Amazon Top Books Review

Import Libraries

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from datetime import datetime
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
d1=pd.read_csv(r'C:\Users\admin\Downloads\amazon.csv')
```

Read the data set

In [3]:
```
d1
```

Out[3]:

	Name	Author	User Rating	Reviews	Price	Year	Genre
0	10-Day Green Smoothie Cleanse	JJ Smith	4.7	17350	8	2016	Non Fiction
1	11/22/63: A Novel	Stephen King	4.6	2052	22	2011	Fiction
2	12 Rules for Life: An Antidote to Chaos	Jordan B. Peterson	4.7	18979	15	2018	Non Fiction
3	1984 (Signet Classics)	George Orwell	4.7	21424	6	2017	Fiction
4	5,000 Awesome Facts (About Everything!) (Natio...	National Geographic Kids	4.8	7665	12	2019	Non Fiction
...	...	...	...	...	...	...	...
545	Wrecking Ball (Diary of a Wimpy Kid Book 14)	Jeff Kinney	4.9	9413	8	2019	Fiction
546	You Are a Badass: How to Stop Doubting Your Gr...	Jen Sincero	4.7	14331	8	2016	Non Fiction
547	You Are a Badass: How to Stop Doubting Your Gr...	Jen Sincero	4.7	14331	8	2017	Non Fiction
548	You Are a Badass: How to Stop Doubting Your Gr...	Jen Sincero	4.7	14331	8	2018	Non Fiction
549	You Are a Badass: How to Stop Doubting Your Gr...	Jen Sincero	4.7	14331	8	2019	Non Fiction

550 rows × 7 columns

Data cleaning

Read the columns names

In [4]:
```
1 d1.columns
```

Out[4]:
```
Index(['Name', 'Author', 'User Rating', 'Reviews', 'Price', 'Year', 'Genre'], d
type='object')
```

In [5]:
```
1 d1.shape
```

Out[5]:
```
(550, 7)
```

check the information about the dataset

In [6]:
```
1 d1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 550 entries, 0 to 549
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Name         550 non-null    object
 1   Author       550 non-null    object
 2   User Rating  550 non-null    float64
 3   Reviews      550 non-null    int64
 4   Price        550 non-null    int64
 5   Year         550 non-null    int64
 6   Genre        550 non-null    object
dtypes: float64(1), int64(3), object(3)
memory usage: 30.2+ KB
```

Check is there any null value in the dataset

In [7]:
```
1 d1.isnull().sum()
```

Out[7]:
```
Name           0
Author         0
User Rating    0
Reviews        0
Price          0
Year           0
Genre          0
dtype: int64
```

Check the duplicate value in Dataset

In [8]:
```
1 d1.duplicated().sum()
```

Out[8]:
```
0
```

Check the statistical information about the dataset

In [9]:
```
1 d1.describe(include='all')
```

Out[9]:

	Name	Author	User Rating	Reviews	Price	Year	Genre
count	550	550	550.000000	550.000000	550.000000	550.000000	550
unique	351	248	NaN	NaN	NaN	NaN	2
top	Publication Manual of the American Psychologic...	Jeff Kinney	NaN	NaN	NaN	NaN	Non Fiction
freq	10	12	NaN	NaN	NaN	NaN	310
mean	NaN	NaN	4.618364	11953.281818	13.100000	2014.000000	NaN
std	NaN	NaN	0.226980	11731.132017	10.842262	3.165156	NaN
min	NaN	NaN	3.300000	37.000000	0.000000	2009.000000	NaN
25%	NaN	NaN	4.500000	4058.000000	7.000000	2011.000000	NaN
50%	NaN	NaN	4.700000	8580.000000	11.000000	2014.000000	NaN
75%	NaN	NaN	4.800000	17253.250000	16.000000	2017.000000	NaN
max	NaN	NaN	4.900000	87841.000000	105.000000	2019.000000	NaN

Arrange the dataset by highest price

In [10]:
```
1 d1.sort_values('Price',ascending=False)
```

Out[10]:

	Name	Author	User Rating	Reviews	Price	Year	Genre
69	Diagnostic and Statistical Manual of Mental Di...	American Psychiatric Association	4.5	6679	105	2013	Non Fiction
70	Diagnostic and Statistical Manual of Mental Di...	American Psychiatric Association	4.5	6679	105	2014	Non Fiction
473	The Twilight Saga Collection	Stephenie Meyer	4.7	3801	82	2009	Fiction
151	Hamilton: The Revolution	Lin-Manuel Miranda	4.9	5867	54	2016	Non Fiction
346	The Book of Basketball: The NBA According to T...	Bill Simmons	4.7	858	53	2009	Non Fiction
...	...	...	...	...	...	...	...
