FR. CONCEICAO RODRIGUES COLLEGE OF ENGINEERIG
Department of Computer Engineering

Experiment 1 - Collecting, Cleaning and Transforming Healthcare Data for a Specific Disease

1. Course Details:

Academic Year	**2023 - 24**	**Estimated Time**	**Experiment No. 1 – 02 Hours**
Course & Semester	**B.E. – Sem. VII**	**Subject Name**	**Data Science for Health and Social Care Lab**
Experiment Type	**Software Performance**	**Subject Code**	**HDSSBL701**

Name of Student	Atharva Prashant Pawar	**Roll No.**	9427
Date of Performance.:		**Date of Submission.:**	
CO Mapping	HDSSBL701.1 Identify sources of data and methods for collecting, sharing and analyzing Healthcare data.		

Aim: Collecting, Cleaning, Integrating, and Transforming Healthcare Data for a Specific Disease:

Objective: The objective of this lab experiment is to familiarize students with the process of collecting, cleaning, integrating, and transforming healthcare data related to a specific disease. Students will gain hands-on experience in working with real-world healthcare datasets and preparing the data for analysis and AI applications.

Materials:

- Data analysis software (e.g., Python, R, or any preferred tool)
- Healthcare dataset(s) related to the chosen disease (e.g., public datasets, research datasets, or simulated data)

Procedure:

1. Choose a Specific Disease: Select a specific disease as the focus of the lab experiment. Consider diseases that have publicly available datasets or research data that can be accessed for analysis.

Examples of diseases could include diabetes, cardiovascular disease, cancer, respiratory disorders, etc.

2. Data Collection: Identify and collect relevant healthcare data related to the chosen disease. Explore public data repositories, research databases, or other reliable sources to gather datasets that contain patient information, medical records, lab results, diagnostic codes, treatment data, and any other relevant variables. Ensure compliance with ethical guidelines and data protection regulations.

3. Data Cleaning: Clean the collected data to ensure its quality and reliability. This process may involve handling missing values, removing duplicates, standardizing formats, correcting errors, and addressing other data quality issues. Document the steps taken during the cleaning process.

4. Data Integration: Integrate multiple datasets if available or necessary. This step involves combining data from different sources that share common variables or patient

identifiers. Apply appropriate techniques to merge the datasets while maintaining data integrity and ensuring consistent representations.

5. Data Transformation: Transform the integrated data into a suitable format for analysis and AI applications. This may involve feature engineering, scaling, normalization, encoding categorical variables, and creating derived variables. Document the transformations applied and their rationale.

6. Exploratory Data Analysis (EDA): Perform exploratory data analysis to gain insights into the dataset and the relationships between variables. Use visualizations, statistical summaries, and other techniques to understand the distribution of data, identify patterns, and uncover any interesting findings.

7. Summary and Documentation: Summarize the entire data preparation process, including data collection, cleaning, integration, and transformation. Document the steps taken, the challenges encountered, and the decisions made during each stage. Include any observations or insights gained from the exploratory data analysis.

Optional: Predictive Modeling: As an extension to the lab experiment, students can apply predictive modeling techniques using the prepared dataset. This can involve training machine learning models to predict disease outcomes, identify risk factors, or estimate treatment effectiveness. Students can evaluate the performance of the models and interpret their results.

Note: It is important to adhere to ethical guidelines and ensure the privacy and confidentiality of patient data throughout the lab experiment. Use de-identified or simulated datasets whenever possible to avoid any privacy concerns.

Data Repositories and Platforms: There are various data repositories and platforms where researchers and organizations share datasets. Some popular ones include:

- Kaggle
- UCI Machine Learning Repository
- Data.gov
- NIH National Library of Medicine

Result:

Dataset : heart_disease_data

Team: Atharva Pawar (9427), Aditya, Harsh

Data Science : Exp-1

In [27]:
```
# !pip install seaborn
```

In [28]:
```
# dataset link:
#     https://www.kaggle.com/datasets/johnsmith88/heart-disease-dataset
```

Importing the Dependencies

In [29]:
```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt
import seaborn as sns
```

Data Collection and Processing

In [30]:
```
# loading the csv data to a Pandas DataFrame
heart_data = pd.read_csv('heart2.csv')
```

In [31]:
```
heart_data.shape
```

Out[31]:
```
(1025, 14)
```

In [3]:
```
print(heart_data)
```

```
      age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  \
0      52    1   0       125   212    0        1      168      0      1.0
1      53    1   0       140   203    1        0      155      1      3.1
2      70    1   0       145   174    0        1      125      1      2.6
3      61    1   0       148   203    0        1      161      0      0.0
4      62    0   0       138   294    1        1      106      0      1.9
...   ...  ...  ..       ...   ...  ...      ...      ...    ...      ...
1020   59    1   1       140   221    0        1      164      1      0.0
1021   60    1   0       125   258    0        0      141      1      2.8
1022   47    1   0       110   275    0        0      118      1      1.0
1023   50    0   0       110   254    0        0      159      0      0.0
1024   54    1   0       120   188    0        1      113      0      1.4

      slope  ca  thal  target
0         2   2     3       0
1         0   0     3       0
2         0   0     3       0
3         2   1     3       0
4         1   3     2       0
...     ...  ..   ...     ...
1020      2   0     2       1
1021      1   1     3       0
1022      1   1     2       0
1023      2   0     2       1
1024      1   1     3       0

[1025 rows x 14 columns]
```

In [4]:
```
# print first 10 rows of the dataset
heart_data.head(10)
```

Out[4]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0
5	58	0	0	100	248	0	0	122	0	1.0	1	0	2	1
6	58	1	0	114	318	0	2	140	0	4.4	0	3	1	0
7	55	1	0	160	289	0	0	145	1	0.8	1	1	3	0
8	46	1	0	120	249	0	0	144	0	0.8	2	0	3	0
9	54	1	0	122	286	0	0	116	1	3.2	1	2	2	0

SEX: 0 = female 1 = male

In [5]:
```
# print last 10 rows of the dataset
heart_data.tail(10)
```

Out[5]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
1015	58	1	0	128	216	0	0	131	1	2.2	1	3	3	0
1016	65	1	3	138	282	1	0	174	0	1.4	1	1	2	0
1017	53	1	0	123	282	0	1	95	1	2.0	1	2	3	0
1018	41	1	0	110	172	0	0	158	0	0.0	2	0	3	0
1019	47	1	0	112	204	0	1	143	0	0.1	2	0	2	1
1020	59	1	1	140	221	0	1	164	1	0.0	2	0	2	1
1021	60	1	0	125	258	0	0	141	1	2.8	1	1	3	0
1022	47	1	0	110	275	0	0	118	1	1.0	1	1	2	0
1023	50	0	0	110	254	0	0	159	0	0.0	2	0	2	1
1024	54	1	0	120	188	0	1	113	0	1.4	1	1	3	0

data info

age: Age of the patient in years (ranging from 29 to 77).

sex: Sex of the patient (0: female, 1: male).

cp: Chest pain type (0 to 3), indicating different levels of chest pain experienced by the patient.

trestbps: Resting blood pressure in mm Hg.

chol: Serum cholesterol level in mg/dL.

fbs: Fasting blood sugar level > 120 mg/dL (1: true, 0: false).

restecg: Resting electrocardiographic results (0 to 2), indicating different types of ECG results.

thalach: Maximum heart rate achieved during exercise.

exang: Exercise-induced angina (1: yes, 0: no).

oldpeak: ST depression induced by exercise relative to rest.

slope: Slope of the peak exercise ST segment (0 to 2).

ca: Number of major vessels (0 to 4) colored by fluoroscopy.

thal: Thalassemia (3 types - 1, 2, 3).

target: The target variable indicating the presence of heart disease (1: yes, 0: no).

In [6]:
```
# Explore the basic statistics of the dataset
print("Basic Statistics:")
print(heart_data.describe())
```

```
Basic Statistics:
                age          sex           cp     trestbps        chol  \
count  1025.000000  1025.000000  1025.000000  1025.000000  1025.00000
mean     54.434146     0.695610     0.942439   131.611707   246.00000
std       9.072290     0.460373     1.029641    17.516718    51.59251
min      29.000000     0.000000     0.000000    94.000000   126.00000
25%      48.000000     0.000000     0.000000   120.000000   211.00000
50%      56.000000     1.000000     1.000000   130.000000   240.00000
75%      61.000000     1.000000     2.000000   140.000000   275.00000
max      77.000000     1.000000     3.000000   200.000000   564.00000

               fbs      restecg      thalach        exang      oldpeak  \
count  1025.000000  1025.000000  1025.000000  1025.000000  1025.000000
mean      0.149268     0.529756   149.114146     0.336585     1.071512
std       0.356527     0.527878    23.005724     0.472772     1.175053
min       0.000000     0.000000    71.000000     0.000000     0.000000
25%       0.000000     0.000000   132.000000     0.000000     0.000000
50%       0.000000     1.000000   152.000000     0.000000     0.800000
75%       0.000000     1.000000   166.000000     1.000000     1.800000
max       1.000000     2.000000   202.000000     1.000000     6.200000

             slope           ca         thal       target
count  1025.000000  1025.000000  1025.000000  1025.000000
mean      1.385366     0.754146     2.323902     0.513171
std       0.617755     1.030798     0.620660     0.500070
min       0.000000     0.000000     0.000000     0.000000
25%       1.000000     0.000000     2.000000     0.000000
50%       1.000000     0.000000     2.000000     1.000000
75%       2.000000     1.000000     3.000000     1.000000
max       2.000000     4.000000     3.000000     1.000000
```

In [20]:
```
# statistical measures about the data
# heart_data.describe()
```

In [8]:
```
# Explore the distribution of target classes
print("\nTarget Class Distribution:")
print(heart_data['target'].value_counts())
```

```
Target Class Distribution:
1    526
0    499
Name: target, dtype: int64
```

1 --> Defective Heart

0 --> Healthy Heart

In [9]:
```
# number of rows and columns in the dataset
heart_data.shape
```

Out[9]: (1025, 14)

In [10]:
```
# getting some info about the data
heart_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1025 entries, 0 to 1024
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       1025 non-null   int64  
 1   sex       1025 non-null   int64  
 2   cp        1025 non-null   int64  
 3   trestbps  1025 non-null   int64  
 4   chol      1025 non-null   int64  
 5   fbs       1025 non-null   int64  
 6   restecg   1025 non-null   int64  
 7   thalach   1025 non-null   int64  
 8   exang     1025 non-null   int64  
 9   oldpeak   1025 non-null   float64
 10  slope     1025 non-null   int64  
 11  ca        1025 non-null   int64  
 12  thal      1025 non-null   int64  
 13  target    1025 non-null   int64  
dtypes: float64(1), int64(13)
memory usage: 112.2 KB
```

In [11]:
```
# checking for missing values
```

```
heart_data.isnull().sum()
```

Out[11]:
```
age         0
sex         0
cp          0
trestbps    0
chol        0
fbs         0
restecg     0
thalach     0
exang       0
oldpeak     0
slope       0
ca          0
thal        0
target      0
dtype: int64
```

In [12]:
```
# checking the distribution of Target Variable
heart_data['target'].value_counts()
```

Out[12]:
```
1    526
0    499
Name: target, dtype: int64
```

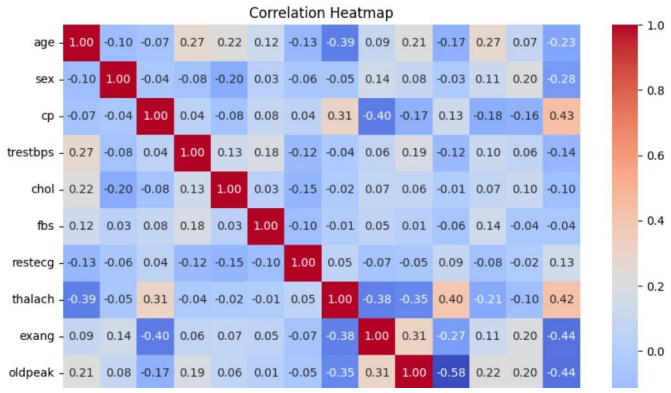
1 --> Defective Heart

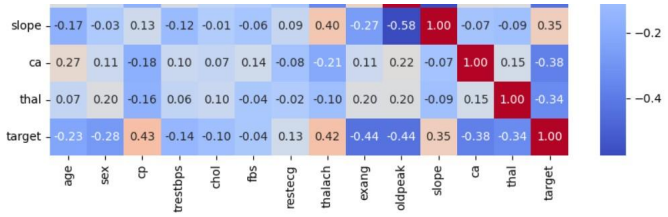
0 --> Healthy Heart

In [32]:
```
# Correlation matrix
print("\nCorrelation Matrix:")
correlation_matrix = heart_data.corr()
print(correlation_matrix)
```

```
Correlation Matrix:
                age       sex        cp  trestbps      chol       fbs  \
age        1.000000 -0.103240 -0.071966  0.271121  0.219823  0.121243
sex       -0.103240  1.000000 -0.041119 -0.078974 -0.198258  0.027200
cp        -0.071966 -0.041119  1.000000  0.038177 -0.081641  0.079294
trestbps   0.271121 -0.078974  0.038177  1.000000  0.127977  0.181767
chol       0.219823 -0.198258 -0.081641  0.127977  1.000000  0.026917
fbs        0.121243  0.027200  0.079294  0.181767  0.026917  1.000000
restecg   -0.132696 -0.055117  0.043581 -0.123794 -0.147410 -0.104051
thalach   -0.390227 -0.049365  0.306839 -0.039264 -0.021772 -0.008866
exang      0.088163  0.139157 -0.401513  0.061197  0.067382  0.049261
oldpeak    0.208137  0.084687 -0.174733  0.187434  0.064880  0.010859
slope     -0.169105 -0.026666  0.131633 -0.120445 -0.014248 -0.061902
ca         0.271551  0.111729 -0.176206  0.104554  0.074259  0.137156
thal       0.072297  0.198424 -0.163341  0.059276  0.100244 -0.042177
target    -0.229324 -0.279501  0.434854 -0.138772 -0.099966 -0.041164

            restecg   thalach     exang   oldpeak     slope        ca  \
age       -0.132696 -0.390227  0.088163  0.208137 -0.169105  0.271551
sex       -0.055117 -0.049365  0.139157  0.084687 -0.026666  0.111729
cp         0.043581  0.306839 -0.401513 -0.174733  0.131633 -0.176206
trestbps  -0.123794 -0.039264  0.061197  0.187434 -0.120445  0.104554
chol      -0.147410 -0.021772  0.067382  0.064880 -0.014248  0.074259
fbs       -0.104051 -0.008866  0.049261  0.010859 -0.061902  0.137156
restecg    1.000000  0.048411 -0.065606 -0.050114  0.086086 -0.078072
thalach    0.048411  1.000000 -0.380281 -0.349796  0.395308 -0.207888
exang     -0.065606 -0.380281  1.000000  0.310844 -0.267335  0.107849
oldpeak   -0.050114 -0.349796  0.310844  1.000000 -0.575189  0.221816
slope      0.086086  0.395308 -0.267335 -0.575189  1.000000 -0.073440
ca        -0.078072 -0.207888  0.107849  0.221816 -0.073440  1.000000
thal      -0.020504 -0.098068  0.197201  0.202672 -0.094090  0.149014
target     0.134468  0.422895 -0.438029 -0.438441  0.345512 -0.382085

              thal    target
age       0.072297 -0.229324
sex       0.198424 -0.279501
cp       -0.163341  0.434854
trestbps  0.059276 -0.138772
chol      0.100244 -0.099966
fbs      -0.042177 -0.041164
restecg  -0.020504  0.134468
thalach  -0.098068  0.422895
exang     0.197201 -0.438029
oldpeak   0.202672 -0.438441
slope    -0.094090  0.345512
ca        0.149014 -0.382085
thal      1.000000 -0.337838
target   -0.337838  1.000000
```

In [33]:
```
# Heatmap of correlation matrix
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Correlation Heatmap")
plt.show()
```




slope -0.17 -0.03 0.13 -0.12 -0.01 -0.06 0.09 0.40 -0.27 -0.58 1.00 -0.07 -0.09 0.35
ca 0.27 0.11 -0.18 0.10 0.07 0.14 -0.08 -0.21 0.11 0.22 -0.07 1.00 0.15 -0.38
thal 0.07 0.20 -0.16 0.06 0.10 -0.04 -0.02 -0.10 0.20 0.20 -0.09 0.15 1.00 -0.34
target -0.23 -0.28 0.43 -0.14 -0.10 -0.04 0.13 0.42 -0.44 -0.44 0.35 -0.38 -0.34 1.00
age sex cp trestbps chol fbs restecg thalach exang oldpeak slope ca thal target
−0.2
−0.4

FR. CONCEICAO RODRIGUES COLLEGE OF ENGINEERIG
Department of Computer Engineering

Experiment 2 - Perform Exploratory Data Analysis of Healthcare Data

1. **Course Details:**

Academic Year	**2023 - 24**	**Estimated Time**	**Experiment No. 1 – 02 Hours**
Course & Semester	**B.E. – Sem. VII**	**Subject Name**	**Data Science for Health and Social Care Lab**
Experiment Type	**Software Performance**	**Subject Code**	**HDSSBL701**

Name of Student	Atharva Prashant Pawar	**Roll No.**	9427
Date of Performance.:		**Date of Submission.:**	
CO Mapping	**HDSSBL701.2 Clean, integrate and transform healthcare data**		

Aim: Perform Exploratory Data Analysis of Healthcare Data

Objective: The objective of this experiment is to familiarize BE Computer students with the process of performing Exploratory Data Analysis (EDA) on healthcare data. Students will learn how to import, explore, visualize, and analyze a healthcare dataset to gain insights and understand the characteristics of the data

Tools and libraries:

- Python programming language
- Jupyter Notebook or any Python IDE
- Required Python libraries: pandas, matplotlib, seaborn

Procedure:

Step 1: Data Loading and Understanding

Start by importing the necessary Python libraries: pandas, matplotlib, and seaborn.
Load the healthcare dataset into a pandas DataFrame.
Display the basic information about the dataset, such as the number of rows and columns, data types, and summary statistics

Step 2: Data Cleaning and Preprocessing

Check for any missing values in the dataset and decide how to handle them (e.g., imputation or removal).

Look for any duplicate entries in the dataset and handle them if found.
Convert data types if necessary (e.g., dates, categorical variables).

Step 3: Exploratory Data Analysis (EDA)
Generate summary statistics for relevant numerical variables (e.g., mean, median, standard deviation, etc.).

Visualize the distribution of numerical variables using histograms, box plots, or kernel density plots.
Analyze the distribution of categorical variables using bar plots or count plots
Explore the correlation between different variables using a correlation heatmap.

Step 4: Data Visualization
Create meaningful visualizations to understand relationships and trends in the data. Use scatter plots, bar plots, line plots, or any other appropriate visualization techniques.
Focus on specific aspects like the relationship between age and health conditions, gender-wise distribution of diseases etc.

Step 5: Insights and Interpretation
Based on the analysis and visualizations, derive insights and observations from the healthcare data.
Identify patterns, trends, and any interesting findings that can help in understanding the dataset better.

