 scaler = StandardScaler()
57 df_scaled = pd.DataFrame(scaler.fit_transform(df[scale_cols]), columns=scale_cols)
58
59 # Step 6: EDA - Side Effects Frequency
60 print("Analyzing side effects...")
61 def extract_side_effects(text):
62     if pd.isna(text) or text == 'Unknown':
63         return []
64     return [s.strip().lower() for s in re.split(r'[;,]', str(text))]
65
66 side_effects_series = df['side_effects'].astype(str).apply(extract_side_effects).explode()
67 side_effect_counts = side_effects_series.value_counts().head(10)
68
69 plt.figure(figsize=(12,6))
70 side_effect_counts.plot(kind='bar', color='salmon')
71 plt.title("Top 10 Side Effects")
72 plt.xlabel("Side Effect")
73 plt.ylabel("Frequency")
74 plt.xticks(rotation=45, ha='right')
75 plt.tight_layout()
76 plt.show()
77
78 # Step 7: Medical Condition Frequency
79 print("Analyzing medical conditions...")
80 plt.figure(figsize=(12,6))
81 df['medical_condition'].value_counts().head(10).plot(kind='bar', color='skyblue')
82 plt.title("Top 10 Medical Conditions")
83 plt.xlabel("Condition")
84 plt.ylabel("Frequency")
85 plt.xticks(rotation=45, ha='right')
86 plt.tight_layout()
87 plt.show()
88
89 # Step 8: Rating Distribution
90 print("Analyzing rating distribution...")
```

```
File Edit Selection View Go Run Terminal Help PYTHON (Workspace)
EXPLORER
PYTHON (WORKSPACE)
  MyPythonHOME
  PROJECT
    pro.py
    pro2.py
  Python
    pythin_ch2.py
    python_ch1.py
  PythonCLASS
    16-5
    nested_if.py
    19-5
    loops.py
    previous_work(12-5--14-5)
    operators.input_if_else_if_el...
  OUTLINE
  TIMELINE
  main
  31°C
  Haze

# Step 8: Rating Distribution
print("\nAnalyzing rating distribution...")
plt.figure(figsize=(10,6))
sns.histplot(df['rating'], bins=20, kde=True)
plt.title("Distribution of Drug Ratings")
plt.xlabel("Rating")
plt.ylabel("Count")
plt.tight_layout()
plt.show()

# Step 9: Save cleaned dataset
print("\nSaving cleaned dataset...")
df.to_csv("cleaned_drug_dataset.csv", index=False)
print("Analysis complete! Cleaned dataset saved as 'cleaned_drug_dataset.csv'")

except FileNotFoundError:
    print("Error: drugs_side_effects_drugs.com.csv file not found. Please make sure the file exists in the same directory as this script.")
except Exception as e:
    print(f"An error occurred: {str(e)}")
    print("\nPlease check if:")
    print("1. The CSV file is properly formatted")
    print("2. All required columns are present")
    print("3. The data types are correct")
```