116	Frozen (Little Golden Book)	RH Disney	4.7	3642	0	2014	Fiction
71	Diary of a Wimpy Kid: Hard Luck, Book 8	Jeff Kinney	4.8	6812	0	2013	Fiction
505	To Kill a Mockingbird	Harper Lee	4.8	26234	0	2013	Fiction
506	To Kill a Mockingbird	Harper Lee	4.8	26234	0	2014	Fiction
358	The Constitution of the United States	Delegates of the Constitutional	4.8	2774	0	2016	Non Fiction

550 rows × 7 columns

max price, reviews,and rating of fiction book

In [11]:
```
1 d1[d1['Genre']=='Fiction'].max()
```

Out[11]:
```
Name           Wrecking Ball (Diary of a Wimpy Kid Book 14)
Author                                        Wizards RPG Team
User Rating                                             4.9
Reviews                                               87841
Price                                                    82
Year                                                   2019
Genre                                               Fiction
dtype: object
```

max price, reviews,and rating of non fiction books

In [12]:
```
1 d1[d1['Genre']=='Non Fiction'].max()
```

Out[12]:
```
Name           You Are a Badass: How to Stop Doubting Your Gr...
Author                                                Zhi Gang Sha
User Rating                                                    4.9
Reviews                                                      61133
Price                                                          105
Year                                                          2019
Genre                                                  Non Fiction