Result:

Data Science in Health Care : Group - 5

Team : Atharva Pawar (9427), Aditya Vyas, Harshvardhan Trivedi

In [2]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [3]:
```
url = "https://raw.githubusercontent.com/Aditya-K-Vyas/data_science_utils/main/heart.csv"
df = pd.read_csv(url)
```

In [4]:
```
df.head()
```

Out[4]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0

In [5]:
```
df.describe()
```

Out[5]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	
count	1025.000000	1025.000000	1025.000000	1025.000000	1025.00000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.00
mean	54.434146	0.695610	0.942439	131.611707	246.00000	0.149268	0.529756	149.114146	0.336585	1.071512	1.385366	0.75
std	9.072290	0.460373	1.029641	17.516718	51.59251	0.356527	0.527878	23.005724	0.472772	1.175053	0.617755	1.03
min	29.000000	0.000000	0.000000	94.000000	126.00000	0.000000	0.000000	71.000000	0.000000	0.000000	0.000000	0.00
25%	48.000000	0.000000	0.000000	120.000000	211.00000	0.000000	0.000000	132.000000	0.000000	0.000000	1.000000	0.00
50%	56.000000	1.000000	1.000000	130.000000	240.00000	0.000000	1.000000	152.000000	0.000000	0.800000	1.000000	0.00
75%	61.000000	1.000000	2.000000	140.000000	275.00000	0.000000	1.000000	166.000000	1.000000	1.800000	2.000000	1.00
max	77.000000	1.000000	3.000000	200.000000	564.00000	1.000000	2.000000	202.000000	1.000000	6.200000	2.000000	4.00

In [6]:
```
duplicates = df.duplicated()
print(duplicates)
```

```
0       False
1       False
2       False
3       False
4       False
        ...
1020     True
1021     True
1022     True
1023     True
1024     True
Length: 1025, dtype: bool
```

In [7]:
```
# removing duplicate data
df = df.drop_duplicates()
```

In [8]:
```
df.shape
```

Out[8]:
```
(302, 14)
```

In [9]:
```
# checking for null values
null_counts = df.isnull().sum()
print(null_counts)
```

```
age         0
sex         0
cp          0
trestbps    0
chol        0
fbs         0
restecg     0
thalach     0
exang       0
oldpeak     0
slope       0
ca          0
thal        0
target      0
dtype: int64
```

In [10]:
```
# final df look
df.head()
```

Out[10]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0

2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0

EDA

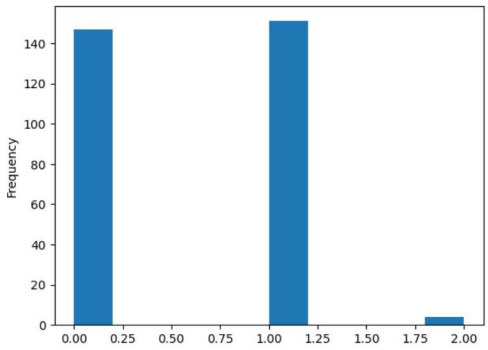
In [12]:
```
# basic df
df.describe()
```

Out[12]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	
count	302.00000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000
mean	54.42053	0.682119	0.963576	131.602649	246.500000	0.149007	0.526490	149.569536	0.327815	1.043046	1.397351	0.718543	2.314
std	9.04797	0.466426	1.032044	17.563394	51.753489	0.356686	0.526027	22.903527	0.470196	1.161452	0.616274	1.006748	0.613
min	29.00000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.000000	0.000000	0.000000	0.000000	0.000000	0.000
25%	48.00000	0.000000	0.000000	120.000000	211.000000	0.000000	0.000000	133.250000	0.000000	0.000000	1.000000	0.000000	2.000
50%	55.50000	1.000000	1.000000	130.000000	240.500000	0.000000	1.000000	152.500000	0.000000	0.800000	1.000000	0.000000	2.000
75%	61.00000	1.000000	2.000000	140.000000	274.750000	0.000000	1.000000	166.000000	1.000000	1.600000	2.000000	1.000000	3.000
max	77.00000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.000000	6.200000	2.000000	4.000000	3.000

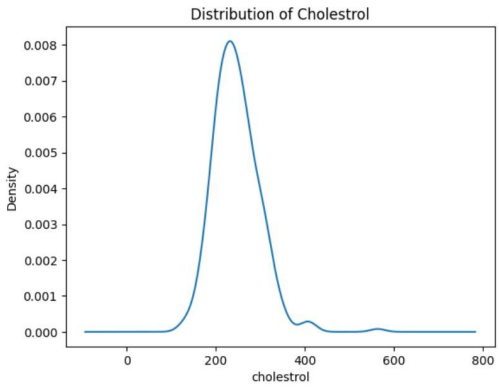
In [13]:
```
# histogram
df['restecg'].plot.hist()  # Histogram
```

Out[13]:
```
<Axes: ylabel='Frequency'>
```

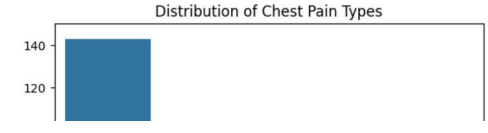


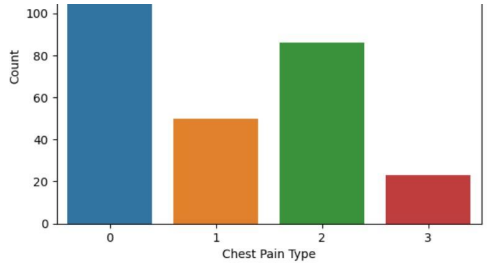
In [14]:
```
df['chol'].plot.kde()  # Kernel density plot
plt.xlabel('cholestrol')
plt.ylabel('Density')
plt.title('Distribution of Cholestrol')
```

Out[14]:
```
Text(0.5, 1.0, 'Distribution of Cholestrol')
```

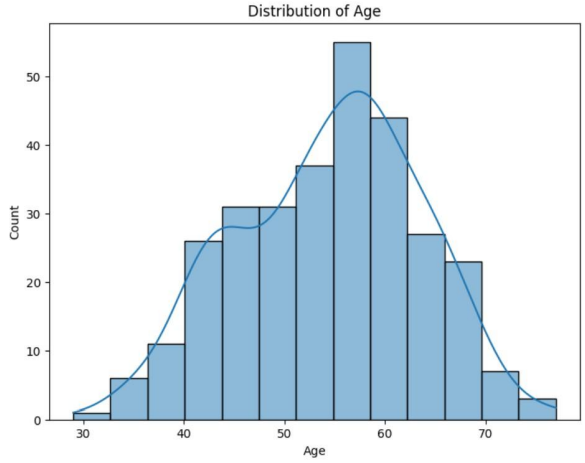


In [15]:
```
sns.countplot(data=df, x='cp')
plt.xlabel('Chest Pain Type')
plt.ylabel('Count')
plt.title('Distribution of Chest Pain Types')
plt.show()
```

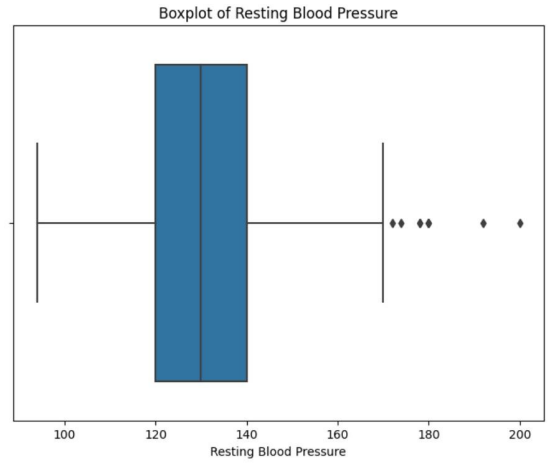




In [16]:
```
plt.figure(figsize=(8,6))
sns.histplot(data=df, x='age', kde=True)
plt.xlabel('Age')
plt.ylabel('Count')
plt.title('Distribution of Age')
plt.show()
```



In [17]:
```
plt.figure(figsize=(8,6))
sns.boxplot(data=df, x='trestbps')
plt.xlabel('Resting Blood Pressure')
plt.title('Boxplot of Resting Blood Pressure')
plt.show()
```



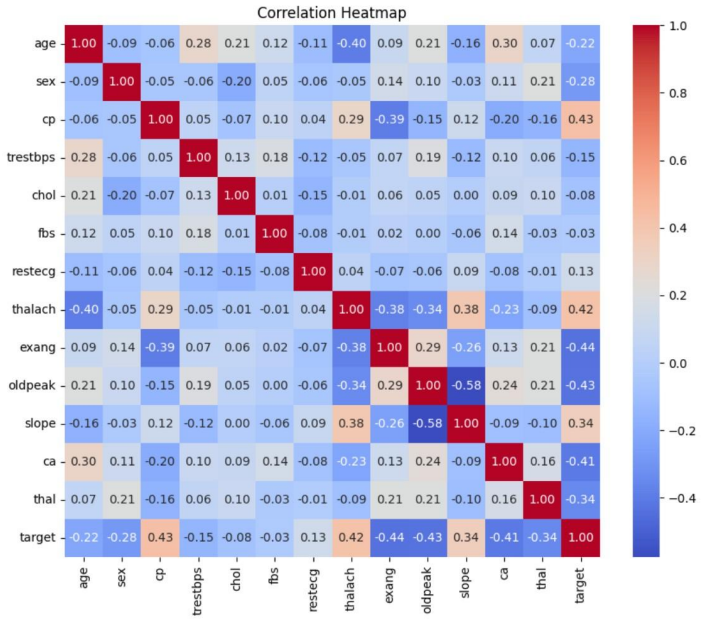
In [18]:
```
# Explore the distribution of target classes
print("\nTarget Class Distribution:")
print(df['target'].value_counts())
```

```
Target Class Distribution:
1    164
0    138
Name: target, dtype: int64
```

In [19]:
```
# Correlation matrix
print("\nCorrelation Matrix:")
correlation_matrix = df.corr()
print(correlation_matrix)
```

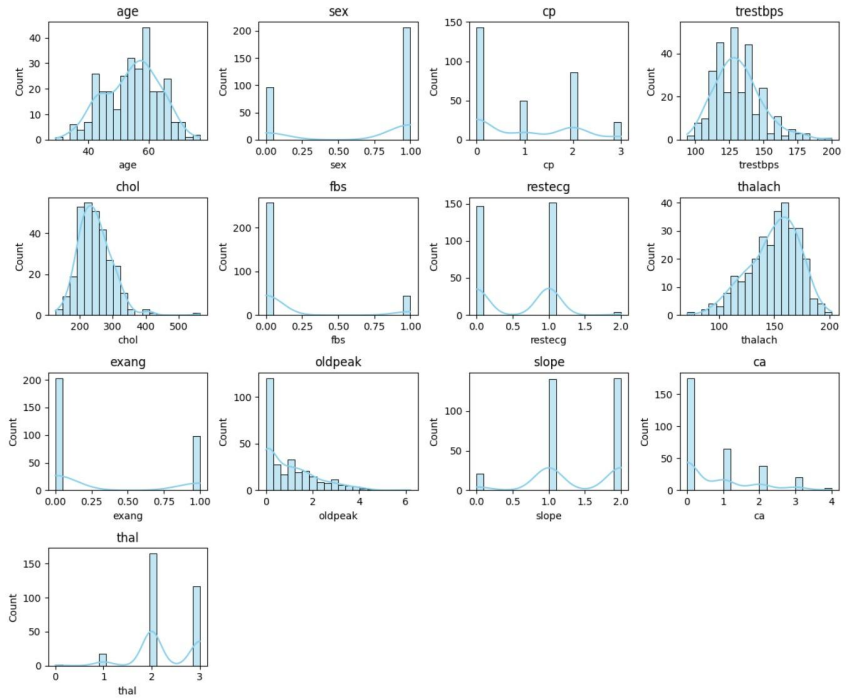
```
Correlation Matrix:
                age       sex        cp  trestbps      chol       fbs  \
age        1.000000 -0.094962 -0.063107  0.283121  0.207216  0.119492
sex       -0.094962  1.000000 -0.051740 -0.057647 -0.195571  0.046022
cp        -0.063107 -0.051740  1.000000  0.046486 -0.072682  0.096018
trestbps   0.283121 -0.057647  0.046486  1.000000  0.125256  0.178125
chol       0.207216 -0.195571 -0.072682  0.125256  1.000000  0.011428
fbs        0.119492  0.046022  0.096018  0.178125  0.011428  1.000000
restecg   -0.111590 -0.060351  0.041561 -0.115367 -0.147602 -0.083081
thalach   -0.395235 -0.046439  0.293367 -0.048023 -0.005308 -0.007169
exang      0.093216  0.143460 -0.392937  0.068526  0.064099  0.024729
oldpeak    0.206040  0.098322 -0.146692  0.194600  0.050086  0.004514
slope     -0.164124 -0.032990  0.116854 -0.122873  0.000417 -0.058654
ca         0.302261  0.113060 -0.195356  0.099248  0.086878  0.144935
thal       0.065317  0.211452 -0.160370  0.062870  0.096810 -0.032752
target    -0.221476 -0.283609  0.432080 -0.146269 -0.081437 -0.026826

             restecg   thalach     exang   oldpeak     slope        ca  \
age       -0.111590 -0.395235  0.093216  0.206040 -0.164124  0.302261
sex       -0.060351 -0.046439  0.143460  0.098322 -0.032990  0.113060
cp         0.041561  0.293367 -0.392937 -0.146692  0.116854 -0.195356
trestbps  -0.115367 -0.048023  0.068526  0.194600 -0.122873  0.099248
chol      -0.147602 -0.005308  0.064099  0.050086  0.000417  0.086878
fbs       -0.083081 -0.007169  0.024729  0.004514 -0.058654  0.144935
restecg    1.000000  0.041210 -0.068807 -0.056251  0.090402 -0.083112
thalach    0.041210  1.000000 -0.377411 -0.342201  0.384754 -0.228311
exang     -0.068807 -0.377411  1.000000  0.286766 -0.256106  0.125377
oldpeak   -0.056251 -0.342201  0.286766  1.000000 -0.576314  0.236560
slope      0.090402  0.384754 -0.256106 -0.576314  1.000000 -0.092236
ca        -0.083112 -0.228311  0.125377  0.236560 -0.092236  1.000000
thal      -0.010473 -0.094910  0.205826  0.209090 -0.103314  0.160085
target     0.134874  0.419955 -0.435601 -0.429146  0.343940 -0.408992

              thal    target
age       0.065317 -0.221476
sex       0.211452 -0.283609
cp       -0.160370  0.432080
trestbps  0.062870 -0.146269
chol      0.096810 -0.081437
fbs      -0.032752 -0.026826
restecg  -0.010473  0.134874
thalach  -0.094910  0.419955
exang     0.205826 -0.435601
oldpeak   0.209090 -0.429146
slope    -0.103314  0.343940
ca        0.160085 -0.408992
thal      1.000000 -0.343101
target   -0.343101  1.000000
```

In [20]:
```
# Heatmap of correlation matrix
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Correlation Heatmap")
plt.show()
```

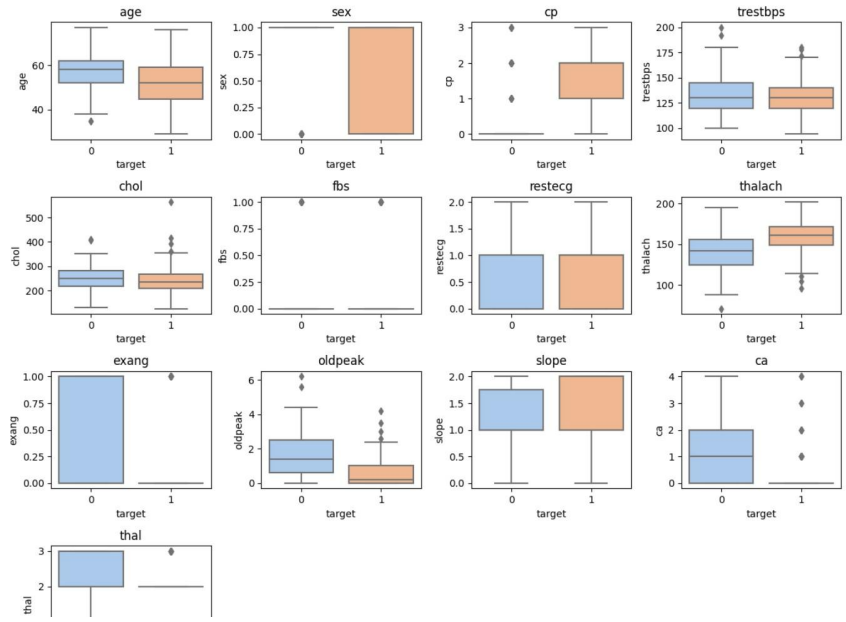


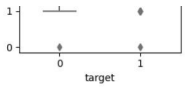
In [21]:
```
# Distribution plots for numerical features
```

```
plt.figure(figsize=(12, 10))
for i, column in enumerate(df.columns[:-1]):
    plt.subplot(4, 4, i + 1)
    sns.histplot(df[column], kde=True, bins=20, color='skyblue')
    plt.title(column)
plt.tight_layout()
plt.show()
```

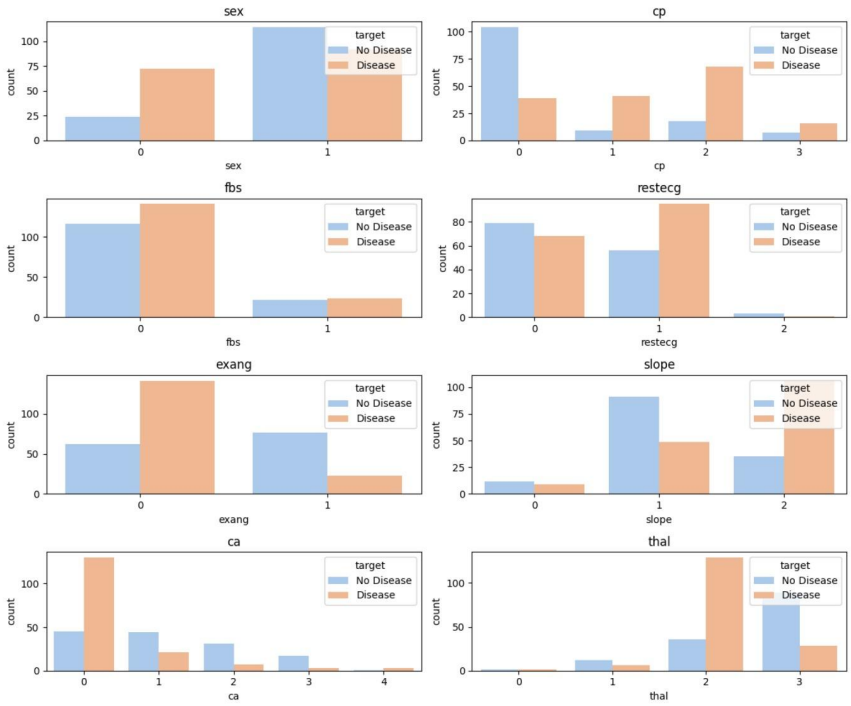


In [22]:
```
# Box plots for numerical features by target class
plt.figure(figsize=(12, 10))
for i, column in enumerate(df.columns[:-1]):
    plt.subplot(4, 4, i + 1)
    sns.boxplot(x='target', y=column, data=df, palette='pastel')
    plt.title(column)
plt.tight_layout()
plt.show()
```

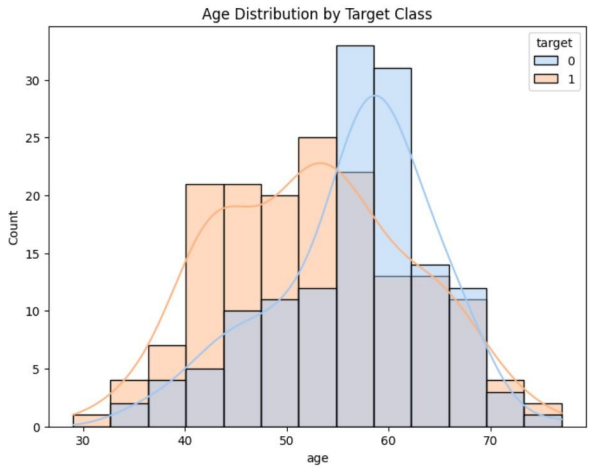




In [23]:
```
# Count plots for categorical features by target class
plt.figure(figsize=(12, 10))
for i, column in enumerate(['sex', 'cp', 'fbs', 'restecg', 'exang', 'slope', 'ca', 'thal']):
    plt.subplot(4, 2, i + 1)
    sns.countplot(x=column, hue='target', data=df, palette='pastel')
    plt.title(column)
    plt.legend(title='target', loc='upper right', labels=['No Disease', 'Disease'])
plt.tight_layout()
plt.show()
```

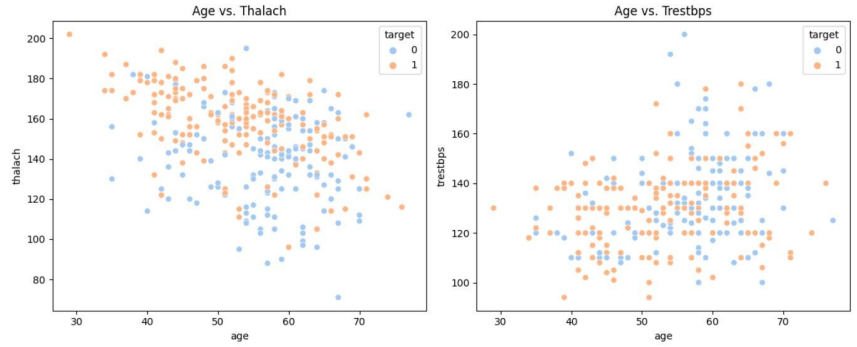


In [29]:
```
# Age distribution by target class
plt.figure(figsize=(8, 6))
sns.histplot(data=df, x='age', hue='target', kde=True, palette='pastel')
plt.title("Age Distribution by Target Class")
plt.show()
```



In [30]:
```
# Scatter plots for age vs. thalach and age vs. trestbps
plt.figure(figsize=(12, 5))
```

```
plt.subplot(1, 2, 1)
sns.scatterplot(x='age', y='thalach', hue='target', data=df, palette='pastel')
plt.title("Age vs. Thalach")

plt.subplot(1, 2, 2)
sns.scatterplot(x='age', y='trestbps', hue='target', data=df, palette='pastel')
plt.title("Age vs. Trestbps")
plt.tight_layout()
plt.show()
```



FR. CONCEICAO RODRIGUES COLLEGE OF ENGINEERIG
Department of Computer Engineering

Experiment 3 - Bio Medial Image Pre processing and Segmentation

1. Course Details:

Academic Year	**2023 - 24**	**Estimated Time**	**Experiment No. 1 – 02 Hours**
Course & Semester	**B.E. (COMP) – Sem. VII**	**Subject Name**	**Data Science for Health and Social Care Lab**
Experiment Type	**Software Performance**	**Subject Code**	**HDSSBL701**

Name of Student	Atharva Pawar	**Roll No.**	9427
Date of Performance.:		**Date of Submission.:**	
CO Mapping	**HDSSBL701.3 - Demonstrate statistical data analysis, Image preprocessing, Segmentation and visualization techniques on healthcare data.**		

Aim: To perform Bio Medial Image Pre processing and Segmentation.
Objective: To explore, visualize and analyze Medical Image dataset to gain insights and understand the characteristics of data.

