prescriber

March 17, 2022

0.0.1 Importing the Libraries

```
In [20]: import numpy as np
         import pandas as pd
         import string

         import matplotlib.pyplot as plt
         import seaborn as sns

         import ipywidgets
         from ipywidgets import interact

         plt.rcParams['figure.figsize'] = (15, 5)
         plt.style.use('fivethirtyeight')
```

0.0.2 Reading the Data

```
In [21]: # reading the Dataset
         data = pd.read_csv('drug.csv')

         # lets print the shape of the dataset
         print("The Shape of the Dataset :", data.shape)
```

```
The Shape of the Dataset : (161297, 7)
```

```
In [22]: # lets check the head of the dataset
         data.head()
```

Out[22]:

	uniqueID	drugName	condition	review	rating	date
0	206461	Valsartan	Left Ventricular Dysfunction	"It has no side effect, I take it in combinati...	9	20-May-12

```
1  "My son is halfway through his fourth week of ...       8  27-Apr-10
2  "I used to take another oral contraceptive, wh...       5  14-Dec-09
3  "This is my first time using any form of birth...       8   3-Nov-15
4  "Suboxone has completely turned my life around...       9  27-Nov-16

   usefulCount
0           27
1          192
2           17
3           10
4           37
```

In [23]:
```
# lets Explore Some of the Important Column in the dataset

print("Number of Unique Drugs present in the Dataset :", data['drugName'].nunique())
print("Number of Unique Medical Conditions present in the Dataset :", data['condition

print("\nThe Time Period of Collecting the Data")
print("Starting Date :", data['date'].min())
print("Ending Date :", data['date'].max())
```

```
Number of Unique Drugs present in the Dataset : 3436
Number of Unique Medical Conditions present in the Dataset : 884

The Time Period of Collecting the Data
Starting Date : 1-Apr-08
Ending Date : 9-Sep-17
```

0.0.3 Summarizing the Dataset

In [24]:
```
# lets summarize the Dataset
data[['rating','usefulCount']].describe()
```

Out[24]:

	rating	usefulCount
count	161297.000000	161297.000000
mean	6.994377	28.004755
std	3.272329	36.403742
min	1.000000	0.000000
25%	5.000000	6.000000
50%	8.000000	16.000000
75%	10.000000	36.000000
max	10.000000	1291.000000

In [25]:
```
# lets check the Number and Name of the Drugs with 0 Useful Count in Details
print("Analysis on Useless Drugs")
print("-------------------------")
print("The Number of Drugs with No Useful Count :", data[data['usefulCount'] == 0].cou
```

```
        # Lets Check the Number of Drugs with No Usesful Count with Review Greater than or Eq
        print("Number of Good Drugs with Lesser Useful Count :", data[(data['usefulCount'] ==
                                                        data['rating'] >= 8].count()[0])

        # Lets Check the Average Rating of the Drugs with No Useful Count
        print("Average Rating of Drugs with No Useful Count : {0:.2f}".format(data[data['usefu

        print("\nAnalysis on Useful Drugs")
        print("----------------------------")
        print("The Number of Drugs with Greater than 1000 Useful Counts :", data[data['useful
        print("Average Rating of Drugs with 1000+ Useful Counts :", data[data['usefulCount'] >
        print("\nName and Condition of these Drugs: \n\n",
            data[data['usefulCount'] > 1000][['drugName','condition']].reset_index(drop = Tru
```

```
Analysis on Useless Drugs
----------------------------
The Number of Drugs with No Useful Count : 6318
Number of Good Drugs with Lesser Useful Count : 0
Average Rating of Drugs with No Useful Count : 5.80

Analysis on Useful Drugs
----------------------------
The Number of Drugs with Greater than 1000 Useful Counts : 4
Average Rating of Drugs with 1000+ Useful Counts : 10.0

Name and Condition of these Drugs:

          drugName      condition
0          Mirena  Birth Control
1      Sertraline     Depression
2  Levonorgestrel  Birth Control
3          Zoloft     Depression
```

In [26]:
```
        # lets summarize Categorical data also
        data[['drugName','condition','review']].describe(include = 'object')
```

Out[26]:

	drugName	condition	review
count	161297	160398	161297
unique	3436	884	112329
top	Levonorgestrel	Birth Control	"Good"
freq	3657	28788	33

In [27]:
```
        # lets check for Missing Values
        data.isnull().sum()
```

