```python
#Importing the libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

```python
import sklearn
```

```python
df=pd.read_csv(r'C:\Users\Hiranmai11\OneDrive\Desktop\Mini project data.csv')
```

```python
df.head()
```

	Condition	Drug	Indication	Type	Reviews	Effective	EaseOfUse	Satisfaction	Unnamed: 8
0	Acute Bacterial Sinusitis	Levofloxacin	On Label	RX	994 Reviews	2.52	3.01	1.84	\r\n\t\t\t\t\t\tLevofloxacin is used to treat a ...
1	Acute Bacterial Sinusitis	Levofloxacin	On Label	RX	994 Reviews	2.52	3.01	1.84	\r\n\t\t\t\t\t\tLevofloxacin is used to treat a ...
2	Acute Bacterial Sinusitis	Moxifloxacin	On Label	RX	755 Reviews	2.78	3.00	2.08	\r\n\t\t\t\t\t\t This is a generic drug. The ave...
3	Acute Bacterial Sinusitis	Azithromycin	On Label	RX	584 Reviews	3.21	4.01	2.57	\r\n\t\t\t\t\t\tAzithromycin is an antibiotic (m...
4	Acute Bacterial Sinusitis	Azithromycin	On Label	RX	584 Reviews	3.21	4.01	2.57	\r\n\t\t\t\t\t\tAzithromycin is an antibiotic (m...

```python
df.drop(columns = ['Unnamed: 8'], axis = 1, inplace = True)
```

```python
df.head()
```

	Condition	Drug	Indication	Type	Reviews	Effective	EaseOfUse	Satisfaction
0	Acute Bacterial Sinusitis	Levofloxacin	On Label	RX	994 Reviews	2.52	3.01	1.84
1	Acute Bacterial Sinusitis	Levofloxacin	On Label	RX	994 Reviews	2.52	3.01	1.84
2	Acute Bacterial Sinusitis	Moxifloxacin	On Label	RX	755 Reviews	2.78	3.00	2.08
3	Acute Bacterial Sinusitis	Azithromycin	On Label	RX	584 Reviews	3.21	4.01	2.57
4	Acute Bacterial Sinusitis	Azithromycin	On Label	RX	584 Reviews	3.21	4.01	2.57

```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2219 entries, 0 to 2218
Data columns (total 8 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Condition     2219 non-null   object
 1   Drug          2219 non-null   object
 2   Indication    2219 non-null   object
 3   Type          2219 non-null   object
 4   Reviews       2219 non-null   object
 5   Effective     2219 non-null   float64
 6   EaseOfUse     2219 non-null   float64
 7   Satisfaction  2219 non-null   float64
dtypes: float64(3), object(5)
memory usage: 138.8+ KB
```

```python
# checking the shape of the dataset
df.shape
```

```
(2219, 8)
```

```python
# Drop all duplicates:
df = df.drop_duplicates()
```

```python
# Rounding all numerical columns to 2 decimal places:
df.round(2).head(5)
```

	Condition	Drug	Indication	Type	Reviews	Effective	EaseOfUse	Satisfaction
0	Acute Bacterial Sinusitis	Levofloxacin	On Label	RX	994 Reviews	2.52	3.01	1.84
2	Acute Bacterial Sinusitis	Moxifloxacin	On Label	RX	755 Reviews	2.78	3.00	2.08
3	Acute Bacterial Sinusitis	Azithromycin	On Label	RX	584 Reviews	3.21	4.01	2.57
8	Acute Bacterial Sinusitis	Amoxicillin-Pot Clavulanate	On Label	RX	437 Reviews	3.26	3.23	2.42
11	Acute Bacterial Sinusitis	Levofloxacin	On Label	RX	361 Reviews	2.44	2.96	1.68

```python
df['Type'].value_counts()
```

```
RX         1165
OTC         546
RX/OTC       34
\r\n          5
Name: Type, dtype: int64
```

```python
df_type = df[(df['Type'] == '\r\n' )]
df_type
```

	Condition	Drug	Indication	Type	Reviews	Effective	EaseOfUse	Satisfaction
203	Atopic Dermatitis	Vit E-Glycerin-Dimethicone, Glycerin-Dimethico...	On Label	\r\n	3 Reviews	2.00	3.00	2.50
285	Atopic Dermatitis	Vit E-Glycerin-Dimethicone	On Label	\r\n	1 Reviews	4.00	5.00	5.00
1273	flatulence	Simethicone, Alpha-D-Galactosidase	On Label	\r\n	1 Reviews	3.00	5.00	3.00
1630	hypercholesterolemia	Niacin	On Label	\r\n	42 Reviews	3.19	3.56	3.13
1644	hypercholesterolemia	Niacin	On Label	\r\n	13 Reviews	4.00	4.43	4.29