Tools :
Jupyter Notebook/ MATLAB

Libraries:
OpenCV, Numpy, Pandas, Tensor Flow

Dataset:
MRI, X-ray or CT scan datasets containing labeled images of various medical conditions (e.g., fractures, tumors, pneumonia, etc.). Datasets can be obtained from publicly available sources

Theory:

Image preprocessing is a vital step when working with image data.Medical Image preprocessing is used to improve the quality of medical images, making it easier to detect diseases or abnormalities. Some powerful image preprocessing techniques include noise reduction, contrast enhancement, image resizing, color correction, segmentation, feature extraction, etc. It is an essential step in image analysis that helps enhance the data in images and reduce clutter.

Techniques for Image Preprocessing

The choice of techniques depends on the nature of the image and the application. Here are a few techniques to improve image quality and suitability:

Noise Reduction: Noise in an image can be caused by various factors such as low light, sensor noise, and compression artifacts. Noise reduction techniques aim to remove noise from the image while preserving its essential features. Some common noise reduction techniques include Gaussian smoothing, median filtering, and wavelet denoising.

Contrast Enhancement: Contrast enhancement techniques aim to increase the contrast of an image, making it easier to distinguish between different image features. These techniques can be helpful in applications such as medical imaging and surveillance. Some standard contrast enhancement techniques include histogram equalization, adaptive histogram equalization, and contrast stretching.

Image Resizing: Image resizing techniques are used to adjust the size of an image. Resizing can be done to make an image smaller or larger or to change its aspect ratio. Some typical image resizing techniques include nearest neighbor interpolation, bilinear interpolation, and bicubic interpolation.

Color Correction: Color correction techniques are used to adjust the color balance of an image. Color correction is important in applications such as photography, where the color accuracy of an image is critical. Some common color correction techniques include gray world assumption, white balance, and color transfer.

Segmentation: Segmentation techniques are used to divide an image into regions based on its content. Segmentation can be helpful in applications such as medical imaging, where specific structures or organs must be isolated from the image. Some standard segmentation techniques include thresholding, edge detection, and region growing.

Feature Extraction: Feature extraction techniques are used to identify and extract relevant features from an image. These features can be used in object recognition and image classification applications. Some standard feature extraction techniques include edge detection, corner detection, and texture analysis.

Procedure:

1. Download any Sample image
2. Convert sample image to gray scale image
3. Apply Thresholding to see the changes (can use any thresholding method such as Binary thresholding or OTSU thresholding)
4. Apply median and gaussian filters for Noise reduction
5. Plot all images and their Histogram
6. Detection of edge

Results:

Advanced Learner Activity:

Study and perform Semantic Segmentation & Instance Segmentation

DSHC - EXP - 3

Demonstrate statistical data analysis, Image preprocessing, Segmentation and visualization techniques on healthcare data.bold text

In [12]:
```
import cv2
import numpy as np
import matplotlib.pyplot as plt
```

In [13]:
```
# Step 1: Read the image
image_path = 'ID_0077_AGE_0074_CONTRAST_0_CT.png'
# image_path = 'ID_0078_AGE_0066_CONTRAST_0_CT.png'
# image_path = 'ID_0079_AGE_0071_CONTRAST_0_CT.png'
image = cv2.imread(image_path, cv2.IMREAD_COLOR)
```

In [15]:
```
# Step 2: Convert to grayscale
if image is None:
    print("Error: Image not loaded.")
else:
    gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    print("Success: Image loaded.")
```

```
Success: Image loaded.
```

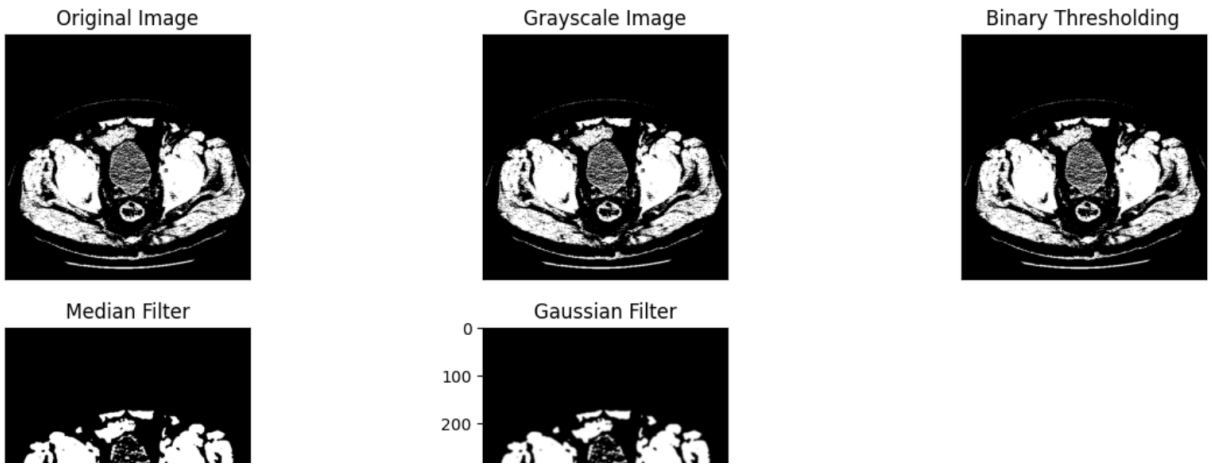
In [16]:
```
# Step 3: Apply Thresholding (Binary thresholding)
ret, binary_image = cv2.threshold(gray_image, 127, 255, cv2.THRESH_BINARY)
```

In [17]:
```
# Step 4: Apply Median Filter for Noise Reduction
median_filtered_image = cv2.medianBlur(binary_image, 5)
```

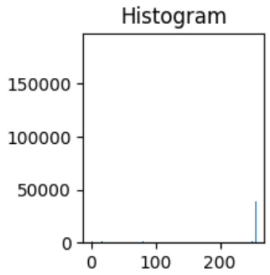
In [18]:
```
# Step 4: Apply Gaussian Filter for Noise Reduction
gaussian_filtered_image = cv2.GaussianBlur(median_filtered_image, (5, 5), 0)
```

In [19]:
```
# Step 5: Plot the Images and their Histograms
plt.figure(figsize=(15, 6))
plt.subplot(2, 3, 1), plt.imshow(cv2.cvtColor(image, cv2.COLOR_BGR2RGB), cmap='gray')
plt.title('Original Image'), plt.xticks([]), plt.yticks([])
plt.subplot(2, 3, 2), plt.imshow(gray_image, cmap='gray')
plt.title('Grayscale Image'), plt.xticks([]), plt.yticks([])
plt.subplot(2, 3, 3), plt.imshow(binary_image, cmap='gray')
plt.title('Binary Thresholding'), plt.xticks([]), plt.yticks([])
plt.subplot(2, 3, 4), plt.imshow(median_filtered_image, cmap='gray')
plt.title('Median Filter'), plt.xticks([]), plt.yticks([])
plt.subplot(2, 3, 5), plt.imshow(gaussian_filtered_image, cmap='gray')
plt.title('Gaussian Filter'), plt.xticks([]), plt.yticks()
```

Out[19]:
```
(Text(0.5, 1.0, 'Gaussian Filter'),
 ([], []),
 (array([-100.,    0.,  100.,  200.,  300.,  400.,  500.,  600.]),
  [Text(0, -100.0, '-100'),
   Text(0, 0.0, '0'),
   Text(0, 100.0, '100'),
   Text(0, 200.0, '200'),
   Text(0, 300.0, '300'),
   Text(0, 400.0, '400'),
   Text(0, 500.0, '500'),
   Text(0, 600.0, '600')]))
```

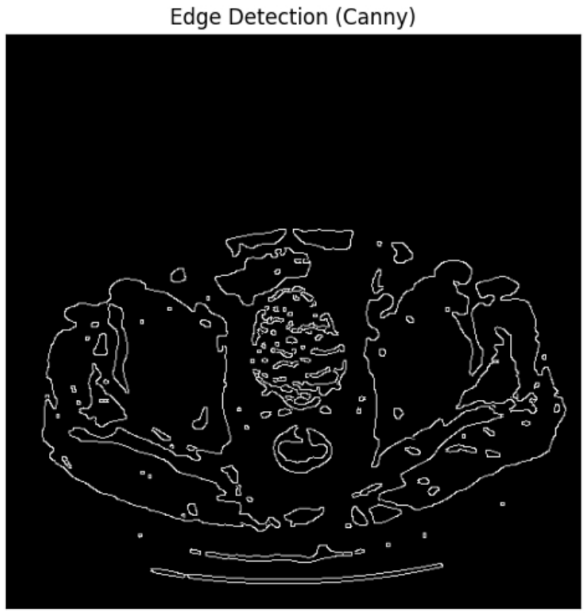


In [20]:
```
# Plot histograms
plt.subplot(2, 3, 6), plt.hist(gaussian_filtered_image.ravel(), 256, [0, 256])
plt.title('Histogram')
plt.show()
```

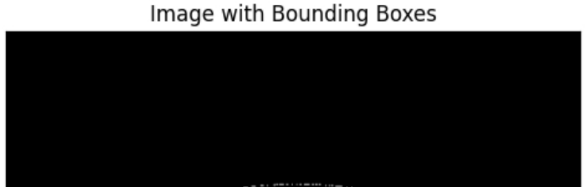


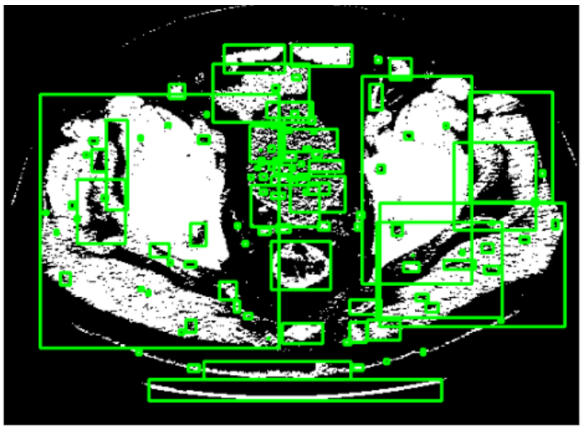
In [21]:
```
# Step 6: Detection of Edges (Canny Edge Detection)
edges = cv2.Canny(gaussian_filtered_image, 100, 200)
```

In [22]:
```
# Display the edges
plt.figure(figsize=(6, 6))
plt.imshow(edges, cmap='gray')
plt.title('Edge Detection (Canny)')
plt.xticks([]), plt.yticks([])
plt.show()
```



In [23]:
```
# Find contours in the edge image
contours, _ = cv2.findContours(edges, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)

# Create a copy of the original image to draw bounding boxes on
image_with_boxes = image.copy()

# Iterate through the contours and draw bounding boxes
for contour in contours:
    x, y, w, h = cv2.boundingRect(contour)
    cv2.rectangle(image_with_boxes, (x, y), (x + w, y + h), (0, 255, 0), 2)  # Green rectangle

# Display the image with bounding boxes
plt.figure(figsize=(6, 6))
plt.imshow(cv2.cvtColor(image_with_boxes, cv2.COLOR_BGR2RGB))
plt.title('Image with Bounding Boxes')
plt.xticks([]), plt.yticks([])
plt.show()
```





In [ ]:

FR. CONCEICAO RODRIGUES COLLEGE OF ENGINEERIG
Department of Computer Engineering

Experiment 4 - Bio Medial Image Analytics

Course Details:

Academic Year	**2023 - 24**	**Estimated Time**	**Experiment No. 4 – 02 Hours**
Course & Semester	**B.E. (COMP) – Sem. VII**	**Subject Name**	**Data Science for Healthand Social Care Lab**
Experiment Type	**Software Performance**	**Subject Code**	**HDSSBL701**

Name of Student	Atharva Pawar	**Roll No.**	9427
Date of Performance.:		**Date of Submission.:**	
CO Mapping	**HDSSBL701.4 - Use algorithms and develop models for healthcare data analytics.**		

Aim: To perform Bio Medial Image Analytics.
Objective: To detect patterns, anomalies, and features within medical images to gain insights and understand the characteristics of data.

Tools :
Jupyter Notebook/MATLAB

Libraries:
OpenCV, Numpy, Pandas, Tensor Flow

Dataset:
Various types of medical imaging modalities, such as X-rays, computed tomography (CT) scans, magnetic resonance imaging (MRI), ultrasound, and microscopy images. Datasets can be obtained from publicly available sources

Procedure:
Step 1: Data Collection and Preparation

Step 2: Feature Extraction

Image Segmentation: If the prognosis requires segmenting specific regions of interest within the images, apply image segmentation techniques to isolate these regions accurately.

Step 3: Model Selection and Training

Choose a Model: Select an appropriate machine learning or deep learning model based on the task (classification, regression) and the nature of the medical image data.

Data Splitting: Divide the dataset into training, validation, and testing subsets.

Model Architecture: Design the architecture of the chosen model, which may include convolutional neural networks (CNNs) for image-based tasks.

Training: Train the model using the training data and monitor its performance on the validation set. Use appropriate loss functions and optimization algorithms.

Step 5: Model Evaluation and Interpretation

Testing: Evaluate the trained model on the testing dataset to assess its performance on unseen data.
Metrics: Calculate relevant metrics such as accuracy, sensitivity, specificity.

Interpretation: Analyze the model's predictions and visually inspect its segmentation results to gain insights into its performance and potential areas for improvement.

Step 6: Documentation and Reporting

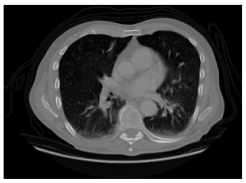
Results: Summarize the experiment's results, including model performance metrics, visualizations, and interpretations.

DSHC : Chest Cancer Detection

Sample Images | Documentation | GITHUB

Choose File | No file chosen

Upload and Predict



Uploaded Image: t1-large.cell.carcinoma.png

Predicted Class: **Large cell carcinoma**

Documentation

```
# Chest Cancer Detection - Documentation


## Project Overview

- Project Name: Chest Cancer Detection
- Dataset: [Chest CTScan Images on Kaggle](https://www.kaggle.com/datasets/mohamedhanyyy/chest-ctscan-images)
- Model Development Code: [Kaggle Notebook](https://www.kaggle.com/code/mrappplg/chest-cancer-detection/notebook)

## Team Members

- Atharva Pawar
- Aditya Vyas
- Harsh Trivedi

##  Classes for Prediction
    Adenocarcinoma: color-Red
    Large Cell Carcinoma: color-Blue
    Squamous Cell Carcinoma: color-Green
    Normal: color-Gray

## Project Description

    This project, "Chest Cancer Detection," is a mini-project in the field of data science applied to healthcare.
    The objective is to develop a deep learning model that can classify chest CT scan images into four categories:
    Adenocarcinoma, Large cell carcinoma, Squamous cell carcinoma, and Normal.
    The project aims to assist in early cancer detection and improve healthcare diagnostics.

## Problem Statement:

    - Problem:
        Chest cancer, including adenocarcinoma, large cell carcinoma, and squamous cell carcinoma, is a major health concern globally.
        Early detection is critical for effective treatment and improved patient outcomes.

    - Objective:
        Develop an automated system that can accurately classify chest CT scan images into four categories: Adenocarcinoma,
        Large cell carcinoma, Squamous cell carcinoma, and Normal, to assist medical professionals in early cancer detection.

## Abstract:

    The Chest Cancer Detection project addresses the pressing need for early cancer diagnosis through the application of deep learning in healthcare.
    The project leverages a Convolutional Neural Network (CNN) model trained on a dataset of chest CT scan images.
    This model classifies the images into four distinct categories, allowing for efficient and accurate detection of chest cancer types.
    The project has been developed and deployed as a web application for easy access and usability.

## Installations:

    ```
    pip install Flask
    pip install tensorflow
    pip install pillow
    pip install numpy

    ```
```

Usage:

    - Access the Web Application:
        Visit the deployed website (e.g., https://dshc-chest-cancer-detection.atharvapawar.repl.co/) where the Chest Cancer Detection system is hosted.

    - Upload an Image:
        On the web application, there should be a file upload form. Use it to upload a chest CT scan image that you want to classify.

    - Prediction:
        After uploading the image, the system will process it using the trained deep learning model.

    - View Results:
        The web application will display the predicted class for the uploaded image, along with the uploaded image itself.

    - Interpretation:
        Interpret the prediction result. If the model predicts a class (e.g., "Adenocarcinoma"), it means the input image is classified as that
        type of chest cancer.

    - Medical Decision Support:
        Medical professionals can use the prediction results as additional information to aid in their diagnoses. However, please note that the
        model's predictions should be considered as supportive information and not a replacement for clinical judgment.

Applications:

    - Early Cancer Detection: The primary application of the project is to assist medical professionals in detecting chest
        cancers at an early stage, which can significantly improve treatment outcomes.

    - Medical Decision Support: The model can serve as a decision support tool for radiologists and oncologists, providing
        them with additional information to make more accurate diagnoses.

    - Reduced Human Error: Automation reduces the risk of human error in the interpretation of medical images, leading to more consistent results.

    - Efficient Triage: The system can help prioritize patients, ensuring that those with a higher likelihood of cancer are assessed more urgently.

    - Telemedicine: The web application can be integrated into telemedicine platforms, enabling remote diagnosis and
        consultation for patients in underserved areas.

Limitations:

    - Data Quality: The accuracy of the model heavily depends on the quality and diversity of the training data.
        Insufficient or biased data may result in misclassifications.

    - Interpretability: Deep learning models are often considered "black boxes," making it challenging to interpret the
        rationale behind specific predictions.

    - Resource Intensive: Running deep learning models can be computationally intensive and may require substantial hardware resources,
        limiting deployment in resource-constrained environments.

Future Scope:

    - Integration with Electronic Health Records (EHR): Integrate the system with electronic health records to provide a comprehensive
        patient history, aiding in more accurate diagnoses.

    - Enhanced Explainability: Develop techniques for better model interpretability to provide insights into why a particular classification
        was made.

    - Real-time Analysis: Enable real-time analysis of streaming medical imaging data for immediate detection and response.

    - Multi-modal Data: Extend the system to handle multiple types of medical imaging data, such as X-rays and MRIs, for a broader
        range of applications.

Model Development and Training

Model Architecture

```python
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Activation, Dropout, Flatten, Dense
from tensorflow.keras.optimizers import Adam

model = Sequential()
model.add(Conv2D(32, (3, 3), input_shape=(img_width, img_height, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(64, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(128, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Flatten())
model.add(Dense(128))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(4, activation='softmax'))  # Four classes

# Compile the model
```

```
model.compile(loss='categorical_crossentropy',
              optimizer=Adam(learning_rate=0.001),
              metrics=['accuracy'])

# Model Training

history = model.fit(
    train_generator,
    steps_per_epoch=train_generator.samples // batch_size,
    epochs=121,  # You can adjust the number of epochs
    validation_data=validation_generator,
    validation_steps=validation_generator.samples // batch_size)

```

CNN Layers

- Input Layer (Conv2D):
  Conv2D(32, (3, 3), input_shape=(img_width, img_height, 3))
    Explanation:
      This is the first layer of your CNN and is responsible for processing the input images.
      32 represents the number of filters or convolutional kernels to be applied in this layer.
      It means there are 32 different feature maps generated.
      (3, 3) defines the size of the convolutional kernel, which is a 3x3 filter in this case.
      input_shape=(img_width, img_height, 3) specifies the shape of the input data.
      In your case, it's images with a width and height of img_width and img_height pixels,
      and 3 channels (RGB color images).
      The Activation('relu') activation function is used to introduce non-linearity into the model.
      ReLU (Rectified Linear Unit) is a commonly used activation function in CNNs.

- MaxPooling Layer (MaxPooling2D):
  MaxPooling2D(pool_size=(2, 2))
    Explanation:
      Max pooling is a downsampling operation used to reduce the spatial dimensions of the feature maps and capture the most
      important information.
      pool_size=(2, 2) specifies that for each 2x2 region in the input, only the maximum value will be retained, reducing the
      spatial dimensions by a factor of 2.

- Convolutional Layer (Conv2D):
  Conv2D(64, (3, 3))
    Explanation:
      This is another convolutional layer with 64 filters and a 3x3 kernel size.
      The Activation('relu') activation function is applied again after this convolution to introduce non-linearity.

- MaxPooling Layer (MaxPooling2D):
  MaxPooling2D(pool_size=(2, 2))
    Explanation:
      Similar to the previous MaxPooling layer, this one further reduces the spatial dimensions of the feature maps.

- Convolutional Layer (Conv2D):
  Conv2D(128, (3, 3))
    Explanation:
    Another convolutional layer with 128 filters and a 3x3 kernel size.
    The Activation('relu') activation function is applied.

- MaxPooling Layer (MaxPooling2D):
  MaxPooling2D(pool_size=(2, 2))
    Explanation:
    Yet another MaxPooling layer to further downsample the feature maps.