dtype: object
```

Count the numbers of book available in the dataset

In [13]:
```
1 d1['Name'].value_counts().sum()
```

Out[13]:
```
550
```

Check the unqiue books in datset

In [14]:
```
1 d2=d1.Name.unique()
2 d2
```

Out[14]:
```
array(['10-Day Green Smoothie Cleanse', '11/22/63: A Novel',
       '12 Rules for Life: An Antidote to Chaos',
       '1984 (Signet Classics)',
       '5,000 Awesome Facts (About Everything!) (National Geographic Kids)',
       'A Dance with Dragons (A Song of Ice and Fire)',
       'A Game of Thrones / A Clash of Kings / A Storm of Swords / A Feast of
Crows / A Dance with Dragons',
       'A Gentleman in Moscow: A Novel',
       'A Higher Loyalty: Truth, Lies, and Leadership',
       'A Man Called Ove: A Novel',
       "A Patriot's History of the United States: From Columbus's Great Disco
very to the War on Terror",
       'A Stolen Life: A Memoir', 'A Wrinkle in Time (Time Quintet)',
       'Act Like a Lady, Think Like a Man: What Men Really Think About Love,
Relationships, Intimacy, and Commitment',
       'Adult Coloring Book Designs: Stress Relief Coloring Book: Garden Desi
gns, Mandalas, Animals, and Paisley Patterns',
       'Adult Coloring Book: Stress Relieving Animal Designs',
       'Adult Coloring Book: Stress Relieving Patterns',
```

Is there any correlation in the datsets

In [15]:
```
1 d1.corr()
```

Out[15]:

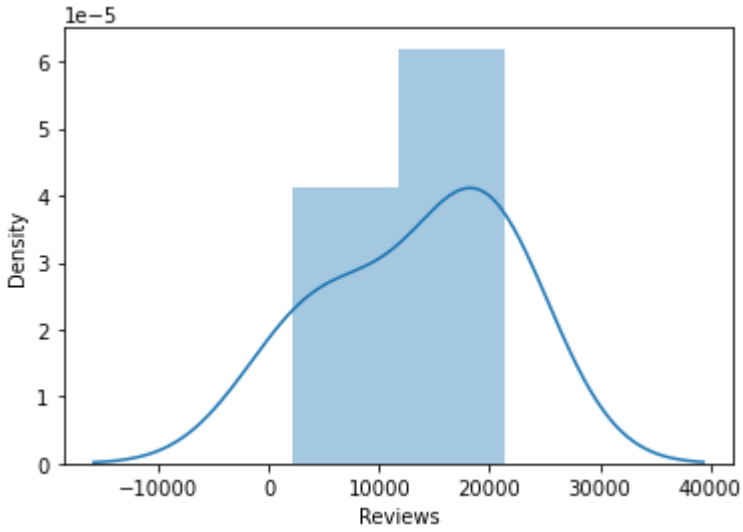
	User Rating	Reviews	Price	Year
User Rating	1.000000	-0.001729	-0.133086	0.242383