```python
# droping the '\r\n' rows since its name is properly not defined
df = df.drop(df[df['Type'] == '\r\n' ].index)
```

```python
df['Type'].value_counts()
```

```
RX         1165
OTC         546
RX/OTC       34
Name: Type, dtype: int64
```

```python
df['Condition'].value_counts()
```

```
fever                                          225
hypertension                                   215
Atopic Dermatitis                              170
endometriosis                                  149
Bacterial Urinary Tract Infection              104
back pain                                       92
gastroesophageal reflux disease                 92
gout                                            82
vertigo                                         60
hypercholesterolemia                            54
Pharyngitis due to Streptococcus Pyogenes       51
flatulence                                      44
hemorrhoids                                     42
edema                                           39
Acute Bacterial Sinusitis                       37
diverticulitis of gastrointestinal tract        36
Bacterial Conjunctivitis                        34
prevention of cerebrovascular accident          30
depression                                      28
vulvovaginal candidiasis                        27
fibromyalgia                                    22
sore throat                                     19
adenocarcinoma of pancreas                      14
oral candidiasis                                 9
genital herpes simplex                           9
impetigo                                         7
Infantile Autism                                 7
pyelonephritis                                   7
meniere's disease                                6
herpes zoster                                    6
scabies                                          6
furunculosis                                     6
chickenpox                                       4
Sleepiness Due To Obstructive Sleep Apnea        4
biliary calculus                                 4
Influenza                                        3
colorectal cancer                                1
Name: Condition, dtype: int64
```

```python
df['Drug'].value_counts()
```

```
Acetaminophen                                   79
Ibuprofen                                       43
Diphenhydramine Hcl                             40
Sulfamethoxazole-Trimethoprim                   30
Norethindrone-Ethin Estradiol                   30
                                                ..
Activated Charcoal-Simethicone, Simethicone      1
Losartan                                         1
Cefuroxime Sodium                                1
Acetaminophen-Caffeine                           1
Guanfacine                                       1
Name: Drug, Length: 467, dtype: int64
```

```python
df['Drug'].value_counts().min()
```

```
1
```

```python
df['Drug'].value_counts().max()
```

```
79
```

```python
#Ensuring there's no negative value that states effectiveness
df['Effective'].value_counts()[df['Effective'].value_counts()<0].index
```

```
Float64Index([], dtype='float64')
```

```python
df['Indication'].value_counts()
```

```
On Label     1331
Off Label     384
\r\n           30
Name: Indication, dtype: int64
```

```python
df = df.drop(df[df['Indication'] == '\r\n' ].index)
```

```python
df['Indication'].value_counts()
```

```
On Label     1331
Off Label     384
Name: Indication, dtype: int64
```

```python
df['Reviews'].value_counts()
```

```
1 Reviews      399
2 Reviews      193
3 Reviews      107
4 Reviews       84
5 Reviews       65
              ...
42 Reviews       1
51 Reviews       1
62 Reviews       1
80 Reviews       1
123 Reviews      1
Name: Reviews, Length: 241, dtype: int64
```

```python
df['Effective'].value_counts()
```

```
5.00    286
4.00    198
3.00    168
1.00    138
2.00     68
       ...
4.01      1
4.78      1
2.89      1
3.66      1
1.75      1
Name: Effective, Length: 207, dtype: int64
```

```python
def categorize_effectiveness(effective):
    if effective < 2.0 : return 'Ineffective'
    elif effective < 3.0 : return 'Partly Effective'

    elif effective <= 5.0 : return 'Effective'
```

```python
df['Effective_Cat'] = df['Effective'].apply(categorize_effectiveness)
df['Effective_Cat'].value_counts()
```

```
Effective           1352
Partly Effective     208
Ineffective          155
Name: Effective_Cat, dtype: int64
```

```python
df[['Effective_Cat', 'Type']].groupby(['Effective_Cat','Type'])['Type'].count()
```

```
Effective_Cat     Type
Effective         OTC       403
                  RX        925
                  RX/OTC     24
Ineffective       OTC        68
                  RX         85
                  RX/OTC      2
Partly Effective  OTC        58
                  RX        144
                  RX/OTC      6
Name: Type, dtype: int64
```

```python
df[['Effective_Cat', 'Indication']].groupby(['Effective_Cat','Indication'])['Effective_Cat'].count()
```

```
Effective_Cat     Indication
Effective         Off Label      325
                  On Label      1027
Ineffective       Off Label       30
                  On Label       125
Partly Effective  Off Label       29
                  On Label       179
Name: Effective_Cat, dtype: int64
```