- Flatten Layer (Flatten):
  Flatten()
    Explanation:
    This layer is used to flatten the 2D feature maps from the previous layers into a 1D vector. It prepares the data for the
    fully connected layers.

- Dense Layer (Fully Connected):
  Dense(128)
    Explanation:
    This is a fully connected layer with 128 neurons.
    The Activation('relu') activation function is applied to introduce non-linearity.

- Dropout Layer (Dropout):
  Dropout(0.5)
    Explanation:
    Dropout is a regularization technique used to prevent overfitting.
    It randomly sets a fraction of input units to zero during each update, in this case, 50% (0.5).

- Output Layer (Dense):
  Dense(4, activation='softmax')
    Explanation:
    This is the final output layer with 4 neurons, one for each class in your classification task.
    The activation='softmax' activation function is used for multi-class classification.
    It computes the probabilities of each class for a given input.

Deployment
  The trained model is deployed on a website accessible at https://dshc-chest-cancer-detection.atharvapawar.repl.co/.
  Users can upload chest CT scan images, and the website provides predictions for the uploaded images.

Conclusion
  The Chest Cancer Detection project demonstrates the application of deep learning in healthcare for early cancer detection.
  The model is capable of classifying chest CT scan images into four distinct categories,
```
```

providing valuable diagnostic assistance to medical professionals.

For more details, refer to the **Kaggle Notebook** containing the code used in model development.

FR. CONCEICAO RODRIGUES COLLEGE OF ENGINEERIG
Department of Computer Engineering

Experiment 5 and 6 - Natural Language Entity Extraction from Medical Reports

Course Details:

Academic Year	**2023 - 24**	**Estimated Time**	**Experiment No. 5 & 6 – 02 Hours**
Course & Semester	**B.E. (COMP) – Sem. VII**	**Subject Name**	**Data Science for Health and Social Care Lab**
Experiment Type	**Software Performance**	**Subject Code**	**HDSSBL701**

Name of Student	Atharva Pawar	**Roll No.**	9427
Date of Performance.:		**Date of Submission.:**	
CO Mapping	HDSSBL701.2 Clean, integrate and transform healthcare data. HDSSBL701.5 Implement data science solutions for solving healthcare problems.		

Aim: Natural Language Entity Extraction from Medical Reports

Objective: To perform the process of entity extraction from medical text data using NLP techniques and tools..

Tools and Libraries:

- Medical text data (e.g., sample medical reports or synthetic data)
- NLP libraries (e.g., spaCy, NLTK, or Hugging Face Transformers)
- Pre-trained NLP models (e.g., spaCy's "en_core_med7" or BERT-based models)
- Evaluation metrics (precision, recall, F1-score)

Procedure:

Step 1: Data Preparation

Collect a dataset of medical reports in text format.

Understand the structure of the data, including the types of entities needed to extract (e.g., medical conditions, medications, dates).

Step 2: Data Cleaning

Remove any irrelevant information such as headers, footers, page numbers, and boilerplate text.

Remove special characters, symbols, and non-alphanumeric characters that don't contribute to the analysis.

Handle or remove noisy text like HTML tags, XML markup, or other formatting artifacts.

Step 3: Text Pre-processing

a. Tokenization: Split the cleaned text into individual words or tokens.

Consider using more specialized tokenization methods for medical terms or abbreviations.

b. Lowercasing: Convert all text to lowercase to ensure consistent processing.

Stopword Removal:

c. Remove common stopwords that don't contribute much to the overall meaning of the text. Create a custom stopword list that considers medical domain-specific terms.
d. Stemming and Lemmatization: Apply stemming or lemmatization to reduce words to their root forms.Consider using a medical-specific stemmer or lemmatizer if available.
e. Spell Checking and Correction: Implement spell checking and correction techniques to fix common typos or misspellings. Utilize medical dictionaries to ensure accurate corrections for domain-specific terms.
f. Handling Numeric Data:Identify and handle numeric values such as measurements, lab values, and vital signs. Normalize numeric values to a consistent format.
g. Handling Dates and Times: Extract and standardize date and time information from the text. Convert dates to a common format for analysis.
h. Removing Personal Identifiers: Anonymize or remove personally identifiable information (PII) to ensure privacy and compliance with data protection regulations.
i. Handling Missing Data: Decide on strategies for dealing with missing data, such as filling with placeholders or removing affected records.

Step 4: Entity Recognition and Normalization

a. Identify and label medical entities like diseases, treatments, medications, and anatomical terms. Use Named Entity Recognition (NER) tools using libraries like spaCy or specialized NER tools, or libraries trained on medical data if available.
b. Standardize abbreviations, acronyms, and variations of medical terms. Map synonyms to a common representation.

Step 5: Visualizing the impact of data cleaning and pre-processing.

Step 6: Evaluation:

Evaluate the performance of the entity extraction system. Use precision, recall, and F1-score for their extracted entities. Fine-tune the models to improve results.

Dataset:
https://drive.google.com/file/d/1VMu8eLNLfk1SQUJ8HldFYYiEX_WYWKW7/view?usp=drive_link

Reference Links for Implementation:
https://www.analyticsvidhya.com/blog/2019/03/learn-to-use-elmo-to-extract-features-from-text/
https://www.analyticsvidhya.com/blog/2023/02/extracting-medical-information-from-clinical-text-with-nlp/

DSHC - exp - 5 and 6

In [66]:
```
import pandas as pd

# Load the CSV file into a DataFrame
df = pd.read_csv('mtsamples.csv')

# Select only the "transcription" column
trans_col = df['transcription']
```

In [67]:
```
df.head
```

Out[67]:
```
<bound method NDFrame.head of       Unnamed: 0                                        description  \
0              0   A 23-year-old white female presents with comp...
1              1           Consult for laparoscopic gastric bypass.
2              2           Consult for laparoscopic gastric bypass.
3              3                                2-D M-Mode. Doppler.
4              4                                    2-D Echocardiogram
...          ...                                                ...
4994        4994   Patient having severe sinusitis about two to ...
4995        4995   This is a 14-month-old baby boy Caucasian who...
4996        4996   A female for a complete physical and follow u...
4997        4997   Mother states he has been wheezing and coughing.
4998        4998   Acute allergic reaction, etiology uncertain, ...

                medical_specialty                              sample_name  \
0           Allergy / Immunology                          Allergic Rhinitis
1                     Bariatrics   Laparoscopic Gastric Bypass Consult - 2
2                     Bariatrics   Laparoscopic Gastric Bypass Consult - 1
3      Cardiovascular / Pulmonary                    2-D Echocardiogram - 1
4      Cardiovascular / Pulmonary                    2-D Echocardiogram - 2
...                           ...                                       ...
4994        Allergy / Immunology                          Chronic Sinusitis
4995        Allergy / Immunology      Kawasaki Disease - Discharge Summary
4996        Allergy / Immunology                          Followup on Asthma
4997        Allergy / Immunology                       Asthma in a 5-year-old
4998        Allergy / Immunology                Allergy Evaluation Consult

                                          transcription  \
0     SUBJECTIVE:,  This 23-year-old white female pr...
1     PAST MEDICAL HISTORY:, He has difficulty climb...
2     HISTORY OF PRESENT ILLNESS: , I have seen ABC ...
3     2-D M-MODE: , ,1.  Left atrial enlargement wit...
4     1.  The left ventricular cavity size and wall ...
...                                                 ...
4994  HISTORY:,  I had the pleasure of meeting and e...
4995  ADMITTING DIAGNOSIS: , Kawasaki disease.,DISCH...
4996  SUBJECTIVE: , This is a 42-year-old white fema...
4997  CHIEF COMPLAINT: , This 5-year-old male presen...
4998  HISTORY: , A 34-year-old male presents today s...

                                             keywords
0     allergy / immunology, allergic rhinitis, aller...
1     bariatrics, laparoscopic gastric bypass, weigh...
2     bariatrics, laparoscopic gastric bypass, heart...
3     cardiovascular / pulmonary, 2-d m-mode, dopple...
4     cardiovascular / pulmonary, 2-d, doppler, echo...
...                                                 ...
4994                                              NaN
4995  allergy / immunology, mucous membranes, conjun...
4996                                              NaN
4997                                              NaN
4998                                              NaN

[4999 rows x 6 columns]>
```

In [68]:
```
df.tail
```

Out[68]:
```
<bound method NDFrame.tail of       Unnamed: 0                                        description  \
0              0   A 23-year-old white female presents with comp...
1              1           Consult for laparoscopic gastric bypass.
2              2           Consult for laparoscopic gastric bypass.
3              3                                2-D M-Mode. Doppler.
4              4                                    2-D Echocardiogram
...          ...                                                ...
4994        4994   Patient having severe sinusitis about two to ...
4995        4995   This is a 14-month-old baby boy Caucasian who...
4996        4996   A female for a complete physical and follow u...
4997        4997   Mother states he has been wheezing and coughing.
4998        4998   Acute allergic reaction, etiology uncertain, ...

                medical_specialty                              sample_name  \
0           Allergy / Immunology                          Allergic Rhinitis
1                     Bariatrics   Laparoscopic Gastric Bypass Consult - 2
```

```
2                    Bariatrics   Laparoscopic Gastric Bypass Consult - 1
3      Cardiovascular / Pulmonary                  2-D Echocardiogram - 1
4      Cardiovascular / Pulmonary                  2-D Echocardiogram - 2
...                          ...                                    ...
4994         Allergy / Immunology                       Chronic Sinusitis
4995         Allergy / Immunology     Kawasaki Disease - Discharge Summary
4996         Allergy / Immunology                     Followup on Asthma
4997         Allergy / Immunology                  Asthma in a 5-year-old
4998         Allergy / Immunology              Allergy Evaluation Consult

                                        transcription  \
0     SUBJECTIVE:,  This 23-year-old white female pr...
1     PAST MEDICAL HISTORY:, He has difficulty climb...
2     HISTORY OF PRESENT ILLNESS: , I have seen ABC ...
3     2-D M-MODE: , ,1.  Left atrial enlargement wit...
4     1.  The left ventricular cavity size and wall ...
...                                                 ...
4994  HISTORY:,  I had the pleasure of meeting and e...
4995  ADMITTING DIAGNOSIS: , Kawasaki disease.,DISCH...
4996  SUBJECTIVE: , This is a 42-year-old white fema...
4997  CHIEF COMPLAINT: , This 5-year-old male presen...
4998  HISTORY: , A 34-year-old male presents today s...

                                              keywords
0     allergy / immunology, allergic rhinitis, aller...
1     bariatrics, laparoscopic gastric bypass, weigh...
2     bariatrics, laparoscopic gastric bypass, heart...
3     cardiovascular / pulmonary, 2-d m-mode, dopple...
4     cardiovascular / pulmonary, 2-d, doppler, echo...
...                                                 ...
4994                                              NaN
4995  allergy / immunology, mucous membranes, conjun...
4996                                              NaN
4997                                              NaN
4998                                              NaN

[4999 rows x 6 columns]>
```

In [69]:
```
# Display the first 5 rows
print(trans_col.head(10))
```

```
0    SUBJECTIVE:,  This 23-year-old white female pr...
1    PAST MEDICAL HISTORY:, He has difficulty climb...
2    HISTORY OF PRESENT ILLNESS: , I have seen ABC ...
3    2-D M-MODE: , ,1.  Left atrial enlargement wit...
4    1.  The left ventricular cavity size and wall ...
5    PREOPERATIVE DIAGNOSIS: , Morbid obesity.,POST...
6    PREOPERATIVE DIAGNOSES:,1.  Deformity, right b...
7    2-D ECHOCARDIOGRAM,Multiple views of the heart...
8    PREOPERATIVE DIAGNOSIS: , Lipodystrophy of the...
9    DESCRIPTION:,1.  Normal cardiac chambers size....
Name: transcription, dtype: object
```

In [70]:
```
trans_col[0]
```

Out[70]:
```
'SUBJECTIVE:,  This 23-year-old white female presents with complaint of allergies.  She used to have allergies when she lived i
n Seattle but she thinks they are worse here.  In the past, she has tried Claritin, and Zyrtec.  Both worked for short time but
then seemed to lose effectiveness.  She has used Allegra also.  She used that last summer and she began using it again two week
s ago.  It does not appear to be working very well.  She has used over-the-counter sprays but no prescription nasal sprays.  Sh
e does have asthma but doest not require daily medication for this and does not think it is flaring up.,MEDICATIONS: , Her only
medication currently is Ortho Tri-Cyclen and the Allegra.,ALLERGIES: , She has no known medicine allergies.,OBJECTIVE:,Vitals:
Weight was 130 pounds and blood pressure 124/78.,HEENT:  Her throat was mildly erythematous without exudate.  Nasal mucosa was
erythematous and swollen.  Only clear drainage was seen.  TMs were clear.,Neck:  Supple without adenopathy.,Lungs:  Clear.,ASSE
SSMENT:,  Allergic rhinitis.,PLAN:,1.  She will try Zyrtec instead of Allegra again.  Another option will be to use loratadine.
She does not think she has prescription coverage so that might be cheaper.,2.  Samples of Nasonex two sprays in each nostril gi
ven for three weeks.  A prescription was written as well.'
```

In [71]:
```
# Display the last 5 rows
print(trans_col.tail(5))
```

```
4994    HISTORY:,  I had the pleasure of meeting and e...
4995    ADMITTING DIAGNOSIS: , Kawasaki disease.,DISCH...
4996    SUBJECTIVE: , This is a 42-year-old white fema...
4997    CHIEF COMPLAINT: , This 5-year-old male presen...
4998    HISTORY: , A 34-year-old male presents today s...
Name: transcription, dtype: object
```

In [72]:
```
trans_col.describe()
```

Out[72]:
```
count                                          4966
unique                                         2357
top       PREOPERATIVE DIAGNOSIS: , Low back pain.,POSTO...
freq                                              5
Name: transcription, dtype: object
```

In [73]:
```
import nltk

# Download the NLTK tokenizer model (if not already downloaded)
nltk.download('punkt')

# Check for NaN values and replace them with empty strings
trans_col = trans_col.fillna('')

# Tokenize the "transcription" column
```

```
trans_col_tokenize = trans_col.apply(lambda x: nltk.word_tokenize(str(x)))
```

```
[nltk_data] Downloading package punkt to C:\Users\Atharva
[nltk_data]     Pawar\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

In [74]:
```
trans_col_tokenize[0]
```

Out[74]:
```
['SUBJECTIVE',
 ':',
 ',',
 'This',
 '23-year-old',
 'white',
 'female',
 'presents',
 'with',
 'complaint',
 'of',
 'allergies',
 '.',
 'She',
 'used',
 'to',
 'have',
 'allergies',
 'when',
```

In [75]:
```
# Convert text to lowercase in the "transcription" column
trans_col_tok_lowC = trans_col_tokenize.apply(lambda x: [word.lower() for word in x])
trans_col_tok_lowC[0]
```

Out[75]:
```
['subjective',
 ':',
 ',',
 'this',
 '23-year-old',
 'white',
 'female',
 'presents',
 'with',
 'complaint',
 'of',
 'allergies',
 '.',
 'she',
 'used',
 'to',
 'have',
 'allergies',
 'when',
```

In [76]:
```
# Download the NLTK stopwords corpus (if not already downloaded)
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to C:\Users\Atharva
[nltk_data]     Pawar\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

Out[76]:
```
True
```

In [77]:
```
# Import the NLTK stopwords
from nltk.corpus import stopwords

# Define a custom list of medical domain-specific stopwords
custom_stopwords = ["medical_term1", "medical_term2", "medical_term3", ...]

# Get the standard English stopwords
english_stopwords = set(stopwords.words('english'))

# Combine the custom medical stopwords and the standard English stopwords
all_stopwords = set(custom_stopwords).union(english_stopwords)

# Remove stopwords from the "transcription" column using the custom list
trans_col_tok_lowC_stopW = trans_col_tok_lowC.apply(lambda x: [word for word in x if word not in all_stopwords])
trans_col_tok_lowC_stopW[0]
```

Out[77]:
```
['subjective',
 ':',
 ',',
 '23-year-old',
 'white',
 'female',
 'presents',
 'complaint',
 'allergies',
 '.',
 'used',
 'allergies',
 'lived',
 'seattle',
 'thinks',
 'worse',
 '.',
```

```
 'past',
 ',',
```

In [78]:
```
import spacy

# Load the spaCy English language model
nlp = spacy.load("en_core_web_sm")
```

In [79]:
```
# Define a function for lemmatization
def lemmatize_text(text):
    doc = nlp(" ".join(text))
    return [token.lemma_ for token in doc]

# Apply lemmatization to the "transcription" column
trans_col_tok_lowC_stopW_Lemm = trans_col_tok_lowC_stopW.apply(lemmatize_text)
trans_col_tok_lowC_stopW_Lemm[0]
```

Out[79]:
```
['subjective',
 ':',
 ',',
 '23',
 '-',
 'year',
 '-',
 'old',
 'white',
 'female',
 'present',
 'complaint',
 'allergy',
 '.',
 'use',
 'allergy',
 'live',
 'seattle',
 'think',
```

In [80]:
```
# !pip install indexer
# !pip install pyspellchecker
```

In [81]:
```
# import re
from spellchecker import SpellChecker
# import dateutil.parser
```

In [82]:
```
# Spell Checking and Correction
def spell_check_and_correct(text):
    spell = SpellChecker()
    words = text.split()
    corrected_words = [spell.correction(word) if word is not None else word for word in words]
    corrected_words = [word for word in corrected_words if word is not None]  # Filter out None values
    corrected_text = ' '.join(corrected_words)
    return corrected_text
```

In [83]:
```
# Handling Numeric Data
def extract_numeric_data(text):
    numeric_values = re.findall(r'\d+\.\d+|\d+', text)
    return [float(value) for value in numeric_values]
```

In [84]:
```
# Handling Dates and Times
def extract_dates_and_times(text):
    try:
        dates = dateutil.parser.parse(text, fuzzy=True)
        return dates
    except:
        return None
```

In [85]:
```
# Removing Personal Identifiers (Names in this example)
def remove_names(text):
    name_pattern = re.compile(r'\b[A-Z][a-z]+\b')
    return name_pattern.sub('XXXX', text)
```

In [86]:
```
# Handling Missing Data (Fill with Placeholder in this example)
def fill_missing_data(text, placeholder="[MISSING]"):
    return text.replace("MISSING", placeholder)
```

In [87]:
```
# Sample Text
sample_text = "A 23-year-old white female presents with complaint of allergies. Allergy / Immunology. Allergic Rhinitis. SUBJECTI

# Apply Functions
sample_text = spell_check_and_correct(sample_text)
numeric_data = extract_numeric_data(sample_text)
dates_and_times = extract_dates_and_times(sample_text)
sample_text = remove_names(sample_text)
sample_text = fill_missing_data(sample_text)

# Output
print("Spell Checked and Corrected Text:")
print(sample_text)
print("\nExtracted Numeric Data:")
print(numeric_data)
```

```
print("\nExtracted Dates and Times: ")
print(dates_and_times)
# This script defines the functions and demonstrates their use with a sample text. You can adapt and use these functions accordir
```

```
Spell Checked and Corrected Text:
A white female presents with complaint of allergies XXXX / immunology XXXX subjective XXXX white female presents with complaint
of allergies XXXX used to have allergies when she lived in XXXX but she thinks they are worse here XXXX has tried clarity and X
XXX worked for a short time but then seemed to lose effectiveness XXXX has used XXXX also XXXX used that last summer and she be
gan using it again two weeks ago XXXX does not appear to be working very well XXXX has used sprays but no prescription nasal sp
rays XXXX does have asthma but does not require daily medication for this and does not think it is flaring up

Extracted Numeric Data:
[]

Extracted Dates and Times:
None
```

In [88]:
```
# Download the NER corpus if not already downloaded
nltk.download('maxent_ne_chunker')
nltk.download('words')
```

```
[nltk_data] Downloading package maxent_ne_chunker to C:\Users\Atharva
[nltk_data]     Pawar\AppData\Roaming\nltk_data...
[nltk_data]   Package maxent_ne_chunker is already up-to-date!
[nltk_data] Downloading package words to C:\Users\Atharva
[nltk_data]     Pawar\AppData\Roaming\nltk_data...
[nltk_data]   Package words is already up-to-date!
```

Out[88]:
```
True
```

In [89]:
```
import nltk


from nltk import word_tokenize, pos_tag, ne_chunk

def custom_ner(text):
    words = word_tokenize(text)
    tagged = pos_tag(words)
    named_entities = ne_chunk(tagged)

    return named_entities

# # Sample text for NER
# sample_text = "Barack Obama was born in Honolulu. He was the President of the United States."

# # Apply the custom NER function
# entities = custom_ner(sample_text)

# # Print the named entities
# print(entities)
```

In [90]:
```
# Apply the custom NER function to the 'description' column
df['NER'] = df['description'].apply(custom_ner)

print(df['NER'])
```

```
0       [(A, DT), (23-year-old, JJ), (white, JJ), (fem...
1       [[(Consult, NN)], (for, IN), (laparoscopic, NN...
2       [[(Consult, NN)], (for, IN), (laparoscopic, NN...
3       [(2-D, JJ), (M-Mode, NNP), (., .), [(Doppler, ...
4                       [(2-D, JJ), (Echocardiogram, NNP)]
                              ...
4994    [(Patient, NNP), (having, VBG), (severe, JJ), ...
4995    [(This, DT), (is, VBZ), (a, DT), (14-month-old...
4996    [(A, DT), (female, NN), (for, IN), (a, DT), (c...
4997    [(Mother, RB), (states, NNS), (he, PRP), (has,...
4998    [[(Acute, NNP)], (allergic, JJ), (reaction, NN...
Name: NER, Length: 4999, dtype: object
```

In [91]:
```
ner_row_2 = df.loc[1, 'NER']  # Access row 2 (index 1) and the 'NER' column
print(ner_row_2)
```

```
(S (GSP Consult/NN) for/IN laparoscopic/NN gastric/JJ bypass/NN ./.)
```

In [ ]:

In [ ]:

FR. CONCEICAO RODRIGUES COLLEGE OF ENGINEERIG
Department of Computer Engineering

Experiment 7 - Social Media Analytics for Drug Review analytics

Course Details:

Academic Year	**2023 - 24**	**Estimated Time**	**Experiment No. 7 – 02 Hours**
Course & Semester	**B.E. (COMP) – Sem. VII**	**Subject Name**	**Data Science for Health and Social Care Lab**
Experiment Type	**Software Performance**	**Subject Code**	**HDSSBL701**

Name of Student	Atharva Pawar	**Roll No.**	9427
Date of Performance.:		**Date of Submission.:**	
CO Mapping	**HDSSBL701.3** - Demonstrate statistical data analysis and visualization techniques on healthcare data.		