Reviews	-0.001729	1.000000	-0.109182	0.263560
Price	-0.133086	-0.109182	1.000000	-0.153979
Year	0.242383	0.263560	-0.153979	1.000000

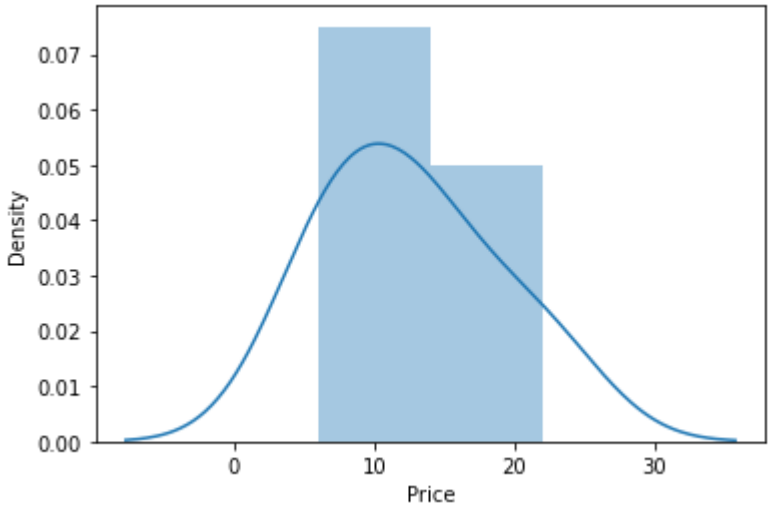
In [16]:
```
1 # Conclusion
2 # There are 550 rows and 7 columns in the dataset and 550 rows in the datset
3 # User Rating of the col are 550 non-null   UserRating: float64, Reviews: in
4 # The min rating of the book is 3.3, with the reviews 37 and price 0in the y
5 # The highest price of the book is $105 of the book name "Diagnostic and Sta
6 # The  max price, reviews,and rating of non fiction books: You Are a Badass:
7 # The max price, reviews,and rating of fiction book Name: Wrecking Ball (Dia
8 # There is neative correlation between reviews and rating, price and rating,
```

Data Visalization

Distplot

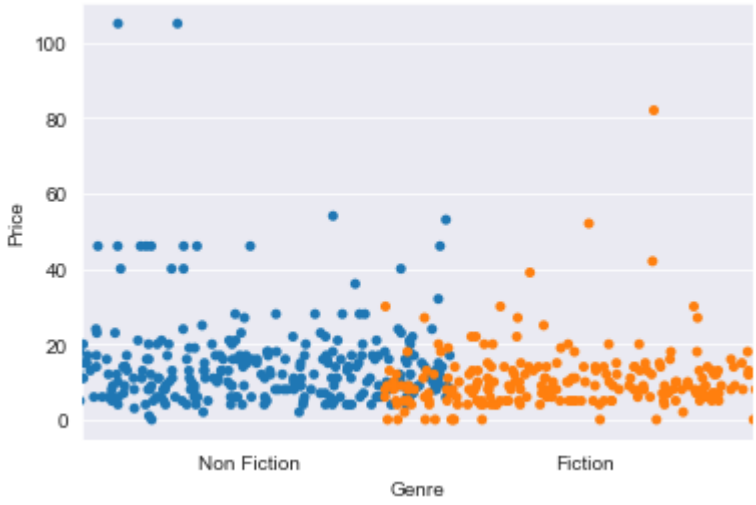
In [17]:
```
sns.distplot(d1['Reviews'].head(5))
plt.show()

sns.distplot(d1['Price'].head(5))
plt.show()
```





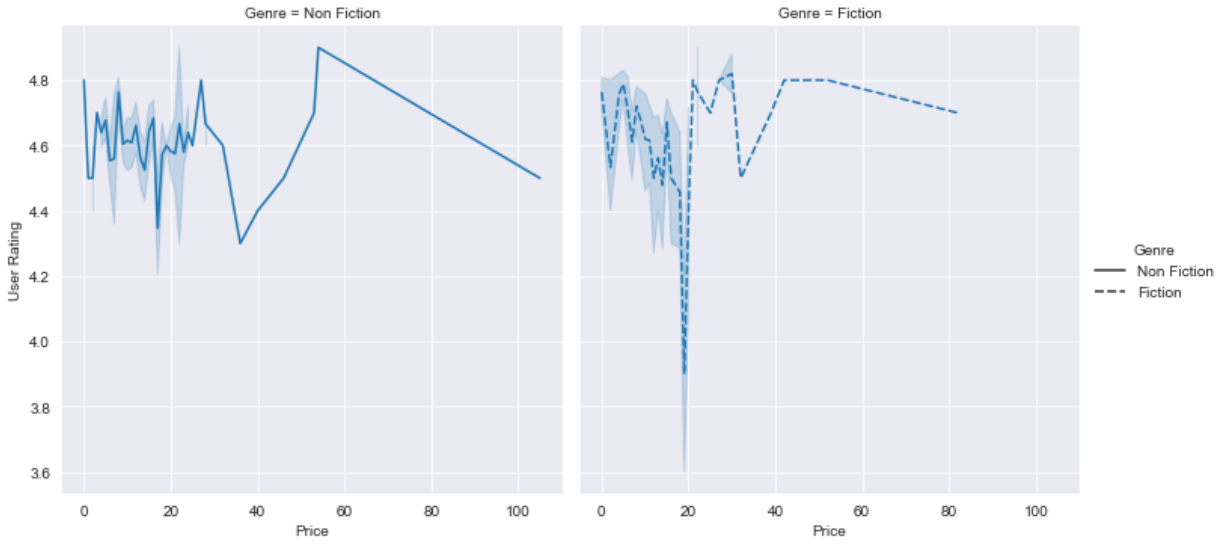
Stripplot

In [18]:
```
sns.set_style('darkgrid')
sns.stripplot(x='Genre',y='Price',data=d1,jitter=0.60,dodge=True)
plt.show()
```



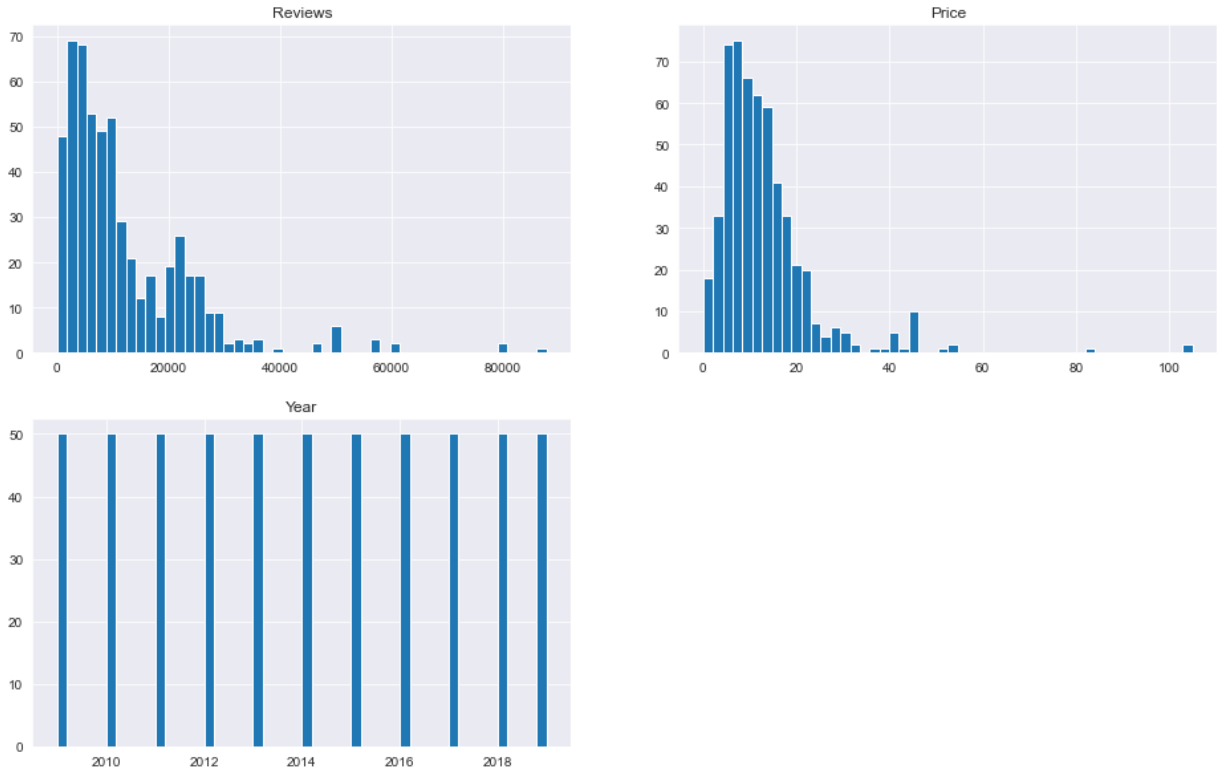
line plot

In [19]:
```
sns.relplot(x='Price',y='User Rating',style='Genre',col='Genre', kind='line'
plt.show()
```



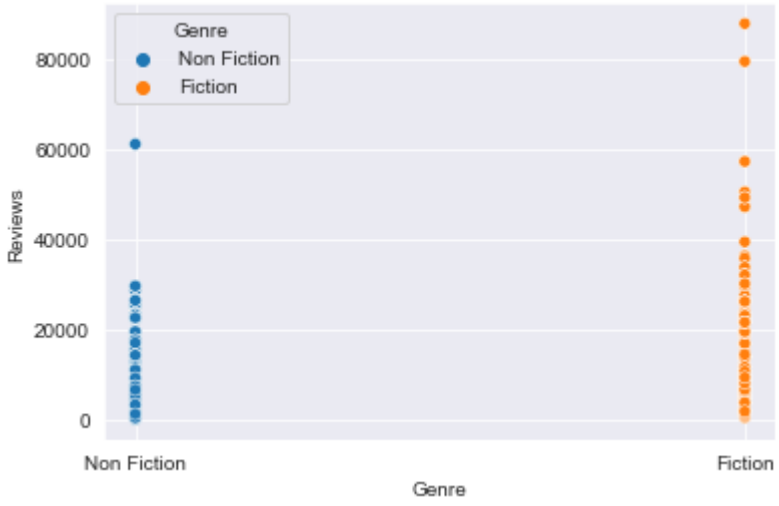
Histrogram

In [20]:
```
fig, axs = plt.subplots(2, 2, figsize=(16,10))
fig.delaxes(axs[1,1])

axs[0,0].hist(d1['Reviews'], bins=50)
axs[0,1].hist(d1['Price'], bins=50)
axs[1,0].hist(d1['Year'], bins=50)
