[1]:
```
# Import libraries used in data preparation
import pandas as pd
```

[2]:
```
# Load the amazon_products_final dataset .. in same folder
dataset = pd.read_csv("amazon_products_final.csv") #load dataset into a pandas DataFrame.
dataset # ensure read it successfully
```

[2]:

	index	page	name	category	image	price	rating	rating_count	delivery	is_best_seller	is_overall_pick
0	1	1	Apple iPad (10th Generation): with A14 Bionic ...	electronics	https://m.media-amazon.com/images/I/61uA2UVnYW...	No featured offers available$276.21(4 used & n...	4.8	19683.0	NaN	0	1
1	2	1	Nintendo Switch™ with Neon Blue and Neon Red J...	electronics	https://m.media-amazon.com/images/I/71wpE+ZIeh...	$299.00	4.7	8012.0	$45.66 delivery	0	0
2	3	1	Canon EOS Rebel T7 DSLR Camera with 18-55mm Le...	electronics	https://m.media-amazon.com/images/I/71Is-Zv6A0...	No featured offers available$453.92(19 used & ...	4.7	7318.0	NaN	1	0
3	4	1	LISEN Retractable Car Charger [69W USB C Car C...	electronics	https://m.media-amazon.com/images/I/71R6ka8Os4...	$16.98	4.5	1154.0	FREE delivery Thu, Dec 12 to Palestinian Terri...	1	0
4	5	1	SAMSUNG 990 EVO SSD 1TB, PCIe Gen 4x4, Gen 5x2...	electronics	https://m.media-amazon.com/images/I/71FHPYQ2Jb...	$69.99	4.7	1383.0	FREE delivery Thu, Dec 12 to Palestinian Terri...	1	0
...	...	...	...	...	...	...	...	...	...	...	...
2010	2011	9	[2024 Upgrade] Nmoiss Windshield Sun Shade Umb...	accessories	https://m.media-amazon.com/images/I/71sL6kb9cL...	$31.99	4.4	1958.0	FREE delivery Thu, Dec 12 to Palestinian Terri...	0	0
2011	2012	9	Spigen Center Console Organizer Tray(Carbon Ed...	accessories	https://m.media-amazon.com/images/I/71rMs+TLI-...	$16.78	4.7	3482.0	FREE delivery Sun, Dec 15 to Palestinian Terri...	0	0
2012	2013	9	TripleAliners Truck Bed Mat Compatible with 20...	accessories	https://m.media-amazon.com/images/I/81g-+3ehqF...	$87.19	4.6	940.0	$92.45 delivery Tue, Dec 17	1	0
2013	2014	9	SponsoredPUCKY CNC Wide Foot Pegs 360° Roating...	accessories	https://m.media-amazon.com/images/I/61eeFHAYdQ...	$24.99	4.7	10.0	FREE delivery Thu, Dec 12 to Palestinian Terri...	0	0
2014	2015	9	SponsoredAZUTO Cup Holder, Drink Folder Bevera...	accessories	https://m.media-amazon.com/images/I/61qbHjGDl3...	$40.99	NaN	NaN	FREE delivery Tue, Dec 17 to Palestinian Terri...	0	0

2015 rows × 11 columns

[3]:
```
# get the dataset Size
size = dataset.shape
size
```

[3]:
```
(2015, 11)
```

[4]:
```
# get the dataset dimension
dimension = dataset.ndim
dimension
```

[4]:
```
2
```

[5]:
```
# get columns' titles
titles = dataset.columns
titles
```

[5]:
```
Index(['index', 'page', 'name', 'category', 'image', 'price', 'rating',
       'rating_count', 'delivery', 'is_best_seller', 'is_overall_pick'],
      dtype='object')
```

[6]:
```
# Step 1 : Data Exploration
# Display the first 5 rows of the dataset using head method
dataset.head()
```

[6]:

	index	page	name	category	image	price	rating	rating_count	delivery	is_best_seller	is_overall_pick
0	1	1	Apple iPad (10th Generation): with A14 Bionic ...	electronics	https://m.media-amazon.com/images/I/61uA2UVnYW...	No featured offers available$276.21(4 used & n...	4.8	19683.0	NaN	0	1
1	2	1	Nintendo Switch™ with	electronics	https://m.media-	$299.00	4.7	8012.0	$45.66	0	0