Aim: To perform Social Media Analytics for Drug Review analytics.
Objective: To perform Twitter / instagram content based analysis.

Tools :

Keyhole - Real-Time Social Media Analytics & Reporting Tool

Result:
(Mention which Drug manufacturing / Pharmaceutical Company considered and Attach screenshots of comparison analytics)

0

Brand Comparison

AUG 12 - SEP 11

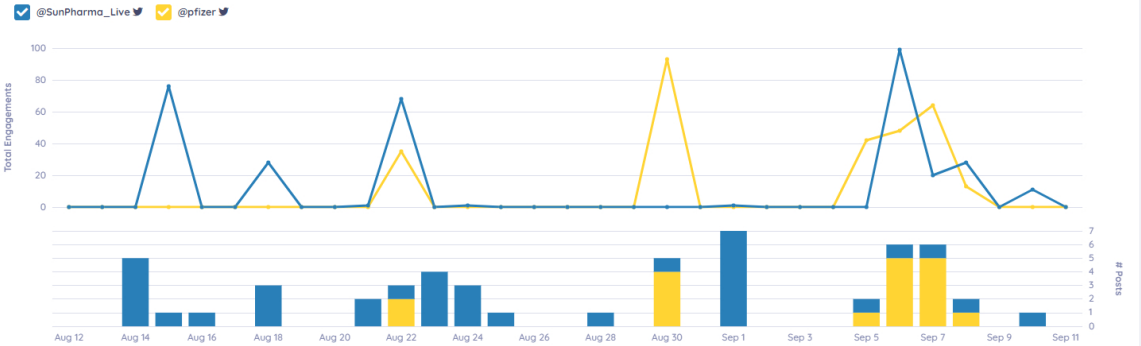
GO TO MY COMPARISON GROUPS | GO TO AGGREGATED DASHBOARD | EXPORT | SHARE

@SunPharma_... ×
Status: Running

@pfizer ×
Status: Running

Timeline

ENGAGEMENTS | GROUP BY DAY



Dates & times are based on your local timezone: (GMT+0530 | IST)

Keyhole

Summary

	53 POSTS	520,760 TOTAL FOLLOWERS	628 TOTAL ENGAGEMENT	0.1% ENG. RATE (BY FOLLOWERS)
@SunPharma_Live...	35	29,532	333	1.1%
@pfizer	18	491,228	295	0.1%

Top Posts

@SunPharma_Live @pfizer

@SunPharma_Live Sep 6 — 14 85
We were voted as the Most Outstanding Company in India – Healthcare as per the 2023 Asiamoney Asia's Outstanding Companies Poll which acknowledges the performance of globally listed companies in finance, management, investor relations and CSR: https://t.co/0aaZ77LvRM #WeAreSun

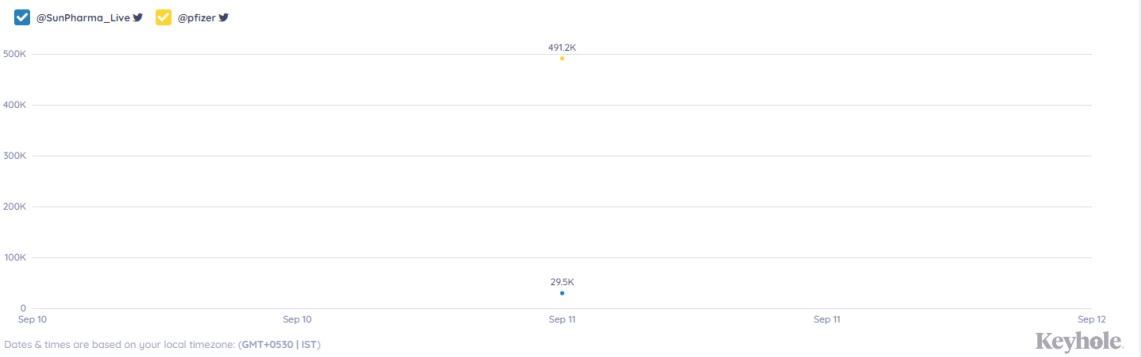
@SunPharma_Live Aug 22 — 9 59
Our MD, Dilip Shanghvi was felicitated with the 'Lifetime Achievement' award at @Moneycontrolcom's Indian Family Business Awards, 2022. He received the award from our Hon'ble Minister of Commerce & Industry, @PiyushGoyal. Watch his acceptance speech here: https://t.co/Zt044aehV4

@SunPharma_Live Aug 15 — 6 70
From reaching communities far and wide to touching lives as a leading provider of valued medicines, we pledge our continued commitment to building a healthier India. Happy Independence Day! #ReachingPeopleTouchingLives

@pfizer Aug 30 — 13 39

Follower Growth

FOLLOWER COUNT | NET CHANGE | GROUP BY DAY

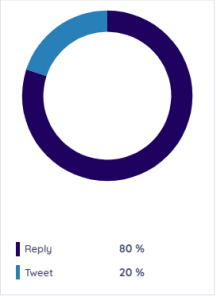


Dates & times are based on your local timezone: (GMT+0530 | IST)

Keyhole

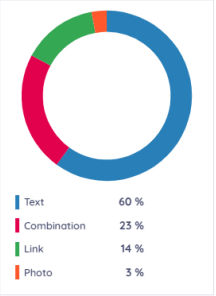
Most Frequent Post Types

What type of content is being posted?



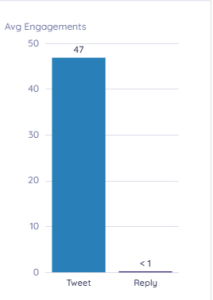
Most Frequent Media Types

What type of media is being posted?



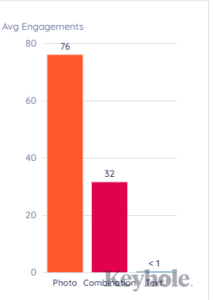
Most Engaging Post Types

Does this match with the profile's most frequent post types?



Most Engaging Media Types

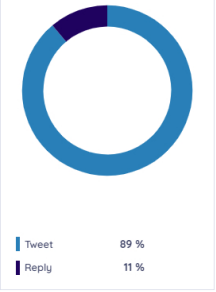
Does this match with the profile's most frequent media types?



pfizer

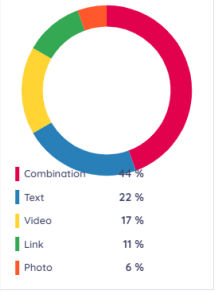
Most Frequent Post Types

What type of content is being posted?



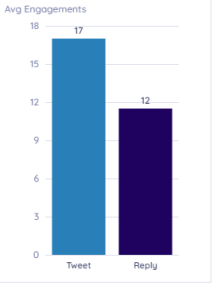
Most Frequent Media Types

What type of media is being posted?



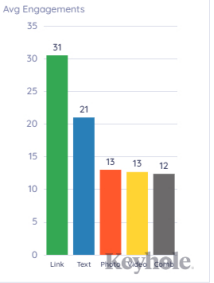
Most Engaging Post Types

Does this match with the profile's most frequent post types?



Most Engaging Media Types

Does this match with the profile's most frequent media types?



FR. CONCEICAO RODRIGUES COLLEGE OF ENGINEERIG
Department of Computer Engineering

Experiment 8 - Exploratory Data Analysis and Sentiment Analysis of Drug Reviews

Course Details:

Academic Year	**2023 - 24**	**Estimated Time**	**Experiment No. 8 – 02 Hours**
Course & Semester	**B.E. (COMP) – Sem. VII**	**Subject Name**	**Data Science for Health and Social Care Lab**
Experiment Type	**Software Performance**	**Subject Code**	**HDSSBL701**

Name of Student	Atharva Pawar	**Roll No.**	9427
Date of Performance.:		**Date of Submission.:**	
CO Mapping	**HDSSBL701.3 -** Demonstrate statistical data analysis and visualization techniques on healthcare data.		

Aim: Exploratory Data Analysis and Sentiment Analysis of Drug reviews.

Objective:

1. To plot a scatter plot on rating verses useful Count.
2. To draw bar graph of number of reviews per condition (depression , panic condition etc.)
3. To Perform sentiment analysis using TextBlob module of python for text classification as positive, negative and neutral.

Steps:

1. Search and collect Drug Review data set from the UCI machine learning repository
2. Perform EDA and SA on the drug review dataset.

Result:

Reference:

http://www.scielo.org.mx/scielo.php?script=sci_arttext&pid=S1405-55462022000301191

Exp - 8

About Dataset

- The Drug Review Dataset is taken from the UCI Machine Learning Repository.**It has 7 features** including the review and **161297 Data Points** or entries.

Attributes :

1. **drugName :** name of drug 2.**condition :** name of condition
2. **review :** patient review
3. **rating :** 10 star patient rating
4. **date :** date of review entry
5. **usefulCount :** number of users who found review useful

In [1]:
```
# Load EDA Pkgs
import pandas as pd
import numpy as np
```

In [2]:
```
# Load Data Viz
import matplotlib.pyplot as plt
import seaborn as sns
import plotly as px
plt.style.use('fivethirtyeight')
%matplotlib inline
```

In [3]:
```
# Load Sentiment Pkgs
from textblob import TextBlob
```

Loading the Datset

In [4]:
```
# Load Dataset
df = pd.read_csv("../input/kuc-hackathon-winter-2018/drugsComTrain_raw.csv")
```

In [5]:
```
# Preview Dataset
df.head()
```

Out[5]:

	uniqueID	drugName	condition	review	rating	date	usefulCount
0	206461	Valsartan	Left Ventricular Dysfunction	"It has no side effect, I take it in combinati...	9	20-May-12	27
1	95260	Guanfacine	ADHD	"My son is halfway through his fourth week of ...	8	27-Apr-10	192
2	92703	Lybrel	Birth Control	"I used to take another oral contraceptive, wh...	5	14-Dec-09	17
3	138000	Ortho Evra	Birth Control	"This is my first time using any form of birth...	8	3-Nov-15	10
4	35696	Buprenorphine / naloxone	Opiate Dependence	"Suboxone has completely turned my life around...	9	27-Nov-16	37

Data Exploration

In [6]:
```
# Columns
df.columns
```

Out[6]:
```
Index(['uniqueID', 'drugName', 'condition', 'review', 'rating', 'date',
       'usefulCount'],
      dtype='object')
```

In [7]:
```
# Missing Values
df.isnull().sum()
```

Out[7]:
```
uniqueID         0
drugName         0
condition      899
review           0
rating           0
date             0
usefulCount      0
dtype: int64
```

Observation :

- Most of the missing values are in the condition column
- This implies that most people don't know their condition by name or privacy

In [8]:
```
df.describe().T
```

Out[8]:

	count	mean	std	min	25%	50%	75%	max
uniqueID	161297.0	115923.585305	67004.445170	2.0	58063.0	115744.0	173776.0	232291.0
rating	161297.0	6.994377	3.272329	1.0	5.0	8.0	10.0	10.0
usefulCount	161297.0	28.004755	36.403742	0.0	6.0	16.0	36.0	1291.0

In [9]:
```
df.info()
print('==============='*10)
df.dtypes
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 161297 entries, 0 to 161296
Data columns (total 7 columns):
 #   Column       Non-Null Count   Dtype 
---  ------       --------------   ----- 
 0   uniqueID     161297 non-null  int64 
 1   drugName     161297 non-null  object
 2   condition    160398 non-null  object
 3   review       161297 non-null  object
 4   rating       161297 non-null  int64 
 5   date         161297 non-null  object
 6   usefulCount  161297 non-null  int64 
dtypes: int64(3), object(4)
memory usage: 8.6+ MB
==========================================================================================================================================
=============
```

Out[9]:
```
uniqueID        int64
drugName       object
condition      object
review         object
rating          int64
date           object
usefulCount     int64
dtype: object
```

EDA and Data Visualization

-- How many drugs do we have?

In [11]:
```
# How many drugs do we have?
num = len(df['drugName'].unique().tolist())
print('Number of Drugs are -',num )
```

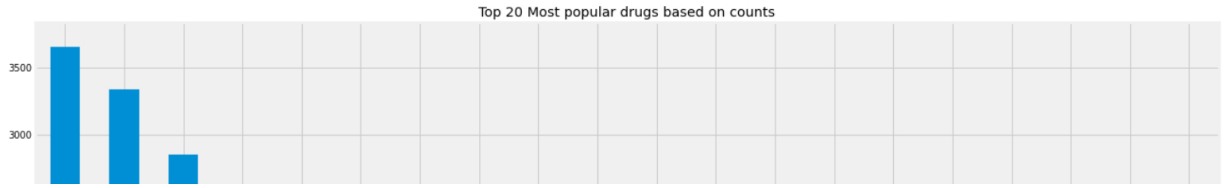
```
Number of Drugs are - 3436
```

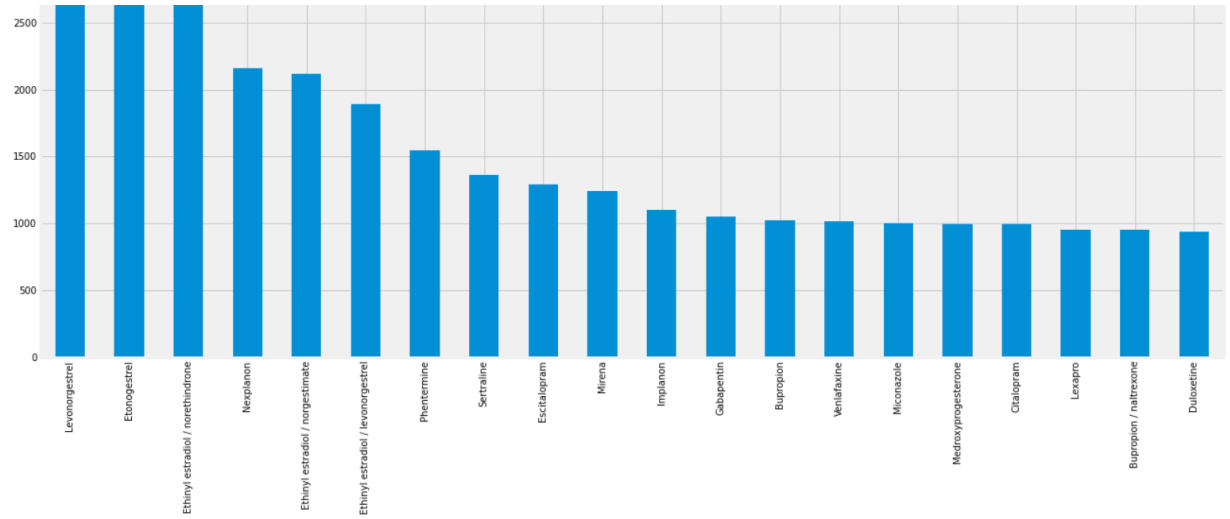
--- What are the Most popular Drug?

In [12]:
```
# What is the most popular drug?
# Top 20 Drugs (Most Popular)
df['drugName'].value_counts().nlargest(20)
```

Out[12]:
```
Levonorgestrel                      3657
Etonogestrel                        3336
Ethinyl estradiol / norethindrone    2850
Nexplanon                           2156
Ethinyl estradiol / norgestimate     2117
Ethinyl estradiol / levonorgestrel   1888
Phentermine                         1543
Sertraline                          1360
Escitalopram                        1292
Mirena                              1242
Implanon                            1102
Gabapentin                          1047
Bupropion                           1022
Venlafaxine                         1016
Miconazole                          1000
Medroxyprogesterone                  995
Citalopram                           995
Lexapro                              952
Bupropion / naltrexone               950
Duloxetine                           934
Name: drugName, dtype: int64
```

In [13]:
```
# Top 20 Drugs (Most Popular)
plt.figure(figsize=(20,10))
df['drugName'].value_counts().nlargest(20).plot(kind='bar')
plt.title("Top 20 Most popular drugs based on counts")
plt.show()
```

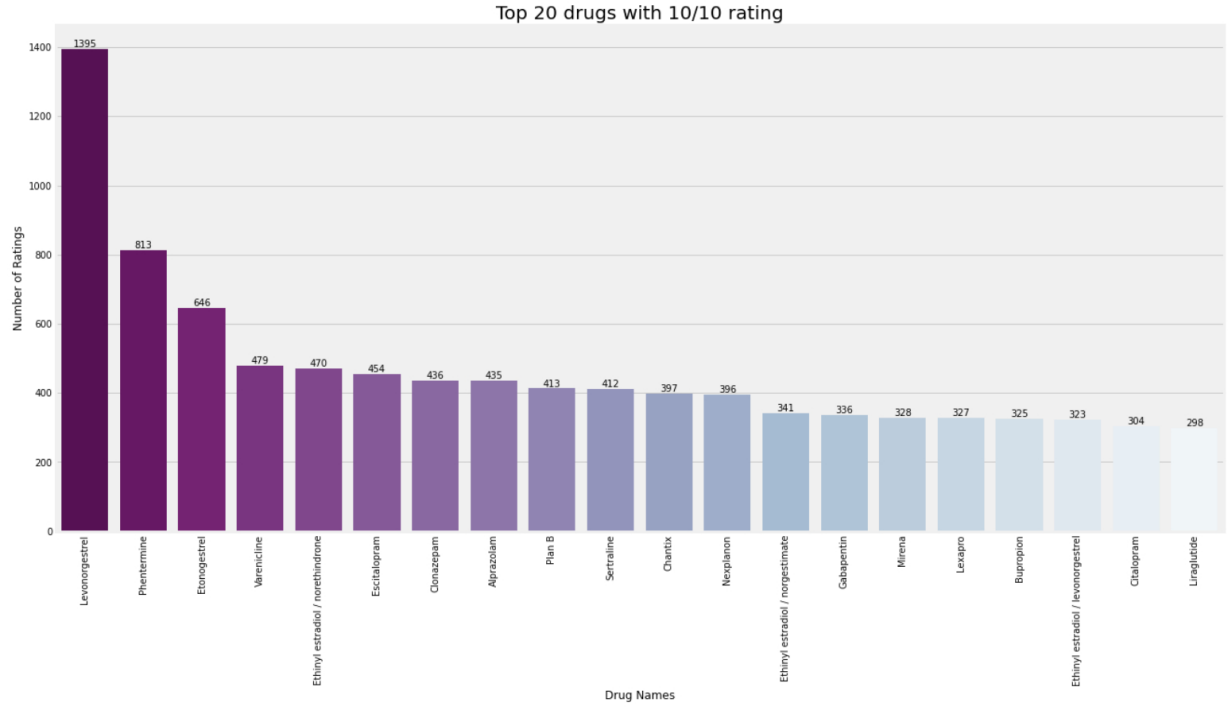




Observation

- Most of the commonest drugs are hormonal drugs

In [14]:
```
# Top 20 Drugs (Most Popular)
plt.figure(figsize=(20,10))
rating = dict(df.loc[df.rating == 10, "drugName"].value_counts())
drugname = list(rating.keys())
drug_rating = list(rating.values())

sns_rating = sns.barplot(x = drugname[0:20], y = drug_rating[0:20], palette = 'BuPu_r')

for i in sns_rating.containers:
    sns_rating.bar_label(i,)

sns_rating.set_title('Top 20 drugs with 10/10 rating', fontsize=20)
sns_rating.set_ylabel("Number of Ratings")
sns_rating.set_xlabel("Drug Names")
plt.setp(sns_rating.get_xticklabels(), rotation=90)
plt.show()
```