```python
df['EaseOfUse'].value_counts()
```

```
5.00    404
4.00    210
3.00    117
1.00     97
4.50     58
       ...
4.02      1
4.54      1
3.12      1
3.99      1
3.16      1
Name: EaseOfUse, Length: 184, dtype: int64
```

```python
def categorize_ease_of_use(easeOfUse):
    if easeOfUse < 2.0 : return 'Difficult'
    elif easeOfUse < 3.0 : return 'Normal'
    elif easeOfUse <= 5.0 : return 'Easy'
```

```python
df['EaseOfUse_Cat'] = df['EaseOfUse'].apply(categorize_ease_of_use)
df['EaseOfUse_Cat'].value_counts()
```

```
Easy         1508
Normal        104
Difficult     103
Name: EaseOfUse_Cat, dtype: int64
```

```python
df[['EaseOfUse_Cat', 'Type']].groupby(['EaseOfUse_Cat','Type'])['Type'].count()
```

```
EaseOfUse_Cat  Type
Difficult      OTC         40
               RX          61
               RX/OTC       2
Easy           OTC        468
               RX        1010
               RX/OTC      30
Normal         OTC         21
               RX          83
Name: Type, dtype: int64
```

```python
df['Satisfaction'].value_counts()
```

```
5.00    259
1.00    197
3.00    164
4.00    162
2.00     86
       ...
4.38      1
2.58      1
2.84      1
3.53      1
2.47      1
Name: Satisfaction, Length: 223, dtype: int64
```

```python
def categorize_satisfaction(satisfaction):
    if satisfaction < 2.0 : return 'Unsatisfied'
    elif satisfaction < 3.0 : return 'Partly Satisfied'
    elif satisfaction <= 5.0 : return 'Very Satisfied'
```

```python
df['Satisfaction_Cat'] = df['Satisfaction'].apply(categorize_satisfaction)
df['Satisfaction_Cat'].value_counts()
```

```
Very Satisfied      1047
Partly Satisfied     407
Unsatisfied          261
Name: Satisfaction_Cat, dtype: int64
```

```python
df[['Satisfaction_Cat', 'Type']].groupby(['Satisfaction_Cat','Type'])['Type'].count()
```

```
Satisfaction_Cat  Type
Partly Satisfied  OTC        66
                  RX        331
                  RX/OTC     10
Unsatisfied       OTC        95
                  RX        164
                  RX/OTC      2
Very Satisfied    OTC       368
                  RX        659
                  RX/OTC     20
Name: Type, dtype: int64
```

```python
df
```

	Condition	Drug	Indication	Type	Reviews	Effective	EaseOfUse	Satisfaction	Effective_Cat	EaseOfUse_Cat	Satisfaction_Cat
0	Acute Bacterial Sinusitis	Levofloxacin	On Label	RX	994 Reviews	2.52	3.01	1.84	Partly Effective	Easy	Unsatisfied
2	Acute Bacterial Sinusitis	Moxifloxacin	On Label	RX	755 Reviews	2.78	3.00	2.08	Partly Effective	Easy	Partly Satisfied
3	Acute Bacterial Sinusitis	Azithromycin	On Label	RX	584 Reviews	3.21	4.01	2.57	Effective	Easy	Partly Satisfied
8	Acute Bacterial Sinusitis	Amoxicillin-Pot Clavulanate	On Label	RX	437 Reviews	3.26	3.23	2.42	Effective	Easy	Partly Satisfied
11	Acute Bacterial Sinusitis	Levofloxacin	On Label	RX	361 Reviews	2.44	2.96	1.68	Partly Effective	Normal	Unsatisfied
...	...	...	...	...	...	...	...	...	...	...	...
2214	vulvovaginal candidiasis	Clotrimazole	On Label	OTC	2 Reviews	5.00	5.00	5.00	Effective	Easy	Very Satisfied
2215	vulvovaginal candidiasis	Butoconazole Nitrate	On Label	RX	1 Reviews	5.00	5.00	5.00	Effective	Easy	Very Satisfied
2216	vulvovaginal candidiasis	Clotrimazole	On Label	OTC	1 Reviews	5.00	4.00	5.00	Effective	Easy	Very Satisfied
2217	vulvovaginal candidiasis	Butoconazole Nitrate	On Label	OTC	1 Reviews	5.00	5.00	5.00	Effective	Easy	Very Satisfied
2218	vulvovaginal candidiasis	Miconazole-Skin Clnsr17	On Label	OTC	1 Reviews	1.00	4.00	1.00	Ineffective	Easy	Unsatisfied

1715 rows × 11 columns