Out[27]:
```
uniqueID         0
drugName         0
condition      899
```

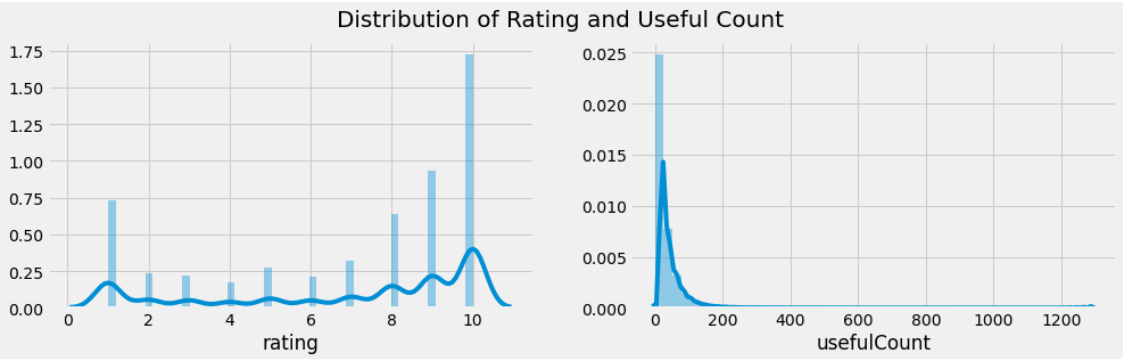
```
review          0
rating          0
date            0
usefulCount     0
dtype: int64
```

In [28]:
```
# as we know that condition is an Important Column, so we will delete all the records
data = data.dropna()

# lets check the Missing values now
data.isnull().sum().sum()
```

Out[28]: 0

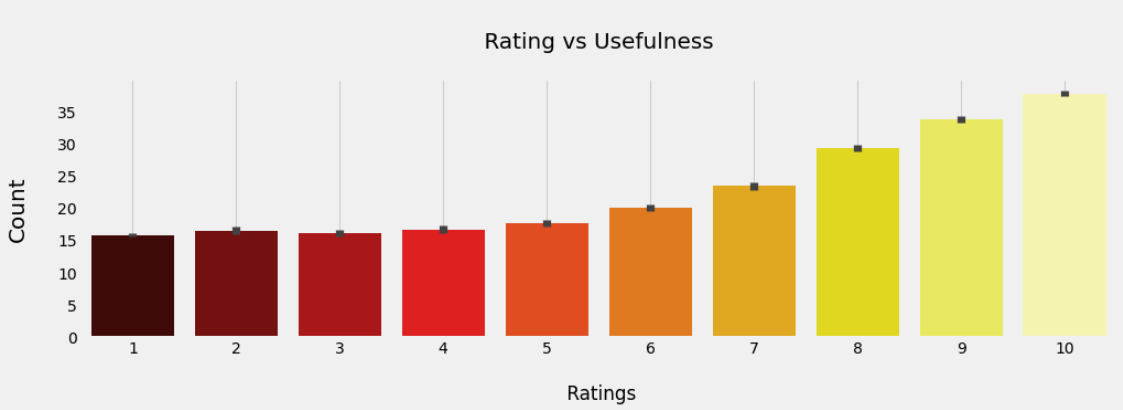
0.0.4 Unveiling Hidden Patterns from the Data

In [29]:
```
# lets check the Distribution of Rating and Useful Count

plt.rcParams['figure.figsize'] = (15, 4)

plt.subplot(1, 2, 1)
sns.distplot(data['rating'])

plt.subplot(1, 2, 2)
sns.distplot(data['usefulCount'])

plt.suptitle('Distribution of Rating and Useful Count \n ', fontsize = 20)
plt.show()
```



In [30]:
```
# lets check the Impact of Ratings on Usefulness

plt.rcParams['figure.figsize'] = (15, 4)
sns.barplot(data['rating'], data['usefulCount'], palette = 'hot')
plt.grid()
plt.xlabel('\n Ratings')
```

```
plt.ylabel('Count\n', fontsize = 20)
plt.title('\n Rating vs Usefulness \n', fontsize = 20)
plt.show()
```



In [31]:
```
# Checking whether Length of Review has any Impact on Ratings of the Drugs

# for that we need to create a new column to calculate length of the reviews
data['len']  = data['review'].apply(len)
```

In [32]:
```
# lets check the Impact of Length of Reviews on Ratings
data[['rating','len']].groupby(['rating']).agg(['min','mean','max'])
```

Out[32]:

	len		
	min	mean	max
rating			
1	5	428.784505	3692
2	9	452.902893	10787
3	8	461.249961	5112
4	7	464.077912	3030
5	6	477.982661	2048
6	4	467.957150	2202
7	6	485.597765	3063
8	3	483.584163	4087
9	3	477.696117	6182
10	3	443.215923	6192

In [33]:
```
# lets check the Highest Length Review
print("Length of Longest Review", data['len'].max())
data['review'][data['len'] == data['len'].max()].iloc[0]
```

```
Length of Longest Review 10787
```

Out[33]:
```
'"Two and a half months ago I was prescribed Venlafaxine to help prevent chronic migra
```

0.0.5 Cleaning the Reviews

In [34]:
```
# as it is clear that the reviews have so many unnecassry things such as Stopwords, P

# First lets remove Punctuations from the Reviews
def punctuation_removal(messy_str):
    clean_list = [char for char in messy_str if char not in string.punctuation]
    clean_str = ''.join(clean_list)
    return clean_str

data['review'] = data['review'].apply(punctuation_removal)
```

In [35]:
```
# Now lets Remove the Stopwords also

from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize

stop = stopwords.words('english')
stop.append("i'm")

stop_words = []

for item in stop:
    new_item = punctuation_removal(item)
    stop_words.append(new_item)

def stopwords_removal(messy_str):
    messy_str = word_tokenize(messy_str)
    return [word.lower() for word in messy_str
            if word.lower() not in stop_words ]

data['review'] = data['review'].apply(stopwords_removal)
```

In [36]:
```
# lets remove the Numbers also

import re
def drop_numbers(list_text):
    list_text_new = []
    for i in list_text:
        if not re.search('\d', i):
            list_text_new.append(i)
    return ' '.join(list_text_new)

data['review'] = data['review'].apply(drop_numbers)
```

0.0.6 Calculating the Sentiment from Reviews

In [37]:
```
# for using Sentiment Analyzer we will have to dowload the Vader Lexicon from NLTK
```

```
import nltk
nltk.download('vader_lexicon')
```

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

Out[37]: True

In [38]:
```
# lets calculate the Sentiment from Reviews

from nltk.sentiment.vader import SentimentIntensityAnalyzer
sid = SentimentIntensityAnalyzer()

train_sentiments = []

for i in data['review']:
    train_sentiments.append(sid.polarity_scores(i).get('compound'))

train_sentiments = np.asarray(train_sentiments)
data['sentiment'] = pd.Series(data=train_sentiments)
```

In [39]:
```
# lets check Impact of Sentiment on Reviews
data[['rating','sentiment']].groupby(['rating']).agg(['min','mean','max'])
```

Out[39]:

	sentiment		
	min	mean	max
rating			
1	-0.9931	0.005311	0.9898
2	-0.9929	0.003867	0.9924
3	-0.9925	0.003170	0.9877
4	-0.9919	0.000697	0.9867
5	-0.9920	0.014445	0.9882
6	-0.9914	0.008838	0.9936
7	-0.9938	-0.000509	0.9911
8	-0.9936	0.008952	0.9923
9	-0.9964	0.009489	0.9911
10	-0.9982	0.005446	0.9923

In [40]:
```
# as we can see that Sentiment and length of the review are not related to Reviews, w

# lets remove the unique Id, date, review, len, and sentiment column also
data = data.drop(['date','uniqueID','sentiment','review','len'], axis = 1)

# lets check the name of columns now
data.columns
```

Out[40]: Index(['drugName', 'condition', 'rating', 'usefulCount'], dtype='object')

0.0.7 Calculating Effectiveness and Usefulness of Drugs

In [41]:
```
# Lets Calculate an Effective Rating


min_rating = data['rating'].min()
max_rating = data['rating'].max()

def scale_rating(rating):
    rating -= min_rating
    rating = rating/(max_rating -1)
    rating *= 5
    rating = int(round(rating,0))

    if(int(rating) == 0 or int(rating)==1 or int(rating)==2):
        return 0
    else:
        return 1

data['eff_score'] = data['rating'].apply(scale_rating)
```

In [42]:
```
# lets also calculate Usefulness Score

data['usefulness'] = data['rating']*data['usefulCount']*data['eff_score']

# lets check the Top 10 Most Useful Drugs with their Respective Conditions
data[['drugName','condition','usefulness']][data['usefulness'] >
                                data['usefulness'].mean()].sort_values(by = 'usefulness',
                                            ascending = False).head(10).reset_index(drop =
```

Out[42]:

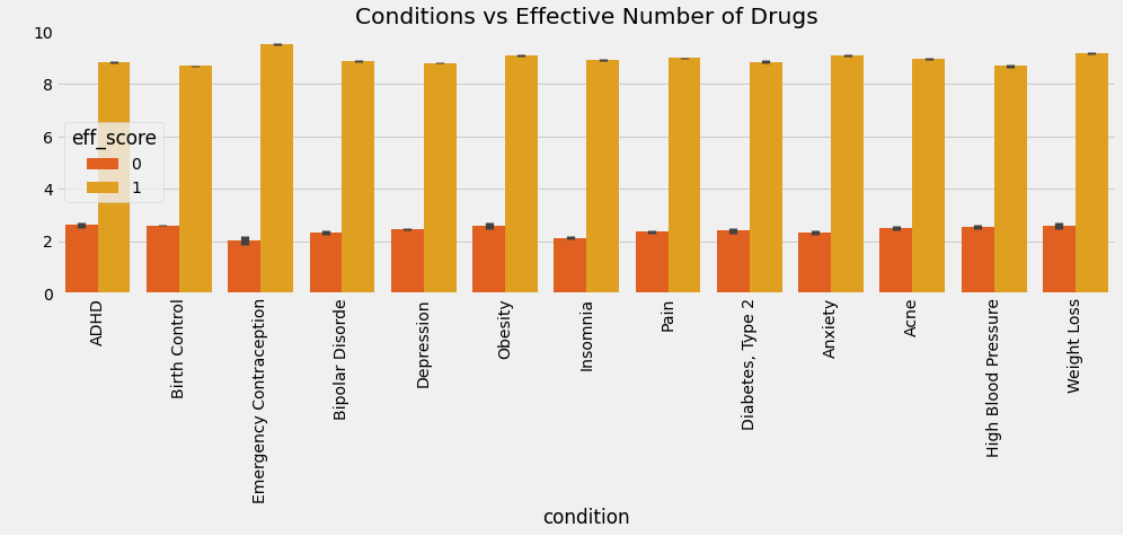
	drugName	condition	usefulness
0	Sertraline	Depression	12910
1	Zoloft	Depression	12910
2	Levonorgestrel	Birth Control	12470
3	Mirena	Birth Control	12470
4	Zoloft	Depression	8541
5	Phentermine	Weight Loss	7960
6	Adipex-P	Weight Loss	7960
7	Implanon	Birth Control	7300
8	Viibryd	Depression	6930
9	Vilazodone	Depression	6930

0.0.8 Analyzing the Medical Condtitions

In [43]:
```
# lets calculate the Number of Useless and Useful Drugs for Each Condition

@interact
def check(condition = list(data['condition'].value_counts().index)):
    return data[data['condition'] == condition]['eff_score'].value_counts()
```

```
interactive(children=(Dropdown(description='condition', options=('Birth Control', 'Depression'
```

In [44]:
```
# lets check this in Graph,

popular_conditions = ('Birth Control','Depression','Pain','Anxiety','Acne','Bipolar Di
                      'Obesity','ADHD', 'Diabetes, Type 2', 'Emergency Contraception'
conditions = data.loc[data['condition'].isin(popular_conditions)]

sns.barplot(x = conditions['condition'], y = conditions['rating'], hue = data['eff_sc
     palette = 'autumn')
plt.title('Conditions vs Effective Number of Drugs')
plt.xticks(rotation = 90)
plt.ylabel(' ')
plt.show()
```



In [45]:
```
# lets check the Most Common Conditions

print("Number of Unique Conditions :", data['condition'].nunique())
data['condition'].value_counts().head(10)
```

```
Number of Unique Conditions : 884
```

Out[45]:
```
Birth Control      28788
Depression          9069
Pain                6145
Anxiety             5904
Acne                5588
Bipolar Disorde     4224
```

```
Insomnia            3673
Weight Loss         3609
Obesity             3568
ADHD                3383
Name: condition, dtype: int64
```

In [46]:
```
# lets check Drugs, which were useful to Highest Number of Poeple
data[['drugName','usefulCount']][data['usefulCount'] >
                    data['usefulCount'].mean()].sort_values(by = 'usefulCount',
                                        ascending = False).head(10).reset_index(drop =
```

Out[46]:

	drugName	usefulCount
0	Zoloft	1291
1	Sertraline	1291
2	Levonorgestrel	1247
3	Mirena	1247
4	Zoloft	949
5	Adipex-P	796
6	Phentermine	796
7	Celexa	771
8	Citalopram	771
9	Implanon	730

0.0.9 Finding Most Useful and Useless Drugs for each Condition

In [47]:
```
# lets remove all the Duplicates from the Dataset
data = data.drop_duplicates()
```

In [48]:
```
# lets find the Highest and Lowest Rated Drugs for each Condition

@interact
def high_low_rate(condition = list(data['condition'].value_counts().index)):
    print("\n Top 5 Drugs")
    print(data[data['condition'] == condition][['drugName','usefulness']].sort_values
                                                ascending = False).head().reset_inde
    print("\n\n Bottom 5 Drugs")
    print(data[data['condition'] == condition][['drugName','usefulness']].sort_values
                                                ascending = True).head().reset_index
```

```
interactive(children=(Dropdown(description='condition', options=('Birth Control', 'Depression'
```