```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from scipy.stats import chi2_contingency
from sklearn.cluster import KMeans
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

df=pd.read_json('flipkart_fashion_products_dataset.json')

df.head()
```

```
                                    _id actual_price average_rating
brand  \
0  fa8e22d6-c0b6-5229-bb9e-ad52eda39a0a        2,999            3.9
York
1  893e6980-f2a0-531f-b056-34dd63fe912c        1,499            3.9
York
2  eb4c8eab-8206-59d0-bcd1-a724d96bf74f        2,999            3.9
York
3  3f3f97bb-5faf-57df-a9ff-1af24e2b1045        2,999            3.9
York
4  750caa3d-6264-53ca-8ce1-94118a1d8951        2,999            3.9
York

                   category          crawled_at  \
0  Clothing and Accessories 2021-02-10 20:11:51
1  Clothing and Accessories 2021-02-10 20:11:52
2  Clothing and Accessories 2021-02-10 20:11:52
3  Clothing and Accessories 2021-02-10 20:11:53
4  Clothing and Accessories 2021-02-10 20:11:53

                                         description discount  \
0  Yorker trackpants made from 100% rich combed c...  69% off
1  Yorker trackpants made from 100% rich combed c...  66% off
2  Yorker trackpants made from 100% rich combed c...  68% off
3  Yorker trackpants made from 100% rich combed c...  69% off
4  Yorker trackpants made from 100% rich combed c...  68% off

                                              images  out_of_stock  \
0  [https://rukminim1.flixcart.com/image/128/128/...         False
1  [https://rukminim1.flixcart.com/image/128/128/...         False
2  [https://rukminim1.flixcart.com/image/128/128/...         False
3  [https://rukminim1.flixcart.com/image/128/128/...         False
4  [https://rukminim1.flixcart.com/image/128/128/...         False

                pid                                    product_details
\
```

```
0  TKPFCZ9EA7H5FYZH  [{'Style Code': '1005COMBO2'}, {'Closure': 'El...

1  TKPFCZ9EJZV2UVRZ  [{'Style Code': '1005BLUE'}, {'Closure': 'Draw...

2  TKPFCZ9EHFCY5Z4Y  [{'Style Code': '1005COMBO4'}, {'Closure': 'El...

3  TKPFCZ9ESZZ7YWEF  [{'Style Code': '1005COMBO3'}, {'Closure': 'El...

4  TKPFCZ9EVXKBSUD7  [{'Style Code': '1005COMBO1'}, {'Closure': 'Dr...


               seller selling_price sub_category  \
0  Shyam Enterprises           921   Bottomwear
1  Shyam Enterprises           499   Bottomwear
2  Shyam Enterprises           931   Bottomwear
3  Shyam Enterprises           911   Bottomwear
4  Shyam Enterprises           943   Bottomwear

                               title  \
0   Solid Men Multicolor Track Pants
1         Solid Men Blue Track Pants
2   Solid Men Multicolor Track Pants
3   Solid Men Multicolor Track Pants
4  Solid Men Brown, Grey Track Pants

                                                 url
0  https://www.flipkart.com/yorker-solid-men-mult...
1  https://www.flipkart.com/yorker-solid-men-blue...
2  https://www.flipkart.com/yorker-solid-men-mult...
3  https://www.flipkart.com/yorker-solid-men-mult...
4  https://www.flipkart.com/yorker-solid-men-brow...
```

```
df.drop(['images','url'],axis=1,inplace=True)

df.drop('description',axis=1,inplace=True)

df['discount_in_percentage']=df['discount'].str.replace("% off",'')

df.drop('discount',axis=1,inplace=True)

df.head(n=1)
```

```
                                    _id actual_price average_rating
brand  \
0  fa8e22d6-c0b6-5229-bb9e-ad52eda39a0a        2,999            3.9
York

                   category          crawled_at  out_of_stock  \
0  Clothing and Accessories 2021-02-10 20:11:51         False

                 pid                                      product_details
\
```