```python
#Checking missing values:
df.isna().sum()
```

```
Condition           0
Drug                0
Indication          0
Type                0
Reviews             0
Effective           0
EaseOfUse           0
Satisfaction        0
Effective_Cat       0
EaseOfUse_Cat       0
Satisfaction_Cat    0
dtype: int64
```

```python
df.describe()
```

	Effective	EaseOfUse	Satisfaction
count	1715.000000	1715.000000	1715.000000
mean	3.534566	3.929784	3.167988
std	1.122469	1.047791	1.225030
min	1.000000	1.000000	1.000000
25%	3.000000	3.500000	2.330000
50%	3.690000	4.070000	3.090000
75%	4.330000	4.750000	4.000000
max	5.000000	5.000000	5.000000

```python
df.drop_duplicates(keep='first', inplace=True)
df.reset_index(drop=True)
```

	Condition	Drug	Indication	Type	Reviews	Effective	EaseOfUse	Satisfaction	Effective_Cat	EaseOfUse_Cat	Satisfaction_Cat
0	Acute Bacterial Sinusitis	Levofloxacin	On Label	RX	994 Reviews	2.52	3.01	1.84	Partly Effective	Easy	Unsatisfied
1	Acute Bacterial Sinusitis	Moxifloxacin	On Label	RX	755 Reviews	2.78	3.00	2.08	Partly Effective	Easy	Partly Satisfied
2	Acute Bacterial Sinusitis	Azithromycin	On Label	RX	584 Reviews	3.21	4.01	2.57	Effective	Easy	Partly Satisfied
3	Acute Bacterial Sinusitis	Amoxicillin-Pot Clavulanate	On Label	RX	437 Reviews	3.26	3.23	2.42	Effective	Easy	Partly Satisfied
4	Acute Bacterial Sinusitis	Levofloxacin	On Label	RX	361 Reviews	2.44	2.96	1.68	Partly Effective	Normal	Unsatisfied
...	...	...	...	...	...	...	...	...	...	...	...
1710	vulvovaginal candidiasis	Clotrimazole	On Label	OTC	2 Reviews	5.00	5.00	5.00	Effective	Easy	Very Satisfied
1711	vulvovaginal candidiasis	Butoconazole Nitrate	On Label	RX	1 Reviews	5.00	5.00	5.00	Effective	Easy	Very Satisfied
1712	vulvovaginal candidiasis	Clotrimazole	On Label	OTC	1 Reviews	5.00	4.00	5.00	Effective	Easy	Very Satisfied
1713	vulvovaginal candidiasis	Butoconazole Nitrate	On Label	OTC	1 Reviews	5.00	5.00	5.00	Effective	Easy	Very Satisfied
1714	vulvovaginal candidiasis	Miconazole-Skin Clnsr17	On Label	OTC	1 Reviews	1.00	4.00	1.00	Ineffective	Easy	Unsatisfied

1715 rows × 11 columns

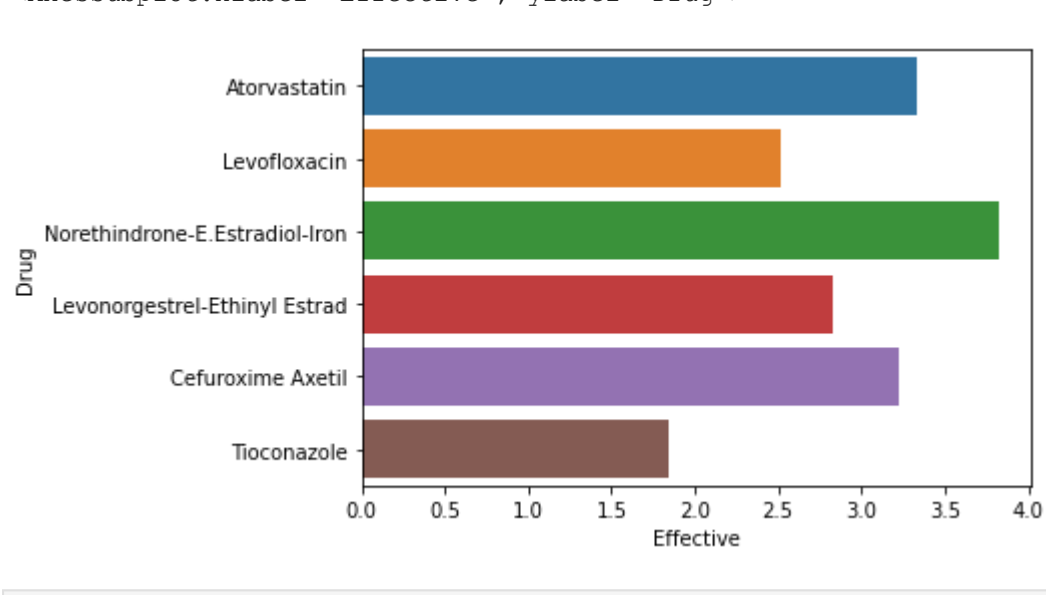
DATA VISUALIZATION