			Neon Blue and Neon Red J...		amazon.com/images/I/71wpL+Zleh...					delivery		
2	3	1	Canon EOS Rebel T7 DSLR Camera with 18-55mm Le...	electronics	https://m.media-amazon.com/images/I/71Is-Zv6A0...	No featured offers available$453.92(19 used & ...	4.7	7318.0	NaN	1	0
3	4	1	LISEN Retractable Car Charger [69W USB C Car C...	electronics	https://m.media-amazon.com/images/I/71R6ka8Os4...	$16.98	4.5	1154.0	FREE delivery Thu, Dec 12 to Palestinian Terri...	1	0
4	5	1	SAMSUNG 990 EVO SSD 1TB, PCIe Gen 4x4, Gen 5x2...	electronics	https://m.media-amazon.com/images/I/71FHPYQ2Jb...	$69.99	4.7	1383.0	FREE delivery Thu, Dec 12 to Palestinian Terri...	1	0

[7]:
```
# Display information about the dataset
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2015 entries, 0 to 2014
Data columns (total 11 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   index            2015 non-null   int64  
 1   page             2015 non-null   int64  
 2   name             2015 non-null   object 
 3   category         2015 non-null   object 
 4   image            2015 non-null   object 
 5   price            1950 non-null   object 
 6   rating           1849 non-null   float64
 7   rating_count     1782 non-null   float64
 8   delivery         1782 non-null   object 
 9   is_best_seller   2015 non-null   int64  
 10  is_overall_pick  2015 non-null   int64  
dtypes: float64(2), int64(4), object(5)
memory usage: 173.3+ KB
```

[8]:
```
# Display descriptive statistics of the dataset "for numeric columns"
dataset.describe()
```

[8]:

	index	page	rating	rating_count	is_best_seller	is_overall_pick
count	2015.000000	2015.000000	1849.000000	1782.000000	2015.00000	2015.00000
mean	1008.000000	4.166749	4.420227	9411.675645	0.08933	0.00397
std	581.824716	2.255741	0.394074	33841.556774	0.28529	0.06290
min	1.000000	1.000000	1.000000	1.000000	0.00000	0.00000
25%	504.500000	2.000000	4.300000	52.000000	0.00000	0.00000
50%	1008.000000	4.000000	4.500000	648.000000	0.00000	0.00000
75%	1511.500000	6.000000	4.600000	5042.750000	0.00000	0.00000
max	2015.000000	9.000000	5.000000	603391.000000	1.00000	1.00000

[9]:
```
# Display summary statistics for all columns
dataset.describe(include='all')
```

[9]:

	index	page	name	category	image	price	rating	rating_count	delivery	is_best_seller	is_overall
count	2015.000000	2015.000000	2015	2015	2015	1950	1849.000000	1782.000000	1782	2015.00000	2015.0
unique	NaN	NaN	1580	8	1652	643	NaN	NaN	67	NaN	
top	NaN	NaN	SponsoredSponsored You're seeing this ad based...	flowers	https://m.media-amazon.com/images/I/61s+9FIpRk...	$19.99	NaN	NaN	Delivery Thu, Dec 12 to Palestinian Territories	NaN	
freq	NaN	NaN	7	380	8	53	NaN	NaN	601	NaN	
mean	1008.000000	4.166749	NaN	NaN	NaN	NaN	4.420227	9411.675645	NaN	0.08933	0.0
std	581.824716	2.255741	NaN	NaN	NaN	NaN	0.394074	33841.556774	NaN	0.28529	0.0
min	1.000000	1.000000	NaN	NaN	NaN	NaN	1.000000	1.000000	NaN	0.00000	0.0
25%	504.500000	2.000000	NaN	NaN	NaN	NaN	4.300000	52.000000	NaN	0.00000	0.0
50%	1008.000000	4.000000	NaN	NaN	NaN	NaN	4.500000	648.000000	NaN	0.00000	0.0
75%	1511.500000	6.000000	NaN	NaN	NaN	NaN	4.600000	5042.750000	NaN	0.00000	0.0
max	2015.000000	9.000000	NaN	NaN	NaN	NaN	5.000000	603391.000000	NaN	1.00000	1.0