axs[0,0].title.set_text('Reviews')
axs[0,1].title.set_text('Price')
axs[1,0].title.set_text('Year')

plt.show()
```



Scatterplot

In [21]:
```
sns.scatterplot(x='Genre',y='Reviews',hue='Genre',data=d1)
plt.show()
```



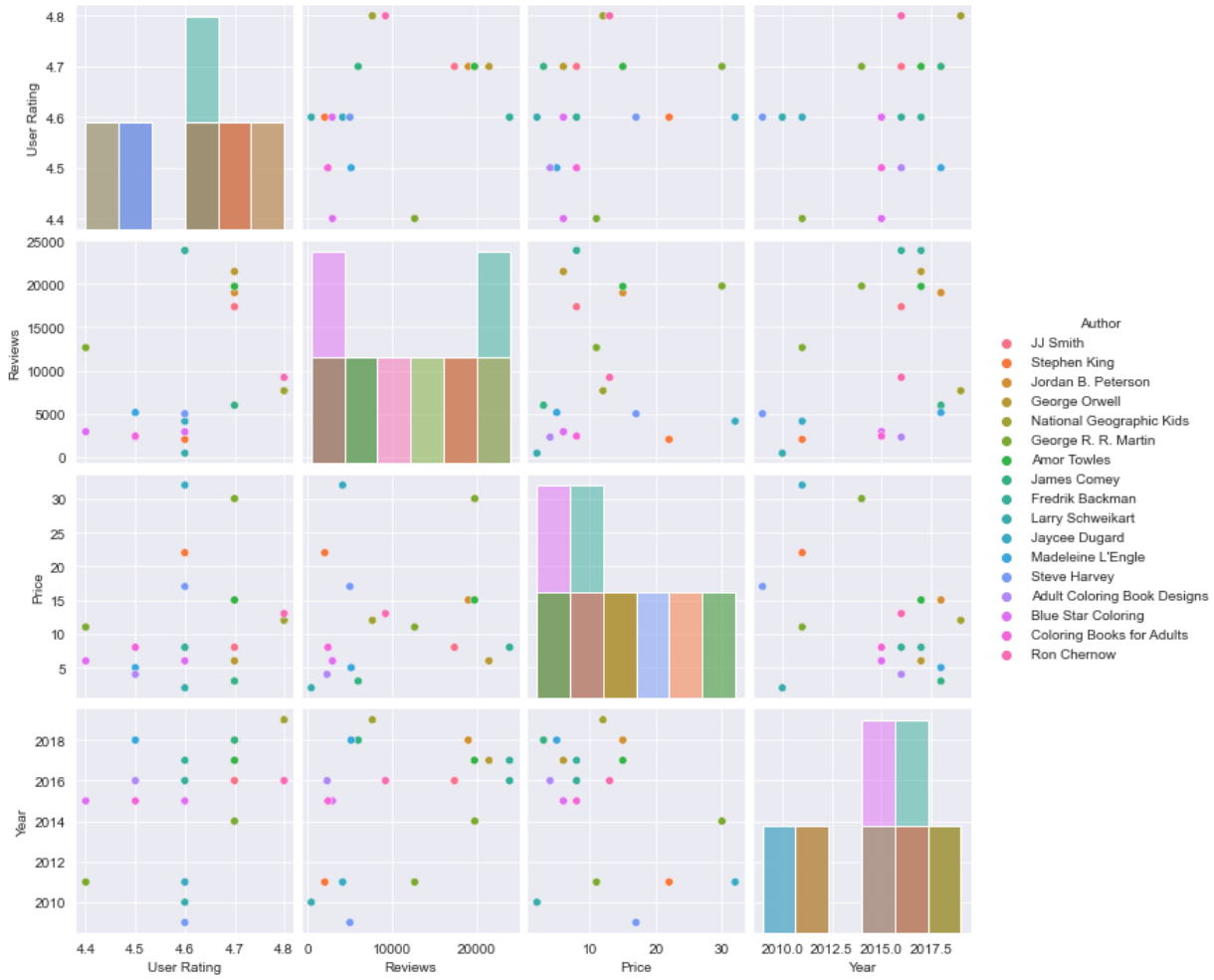
In [22]:
```
d3=d1.head(20)
d3
```

Out[22]:

	Name	Author	User Rating	Reviews	Price	Year	Genre
0	10-Day Green Smoothie Cleanse	JJ Smith	4.7	17350	8	2016	Non Fiction
1	11/22/63: A Novel	Stephen King	4.6	2052	22	2011	Fiction
2	12 Rules for Life: An Antidote to Chaos	Jordan B. Peterson	4.7	18979	15	2018	Non Fiction
3	1984 (Signet Classics)	George Orwell	4.7	21424	6	2017	Fiction
4	5,000 Awesome Facts (About Everything!) (Natio...	National Geographic Kids	4.8	7665	12	2019	Non Fiction
5	A Dance with Dragons (A Song of Ice and Fire)	George R. R. Martin	4.4	12643	11	2011	Fiction
6	A Game of Thrones / A Clash of Kings / A Storm...	George R. R. Martin	4.7	19735	30	2014	Fiction
7	A Gentleman in Moscow: A Novel	Amor Towles	4.7	19699	15	2017	Fiction
8	A Higher Loyalty: Truth, Lies, and Leadership	James Comey	4.7	5983	3	2018	Non Fiction
9	A Man Called Ove: A Novel	Fredrik Backman	4.6	23848	8	2016	Fiction
10	A Man Called Ove: A Novel	Fredrik Backman	4.6	23848	8	2017	Fiction
11	A Patriot's History of the United States: From...	Larry Schweikart	4.6	460	2	2010	Non Fiction
12	A Stolen Life: A Memoir	Jaycee Dugard	4.6	4149	32	2011	Non Fiction
13	A Wrinkle in Time (Time Quintet)	Madeleine L'Engle	4.5	5153	5	2018	Fiction