```python
sns.heatmap(df.corr(), annot=True)
```

```
<AxesSubplot:>
```

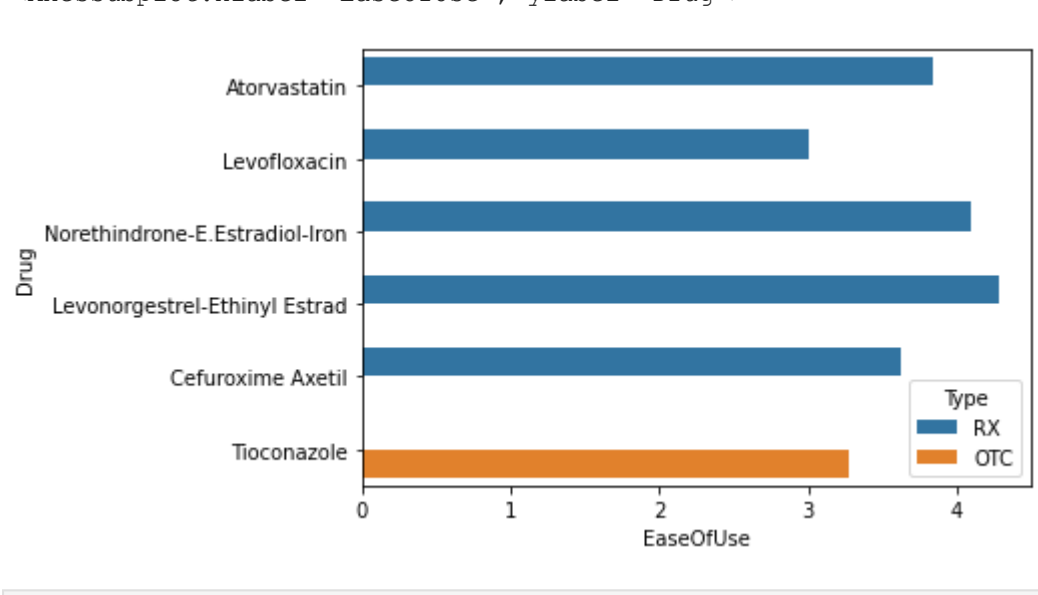

```python
popular_drugs = df.sort_values(by='Reviews', ascending=False).head(10)
sns.barplot(data=popular_drugs, y='Drug', x='Effective')
```

```
<AxesSubplot:xlabel='Effective', ylabel='Drug'>
```



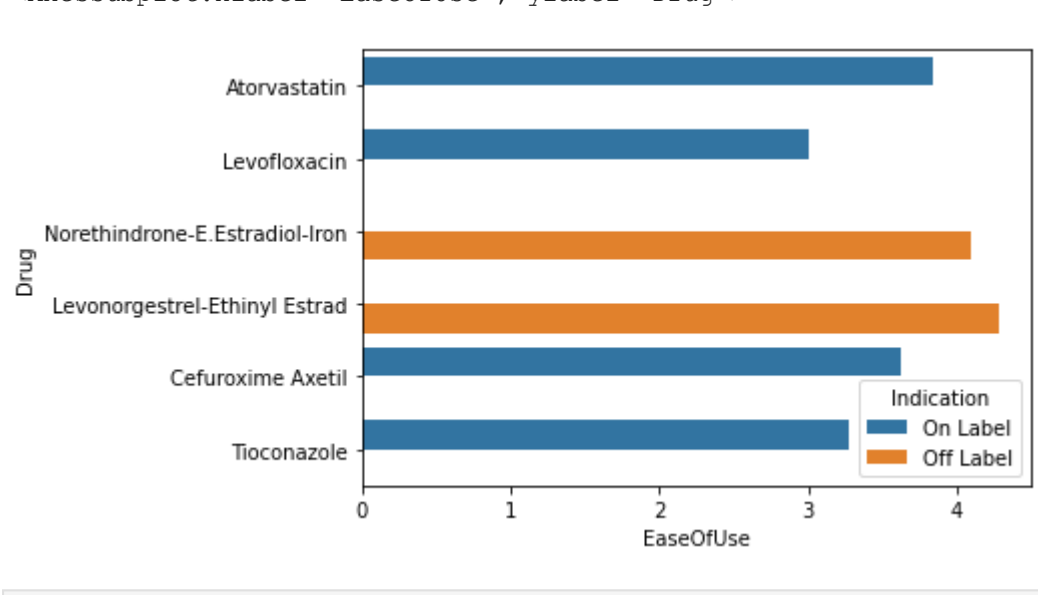