[10]:
```
dataset.dtypes
```

[10]:
```
index               int64
page                int64
name               object
category           object
image              object
price              object
rating            float64
rating_count      float64
delivery           object
is_best_seller      int64
is_overall_pick     int64
dtype: object
```

[11]:
```
# Step 2 : Data Cleaning

#count missing values in each attribute
dataset.isnull().sum() #or pd.DataFrame({'missing': dataset.isnull().sum()})
```

[11]:
```
index              0
page               0
name               0
category           0
image              0
```

```
price              65
rating            166
rating_count      233
delivery          233
is_best_seller      0
is_overall_pick     0
dtype: int64
```

[12]:
```
# considerd as data transformation and cleaning
import numpy as np
import re  #regular wxspersion

# extract the numeric price from the string No featured offers available$276.21(4 used & n...   be 276.21
def extract_price(value):
    if isinstance(value, str):
        match = re.search(r'(\d+\.\d{2})', value)
        return float(match.group(1)) if match else np.nan
    else:
        return np.nan

dataset['price'] = dataset['price'].apply(extract_price)

# processed price column
dataset['price']
```

[12]:
```
0       276.21
1       299.00
2       453.92
3        16.98
4        69.99
         ...
2010     31.99
2011     16.78
2012     87.19
2013     24.99
2014     40.99
Name: price, Length: 2015, dtype: float64
```

[13]:
```
# Handle missing values
dataset['price'] = dataset['price'].fillna(dataset['price'].mode()[0])# most frequent value
dataset['rating'] = dataset['rating'].fillna(dataset['rating'].median()) #median value
dataset['rating_count'] = dataset['rating_count'].fillna(0) # with 0
dataset['delivery'] = dataset['delivery'].fillna('Unknown') # with 'Unknown'
```

[14]:
```
dataset.isnull().sum()
```

[14]:
```
index             0
page              0
name              0
category          0
image             0
price             0
rating            0
rating_count      0
delivery          0
is_best_seller    0
is_overall_pick   0
dtype: int64
```

[15]:
```
dataset.duplicated() # duplicates in datset
```

[15]:
```
0       False
1       False
2       False
3       False
4       False
        ...
2010    False
2011    False
2012    False
2013    False
2014    False
Length: 2015, dtype: bool
```

[16]:
```
# dataset.drop_duplicates() don't nedded
dataset.describe()
```

[16]:

	index	page	price	rating	rating_count	is_best_seller	is_overall_pick
count	2015.000000	2015.000000	2015.000000	2015.000000	2015.000000	2015.00000	2015.00000
mean	1008.000000	4.166749	40.922501	4.426799	8323.377667	0.08933	0.00397
std	581.824716	2.255741	67.808313	0.378121	31965.911534	0.28529	0.06290
min	1.000000	1.000000	0.000000	1.000000	0.000000	0.00000	0.00000
25%	504.500000	2.000000	14.990000	4.300000	20.000000	0.00000	0.00000
50%	1008.000000	4.000000	22.390000	4.500000	373.000000	0.00000	0.00000
75%	1511.500000	6.000000	35.990000	4.600000	3915.500000	0.00000	0.00000
max	2015.000000	9.000000	999.990000	5.000000	603391.000000	1.00000	1.00000

[17]:
```
dataset.drop(columns=['page'], inplace=True) #usless coulmn
dataset.describe()
```

[17]:

	index	price	rating	rating_count	is_best_seller	is_overall_pick
count	2015.000000	2015.000000	2015.000000	2015.000000	2015.00000	2015.00000
mean	1008.000000	40.922501	4.426799	8323.377667	0.08933	0.00397
std	581.824716	67.808313	0.378121	31965.911534	0.28529	0.06290
min	1.000000	0.000000	1.000000	0.000000	0.00000	0.00000
25%	504.500000	14.990000	4.300000	20.000000	0.00000	0.00000
50%	1008.000000	22.390000	4.500000	373.000000	0.00000	0.00000
75%	1511.500000	35.990000	4.600000	3915.500000	0.00000	0.00000
max	2015.000000	999.990000	5.000000	603391.000000	1.00000	1.00000