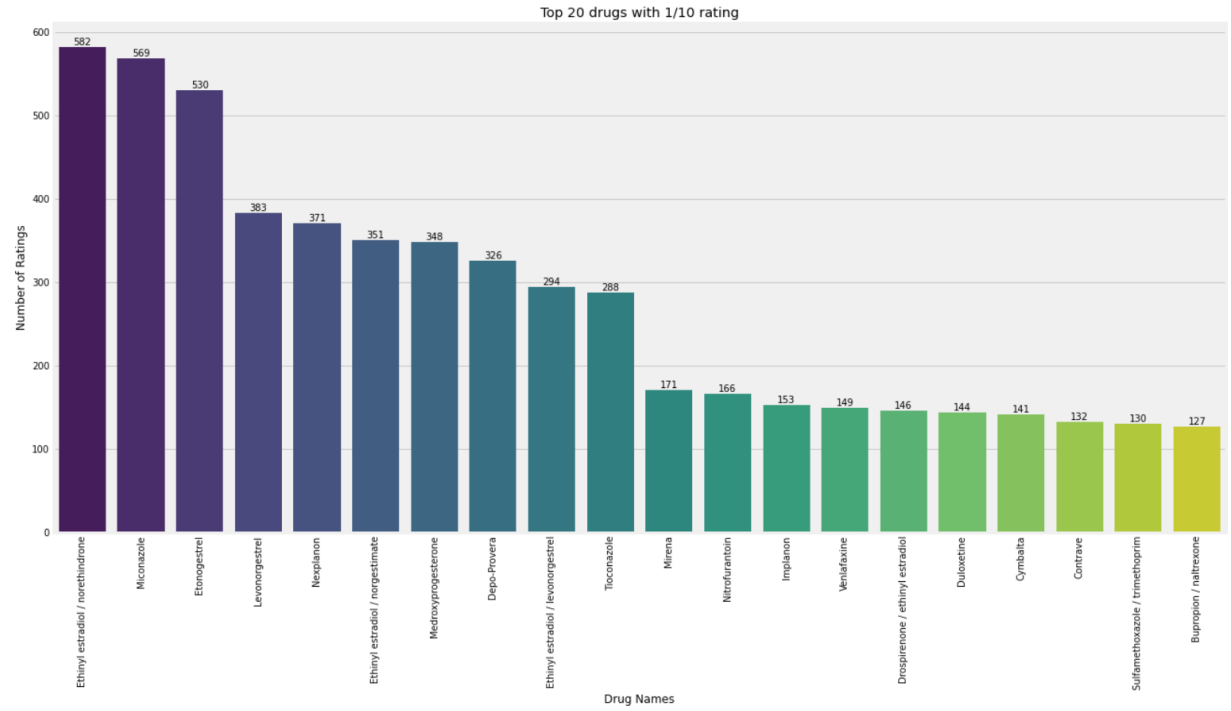


Observation :

Levonorgestrel, Phentemine, Etonogestrel are the Durgs with the **Heighest Ratings**

In [15]:
```
# Least 20 Drugs (Most Popular)
plt.figure(figsize=(20,10))
rating = dict(df.loc[df.rating == 1, "drugName"].value_counts())
drugname = list(rating.keys())
drug_rating = list(rating.values())

sns_rating = sns.barplot(x = drugname[0:20], y = drug_rating[0:20], palette = 'viridis')
```

```
for i in sns_rating.containers:
    sns_rating.bar_label(i,)

sns_rating.set_title('Top 20 drugs with 1/10 rating')
sns_rating.set_ylabel("Number of Ratings")
sns_rating.set_xlabel("Drug Names")
plt.setp(sns_rating.get_xticklabels(), rotation=90)
plt.show()
```

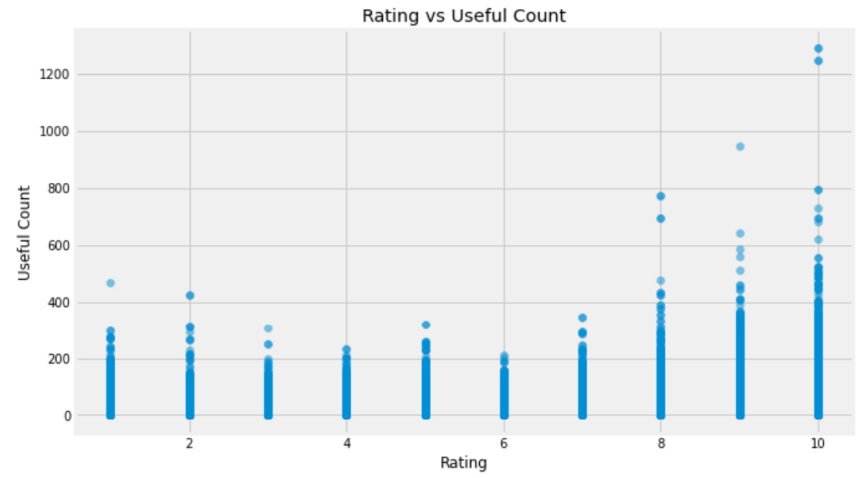


Observation :

Norethindrone, Miconzole are the Drugs with the **Least ratings** given by users

Plot a scatter plot on rating verses useful Count.

In [72]:
```
def plot_scatter_rating_vs_useful_count(dataframe):
    plt.figure(figsize=(10, 6))  # Adjust the figure size as needed
    plt.scatter(dataframe['rating'], dataframe['usefulCount'], alpha=0.5)
    plt.title('Rating vs Useful Count')
    plt.xlabel('Rating')
    plt.ylabel('Useful Count')
    plt.grid(True)
    plt.show()

# Call the function to create the scatter plot
plot_scatter_rating_vs_useful_count(df)
```



-- What are the groups/classification of drugs used?

- suffix or endings

In [16]:
```
drug_suffix = {"azole":"antifungal (except metronidazole)",
"caine":"anesthetic",
"cillin":"antibiotic(penicillins)",
"mycin":"antibiotic",
"micin":"antibiotic",
"cycline":"antibiotic",
"oxacin":"antibiotic",
"ceph":"antibiotic(cephalosporins)",
"cef":"antibiotic (cephalosporins)",
"dine":"h2 blockers (anti-ulcers)",
"done":"opiod analgesics",
"ide":"oral hypoglycemics",
"lam":"anti-anxiety",
"pam":"anti-anxiety",
"mide":"diuretics",
"zide":"diuretics",
"nium":"neuromuscular blocking agents",
"olol":"beta blockers",
"tidine":"h2 antagonist",
"tropin":"pituitary hormone",
"zosin":"alpha blocker",
"ase":"thrombolytics",
"plase":"thrombolytics",
"azepam":"anti-anziety(benzodiazepine)",
"azine":"antipyschotics (phenothiazine)",
"barbital":"barbiturate",
"dipine":"calcium channel blocker",
"lol":"beta blocker",
"zolam":"cns depressants",
"pril":"ace inhibitor",
"artan":"arb blocker",
"statins":"lipid-lowering drugs",
"parin":"anticoagulants",
"sone":"corticosteroid (prednisone)"}
```

In [17]:
```
def classify_drug(drugname):
    for i in drug_suffix.keys():
        if drugname.endswith(i):
            return drug_suffix[i]
```

In [18]:
```
classify_drug('valsartan')
```

Out[18]:
```
'arb blocker'
```

In [19]:
```
df['drug_class'] = df['drugName'].apply(classify_drug)
```

In [20]:
```
df[['drugName','drug_class']]
```

Out[20]:

	drugName	drug_class
0	Valsartan	arb blocker
1	Guanfacine	None
2	Lybrel	None
3	Ortho Evra	None
4	Buprenorphine / naloxone	None
...	...	...
161292	Campral	None
161293	Metoclopramide	oral hypoglycemics
161294	Orencia	None
161295	Thyroid desiccated	None
161296	Lubiprostone	None

161297 rows × 2 columns

In [21]:
```
# How many Groups of Drugs By Class
df['drug_class'].unique().tolist()
```

Out[21]:
```
['arb blocker',
 None,
 'antifungal (except metronidazole)',
 'oral hypoglycemics',
 'opiod analgesics',
 'antibiotic',
 'anti-anxiety',
 'h2 blockers (anti-ulcers)',
 'beta blockers',
 'ace inhibitor',
 'thrombolytics',
 'alpha blocker',
 'corticosteroid (prednisone)',
 'antipyschotics (phenothiazine)',
 'antibiotic(penicillins)',
 'barbiturate',
 'calcium channel blocker',
 'anesthetic',
 'pituitary hormone',
 'antibiotic (cephalosporins)',
 'beta blocker',
```

```
 'neuromuscular blocking agents',
 'anticoagulants']
```

In [22]:
```
# How many Groups of Drugs By Class
grp_drugs = len(df['drug_class'].unique().tolist())
print('Groups of Drugs by Class - ',grp_drugs)
```

```
Groups of Drugs by Class -  23
```

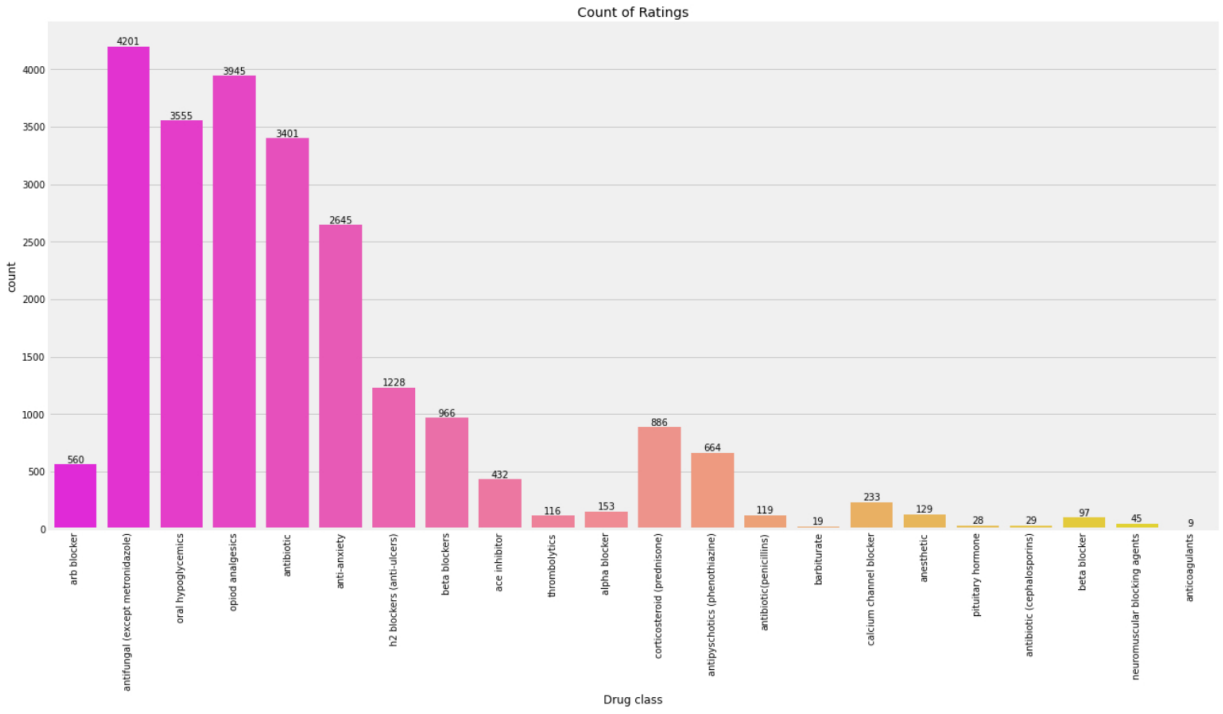
-- Which class of Drug is the most common ?

In [23]:
```
# Which of class of drug  is the most commonest
df['drug_class'].value_counts()
```

Out[23]:
```
antifungal (except metronidazole)    4201
opiod analgesics                     3945
oral hypoglycemics                   3555
antibiotic                           3401
anti-anxiety                         2645
h2 blockers (anti-ulcers)            1228
beta blockers                         966
corticosteroid (prednisone)           886
antipyschotics (phenothiazine)        664
arb blocker                           560
ace inhibitor                         432
calcium channel blocker               233
alpha blocker                         153
anesthetic                            129
antibiotic(penicillins)               119
thrombolytics                         116
beta blocker                           97
neuromuscular blocking agents          45
antibiotic (cephalosporins)            29
pituitary hormone                      28
barbiturate                            19
anticoagulants                          9
Name: drug_class, dtype: int64
```

In [24]:
```
# Which of class of drug  is the most commonest

plt.figure(figsize=(20,10))
sns_1 = sns.countplot(df['drug_class'], palette = 'spring')

for i in sns_1.containers:
    sns_1.bar_label(i,)

sns_1.set_title('Count of Ratings')
sns_1.set_xlabel("Rating")
sns_1.set_xlabel("Drug class")
plt.setp(sns_1.get_xticklabels(), rotation=90)
plt.show()
```

```
/opt/conda/lib/python3.7/site-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg:
x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keywor
d will result in an error or misinterpretation.
  FutureWarning
```



Observations :

Observations :

- The most commonest class/group of drugs used are
  - **Antifungal**
  - **Opiod Analgesics(Pain Killers)**
  - **Oral Hypoglycemics (DM)**
  - **Antibiotic**

-- Distribution of Drugs(class) Per Drug Group based on size

In [25]:
```
# Distribution of Drugs Per Drug Group based on size
drug_groups = df.groupby('drug_class').size()
```

In [26]:
```
type(drug_groups)
```

Out[26]:
```
pandas.core.series.Series
```

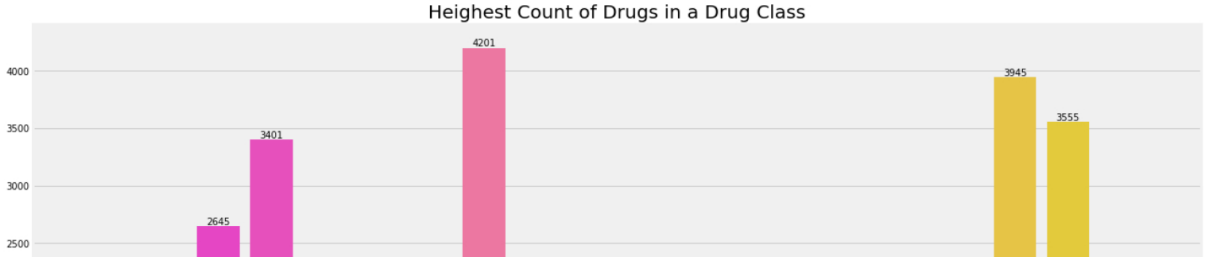
In [27]:
```
# Convert to DF
# Method 1
drug_groups.to_frame()
```

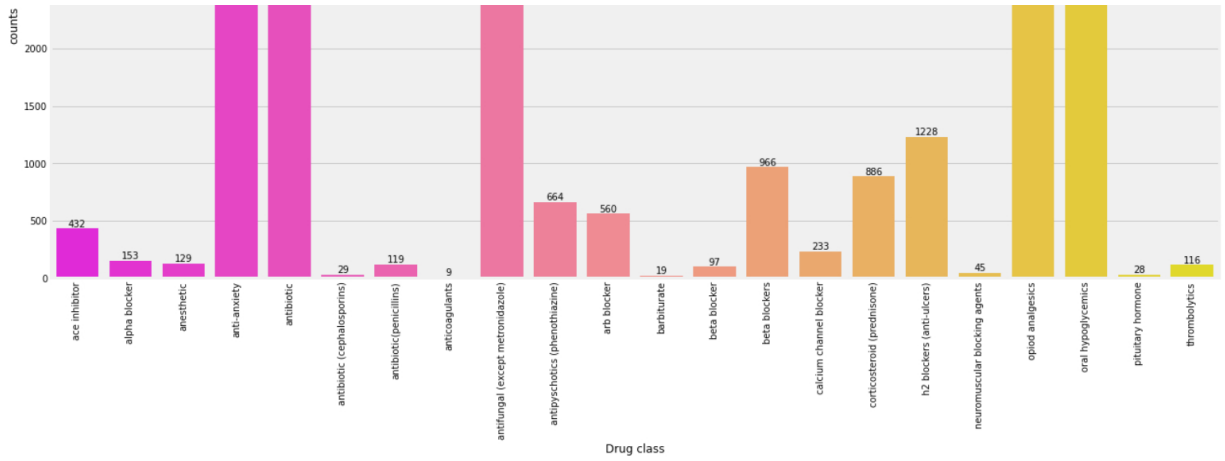
Out[27]:

drug_class	0
ace inhibitor	432
alpha blocker	153
anesthetic	129
anti-anxiety	2645
antibiotic	3401
antibiotic (cephalosporins)	29
antibiotic(penicillins)	119
anticoagulants	9
antifungal (except metronidazole)	4201
antipyschotics (phenothiazine)	664
arb blocker	560
barbiturate	19
beta blocker	97
beta blockers	966
calcium channel blocker	233
corticosteroid (prednisone)	886
h2 blockers (anti-ulcers)	1228
neuromuscular blocking agents	45
opiod analgesics	3945
oral hypoglycemics	3555
pituitary hormone	28
thrombolytics	116

In [28]:
```
# Convert to DF
# Method 2
drug_groups_df = pd.DataFrame({'drug_class':drug_groups.index,'counts':drug_groups.values})
```

In [29]:
```
# Seaborn Plot
plt.figure(figsize=(20,10))
# g = sns.barplot(data=drug_groups_df,x='drug_class',y='counts', palette='gnuplot_r')

sns_1 = sns.barplot(data=drug_groups_df,x='drug_class',y='counts', palette = 'spring')

for i in sns_1.containers:
    sns_1.bar_label(i,)

sns_1.set_title('Heighest Count of Drugs in a Drug Class', fontsize=20)
sns_1.set_xlabel("count")
sns_1.set_xlabel("Drug class")


sns_1.set_xticklabels(drug_groups_df['drug_class'].values,rotation=90)
plt.show()
```





Observations :

Here we can see **Antifungal, Opiod Analgesics, Oral Hypoglycemeics** is the Drug Class with **heighest count of Drugs**

-- How many Conditions are suffered by patients?

In [30]:
```
len(df['condition'].unique().tolist())
```

Out[30]:
```
885
```

Observation :

- We have 885 different conditions

In [31]:
```
#### Distribution of Conditions
df['condition'].value_counts()
```

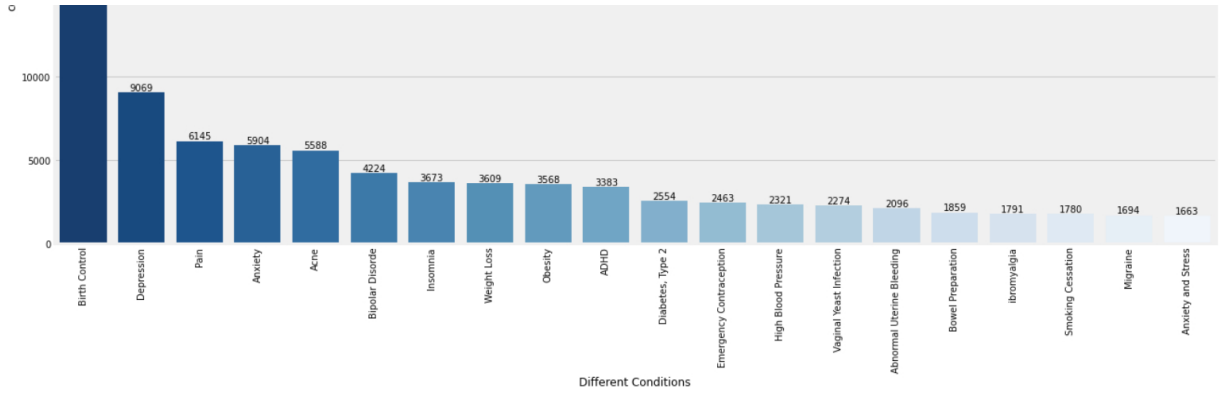
Out[31]:
```
Birth Control                                        28788
Depression                                            9069
Pain                                                  6145
Anxiety                                               5904
Acne                                                  5588
                                                     ...
Dissociative Identity Disorde                            1
Hydrocephalus                                            1
Hyperlipoproteinemia Type III, Elevated beta-VLDL   IDL      1
Q Feve                                                   1
Neutropenia                                              1
Name: condition, Length: 884, dtype: int64
```

-- Most common condition suffered

In [32]:
```
#### Most commonest Conditions
common_conditions = df['condition'].value_counts().nlargest(20)
```

In [33]:
```
# Change the Variable Name
new = pd.DataFrame({'drug_class':common_conditions.index,'counts':common_conditions.values})
```

In [34]:
```
#### Most commonest Conditions
plt.figure(figsize=(20,10))

sns_1 = sns.barplot(data=new,x='drug_class',y='counts', palette = 'Blues_r')

for i in sns_1.containers:
    sns_1.bar_label(i,)

sns_1.set_title('Most suffered Conditions', fontsize=20)
sns_1.set_xlabel("Rating")
sns_1.set_xlabel("Different Conditions")
plt.setp(sns_1.get_xticklabels(), rotation=90)
plt.show()
```



Observations :

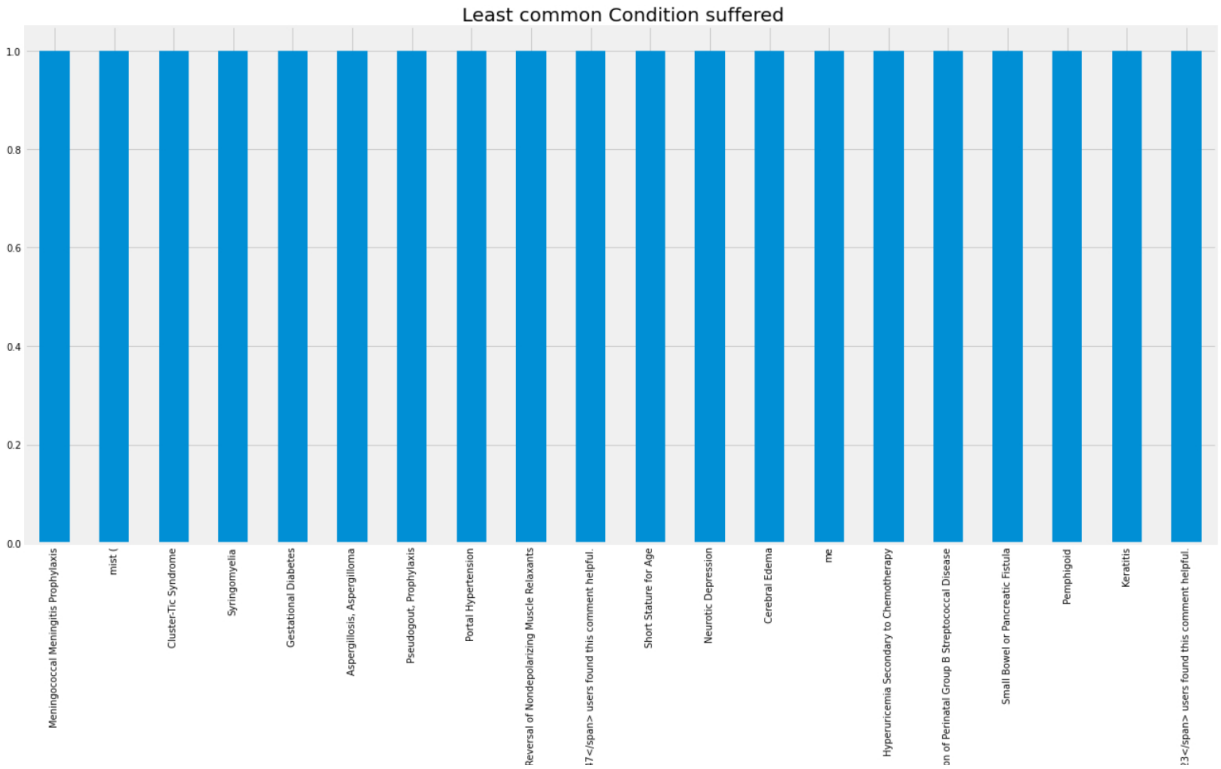
- The most commonest condition is **Birth Control**,followed by **Depression** and **Pain and Anxiety**
- Makes sense as compared to the drug distribution

-- Least common Condition suffered

In [35]:
```
df['condition'].value_counts().nsmallest(20)
```

Out[35]:
```
Meningococcal Meningitis Prophylaxis                    1
mist (                                                  1
Cluster-Tic Syndrome                                    1
Syringomyelia                                           1
Gestational Diabetes                                    1
Aspergillosis, Aspergilloma                             1
Pseudogout, Prophylaxis                                 1
Portal Hypertension                                     1
Reversal of Nondepolarizing Muscle Relaxants            1
47</span> users found this comment helpful.             1
Short Stature for Age                                   1
Neurotic Depression                                     1
Cerebral Edema                                          1
me                                                      1
Hyperuricemia Secondary to Chemotherapy                 1
Prevention of Perinatal Group B Streptococcal Disease   1
Small Bowel or Pancreatic Fistula                       1
Pemphigoid                                              1
Keratitis                                               1
123</span> users found this comment helpful.            1
Name: condition, dtype: int64
```

In [36]:
```
#### Least commonest Conditions
df['condition'].value_counts().nsmallest(20).plot(kind='bar',figsize=(20,10))
plt.title(' Least common Condition suffered', fontsize=20)
plt.show()
```

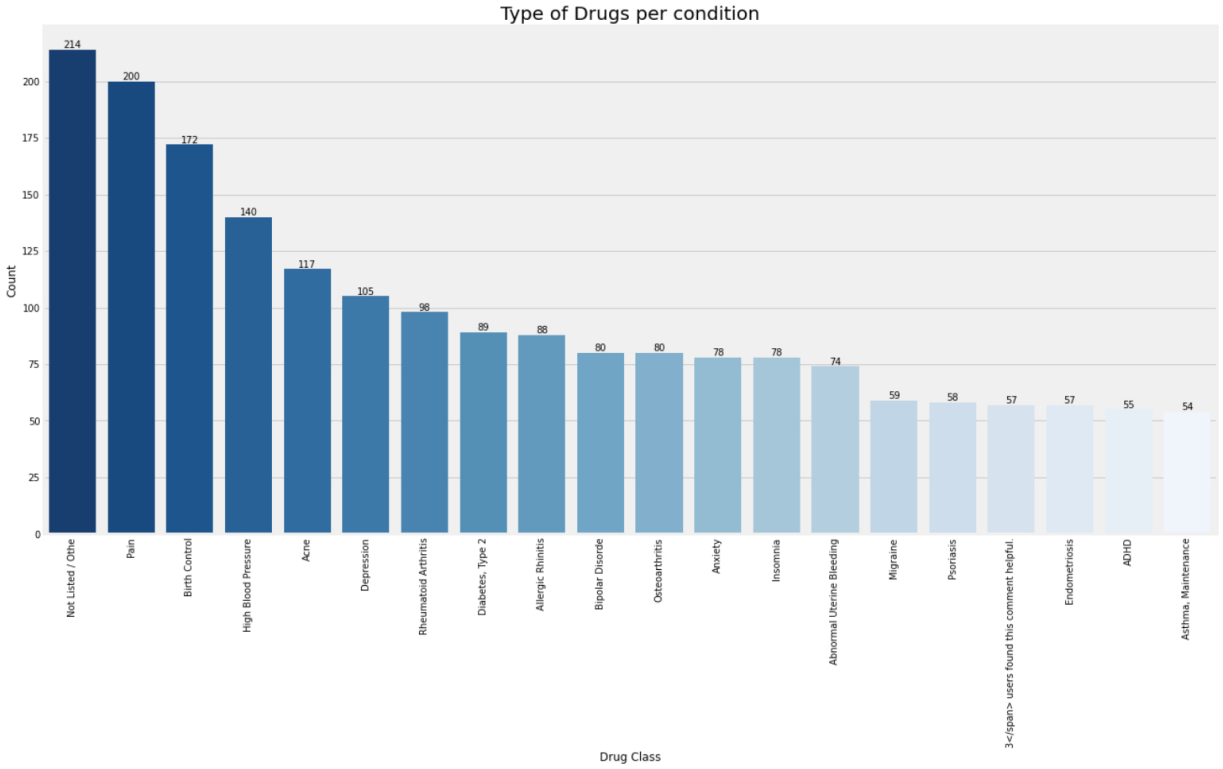


-- How many type of Drugs per condition (Top 20)

In [37]:
```
drug_per_cond = df.groupby('condition')['drugName'].nunique().nlargest(20)
```

In [38]:
```
drug_per_cond_df = pd.DataFrame({'drug_class':drug_per_cond.index,'counts':drug_per_cond.values})
```

In [39]:
```
# How many Drugs per condition (Top 20)
plt.figure(figsize=(20,10))

sns_1 = sns.barplot(data=drug_per_cond_df,x='drug_class',y='counts', palette = 'Blues_r')

for i in sns_1.containers:
    sns_1.bar_label(i,)

sns_1.set_title('Type of Drugs per condition', fontsize=20)
sns_1.set_xlabel("Drug Class")
sns_1.set_ylabel("Count")
plt.setp(sns_1.get_xticklabels(), rotation=90)
plt.show()
```

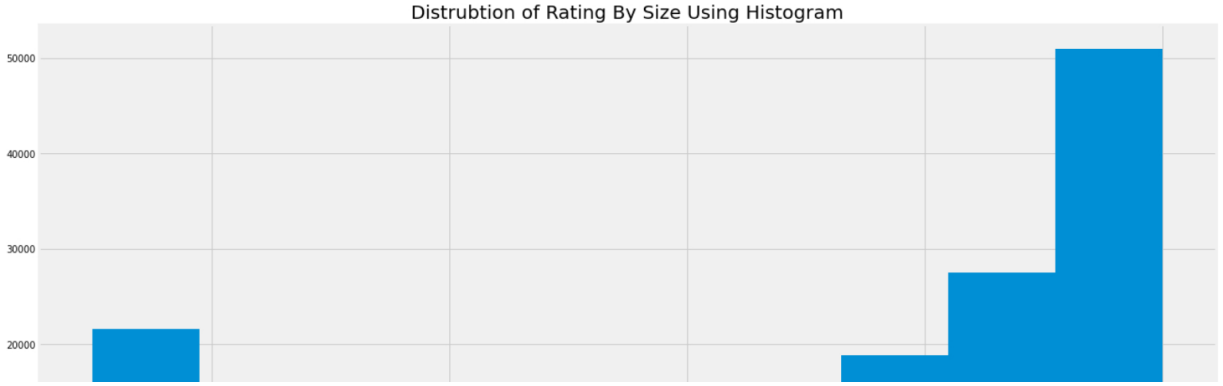


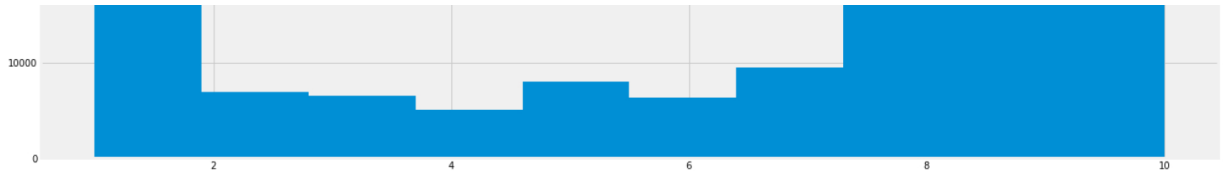
Observation :

- **Pain, Birth Control** and **HBP** have the highest number of different/unique drugs for their condition

-- Distrubtion of Rating given by Users

In [40]:
```
plt.figure(figsize=(20,10))
df['rating'].hist()
plt.title("Distrubtion of Rating By Size Using Histogram", fontsize=20)
plt.show()
```





Observation :

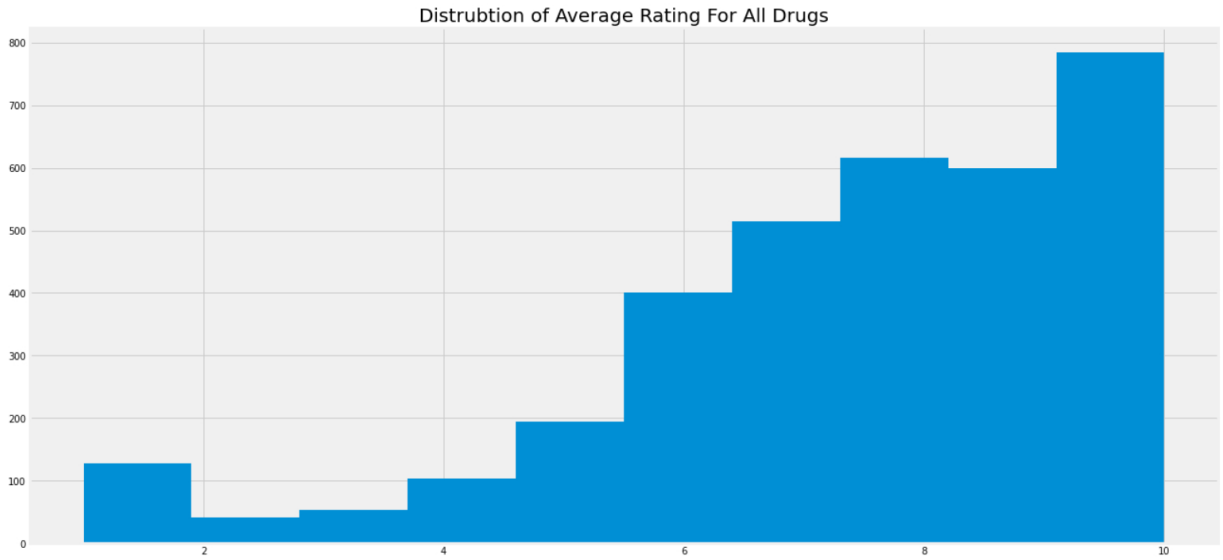
- Most people rated at the **extremes i.e (0 and 10)**

-- Average Rating of Drugs

In [41]:
```
# Average Rating of Drugs
avg_rating = (df['rating'].groupby(df['drugName']).mean())
avg_rating
```

Out[41]:
```
drugName
A + D Cracked Skin Relief              10.000000
A / B Otic                             10.000000
Abacavir / dolutegravir / lamivudine    8.211538
Abacavir / lamivudine / zidovudine      9.000000
Abatacept                               7.157895
                                          ...
Zyvox                                   9.000000
ZzzQuil                                 2.500000
depo-subQ provera 104                   1.000000
ella                                    6.980392
femhrt                                  4.000000
Name: rating, Length: 3436, dtype: float64
```

In [42]:
```
# Average Rating For All Drugs
plt.figure(figsize=(20,10))
avg_rating.hist()
plt.title("Distrubtion of Average Rating For All Drugs", fontsize=20)
plt.show()
```



Observation :

The **Majority of Rating** given by the users are **from 6 to 10**

Insights on Date Column :

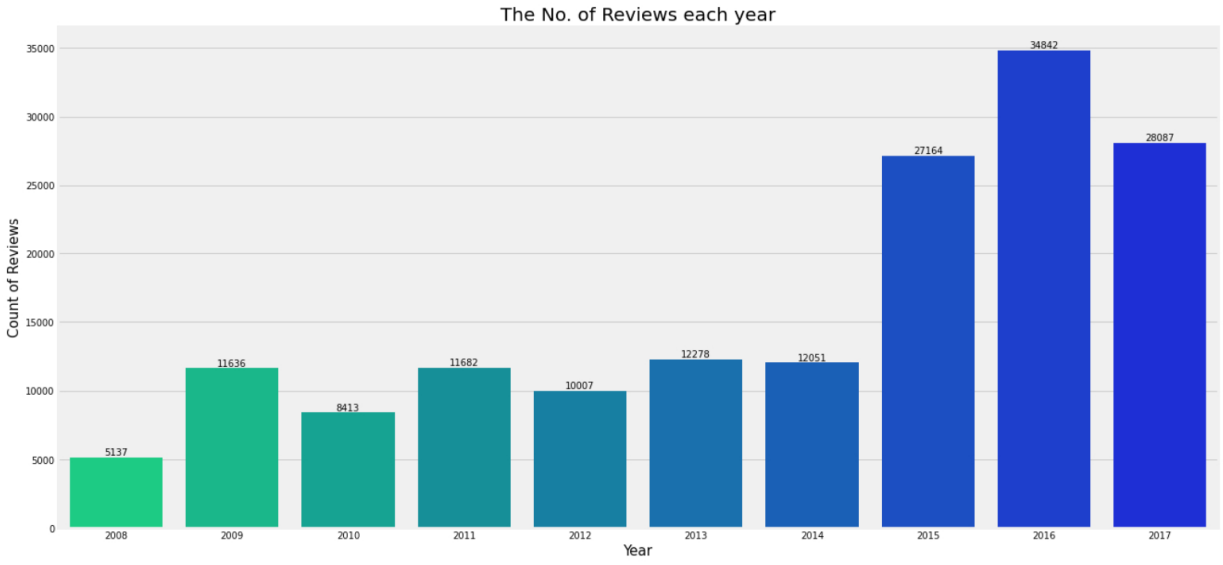
Transforming date columns

In [43]:
```
# converting the date into datetime format
df['date'] = pd.to_datetime(df['date'], errors = 'coerce')

# extracting year from date
df['Year'] = df['date'].dt.year

# extracting the month from the date
df['month'] = df['date'].dt.month

# extracting the days from the date
df['day'] = df['date'].dt.day
```

-- Number of Reviews each Year

In [44]:
```
plt.figure(figsize=(20,10))
sns_=sns.countplot(df['Year'], palette ='winter_r')

for i in sns_.containers:
    sns_.bar_label(i,)

plt.title('The No. of Reviews each year', fontsize = 20)
plt.xlabel('Year', fontsize = 15)
plt.ylabel('Count of Reviews', fontsize = 15)
plt.show()
```

```
/opt/conda/lib/python3.7/site-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg:
x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keywor
d will result in an error or misinterpretation.
  FutureWarning
```



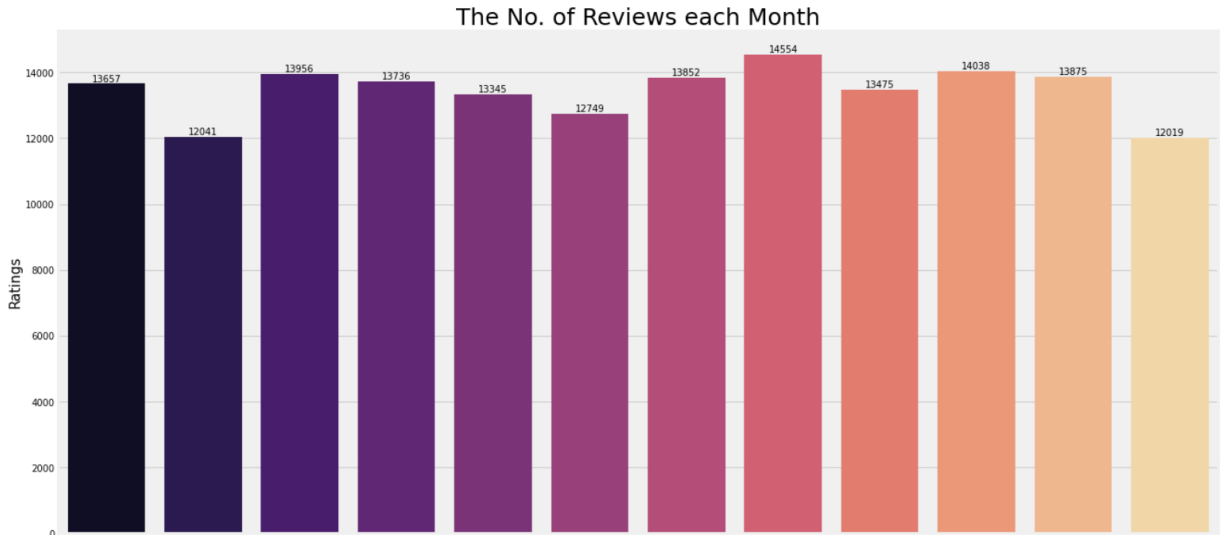
Observations :

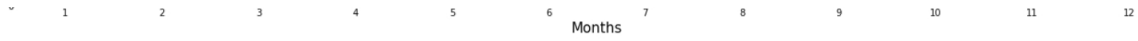
From above plot we can see that **Year 2016 had the heighest number of reviews** followed by **Year 2017 and 2015**

-- Number of Reviews each Month

In [45]:
```
plt.figure(figsize=(20,10))
sns_=sns.countplot(df['month'], palette ='magma')

for i in sns_.containers:
    sns_.bar_label(i,)

plt.title('The No. of Reviews each Month', fontsize = 25)
plt.xlabel('Months', fontsize = 15)
plt.ylabel('Ratings', fontsize = 15)
plt.show()
```

```
/opt/conda/lib/python3.7/site-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg:
x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keywor
d will result in an error or misinterpretation.
  FutureWarning
```



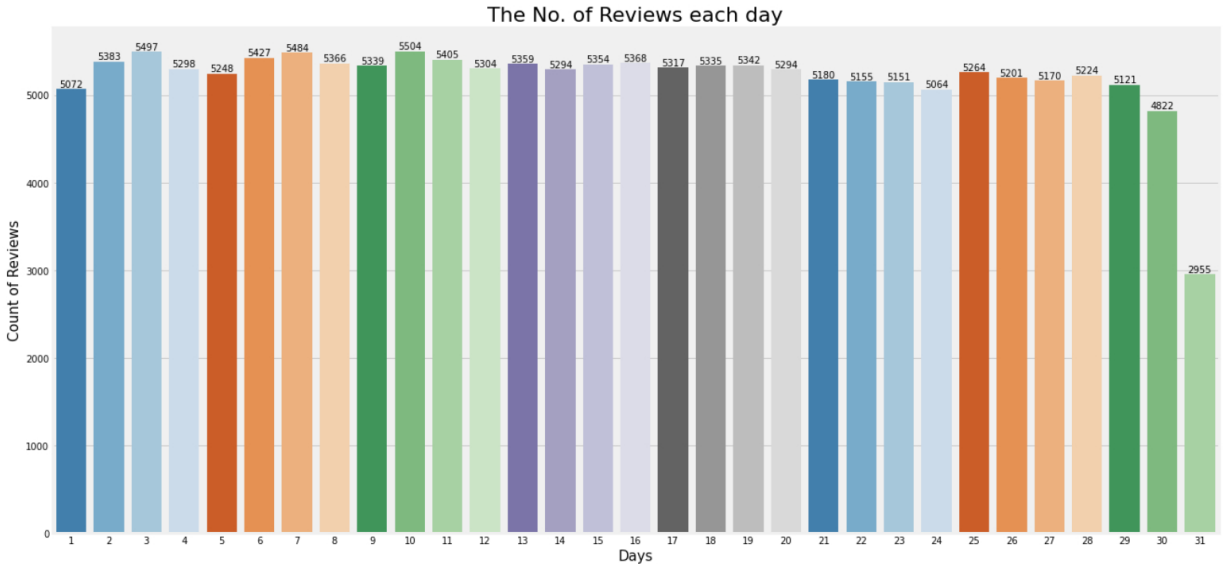


Observations :

August month has the heighest reviwes **followed by October** month.