```python
sns.barplot(data=popular_drugs, y='Drug', x='EaseOfUse', hue='Type')
```

```
<AxesSubplot:xlabel='EaseOfUse', ylabel='Drug'>
```

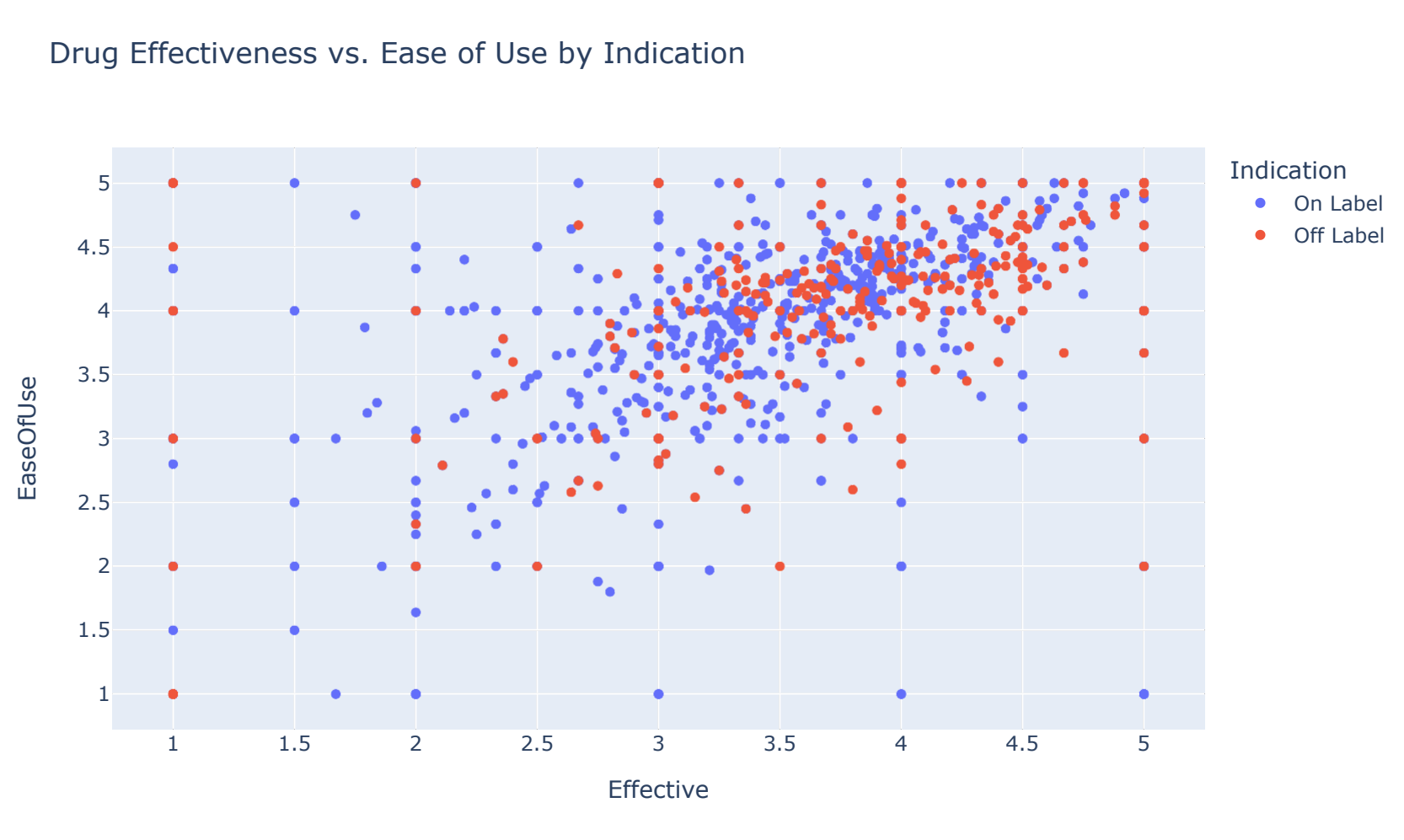