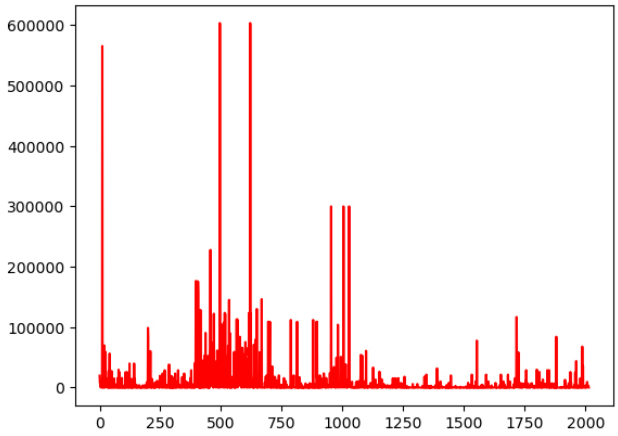
[18]:
```
dataset.describe(include= 'all')
```

[18]:

	index	name	category	image	price	rating	rating_count	delivery	is_best_seller	is_overall_pick
count	2015.000000	2015	2015	2015	2015.000000	2015.000000	2015.000000	2015	2015.00000	2015.00000
unique	NaN	1580	8	1652	NaN	NaN	NaN	68	NaN	NaN
top	NaN	SponsoredSponsored You're seeing this ad based...	flowers	https://m.media-amazon.com/images/I/61s+9FIpRk...	NaN	NaN	NaN	Delivery Thu, Dec 12 to Palestinian Territories	NaN	NaN
freq	NaN	7	380	8	NaN	NaN	NaN	601	NaN	NaN
mean	1008.000000	NaN	NaN	NaN	40.922501	4.426799	8323.377667	NaN	0.08933	0.00397
std	581.824716	NaN	NaN	NaN	67.808313	0.378121	31965.911534	NaN	0.28529	0.06290
min	1.000000	NaN	NaN	NaN	0.000000	1.000000	0.000000	NaN	0.00000	0.00000
25%	504.500000	NaN	NaN	NaN	14.990000	4.300000	20.000000	NaN	0.00000	0.00000
50%	1008.000000	NaN	NaN	NaN	22.390000	4.500000	373.000000	NaN	0.00000	0.00000
75%	1511.500000	NaN	NaN	NaN	35.990000	4.600000	3915.500000	NaN	0.00000	0.00000
max	2015.000000	NaN	NaN	NaN	999.990000	5.000000	603391.000000	NaN	1.00000	1.00000

[19]:
```
dataset['rating_count'].plot.line(color = 'red')
```

[19]:
```
<Axes: >
```

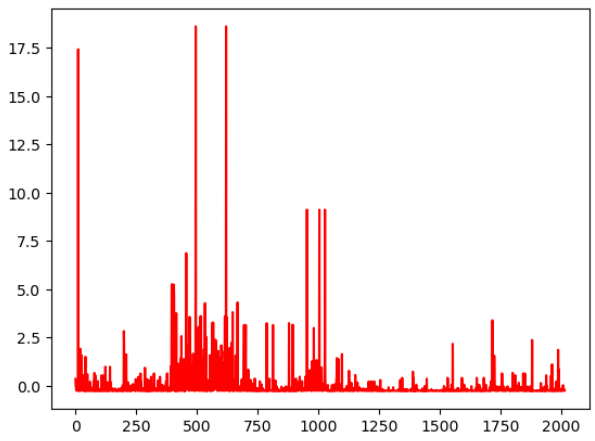


[20]:
```
# Step 3 : Data Transformaition

from sklearn.preprocessing import StandardScaler

# Normalize rating_count column
scaler = StandardScaler() # initializes the scaler.
dataset['rating_count'] = scaler.fit_transform(dataset[['rating_count']]) #The fit_transform method is applied to the rating_count column.
#fit: calculates the mean and standard deviation of rating_count
#transform:  applies the transformation to standardize the data

dataset['rating_count']
```

[20]:
```
0       0.355455
1      -0.009743
2      -0.031459
3      -0.224338
4      -0.217172
          ...
2010   -0.199180
2011   -0.151492
2012   -0.231034
2013   -0.260135
2014   -0.260448
Name: rating_count, Length: 2015, dtype: float64
```