In [46]:
```
plt.figure(figsize=(20,10))
sns_=sns.countplot(df['day'], palette ='tab20c')

for i in sns_.containers:
    sns_.bar_label(i)

plt.title('The No. of Reviews each day', fontsize = 22)
plt.xlabel('Days', fontsize = 15)
plt.ylabel('Count of Reviews', fontsize = 15)
plt.show()
```

```
/opt/conda/lib/python3.7/site-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  FutureWarning
```



-- How genuine is the review? (Using sentiment analysis)

In [47]:
```
from textblob import TextBlob
```

In [48]:
```
df['review'].head()
```

Out[48]:
```
0    "It has no side effect, I take it in combinati...
1    "My son is halfway through his fourth week of ...
2    "I used to take another oral contraceptive, wh...
3    "This is my first time using any form of birth...
4    "Suboxone has completely turned my life around...
Name: review, dtype: object
```

In [49]:
```
def get_sentiment(text):
    blob = TextBlob(text)
    return blob.polarity

def get_sentiment_label(text):
    blob = TextBlob(text)
    if blob.polarity > 0:
        result = 'positive'
    elif blob.polarity < 0:
        result = 'negative'
    else:
        result = 'neutral'
    return result
```

In [50]:
```
# text fxn
get_sentiment("I love apples")
```

Out[50]:
```
0.5
```

In [51]:
```
# text fxn
get_sentiment_label("I love apples")
```

Out[51]:
```
'positive'
```

In [52]:
```
# Sentiment Score for Review
df['sentiment'] = df['review'].apply(get_sentiment)
```

In [53]:
```
# Sentiment Labels for Review
df['sentiment_label'] = df['review'].apply(get_sentiment_label)
```

In [54]:
```
df[['review','sentiment','sentiment_label']].head()
```

Out[54]:

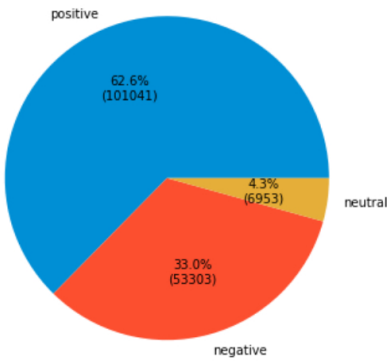
	review	sentiment	sentiment_label
0	"It has no side effect, I take it in combinati...	0.000000	neutral
1	"My son is halfway through his fourth week of ...	0.168333	positive
2	"I used to take another oral contraceptive, wh...	0.067210	positive
3	"This is my first time using any form of birth...	0.179545	positive
4	"Suboxone has completely turned my life around...	0.194444	positive

-- How many positive and negative and neutral reviews?

In [55]:
```
df['sentiment_label'].value_counts()
```

Out[55]:
```
positive    101041
negative     53303
neutral       6953
Name: sentiment_label, dtype: int64
```

In [56]:
```
plt.figure(figsize=(12,6))

def autopct_format(values):
        def my_format(pct):
            total = sum(values)
            val = int(round(pct*total/100.0))
            return '{:.1f}%\n({v:d})'.format(pct, v=val)
        return my_format

s = df['sentiment_label'].value_counts()
plt.pie(s,labels = s.index, autopct=autopct_format(s))
plt.show()
```

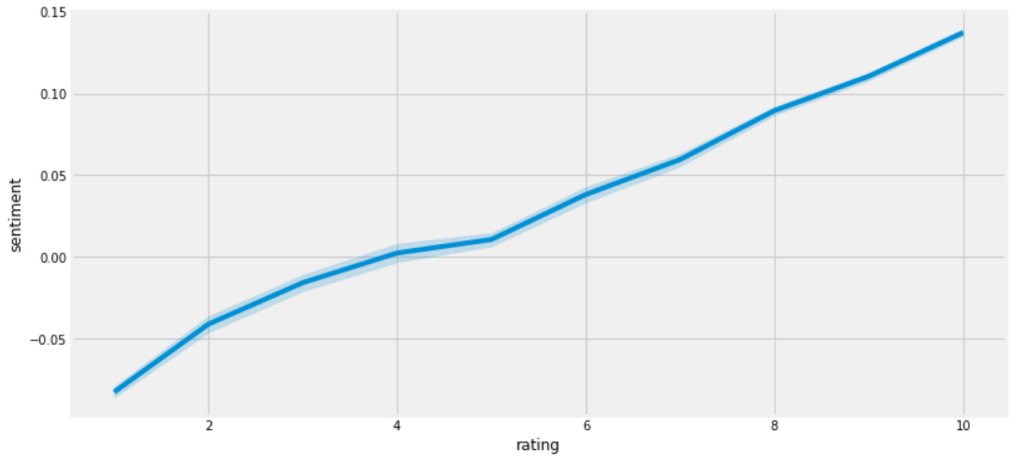


Obserevation :

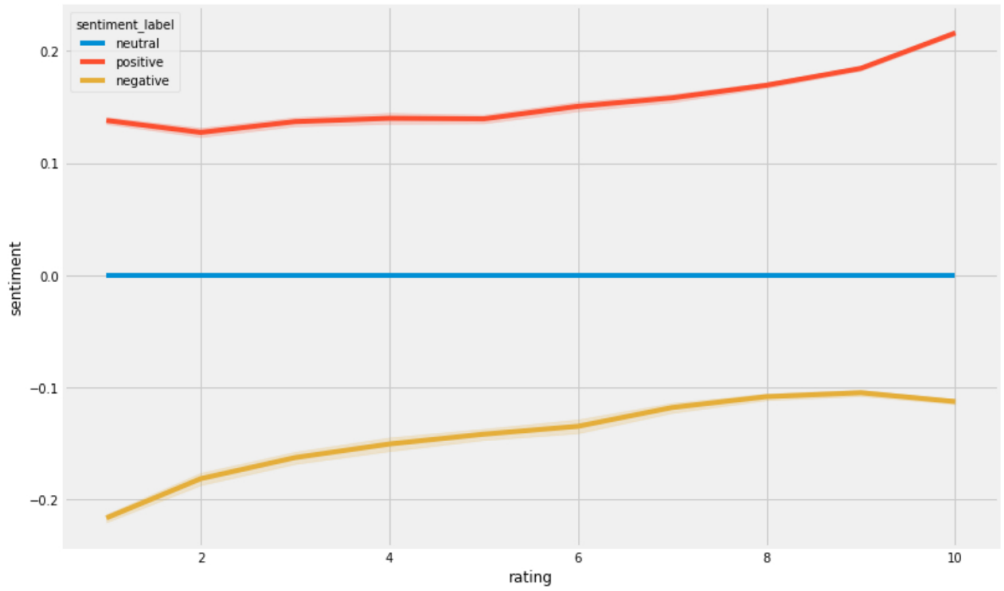
We can see that Majority of reviews given by users are **Positive(62.6%)** and **Negative(33%)**, while **Neutral reviews are (4.3%)**

-- Correlation Between Our sentiment and rating

In [57]:
```
#### Correlation Between Our sentiment and rating
plt.figure(figsize=(12,6))
sns.lineplot(data=df,x='rating',y='sentiment')
plt.show()
```



In [58]:
```
plt.figure(figsize=(12,8))
sns.lineplot(data=df,x='rating',y='sentiment',hue='sentiment_label')
plt.show()
```



Observation :

- The positive Ratings are increasing gradually which indicates that the patients are getting better by the use of the prescribed Drugs
- The Negative Ratings are decreasing which is a good sign that the Drugs given to patients are effective

-- How many reviews are genuine as compared to the rating

- Genuine **good rating** =positive + rating 10-6
- Genuine **bad rating** = negative + rating 4-1

In [59]:
```
# Genuine Good  Rating Per Review
good_review =  len(df[(df['rating'] >= 6) & (df['sentiment_label'] == 'positive')])
print('According to Sentiment and Rating the number of Good Ratings are -', good_review)
```

```
According to Sentiment and Rating the number of Good Ratings are - 81044
```

In [60]:
```
# Genuine Bad  Rating Per Review
bad_review = len(df[(df['rating'] <= 4) & (df['sentiment_label'] == 'negative')])
print('According to Sentiment and Rating the number of Good Ratings are -', bad_review)
```

```
According to Sentiment and Rating the number of Good Ratings are - 21995
```

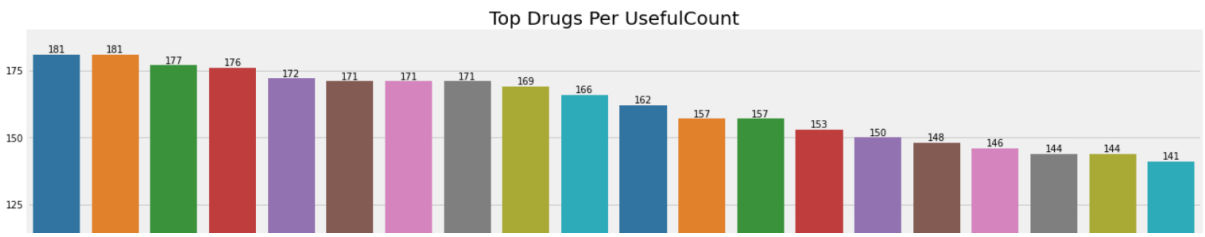
In [61]:
```
# df.groupby('drugName')['usefulCount'].value_counts()
```

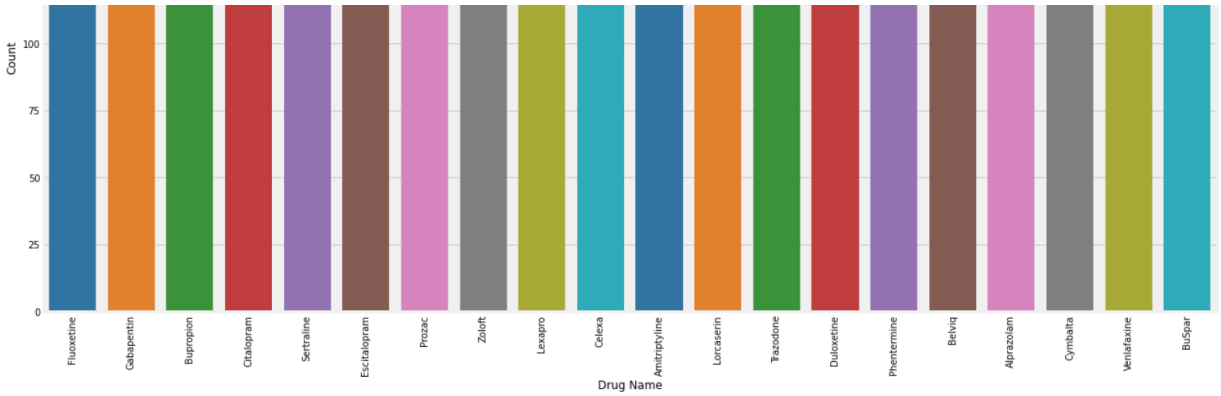
-- Top Drugs name Per UsefulCount

In [62]:
```
drug_usefulcount = df.groupby('drugName')['usefulCount'].nunique().nlargest(20)
```

In [63]:
```
drug_usefulcount_df = pd.DataFrame({'drug_name':drug_usefulcount.index,'counts':drug_usefulcount.values})
```

In [64]:
```
plt.figure(figsize=(20,10))

sns_1 = sns.barplot(data=drug_usefulcount_df,x='drug_name',y='counts', palette = 'tab10')

for i in sns_1.containers:
    sns_1.bar_label(i,)

sns_1.set_title('Top Drugs Per UsefulCount', fontsize=20)
sns_1.set_xlabel("Drug Name")
sns_1.set_ylabel("Count")
plt.setp(sns_1.get_xticklabels(), rotation=90)
plt.show()
```





Observation :

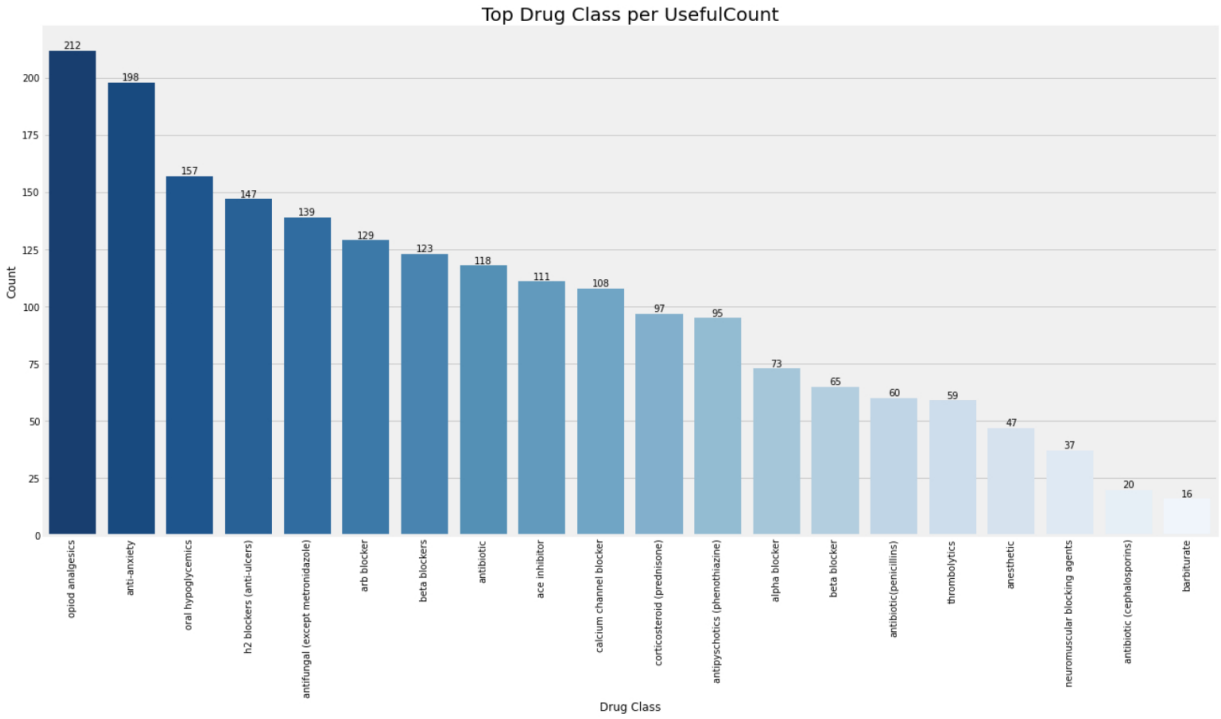
Fluoxetine, Gabapentin and **Bupropion** are the Drugs which are most useful for treating people

-- Top Drug Class per UsefulCount

In [65]:
```
# Top Drugs Class Per UsefulCount
drug_class_usefulcount = df.groupby('drug_class')['usefulCount'].nunique().nlargest(20)
```

In [66]:
```
drug_class_usefulcount_df = pd.DataFrame({'drug_class':drug_class_usefulcount.index,'counts':drug_class_usefulcount.values})
```

In [67]:
```
plt.figure(figsize=(20,10))

sns_1 = sns.barplot(data=drug_class_usefulcount_df,x='drug_class',y='counts', palette = 'Blues_r')

for i in sns_1.containers:
    sns_1.bar_label(i,)

sns_1.set_title('Top Drug Class per UsefulCount', fontsize=20)
sns_1.set_xlabel("Drug Class")
sns_1.set_ylabel("Count")
plt.setp(sns_1.get_xticklabels(), rotation=90)
plt.show()
```



Observations :

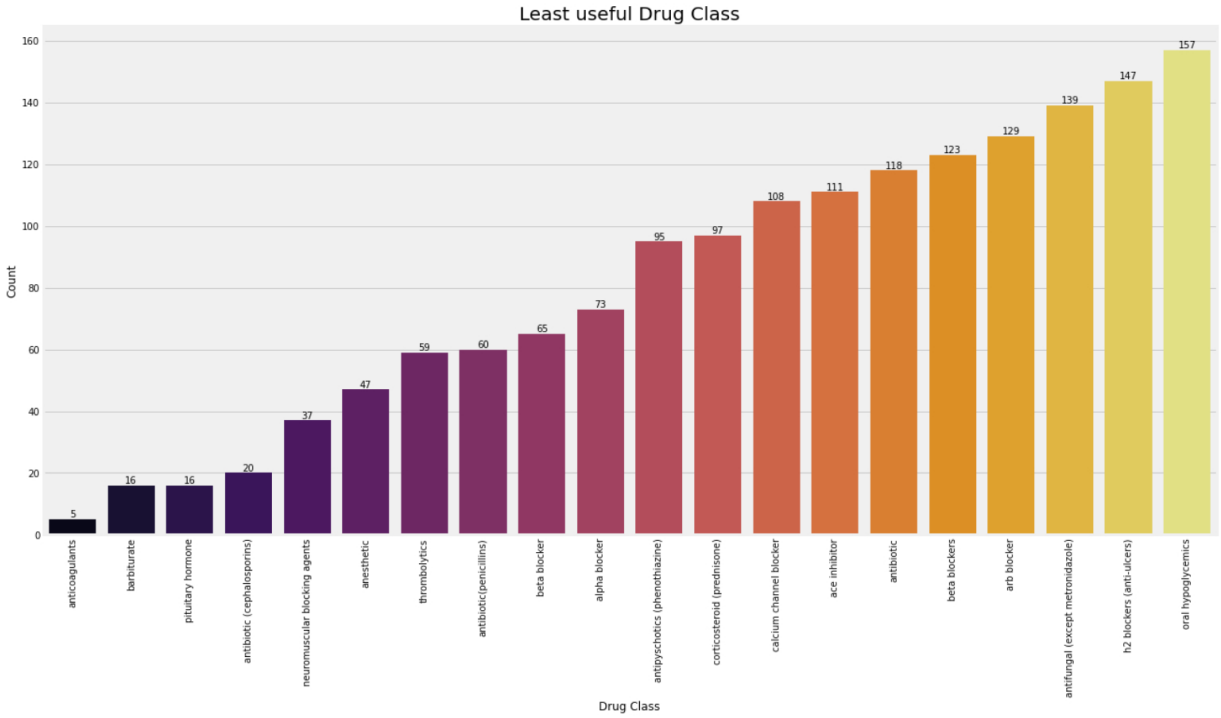
Opiod analgesics, anti-anxiety and **oral hypoglycemics** are the top 3 Drug Classes used to treat the people

-- Least Useful Drug Class

In [68]:
```
# Top Drugs Class Per UsefulCount
least_drug_class_usefulcount = df.groupby('drug_class')['usefulCount'].nunique().nsmallest(20)
```

In [69]:
```
least_drug_class_usefulcount_df = pd.DataFrame({'drug_class':least_drug_class_usefulcount.index,'counts':least_drug_class_usefulc
```

In [70]:
```
plt.figure(figsize=(20,10))

sns_1 = sns.barplot(data=least_drug_class_usefulcount_df,x='drug_class',y='counts', palette = 'inferno')

for i in sns_1.containers:
    sns_1.bar_label(i,)

sns_1.set_title('Least useful Drug Class', fontsize=20)
sns_1.set_xlabel("Drug Class")
sns_1.set_ylabel("Count")
plt.setp(sns_1.get_xticklabels(), rotation=90)
plt.show()
```

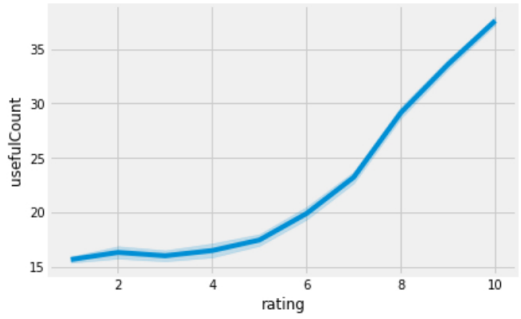


Observation :

Anticoagulant, barbiturate and **pitutary harmone** is the Least used Drug Class for perscribed for peoples

In [71]:
```
### Correlation between Rating and Usefulcount
sns.lineplot(data=df,x='rating',y='usefulCount')
```

Out[71]:
```
<AxesSubplot:xlabel='rating', ylabel='usefulCount'>
```



Observation :

- As the rating goes up the usefulcount goes up