14	Act Like a Lady, Think Like a Man: What Men Re...	Steve Harvey	4.6	5013	17	2009	Non Fiction
15	Adult Coloring Book Designs: Stress Relief Col...	Adult Coloring Book Designs	4.5	2313	4	2016	Non Fiction
16	Adult Coloring Book: Stress Relieving Animal D...	Blue Star Coloring	4.6	2925	6	2015	Non Fiction
17	Adult Coloring Book: Stress Relieving Patterns	Blue Star Coloring	4.4	2951	6	2015	Non Fiction
18	Adult Coloring Books: A Coloring Book for Adul...	Coloring Books for Adults	4.5	2426	8	2015	Non Fiction
19	Alexander Hamilton	Ron Chernow	4.8	9198	13	2016	Non Fiction

Pairplot

In [23]:
```
1 sns.pairplot(d3,hue='Author',diag_kind='hist')
2 plt.show()
```



Heat map

In [24]:
```
sns.heatmap(d1.corr(),cmap='RdBu',vmin=-1,vmax=1, annot=True)
plt.show()
```

Conclusion

There is a negative relation between the price and year, price and review, price and rating

Review of fiction book is high as compare to the non fiction book i.e. the highest review of the fiction book is more than 80,000

The 10,000(approx) is the common reviews of the book that came 70 times in the dataset.

heighest count reviews which is came mostly in the is between 10,000 come 70 times in the dataset

10 is the common price of the book that came 80 times in the datset

The rating of non fiction book is high with the approx price 50 as compare tofiction books

Drop the unwanted columns

In [25]:
```
d=d1.drop(['Name','Author','Genre','Year'],axis=1)
d
```

Out[25]:

	User Rating	Reviews	Price
0	4.7	17350	8
1	4.6	2052	22
2	4.7	18979	15
3	4.7	21424	6
4	4.8	7665	12
...	...	...	...