[21]:
```
dataset['rating_count'].plot.line(color = 'red')
```

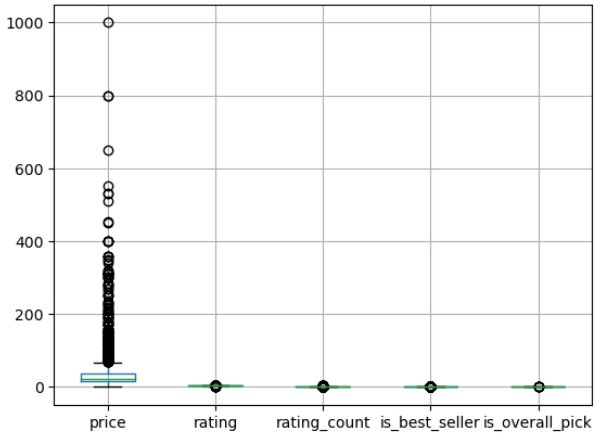
[21]:
```
<Axes: >
```



[22]:
```
# Data cleaning "outliers"

# indices of highest 20 rows rating_count values
indices = dataset['rating_count'].nlargest(20).index
```

```
# drop it rows from the dataset
dataset = dataset.drop(indices)

columns = ['price','rating', 'rating_count', 'is_best_seller', 'is_overall_pick']
dataset.boxplot(columns)
```

[22]: <Axes: >



[23]:
```
dataset['rating'].plot.line(color = 'pink')
```

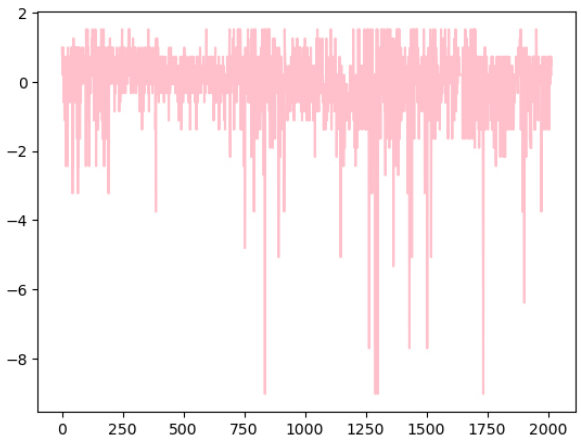
[23]: <Axes: >

[24]:
```
# Normalize rating column
scaler = StandardScaler() # initializes the scaler.
dataset['rating'] = scaler.fit_transform(dataset[['rating']])

dataset['rating']
```

[24]:
```
0       0.987575
1       0.724037
2       0.724037
3       0.196960
4       0.724037
          ...
2010   -0.066578
2011    0.724037
2012    0.460499
2013    0.724037
2014    0.196960
Name: rating, Length: 1995, dtype: float64
```

[25]:
```
dataset['rating'].plot.line(color = 'pink')
```

[25]: <Axes: >

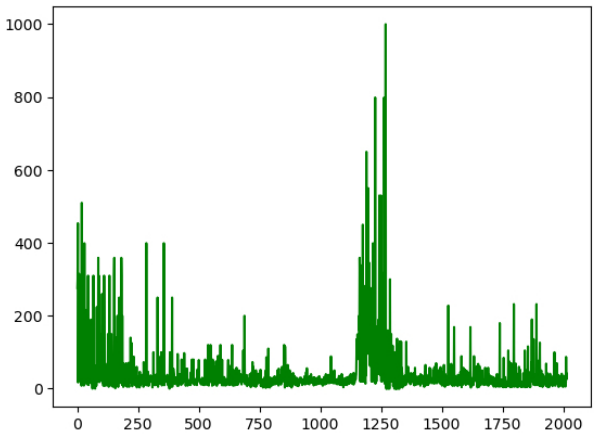


[26]:
```
# Data cleaning "outliers"

# indices of smallest 20 rows rating values
indices = dataset['rating'].nsmallest(20).index
```

```
# drop it rows from the dataset
dataset = dataset.drop(indices)
```