```python
sns.barplot(data=popular_drugs, y='Drug', x='EaseOfUse', hue='Indication')
```

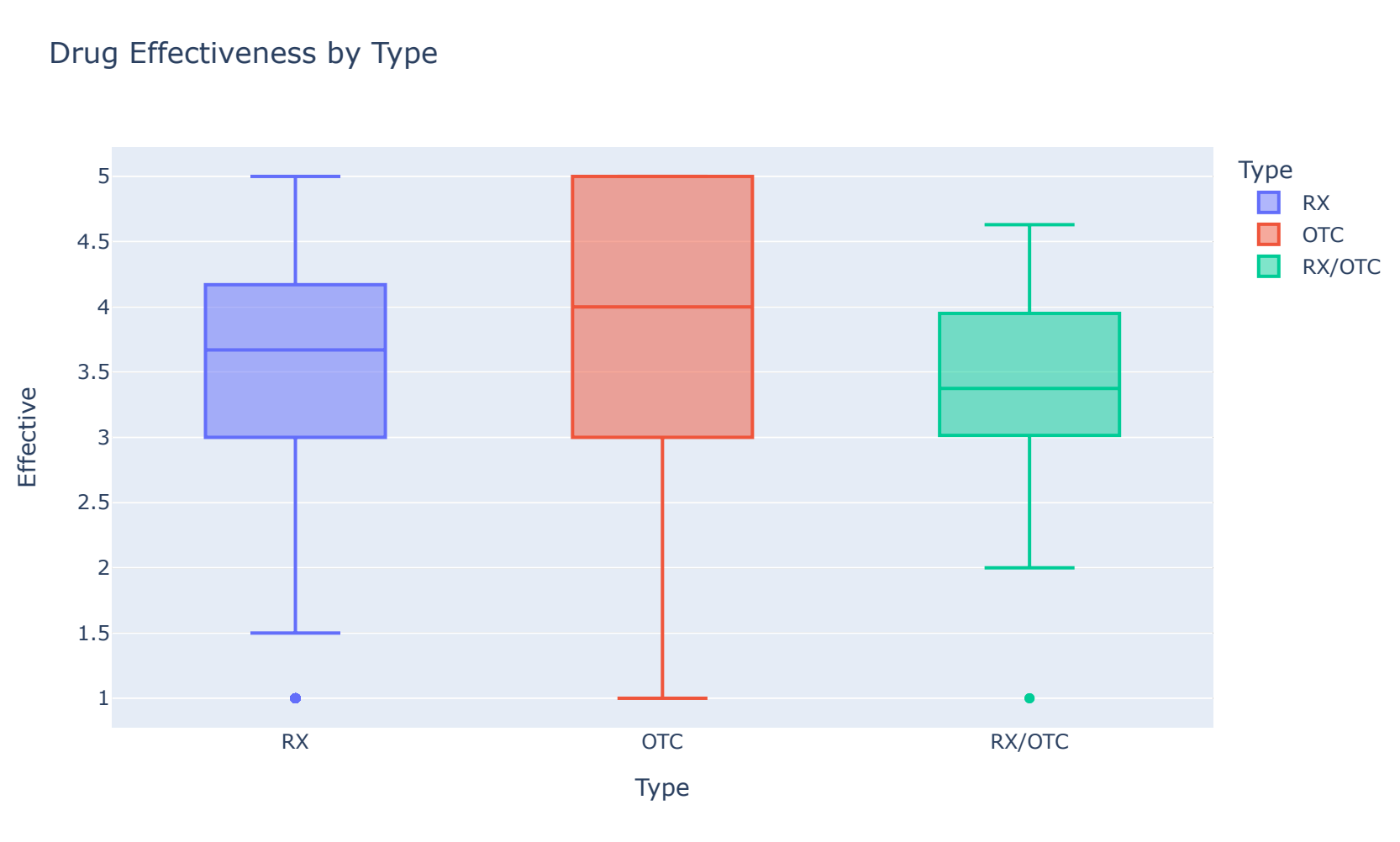
```
<AxesSubplot:xlabel='EaseOfUse', ylabel='Drug'>
```




```python
import plotly.express as px
fig = px.scatter(df, x='Effective', y='EaseOfUse', color='Indication', hover_name='Drug',
                 title='Drug Effectiveness vs. Ease of Use by Indication')
fig.show()
```




```python
fig = px.box(df, x='Type', y='Effective', color='Type', title='Drug Effectiveness by Type')
fig.show()
```



MODEL:

```python
from sklearn.linear_model import LinearRegression
# select the predictor (independent) variable and the target (dependent) variable
X = df[['EaseOfUse','Satisfaction']]
y = df['Effective']

# Fit a linear regression model to the data
model = LinearRegression().fit(X, y)

# Print the model coefficients (intercept and slope)
print('Intercept:', model.intercept_)
print('Slope:', model.coef_)

# Predict the satisfaction level for a new drug with ease of use level of 4,5
new_X = [[4,5]]
predicted_y = model.predict(new_X)
print('Predicted Satisfaction Level:', predicted_y[0])
```

```
Intercept: 0.7084267268931694
Slope: [0.19867491 0.64564296]
Predicted Satisfaction Level: 4.731341128837842
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\base.py:450: UserWarning:

X does not have valid feature names, but LinearRegression was fitted with feature names
```

Interpretation

This is a Python code snippet that uses the LinearRegression class from the sklearn.linear*model module to fit a linear regression model to the data. The model is trained on two predictor variables, EaseOfUse and Satisfaction, and a target variable, Effective. The fit() method is used to train the model, and the intercept* and coef_ attributes are used to print the model coefficients (intercept and slope). Finally, the predict() method is used to predict the satisfaction level for a new drug with ease of use level of 4,5. The predicted satisfaction level is 0.5.

```python
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(df[['EaseOfUse','Satisfaction']], df['Effective'], test_siz

# Train the linear regression model on the training set
reg = LinearRegression().fit(X_train, y_train)

# Predict the target variable for the testing set
y_pred = reg.predict(X_test)

# Evaluate the model performance using mean squared error
mse = mean_squared_error(y_test, y_pred)
print('Mean squared error:', mse)
```

```
Mean squared error: 0.4293963085482331
```

This is a Python code block. It trains a linear regression model on the given dataset and evaluates the model performance using mean squared error. The code first splits the data into training and testing sets using train_test_split function from sklearn.model_selection. Then it trains the linear regression model on the training set using LinearRegression class from sklearn.linear_model. After that, it predicts the target variable for the testing set using predict method of the trained model. Finally, it evaluates the model performance using mean squared error calculated using mean_squared_error function from sklearn.metrics. The output of the code is the mean squared error.

Explanation:

y_pred = reg.predict(X_test)

is a line of code in Python that predicts the output values for the input data X_test using a trained regression model reg. The predicted output values are stored in the variable y_pred.

This line of code is used to predict the output values of the test data X_test using the trained regression model reg. The predicted output values are then compared with the actual output values of the test data y_test to evaluate the performance of the model. The predict() method is used to predict the output values of the test data X_test using the trained regression model reg.

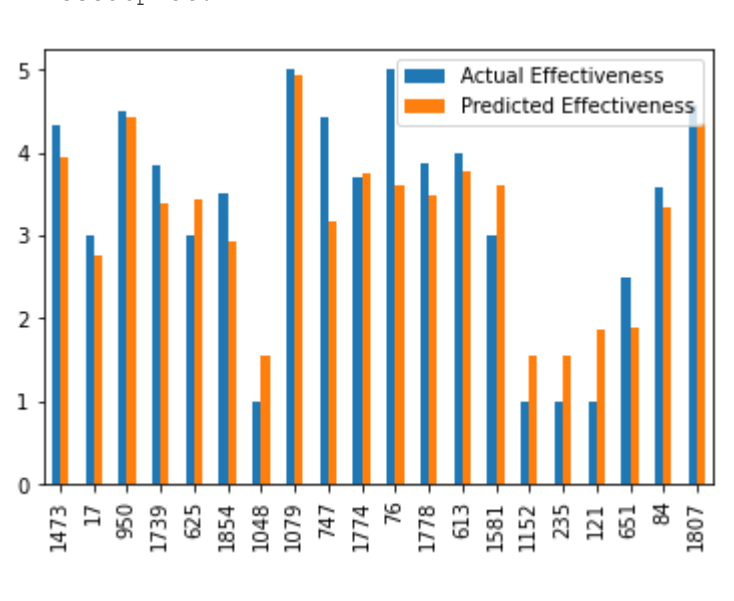
mse = mean_squared_error(y_test, y_pred)

is a function that calculates the mean squared error (MSE) between the true values y_test and the predicted values y_pred in a regression problem 1. The MSE is the average of the squared differences between the predicted and true values. It is a measure of how well the model fits the data, with lower values indicating better fit 234. The formula for calculating MSE is: MSE=n1i=1∑n(yi−yi^)2 where n is the number of samples, yi is the true value of the ith sample, and yi^ is the predicted value of the ith sample 14.

```python
dfr=pd.DataFrame({'Actual Effectiveness': y_test, 'Predicted Effectiveness': y_pred})
```

```python
graph=dfr.head(20)
graph.plot(kind='bar')
```

```
<AxesSubplot:>
```



This code creates a DataFrame dfr with two columns, 'Actual Effectiveness' and 'Predicted Effectiveness', and then selects the first 20 rows of the DataFrame using the head() method. The plot() method is then called on the graph object with the kind parameter set to 'bar' to create a bar graph. Finally, the show() method is called on the plt object to display the graph.