545	4.9	9413	8
546	4.7	14331	8
547	4.7	14331	8
548	4.7	14331	8
549	4.7	14331	8

550 rows × 3 columns

In [26]:
```
x=d.iloc[:,:-1].values
x
```

Out[26]:
```
array([[4.7000e+00, 1.7350e+04],
       [4.6000e+00, 2.0520e+03],
       [4.7000e+00, 1.8979e+04],
       ...,
       [4.7000e+00, 1.4331e+04],
       [4.7000e+00, 1.4331e+04],
       [4.7000e+00, 1.4331e+04]])
```

In [27]:
```
y=d.iloc[:,-1].values
y
```

Out[27]:
```
array([  8,  22,  15,   6,  12,  11,  30,  15,   3,   8,   8,   2,  32,
         5,  17,   4,   6,   6,   8,  13,  14,  14,  13,   9,  13,   5,
         9,  14,   5,  11,  24,  21,  11,  11,  15,  13,  13,  18,  13,
         8,   5,   5,   0,   4,  18,  28,  11,  11,  11,  16,  14,  14,
        14,   8,   4,   5,  11,  11,  10,  13,   4,   8,   4,   5,   5,
         5,   5,  17,  15, 105, 105,   0,  15,  22,   5,  15,  15,   6,
         6,  13,  12,   6,   8,   4,   4,   8,   8,   6,  20,   5,  16,
         1,  14,   9,   9,   7,  18,  15,  15,   9,   8,   8,  15,   2,
         7,  11,  14,  14,  32,   6,   4,   4,   4,   4,   4,   9,   0,
         9,   5,   5,   5,  20,  16,   4,   4,   4,   4,   4,  12,  12,
        12,  11,  19,   9,   6,  10,  10,   9,   6,  14,  14,  14,  14,
         5,   5,   5,   7,   7,  10,  14,   7,  54,  11,  30,  12,  18,
        30,  22,   9,  52,   4,  10,  10,  14,  14,  22,  11,  11,  11,
        11,  11,  16,  15,  15,  17,  17,   7,   7,   4,   4,   9,  21,
        14,  20,  13,  13,  12,   8,   8,   8,   8,   8,   8,   0,  12,
        17,  12,  25,  10,  10,   6,   5,   6,   8,   4,   4,   4,  13,
         4,   4,   4,   4,   4,  13,  21,   6,  15,  18,  10,   0,  12,
         7,  13,   5,   5,  16,  20,  11,  27,   8,   8,   8,   6,  10,
        10,  10,   8,   8,   8,  16,  11,  10,   9,  14,  22,   8,   8,
         8,   8,   8,   8,   8,   8,   7,  12,  13,  13,   9,  13,  11,
        20,  20,   5,   5,   2,  27,  27,  27,   9,  10,  10,  46,  46,
        46,  46,  46,  46,  46,  46,  46,   4,  20,   7,   9,   9,
        12,  12,  12,  12,  20,  20,  10,   6,   6,   9,  11,  16,   6,
        25,  17,  20,  20,   6,  17,  17,  17,  17,  17,  17,  17,  17,
        17,  18,  13,  18,  20,  20,  22,  21,  28,  28,  28,  28,  28,
         8,   8,   8,   8,   8,  12,  24,  24,  24,  24,  16,  16,  16,
        39,   9,  10,  10,  17,  11,  14,  12,  53,   6,   6,   8,   8,
        12,  12,  12,  21,   6,   6,  13,   0,  11,   9,   9,   9,  15,
        11,  13,  13,   7,  13,   6,   6,   6,   6,   6,   9,   6,   6,
         6,   6,   6,   6,   0,  18,   7,  14,  14,   9,  16,   2,   2,
         5,   5,  20,  20,   7,   7,   7,  14,  10,   7,  13,  11,   6,
         6,   8,   7,  14,  14,   8,   8,  30,  30,  13,   9,   9,   7,
         7,   9,   7,   7,  20,  13,  13,  11,  11,  11,  11,  18,  14,
        19,  13,   5,  10,   9,   8,   8,  12,  11,  11,  40,  40,  40,
        40,  40,  36,  16,  17,  14,  21,  18,  17,   5,   5,  21,  18,
        18,  12,  14,  12,   8,   8,   0,  14,  10,  15,  15,  15,   9,
         7,  10,   6,   9,  10,  82,  12,  10,   5,   5,   5,   5,   5,
         5,   5,  16,  16,  16,  10,  10,  10,  10,   4,   4,  16,  15,
        19,  19,   9,  23,  11,  11,  23,  23,  23,  23,  12,   0,   0,
         0,   0,   7,  21,  21,  15,   9,   5,  16,  16,  16,  13,  16,
        20,  11,   4,   9,  42,  12,  18,  17,  14,  13,   9,   6,   6,
        14,  15,  13,  12,  18,  15,  11,   9,   9,   9,   9,   9,   8,
         8,   8,   8,   8], dtype=int64)
```

In [28]:
```