[27]:
```
dataset['price'].plot.line(color = 'green')
```

[27]:
```
<Axes: >
```

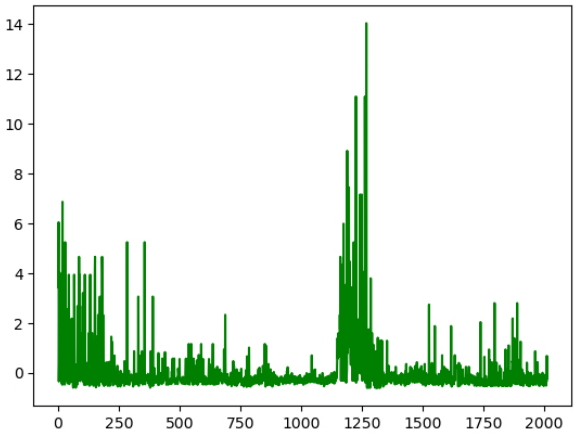


[28]:
```
# Normalize price column
scaler = StandardScaler() # initializes the scaler.
dataset['price'] = scaler.fit_transform(dataset[['price']])

dataset['price']
```

[28]:
```
0       3.443591
1       3.777381
2       6.046388
3      -0.353173
4       0.423229
          ...
2010   -0.133331
2011   -0.356102
2012    0.675145
2013   -0.235856
2014   -0.001515
Name: price, Length: 1975, dtype: float64
```

[29]:
```
dataset['price'].plot.line(color = 'green')
```

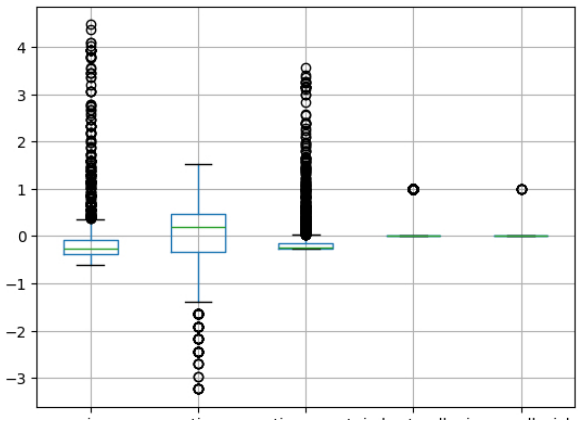
[29]:
```
<Axes: >
```



[30]:
```
# Data cleaning "outliers"

# indices of highest 20 rows price values
indices = dataset['price'].nlargest(20).index

# drop it rows from the dataset
dataset = dataset.drop(indices)
```

[31]:
```
columns = ['price','rating', 'rating_count', 'is_best_seller', 'is_overall_pick']
dataset.boxplot(columns)
```

[31]:
```
<Axes: >
```



price rating rating_count is_best_seller is_overall_pick

[32]:
```
# Step 4 : Feature Engineering

# price and rating columns should be numeric
# price_per_rating feature
dataset['price_per_rating'] = dataset['price'] / dataset['rating']
```

[33]:
```
# rating_density feature
dataset['rating_density'] = dataset['rating_count'] / dataset['price']
```