```
0  TKPFCZ9EA7H5FYZH  [{'Style Code': '1005COMBO2'}, {'Closure': 'El...


              seller selling_price sub_category  \
0  Shyam Enterprises           921   Bottomwear

                              title discount_in_percentage
0  Solid Men Multicolor Track Pants                     69
```

```
df['actual_price']=df['actual_price'].str.replace(',','')
```

```
def remove_comma(value):
    return  value.replace(',','') if isinstance(value,str) else value
```

```
df['selling_price']=df['selling_price'].apply(remove_comma)
```

```
df.isnull().sum()
```

```
_id                        0
actual_price               0
average_rating             0
brand                      0
category                   0
crawled_at                 0
out_of_stock               0
pid                        0
product_details            0
seller                     0
selling_price              0
sub_category               0
title                      0
discount_in_percentage     0
dtype: int64
```

```
df.isna().sum()
```

```
_id                        0
actual_price               0
average_rating             0
brand                      0
category                   0
crawled_at                 0
out_of_stock               0
pid                        0
product_details            0
seller                     0
selling_price              0
sub_category               0
title                      0
discount_in_percentage     0
dtype: int64
```

```
df[['actual_price','selling_price','discount_in_percentage']]=df[['actual_price','selling_price','discount_in_percentage']].replace('',np.nan)

df=df.dropna(subset=['actual_price','selling_price','discount_in_percentage'])

df[['actual_price','selling_price','discount_in_percentage']]=df[['actual_price','selling_price','discount_in_percentage']].astype(int)
```

```
C:\Users\Prem M\AppData\Local\Temp\ipykernel_5628\2231274475.py:1:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
```

```
df[['actual_price','selling_price','discount_in_percentage']]=df[['actual_price','selling_price','discount_in_percentage']].astype(int)

df['average_rating']=df['average_rating'].replace('',np.nan)
```

```
C:\Users\Prem M\AppData\Local\Temp\ipykernel_5628\1678897527.py:1:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  df['average_rating']=df['average_rating'].replace('',np.nan)
```

```
df=df.dropna(subset=['average_rating'])

df['average_rating']=df['average_rating'].astype(float)

df.isnull().sum()
```

```
_id                     0
actual_price            0
average_rating          0
brand                   0
category                0
crawled_at              0
out_of_stock            0
pid                     0
product_details         0
seller                  0
selling_price           0
sub_category            0
```

```
title                       0
discount_in_percentage      0
dtype: int64
```

```
df.shape[0]
```

```
26869
```

```
df.dtypes
```

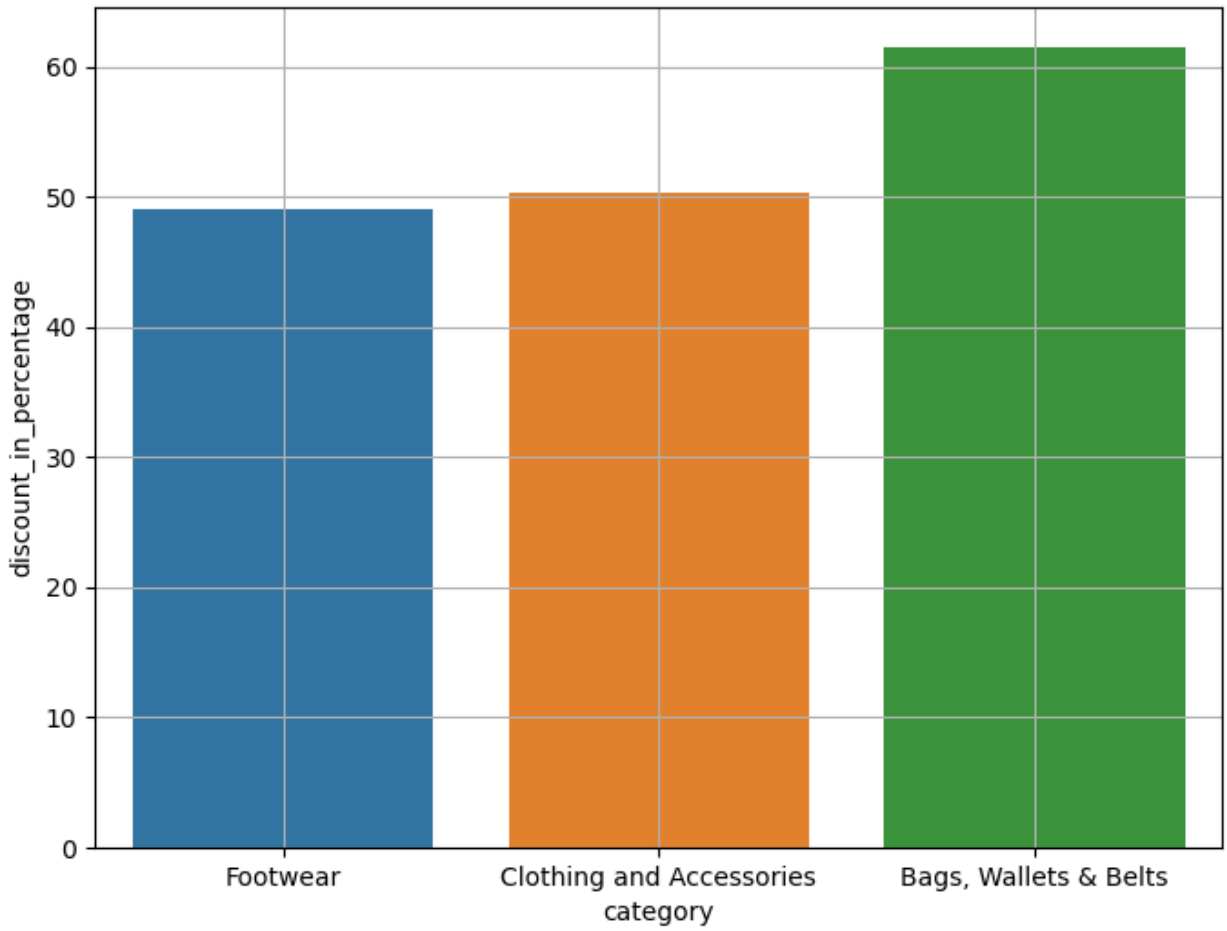
```
_id                                 object
actual_price                         int32
average_rating                     float64
brand                               object
category                            object
crawled_at                  datetime64[ns]
out_of_stock                          bool
pid                                 object
product_details                     object
seller                              object
selling_price                        int32
sub_category                        object
title                               object
discount_in_percentage               int32
dtype: object
```

```
df.describe()
```

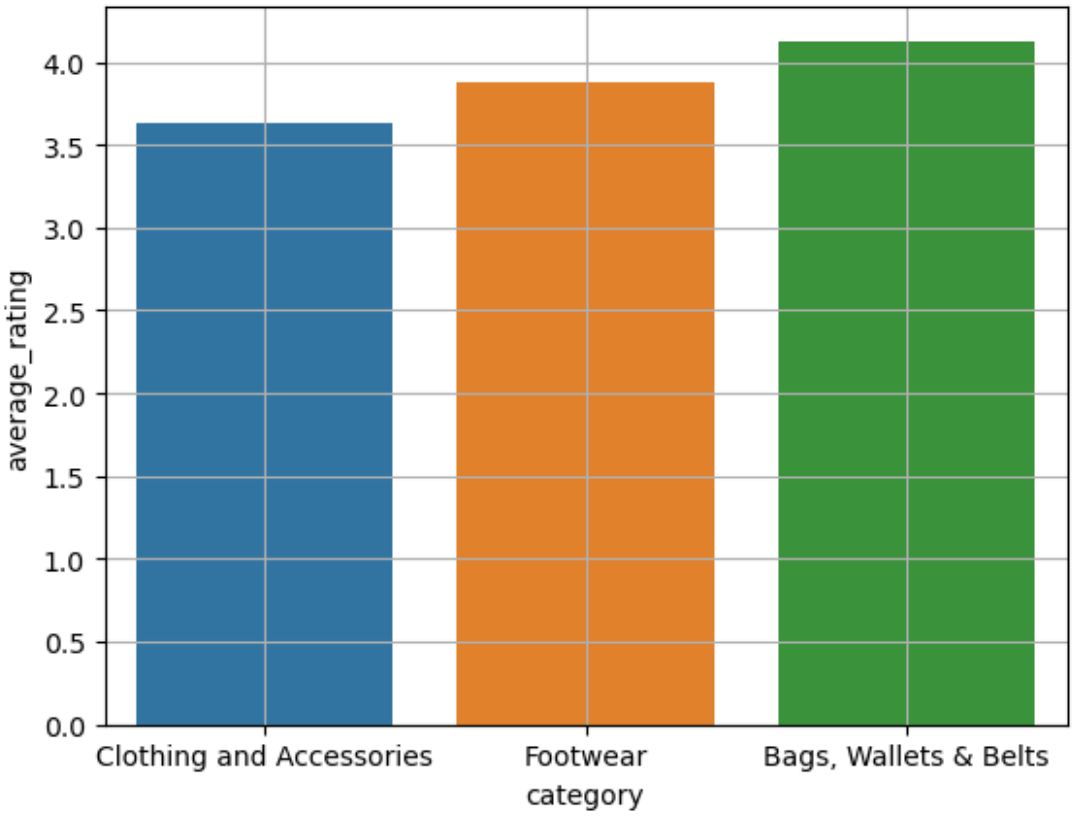
```
       actual_price  average_rating                       crawled_at  \
count  26869.000000    26869.000000                            26869
mean    1476.516059        3.641312  2021-02-10 22:57:07.081096960
min      150.000000        1.000000            2021-02-10 20:11:51
25%      870.000000        3.300000            2021-02-10 21:32:33
50%     1299.000000        3.800000            2021-02-10 22:59:28
75%     1799.000000        4.100000            2021-02-11 00:20:48
max    12999.000000        5.000000            2021-02-11 01:31:55
std      956.356077        0.664691                            NaN

       selling_price  discount_in_percentage
count   26869.000000            26869.000000
mean      701.751796               50.281923
min        99.000000                1.000000
25%       389.000000               40.000000
50%       549.000000               52.000000
75%       824.000000               63.000000
max      7999.000000               87.000000
std       527.823998               16.882599
```

```
df.columns
```

```
Index(['_id', 'actual_price', 'average_rating', 'brand', 'category',
       'crawled_at', 'out_of_stock', 'pid', 'product_details',
'seller',
       'selling_price', 'sub_category', 'title',
'discount_in_percentage'],
      dtype='object')

df.groupby('category').size()
#we have to eliminate toys sections since we dont have sufficient_data
df=df[df['category']!='Toys']

category_discount_avg=df.groupby('category')
['discount_in_percentage'].mean().reset_index()

category_discount_avg_sort=category_discount_avg.sort_values(by='disco
unt_in_percentage',ascending=True)

plt.figure(figsize=(8,6))
sns.barplot(x='category',y='discount_in_percentage',data=category_disc
ount_avg_sort)
plt.grid()
```

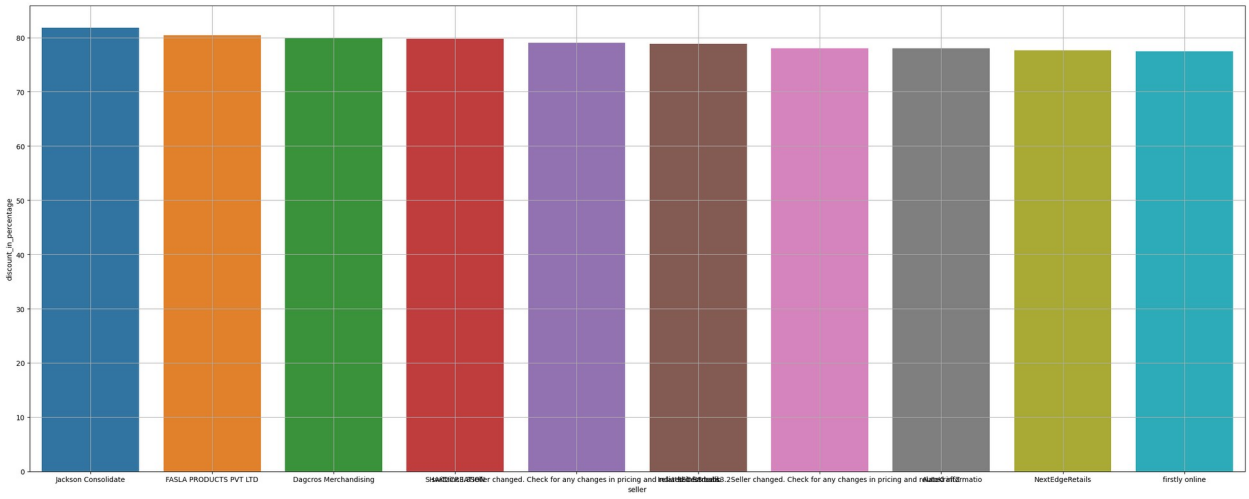


```
rating_category=df.groupby('category')
['average_rating'].mean().reset_index()

rating_category_sort=rating_category.sort_values(by='average_rating',a
scending=True)

sns.barplot(x='category',y='average_rating',data=rating_category_sort)
plt.grid()
```



so from this analysis , bags,wallets and belts has highest average discount percentage and high average rating,which is good, there isn't need to deep further

```
seller_discount=df.groupby('seller')
['discount_in_percentage'].mean().reset_index()

seller_discount_sort=seller_discount.sort_values(by='discount_in_perce
ntage',ascending=False)

top_10_sellers_discount=seller_discount_sort.head(n=10)

plt.figure(figsize=(25,10))
sns.barplot(x='seller',y='discount_in_percentage',data=top_10_sellers_
discount)
plt.tight_layout()
plt.grid()
```



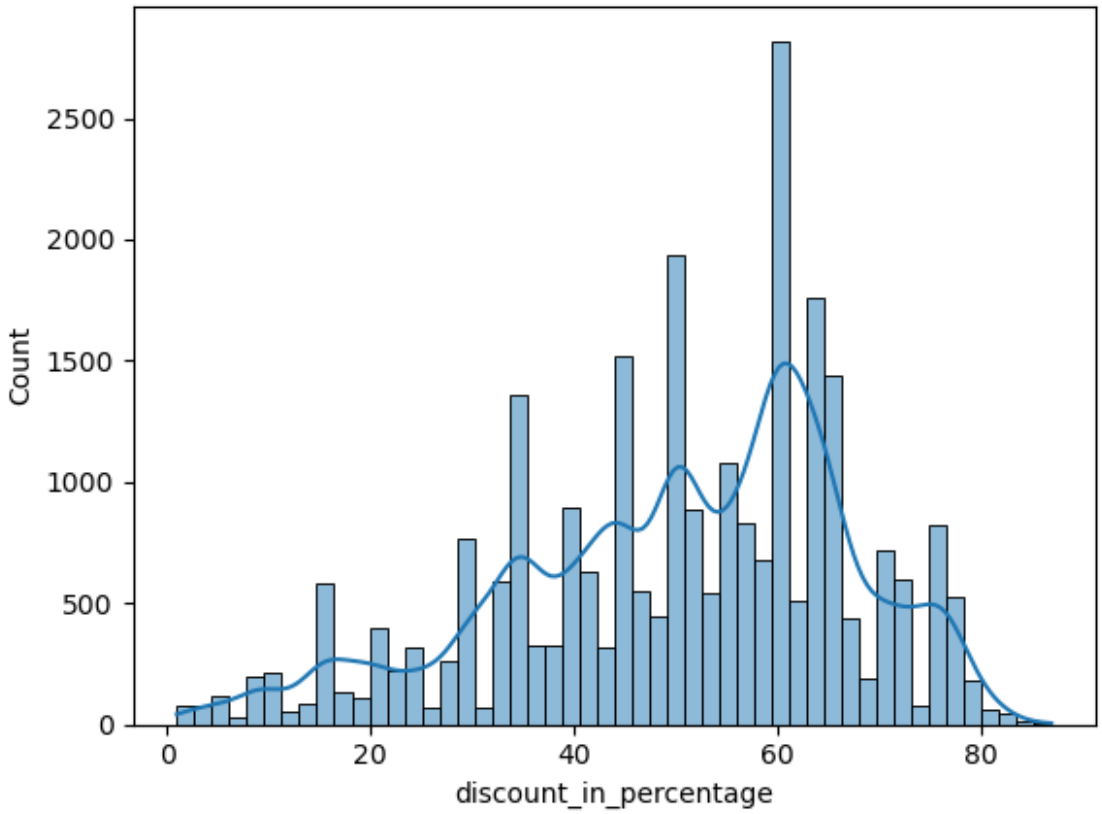

```
seller_80_discount=df[df['discount_in_percentage'] >=80]

seller_80_discount.groupby('discount_in_percentage')
['seller'].count().reset_index()
```

	discount_in_percentage	seller
0	80	96
1	81	64
2	82	35
3	83	9
4	84	8
5	85	1
6	86	2
7	87	1

possible interpretation - aggresive price competition

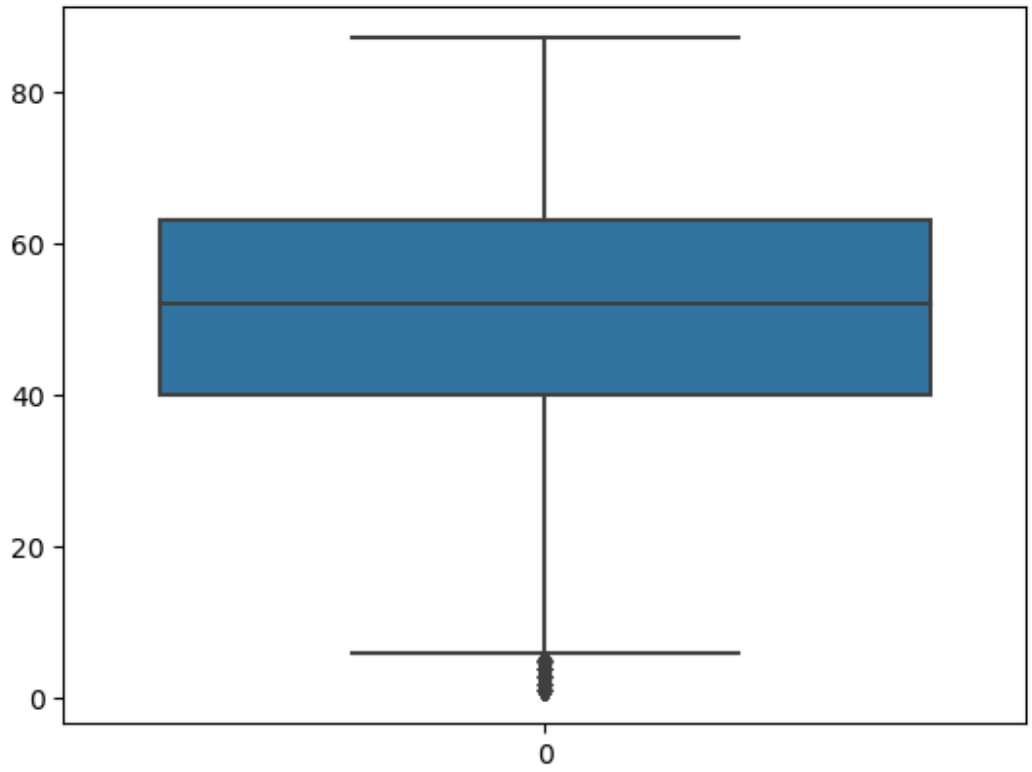
```
#lets plot histogram to see how discounts are distributed

sns.histplot(df['discount_in_percentage'],bins=50,kde=True)
```

```
<Axes: xlabel='discount_in_percentage', ylabel='Count'>
```




```
sns.boxplot(df['discount_in_percentage'])
```

```
<Axes: >
```

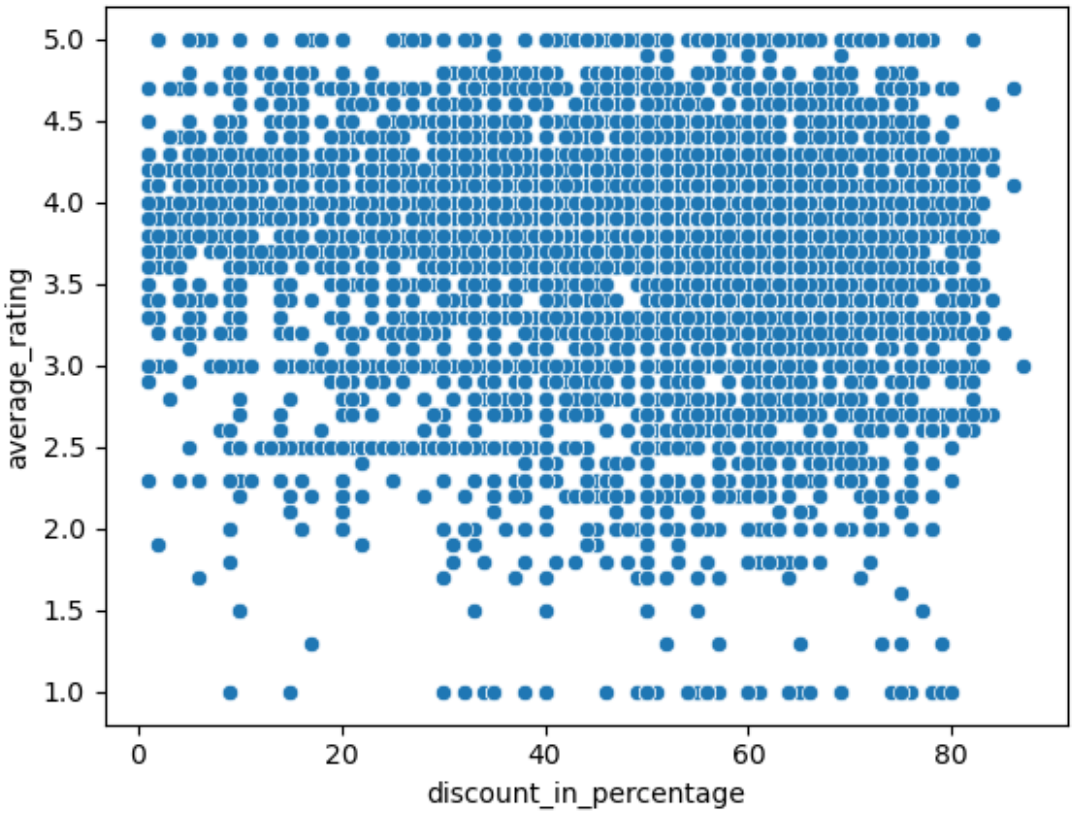



```
#sweet spot - discount rating is 50-55%

df[['discount_in_percentage','average_rating']].corr()
```

	discount_in_percentage	average_rating
discount_in_percentage	1.000000	-0.042124
average_rating	-0.042124	1.000000

```
#there is a weak negative correlation, as discounts increase , ratings slightly decrease

sns.scatterplot(x='discount_in_percentage',y='average_rating',data=df)
```

```
<Axes: xlabel='discount_in_percentage', ylabel='average_rating'>
```



products with low and high discounts receives similar ratings

```
actual_price=df.groupby('category')
['actual_price'].sum().reset_index()

discounted_price=df.groupby('category')
['selling_price'].sum().reset_index()

revenue_loss=actual_price.merge(discounted_price, on='category')

revenue_loss['loss']=revenue_loss['actual_price'] -
revenue_loss['selling_price']

revenue_loss['loss_percentage']=
(revenue_loss['loss']*100)/revenue_loss['actual_price']

revenue_loss
```

```
                   category  actual_price  selling_price      loss  \
0     Bags, Wallets & Belts         29330          10599     18731
1  Clothing and Accessories      38726811       18366453  20360358
2                  Footwear        915870         478128    437742

   loss_percentage
0        63.862939
```

```
1        52.574321
2        47.795211
```

```
revenue_loss['loss'].sum()
```

```
20816831
```

```
# 2.08 crore ruppes loss
```

```
df.head()
```

```
                                    _id  actual_price  average_rating
brand  \
0  fa8e22d6-c0b6-5229-bb9e-ad52eda39a0a          2999             3.9
York
1  893e6980-f2a0-531f-b056-34dd63fe912c          1499             3.9
York
2  eb4c8eab-8206-59d0-bcd1-a724d96bf74f          2999             3.9
York
3  3f3f97bb-5faf-57df-a9ff-1af24e2b1045          2999             3.9
York
4  750caa3d-6264-53ca-8ce1-94118a1d8951          2999             3.9
York

                   category          crawled_at  out_of_stock  \
0  Clothing and Accessories 2021-02-10 20:11:51         False
1  Clothing and Accessories 2021-02-10 20:11:52         False
2  Clothing and Accessories 2021-02-10 20:11:52         False
3  Clothing and Accessories 2021-02-10 20:11:53         False
4  Clothing and Accessories 2021-02-10 20:11:53         False

                pid                                      product_details
\
0  TKPFCZ9EA7H5FYZH  [{'Style Code': '1005COMBO2'}, {'Closure': 'El...

1  TKPFCZ9EJZV2UVRZ  [{'Style Code': '1005BLUE'}, {'Closure': 'Draw...

2  TKPFCZ9EHFCY5Z4Y  [{'Style Code': '1005COMBO4'}, {'Closure': 'El...

3  TKPFCZ9ESZZ7YWEF  [{'Style Code': '1005COMBO3'}, {'Closure': 'El...

4  TKPFCZ9EVXKBSUD7  [{'Style Code': '1005COMBO1'}, {'Closure': 'Dr...


              seller  selling_price sub_category  \
0  Shyam Enterprises            921   Bottomwear
1  Shyam Enterprises            499   Bottomwear
2  Shyam Enterprises            931   Bottomwear
3  Shyam Enterprises            911   Bottomwear
4  Shyam Enterprises            943   Bottomwear
```

```
                                title  discount_in_percentage  year
month_name  \
0   Solid Men Multicolor Track Pants                      69  2021
February
1         Solid Men Blue Track Pants                      66  2021
February
2   Solid Men Multicolor Track Pants                      68  2021
February
3   Solid Men Multicolor Track Pants                      69  2021
February
4  Solid Men Brown, Grey Track Pants                      68  2021
February

    day_name
0  Wednesday
1  Wednesday
2  Wednesday
3  Wednesday
4  Wednesday
```

```
contingency_table=pd.crosstab(df['category'],df['out_of_stock'])

chi2,p_value,dof,expected=chi2_contingency(contingency_table)

if p_value < 0.05:
    print("reject null hypothesis, there is siginificant association
between category and out of stock")
else:
    print("failed to reject null hypothesis, there is no siginificant
association between category and out of stock")
```

```
reject null hypothesis, there is siginificant association between
category and out of stock
```

```
stock_df=df[df['out_of_stock']==True]

stock_df.groupby('category').size()
```

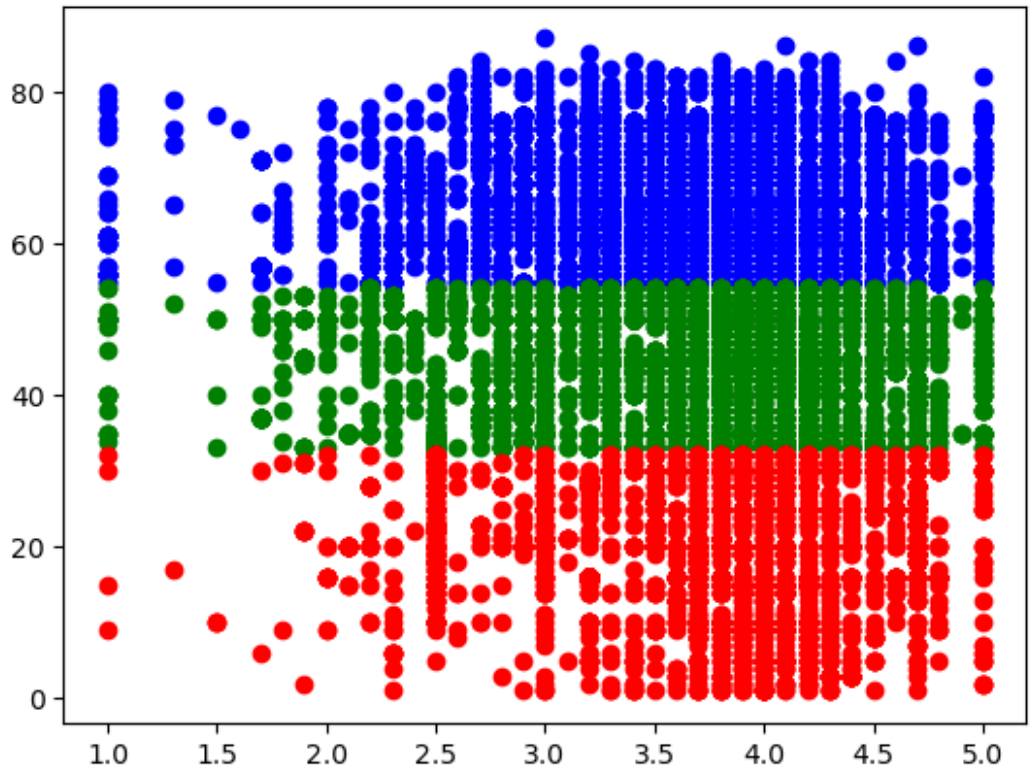
```
category
Clothing and Accessories    776
Footwear                      9
dtype: int64
```

```
# focus more on clothing and accessories

x=df[['discount_in_percentage','average_rating','out_of_stock']]

kmeans=KMeans(n_clusters=3,random_state=42)

df['cluster']=kmeans.fit_predict(x)
```

```
C:\Users\Prem M\anaconda3\envs\pandas_playground\Lib\site-packages\
sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of
`n_init` will change from 10 to 'auto' in 1.4. Set the value of
`n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
C:\Users\Prem M\AppData\Local\Temp\ipykernel_5628\589867116.py:1:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  df['cluster']=kmeans.fit_predict(x)
```

```
df1=df[df['cluster']==0]
df2=df[df['cluster']==1]
df3=df[df['cluster']==2]

plt.scatter(df1['average_rating'],df1['discount_in_percentage'],color=
'blue')
plt.scatter(df2['average_rating'],df2['discount_in_percentage'],color=
'green')
plt.scatter(df3['average_rating'],df3['discount_in_percentage'],color=
'red')
```

```
<matplotlib.collections.PathCollection at 0x2b6ec24b7d0>
```




```
#moderate discount i.e. cluster 2 offer a more efficient stratergy

x=df[['average_rating','actual_price','selling_price']]
y=df['discount_in_percentage']

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,rando
m_state=42)

model=LinearRegression()

model.fit(x,y)

LinearRegression()

y_pred=model.predict(x_test)

mse=mean_squared_error(y_test,y_pred)

rmse= np.sqrt(mse)
rmse

9.01913408777546

r2_score(y_test,y_pred)

0.7123475150655316
```

```
# accurate prediction, since rmse less than 10 and r_soce > 0.7 which
is great in real terms
```