from sklearn.model_selection import train_test_split
```

In [29]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y)
```

In [30]:
```
print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(412, 2)
(138, 2)
(412,)
(138,)
```

In [31]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_stat
```

In [32]:
```
print(np.mean(x_train))
print(np.mean(x_test))
print(np.mean(y_train))
print(np.mean(y_test))
```

```
5619.78034090909
7415.629090909091
13.352272727272727
12.090909090909092
```

Linear Regression

In [33]:
```
from sklearn.linear_model import LinearRegression
```

In [34]:
```
l=LinearRegression()
l
```

Out[34]:
```
LinearRegression()
```

In [35]:
```
l.fit(x_train,y_train)
print("trai test complete")
```

```
trai test complete
```

In [36]:
```
y_pred=l.predict(x_test)
y_pred
```

Out[36]:
```
array([12.93573502, 18.71447085, 11.99167319, 13.05191734, 13.23383076,
       11.12745159, 18.71447085, 12.91857327, 15.52651795, 19.05148581,
       14.44265126,  3.91601794, 10.3195606 , 15.40540908, 12.10243667,
       11.97208876,  9.75913681, 14.40307831, 13.402048  , 14.44107642,
       12.08034044, 17.5317609 , 15.45149929, 10.32271027, 14.32873811,
       13.50698855, 11.45481548, 14.07812916, 10.0230229 , 12.16165845,
       12.24527829, 10.52944171, 13.57082192, 13.14062518, 17.58289059,
       13.05191734, 11.63591372, 12.08317514, 11.88427729, 10.86886348,
       12.84712424, 12.96460702, 16.0649505 , 10.3195606 , 11.52767791,
       15.86568122, 15.81142566, 11.18420216, 15.08393251, 15.16387773,
       10.08249525, 14.20768569, 13.87718006, 11.97162028,  7.96366204,
       12.59505338, 12.6214621 , 14.97295905, 12.77846138, 10.3195606 ,
       11.35890399, 13.748302  , 17.49973923, 10.45547293, 13.81182039,
       13.83733274, 14.53751286, 12.65247448, 14.0691001 , 14.97133568,
       10.08249525, 12.6214621 , 13.50283752, 12.09193777, 12.09403755,
       12.74192518, 11.44651341, 12.77195206,  9.96947847, 11.65261492,
       14.49210112, 11.38656465, 12.13382841, 12.33362263, 12.95384961,
        8.60184215, 10.14216078, 10.45547293, 12.17267437, 11.39103065,
       12.89877887, 13.00225751, 12.29940413, 12.77195206, 14.6615535 ,
       10.0230229 ,  7.96366204, 20.72786235, 13.42052608, 12.66538814,
       16.06642035, 18.27073053, 14.86513526, 12.85877803, 15.1574734 ,
       12.66528315, 10.52944171, 15.88268945, 10.08249525, 11.66452721])
```

In [37]:
```
d1=pd.DataFrame({'Actual':y_test,'Predicted':y_pred})
d1
```

Out[37]:

	Actual	Predicted
0	4	12.935735
1	17	18.714471
2	5	11.991673
3	11	13.051917
4	10	13.233831
...	...	...
105	14	12.665283
106	8	10.529442
107	6	15.882689
108	8	10.082495
109	4	11.664527

110 rows × 2 columns

In [38]:
```
from sklearn.metrics import mean_absolute_error
print('mean absolute error',mean_absolute_error(y_test,y_pred))
```

```
mean absolute error 5.835107313220198
```

In [39]:
```
sns.regplot(x=y_test,y=y_pred)
plt.title("Regression Plot",size=15)
plt.ylabel('Salary',size=12)
plt.xlabel('Year of Experience', size=12)
plt.show()
```

Conclusion

After droping the unwanted columns Name','Author','Genre','Year' and assigned the new dataset into new variable i.e. d we distributed the data into depend(y) and indepent(x) varibale and check that Is price depend on rating and reviews of the users.

We divided the data into train and test model and fit the train model into linesr regression.

Then we make a dataframe of actual and predicted values and concluded that there is a there is a actual and predicted values are differ