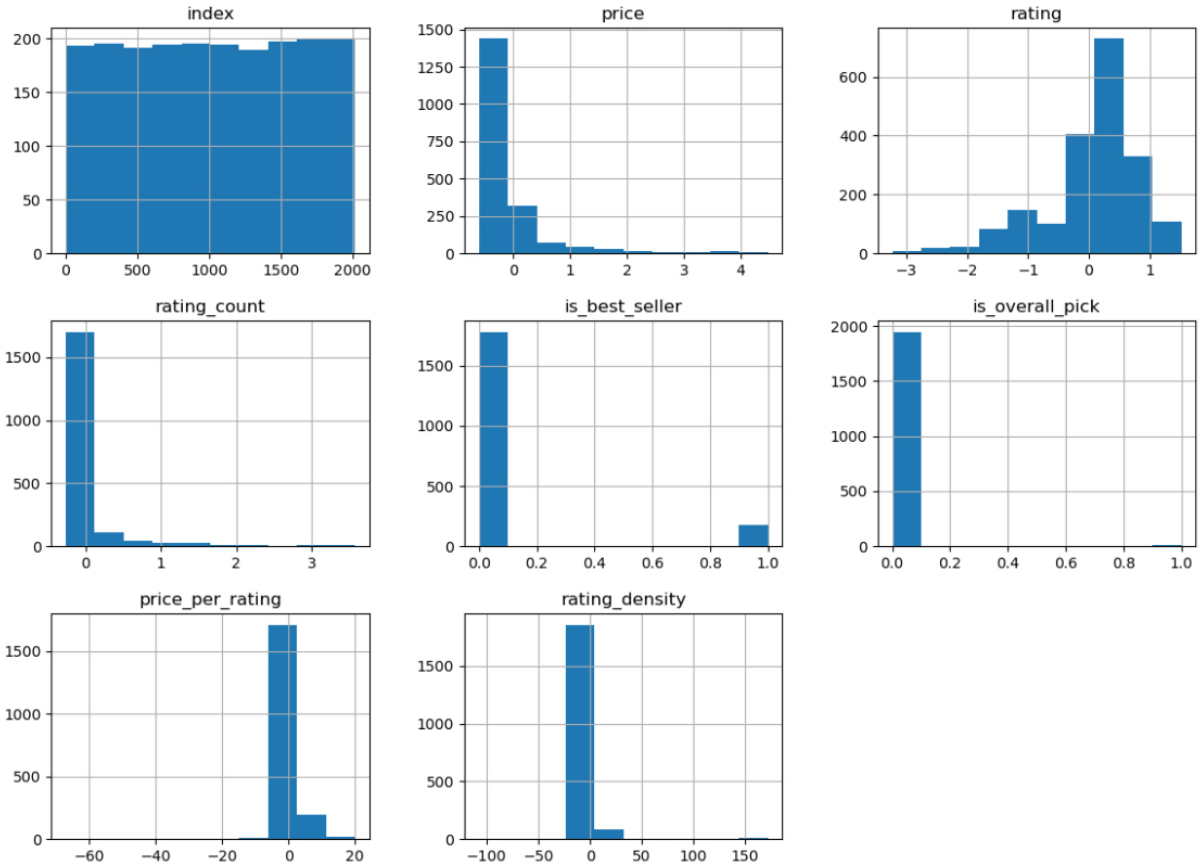
[34]:
```
dataset.head()
```

[34]:

	index	name	category	image	price	rating	rating_count	delivery	is_best_seller	is_overall_pick	price_per_rating	rating_d
0	1	Apple iPad (10th Generation): with A14 Bionic ...	electronics	https://m.media-amazon.com/images/I/61uA2UVnYW...	3.443591	0.987575	0.355455	Unknown	0	1	3.486915	0.1
1	2	Nintendo Switch™ with Neon Blue and Neon Red J...	electronics	https://m.media-amazon.com/images/I/71wpE+ZIeh...	3.777381	0.724037	-0.009743	$45.66 delivery	0	0	5.217110	-0.0
3	4	LISEN Retractable Car Charger [69W USB C Car C...	electronics	https://m.media-amazon.com/images/I/71R6ka8Os4...	-0.353173	0.196960	-0.224338	FREE delivery Thu, Dec 12 to Palestinian Terri...	1	0	-1.793116	0.6
4	5	SAMSUNG 990 EVO SSD 1TB, PCIe Gen 4x4, Gen 5x2...	electronics	https://m.media-amazon.com/images/I/71FHPYQ2Jb...	0.423229	0.724037	-0.217172	FREE delivery Thu, Dec 12 to Palestinian Terri...	1	0	0.584540	-0.5
5	6	SAMSUNG 990 PRO SSD 4TB PCIe 4.0 M.2 2280 Inte...	electronics	https://m.media-amazon.com/images/I/81WuG6lQuD...	3.352491	0.987575	0.048646	Delivery Thu, Dec 12 to Palestinian Territories	0	0	3.394669	0.0

[35]:
```
# Generate histograms for each column
dataset.hist(figsize=(14, 10))
```

[35]:
```
array([[<Axes: title={'center': 'index'}>,
        <Axes: title={'center': 'price'}>,
        <Axes: title={'center': 'rating'}>],
       [<Axes: title={'center': 'rating_count'}>,
        <Axes: title={'center': 'is_best_seller'}>,
        <Axes: title={'center': 'is_overall_pick'}>],
       [<Axes: title={'center': 'price_per_rating'}>,
        <Axes: title={'center': 'rating_density'}>, <Axes: >]],
      dtype=object)
```



[ ]: