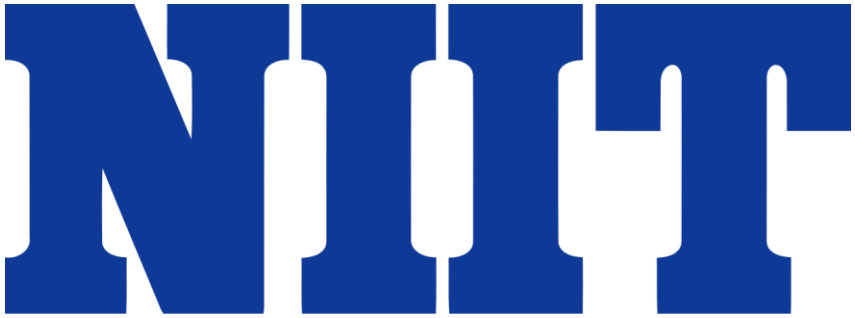


AMAZON PRODUCT REVIEW ANALYSIS ON TOOLS & HOME IMPROVEMENT AND PATIO LAWN & GARDEN CATEGORIES (1998-2018)

BY: ANIMESH GOUR, NAVEEN SINGH and ADRIRAJ DAS

ABSTRACT

The goal of this project is to monitor enormous number of reviews. By monitoring the entire review history, we will analyse the tone, language, keywords, and trends over time to provide valuable insights which will in turn increase the success rate of existing and new products and help Amazon make smarter business decisions.

Identification of current trends in data to classify products which are popular among customers.

Text processing of user reviews and clustering based on product prices and ratings, to identify trends in data which will help in inventory management.

Time Series Analysis to predict future trends with respect to product sales based on past data.

TABLE OF CONTENTS

INTRODUCTION

The year was 1994 when Jeff Bezos launched Amazon from his garage. In 1995, the first product launched by Amazon was a book in 50 states and in 45 countries within 30 days.

Within 26 years, Amazon became the world's largest online retailer and a household name. The Amazon name has become synonymous with online shopping and continues to grow by developing new products, acquisitions, and numerous service offerings to enlarge the customer base.

Nowadays, almost 150.6 million people turn to the Amazon app for most everything. Several types of research have proven that customers trust Amazon. On average, the small and medium-sized businesses located in the USA sell more than 4,000 items per minute (Amazon 2019), which leads to millions of product reviews on Amazon.

Reviews tell which products and features are trending, what's in demand, what's no longer relevant, how products and competitors are doing, and much more.

It's observed that a significant number of shoppers look at product reviews before they make a purchase. Survey results show that positive product reviews are a key factor for purchasing by 57 percent of Amazon buyers.

As product reviews are often the deciding factor for many customers, it is therefore, very important to have a well-automated system for monitoring them.

DATA EXPLORATION

Files provided for data analysis:

A) Tool and Home Improvement:
   1) Metadata:
      https://jmcauley.ucsd.edu/data/amazon_v2/metaFiles2/meta_Tools_and_Home_Improvement.json.gz
   2) 5-core:
      https://jmcauley.ucsd.edu/data/amazon_v2/categoryFilesSmall/Tools_and_Home_Improvement_5.json.gz
   3) Ratings only:
      https://jmcauley.ucsd.edu/data/amazon_v2/categoryFilesSmall/Tools_and_Home_Improvement.csv

B) Patio Lawn and Garden:
   1) Metadata:
      https://jmcauley.ucsd.edu/data/amazon_v2/metaFiles2/meta_Patio_Lawn_and_Garden.json.gz
   2) 5-core:
      https://jmcauley.ucsd.edu/data/amazon_v2/categoryFilesSmall/Patio_Lawn_and_Garden_5.json.gz
   3) Ratings only:
      https://jmcauley.ucsd.edu/data/amazon_v2/categoryFilesSmall/Patio_Lawn_and_Garden.csv

DATA DICTIONARY (Same for both Categories):

1) Metadata:
   i) Category: Sub-category the product belongs to
   ii) Tech_1
   iii) Description: Product Description
   iv) Fit
   v) Title: Product Name
   vi) Also_buy: Product Recommendations
   vii) Tech_2
   viii) Brand: Brand the product belongs to
   ix) Feature: Product features
   x) Rank: Product rank
   xi) Also_view: Similar Product Recommendation
   xii) Main_cat: Main Category of the Product
   xiii) Similar_Item: Similar Product
   xiv) Date: Purchase Date
   xv) Price: Price of Product
   xvi) Asin: Product ID
   xvii) Image_url: Image URL of Product
   xviii) Image_url_high_res: Image URL of Product
   xix) Details: Details of the product

2) 5-Core:
   i) Overall: Rating of the product
   ii) Verified: If account is verified or not
   iii) ReviewTime: Time of the review (raw)
   iv) ReviewID: ID of the reviewer
   v) Asin: ID of the product
   vi) Style: A dictionary of the product metadata
   vii) ReviewerName: Name of the reviewer
   viii) ReviewText: Text of the review
   ix) Summary: Summary of the review
   x) UnixReviewTime: Time of the review (unix time)
   xi) Vote: Helpful votes of the review

3) Ratings Only:
   i) Product_ID: Product ID of the product
   ii) Reviewer_ID: Reviewer ID of the customer
   iii) Rating: Rating given by customer
   iv) Date_Time: Date and time of the review given

DATA WRANGLING / CLEANING

What is Data Cleaning and why is it needed?

Data cleaning is the process of fixing or removing incorrect, corrupted, incorrectly formatted, duplicate, or incomplete data within a dataset. When combining multiple data sources, there are many opportunities for data to be duplicated or mislabelled. If data is incorrect, outcomes and algorithms are unreliable, even though they may look correct. There is no one absolute way to prescribe the exact steps in the data cleaning process because the processes will vary from dataset to dataset. But it is crucial to establish a template for your data cleaning process, so you know you are doing it the right way every time.

A) Tools and Home Improvement:

i) Metadata and Ratings only:

- Read metadata
- Check shape:
  `(571535, 19)`
- Check for null values:

```
category              0
tech1            130672
description           0
fit              571088
title                 8
also_buy              0
tech2            571520
brand             16123
feature               0
rank                  0
also_view             0
main_cat           1732
similar_item     342787
date             112869
price            238658
asin                  0
imageURL              0
imageURLHighRes       0
details              91
dtype: int64
```

- Considering only the columns which will be helpful for our analysis:
  'title','brand','main_cat','price','asin'

- Checking null values for the columns taken into consideration:

```
title            8
brand        16123
main_cat      1732
price       238658
asin             0
dtype: int64
```

- Performing manipulations on main_cat column:
  Checking for value_counts:

```
Tools & Home Improvement
433349
Amazon Home
58007
Industrial & Scientific
32273
Sports & Outdoors
14955
Automotive
12056
Office Products
3513
Baby
3487
Arts, Crafts & Sewing
2843
Toys & Games
1483
<img src="https://images-na.ssl-images-amazon.com/images/G/01/nav2/images/gui/amazon-fashion-store-new._CB520838675_.png" class
="nav-categ-image" alt="AMAZON FASHION"/>       1199
All Electronics
1055
All Beauty
985
Health & Personal Care
784
```

```
Home Audio & Theater
722
Musical Instruments
556
Computers
535
Pet Supplies
475
Camera & Photo
467
Cell Phones & Accessories
438
Car Electronics
166
Grocery
102
Appliances
66
Books
66
Video Games
57
<img src="https://images-na.ssl-images-amazon.com/images/G/01/handmade/brand/logos/2018/subnav_logo._CB502360610_.png" class="n
av-categ-image" alt="Handmade"/>                  36
Software
34
Portable Audio & Accessories
26
Movies & TV
22
GPS & Navigation
20
<img src="https://images-na.ssl-images-amazon.com/images/G/01/digital/music/logos/amzn_music_logo_subnav._CB471835632_.png" cla
ss="nav-categ-image" alt="Digital Music"/>        12
```

- Considering only those sub-categories which are relevant to Tools and Home Improvement and dropping the rest. The sub-categories taken into consideration are:
  i) Tools and home improvement
  ii) Amazon home
  iii) Industrial and scientific
  iv) Automotive
  v) Arts, crafts and sewing
  vi) All electronics
  vii) Computers
  viii) Appliances
- Now checking for null values with respect to the different sub-categories:

```
Tools & Home Improvement
(433349, 5)
title            5
brand         9895
main_cat         0
price       183816
asin             0
dtype: int64
**************************************************
Amazon Home
(58007, 5)
title            1
brand         4070
main_cat         0
price        25685
asin             0
dtype: int64
**************************************************
Industrial & Scientific
(32273, 5)
title            0
brand          264
main_cat         0
price         7694
asin             0
dtype: int64
**************************************************
Automotive
(12056, 5)
title            0
brand          130
main_cat         0
price         4009
asin             0
dtype: int64
**************************************************
Arts, Crafts & Sewing
(2843, 5)
title            0
brand           80
main_cat         0
price         1029
asin             0
dtype: int64
**************************************************
```

```
All Electronics
(1055, 5)
title          0
brand         13
main_cat       0
price        604
asin           0
dtype: int64
**************************************************
Computers
(535, 5)
title          0
brand          4
main_cat       0
price        302
asin           0
dtype: int64
**************************************************
Appliances
(66, 5)
title         0
brand         2
main_cat      0
price        34
asin          0
dtype: int64
**************************************************
```

- Categories like Computers and All electronics have plenty of null values in their price column (>50%) which is an essential factor to take into consideration for analysis. Therefore, we shall remove those sub-categories as well.
- A check for any noise values among the brand column was performed, but no noise values were found.
- Therefore, we proceed to remove all null values from data frame which belonged to the brand subset.
- A check for any noise values among the title column was performed, but no noise values were found.
- Therefore, we proceed to remove all null values from data frame which belonged to the title subset.
- Checking for null values again and observing the dtypes of the columns in our data frame.

```
In [29]: df6.isnull().sum()

Out[29]: title            0
         brand            0
         main_cat         0
         price       209341
         asin             0
         dtype: int64

In [30]: df6.dtypes

Out[30]: title       object
         brand       object
         main_cat    object
         price       object
         asin        object
         dtype: object
```

- An observation was made on the price column and we found that it was of an object datatype with prices being denoted as: '$115.00' for example. There was also presence of URL links within our price column.
- We therefore performed certain string operations before converting our price column to float dtype.

```
In [31]: df6['price']=df6['price'].fillna('0')

In [32]: df6['price'].isnull().sum()

Out[32]: 0

In [33]: df6['price']=df6['price'].str.replace('$','')

In [34]: df6['price']=df6['price'].str.replace(',','')

In [35]: p=[]
         for i in df6['price']:
             if len(i)<10:
                 p.append(i)
             else:
                 p.append('0')
         p

In [36]: df6['price']=p

In [38]: df6.dtypes

Out[38]: title       object
         brand       object
         main_cat    object
         price       object
         asin        object
         dtype: object

In [39]: df6['price']=df6['price'].astype('float64')
```

- We now perform imputation on our price column with the category wise price mean for prices labelled as 0.00:

In [47]:
```
df_THI=df6[(df6['main_cat']=='Tools & Home Improvement')&(df6['price']!=0.00)]
print(df_THI['price'].mean())
df_AH=df6[(df6['main_cat']=='Amazon Home')&(df6['price']!=0.00)]
print(df_AH['price'].mean())
df_IS=df6[(df6['main_cat']=='Industrial & Scientific')&(df6['price']!=0.00)]
print(df_IS['price'].mean())
df_A=df6[(df6['main_cat']=='Automotive')&(df6['price']!=0.00)]
print(df_A['price'].mean())
df_ACS=df6[(df6['main_cat']=='Arts, Crafts & Sewing')&(df6['price']!=0.00)]
print(df_ACS['price'].mean())
df_AP=df6[(df6['main_cat']=='Appliances')&(df6['price']!=0.00)]
print(df_AP['price'].mean())
```

```
45.902273820588185
39.54119254461378
74.16798130222065
38.55676578514486
22.251686340640994
48.150000000000006
```

In [48]:
```
THI=df_THI['price'].mean()
AH=df_AH['price'].mean()
IS=df_IS['price'].mean()
A=df_A['price'].mean()
ACS=df_ACS['price'].mean()
AP=df_AP['price'].mean()
```

In [51]:
```
df_THI2=df6[df6['main_cat']=='Tools & Home Improvement']
df_AH2=df6[df6['main_cat']=='Amazon Home']
df_IS2=df6[df6['main_cat']=='Industrial & Scientific']
df_A2=df6[df6['main_cat']=='Automotive']
df_ACS2=df6[df6['main_cat']=='Arts, Crafts & Sewing']
df_AP2=df6[df6['main_cat']=='Appliances']
```

In [52]:
```
df_THI2['price'].replace(to_replace=0.00,value=THI,inplace=True)
df_AH2['price'].replace(to_replace=0.00,value=AH,inplace=True)
df_IS2['price'].replace(to_replace=0.00,value=IS,inplace=True)
df_A2['price'].replace(to_replace=0.00,value=A,inplace=True)
df_ACS2['price'].replace(to_replace=0.00,value=ACS,inplace=True)
df_AP2['price'].replace(to_replace=0.00,value=AP,inplace=True)
```

In [53]:
```
df_meta_concat_tools=pd.concat([df_THI2,df_AH2,df_IS2,df_A2,df_ACS2,df_AP2],axis=0)
```

- We now don't have any null values in our data and is now ready.
- Now, we read our Ratings only file. Since, there were no headers in our file we must manually add them.

In [56]:
```
header_info = ['product_id', 'Reviewer_id', 'rating', 'date_time']
df_ToolsAndImprovementRating=pd.read_csv('Tools_and_Home_Improvement (1).csv',names=header_info,header=None)
```

- Since our Ratings only file has no null values, we can proceed to merge the two files.

In [60]:
```
tools=pd.merge(df_meta_concat_tools,df_ToolsAndImprovementRating,on='product_id',how='inner')
```

ii) 5-core file:

- Reading the file:

```
In [4]: Tools_and_Home_Improvement_5.head()
```

Out[4]:

	overall	verified	reviewTime	reviewerID	asin	style	reviewerName	reviewText	summary	unixReviewTime	vote	image
0	5.0	True	01 28, 2018	AL19QO4XLBQPU	0982085028	{'Style:': ' 1) IR30 POU (30A/3.4kW/110v)'}	J. Mollenkamp	returned, decided against this product	Five Stars	1517097600	NaN	NaN
1	5.0	True	11 30, 2017	A1I7CVB7X3T81E	0982085028	{'Style:': ' 3) IR260 POU (30A/6kW/220v)'}	warfam	Awesome heater for the electrical requirements...	Five Stars	1512000000	NaN	NaN
2	5.0	True	09 12, 2017	A1AQXO4P5U674E	0982085028	{'Style:': ' Style64'}	gbieber2	Keeps the mist of your wood trim and on you. B...	Five Stars	1505174400	NaN	NaN
3	4.0	True	07 19, 2017	AIRV678P7C4NK	0982085028	NaN	Justin Banner	So far I hooked it up and tested it , filled a...	it is the perfect temp for a shower	1500422400	NaN	NaN
4	1.0	True	05 25, 2017	A22I5QDNTNECDW	0982085028	{'Style:': ' 3) IR260 POU (30A/6kW/220v)'}	daveparker	i installed this 10 months ago, instructions w...	worked well...for 10 months.	1495670400	16	NaN

- Taking only the necessary columns into consideration which include: "overall","verified","reviewerID","asin","reviewText"
- Checking for null values in our data:

```
In [10]: Tools_and_Home_Improvement_5.isnull().sum()

Out[10]: overall         0
         verified        0
         reviewerID      0
         asin            0
         reviewText    522
         dtype: int64
```

- Null values were dropped as we cannot impute review text.
- Now we rename the our asin, overall and reviewer_id column to follow a single syntax across all our files.

```
In [20]: Tools_and_Home_Improvement_5.rename(columns={'asin':'product_id'},inplace=True)

In [21]: Tools_and_Home_Improvement_5.rename(columns={'overall':'rating'},inplace=True)

In [22]: Tools_and_Home_Improvement_5.rename(columns={'reviewerID':'Reviewer_id'},inplace=True)
```

- Our 5-core file is now ready for merging with our merged file of metadata and ratings only. We perform an inner merge with respect to product id,

```
Out[26]: (8537571, 8)

In [28]: tools_core_merged_rating_csv=pd.merge(tools,Tools_and_Home_Improvement_5,on=('product_id',"Reviewer_id","rating"),how='inner')
```

- Checking our data obtained as a result of merging.

	title	brand	main_cat	price	product_id	Reviewer_id	rating	date_time	verified	reviewText
0	Breeding Organic Vegetables: A Step-by-Step Gu...	SioGreen	Tools & Home Improvement	45.902274	0982085028	AL19QO4XLBQPU	5.0	1517097600	True	returned, decided against this product
1	Breeding Organic Vegetables: A Step-by-Step Gu...	SioGreen	Tools & Home Improvement	45.902274	0982085028	A1I7CVB7X3T81E	5.0	1512000000	True	Awesome heater for the electrical requirements...
2	Breeding Organic Vegetables: A Step-by-Step Gu...	SioGreen	Tools & Home Improvement	45.902274	0982085028	A1AQXO4P5U674E	5.0	1505174400	True	Keeps the mist of your wood trim and on you. B...
3	Breeding Organic Vegetables: A Step-by-Step Gu...	SioGreen	Tools & Home Improvement	45.902274	0982085028	AIRV678P7C4NK	4.0	1500422400	True	So far I hooked it up and tested it , filled a...
4	Breeding Organic Vegetables: A Step-by-Step Gu...	SioGreen	Tools & Home Improvement	45.902274	0982085028	A22I5QDNTNECDW	1.0	1495670400	True	i installed this 10 months ago, instructions w...

- Checking for null values:

```
In [30]: tools_core_merged_rating_csv.isnull().sum()

Out[30]: title          0
         brand          0
         main_cat       0
         price          0
         product_id     0
         Reviewer_id    0
         rating         0
         date_time      0
         verified       0
         reviewText     0
         dtype: int64
```

- Converting date_time to datetime dtype:

```
In [33]: def date_time(epoch_time):
             date_time = datetime.datetime.fromtimestamp( epoch_time )
             return date_time
```

```
In [34]: tools_core_merged_rating_csv['date_time'] = tools_core_merged_rating_csv['date_time'].apply(date_time)
```

- Our Data is cleaned and merged appropriately and is now ready for modelling.

A similar data cleaning procedure was performed for our Patio and Lawn Garden data.

EXPLORATORY DATA ANALYSIS (EDA)

What is EDA and why is it needed?

Exploratory data analysis (EDA) is used by data scientists to analyze and investigate data sets and summarize their main characteristics, often employing data visualization methods. It helps determine how best to manipulate data sources to get the answers you need, making it easier for data scientists to discover patterns, spot anomalies, test a hypothesis, or check assumptions.

The main purpose of EDA is to help look at data before making any assumptions. It can help identify obvious errors, as well as better understand patterns within the data, detect outliers or anomalous events, find interesting relations among the variables.

Data scientists can use exploratory analysis to ensure the results they produce are valid and applicable to any desired business outcomes and goals. EDA also helps stakeholders by confirming they are asking the right questions. EDA can help answer questions about standard deviations, categorical variables, and confidence intervals. Once EDA is complete and insights are drawn, its features can then be used for more sophisticated data analysis or modelling, including machine learning.

Using EDA we have solved a wide variety of business questions for Tools & Home Improvement and Patio Lawn and Garden category. We shall discuss them in detail.

A) Tools & Home Improvement

1) Analysis on style column

Even though we observe that the style column had plenty of missing values, EDA was conducted to observe trends among the existing values.

```
In [218]: # Dropping null values to perform analysis on the style column
          df_style_eda=df_tools_and_home.dropna()

In [219]: df_style_eda['style']

Out[219]: 0                {'Style:': ' 1) IR30 POU (30A/3.4kW/110v)'}
          1                 {'Style:': ' 3) IR260 POU (30A/6kW/220v)'}
          2                                    {'Style:': ' Style64'}
          4                 {'Style:': ' 3) IR260 POU (30A/6kW/220v)'}
          5                 {'Style:': ' 3) IR260 POU (30A/6kW/220v)'}
                                             ...
          2070817    {'Size:': ' 1 Pack', 'Color:': ' 4000k (Daylig...
          2070818    {'Size:': ' 1 Pack', 'Color:': ' 4000k (Daylig...
          2070819    {'Size:': ' 1 Pack', 'Color:': ' 4000k (Daylig...
          2070820    {'Size:': ' 1 Pack', 'Color:': ' 4000k (Daylig...
          2070829                              {'Color:': ' warm white'}
          Name: style, Length: 1073363, dtype: object

In [220]: # Checking type of data in style column
          for i in df_style_eda['style'][:3]:
              print(type(i))

          <class 'str'>
          <class 'str'>
          <class 'str'>
```

We observe that our style column is in a dictionary format but is of string type. We must therefore, first convert it to dictionary type first by using the ast module.

What is the ast library and what does it do?

The ast module helps Python applications to process trees of the Python abstract syntax grammar. The abstract syntax itself might change with each Python release; this module helps to find out programmatically what the current grammar looks like.

```
In [221]: # Converting string to dictionary and storing them in a list
          import ast
          style_dict=[]
          for i in df_style_eda['style']:
              b=ast.literal_eval(i)
              style_dict.append(b)
          style_dict
```

```
Out[221]: [{'Style:': ' 1) IR30 POU (30A/3.4kW/110v)'},
           {'Style:': ' 3) IR260 POU (30A/6kW/220v)'},
           {'Style:': ' Style64'},
           {'Style:': ' 3) IR260 POU (30A/6kW/220v)'},
           {'Style:': ' 3) IR260 POU (30A/6kW/220v)'},
           {'Style:': ' 3) IR260 POU (30A/6kW/220v)'},
           {'Style:': ' 3) IR260 POU (30A/6kW/220v)'},
           {'Style Name:': ' Style17'},
           {'Style:': ' 3) IR260 POU (30A/6kW/220v)'},
           {'Style:': ' Style69'},
           {'Style:': ' 3) IR260 POU (30A/6kW/220v)'},
           {'Style:': ' Style20'},
           {'Style:': ' 3) IR260 POU (30A/6kW/220v)'},
           {'Style:': ' Style41'},
           {'Style:': ' 3) IR260 POU (30A/6kW/220v)'},
           {'Style:': ' Style24'},
           {'Style:': ' Style20'},
           {'Style:': ' Style20'},
           {'Style:': ' Style56'},
```

We now observe the style parameters available in our data.

Checking for different style parameters from our Dictionaries

```
In [222]: # Checking for Style parameters
          features=[]
          for i in style_dict:
              for j,k in i.items():
                  features.append(j)
          set(features)
```

```
Out[222]: {'Bore Diameter:',
           'Capacity:',
           'Color Name:',
           'Color:',
           'Configuration:',
           'Connector Type:',
           'Design:',
           'Format:',
           'Gauge:',
           'Grit Type:',
           'Hand Orientation:',
           'Inside Diameter:',
           'Item Display Length:',
           'Item Package Quantity:',
           'Item Shape:',
           'Length:',
           'Line Weight:',
           'Material Type:',
           'Material:',
```

We shall consider only the style parameters whuch occur frequently i.e. : style, format, color, size, wattage

```
In [223]: style_count,format_count,color_count,size_count,wattage_count=[],[],[],[],[]
```

For our analysis we will be conducting EDA on only those style parameters which were occurring frequently in our data. The following parameters were taken into consideration:

i) Style
ii) Format
iii) Color
iv) Size
v) Wattage

The details of the styles were segregated and placed in a data frame as such:

```
In [250]: for i in style_dict:
              for j,k in i.items():
                  if (j=='Color:')|(j=='Color Name:'):
                      color_count.append(k)
                  else:
                      color_count.append(np.nan)

                  if j=='Format:':
                      format_count.append(k)
                  else:
                      format_count.append(np.nan)

                  if (j=='Style:')|(j=='Style Name:'):
                      style_count.append(k)
                  else:
                      style_count.append(np.nan)

                  if (j=='Size:')|(j=='Size Name:'):
                      size_count.append(k)
                  else:
                      size_count.append(np.nan)

                  if j=='Wattage:':
                      wattage_count.append(k)
                  else:
                      wattage_count.append(np.nan)
```

```
In [252]: # Storing in a dataframe to conduct EDA
          df_style=pd.DataFrame()
          df_style['Color']=color_count
          df_style['Format']=format_count
          df_style['Style']=style_count
          df_style['Size']=size_count
          df_style['Wattage']=wattage_count
```

On, viewing our columns in the df_style data frame we observe that there is need to perform some noise treatment in our color, size and wattage columns.

i) Our color column contained numbers for example: 293, 294, 11-D.
ii) There was no defined format in which data existed in our size column. For example: ‘1-pack’ being represented as '1 Pack','1-(Pack)',' Pack of 1',' 1'.
iii) Our values in wattage column had a similar issue. For example: 50 being represented as 50.0,50.00,50W.

The above issues were solved by the following pieces of code:

```
In [255]: c=[]
          for i in df_style['Color']:
              str1=str(i)
              if any(map(str.isdigit,str1))==True:
                  c.append(np.nan)
              else:
                  c.append(i)
```

```
In [256]: df_style['Color']=c
```

```
In [257]: c1=[]
          for i in df_style['Color']:
              c1.append(i)
          set(c1)
```

(to remove numbers from our color column)

```
In [258]: df_style['Size'].replace(to_replace=[' 1 Pack',' 1-(Pack)',' Pack of 1',' 1',' 1-Pack'],value='1-Pack',inplace=True)

In [ ]:
```

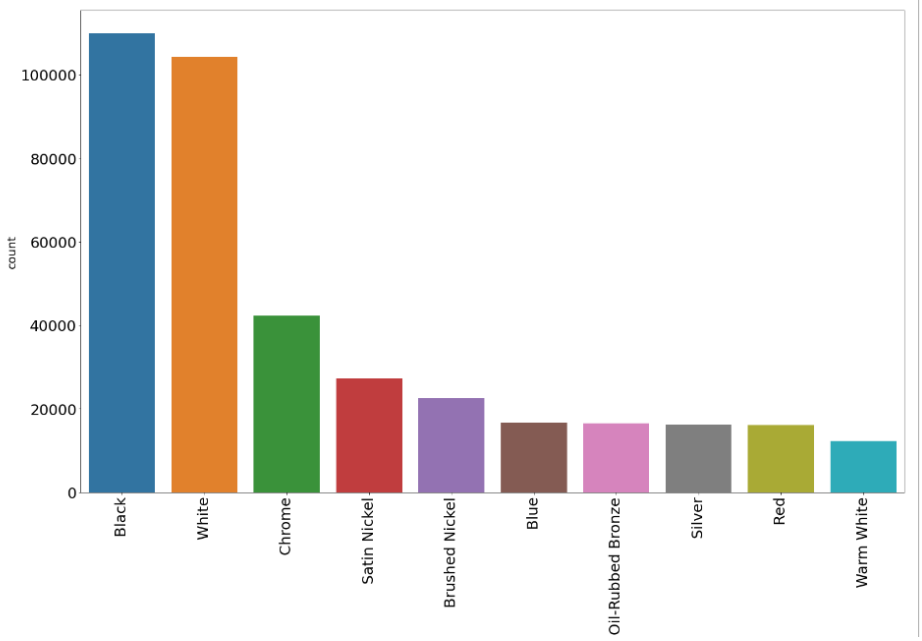
(to make a single annotation in size column)

```
In [262]: w3=[]
          for i in df_style['Wattage']:
              if i!=np.nan:
                  str_w=remove(str(i))
                  w3.append(str_w)
              else:
                  w3.append(i)

In [269]: df_style['Wattage']=w3

In [279]: df_style['Wattage']=df_style['Wattage'].str.replace('[wW]','')

In [280]: df_style['Wattage']=df_style['Wattage'].astype('float64')
```

Now, that all noise is removed from our style column, we shall solve some business questions with respect to our style column.

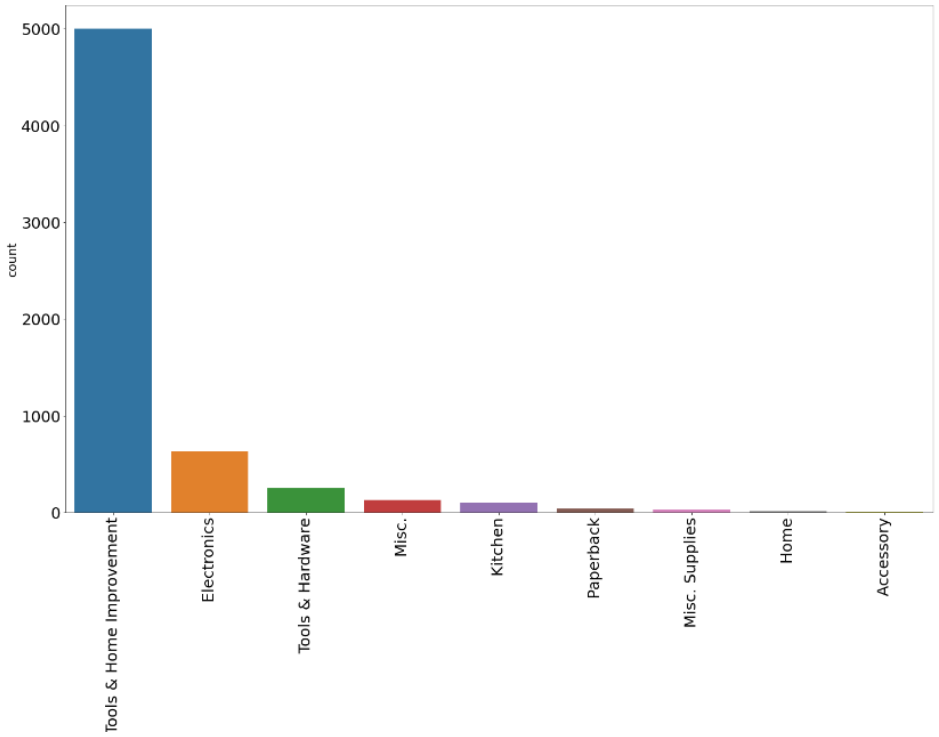
a) What are the top 10 color preferences of customers in the Tools and Home Improvement Category

```
In [259]: plt.figure(figsize=(25,15))
          plt.xlabel('Style',fontsize=20)
          plt.ylabel('Count',fontsize=20)
          plt.yticks(fontsize=25)
          plt.xticks(rotation=90,fontsize=25)
          sb.countplot(x=df_style['Color'],order=df_style['Color'].value_counts().iloc[:10].index)

Out[259]: <AxesSubplot:xlabel='Color', ylabel='count'>
```

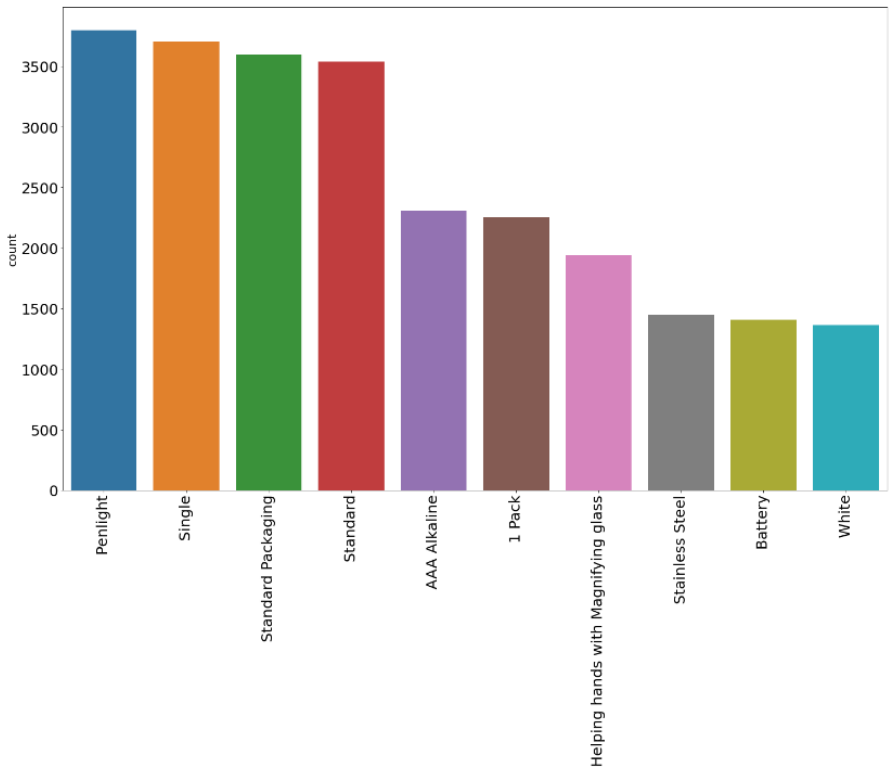


b) What are the top 10 format preferences of customers in the Tools and Home Improvement Category

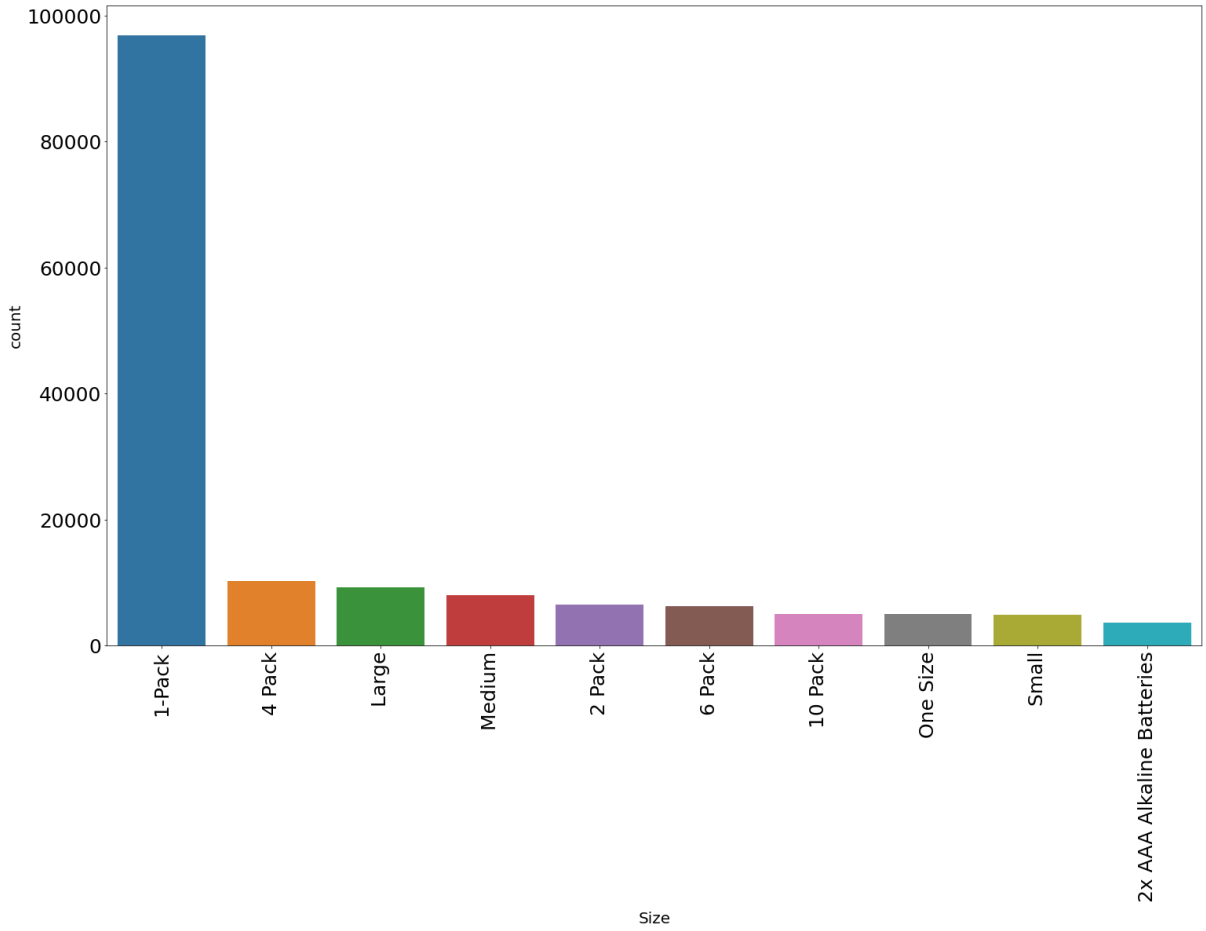
```
In [148]: plt.figure(figsize=(25,15))
          plt.xlabel('Format',fontsize=20)
          plt.ylabel('Count',fontsize=20)
          plt.yticks(fontsize=25)
          plt.xticks(rotation=90,fontsize=25)
          sb.countplot(x=df_style['Format'],order=df_style['Format'].value_counts().iloc[:10].index)

Out[148]: <AxesSubplot:xlabel='Format', ylabel='count'>
```



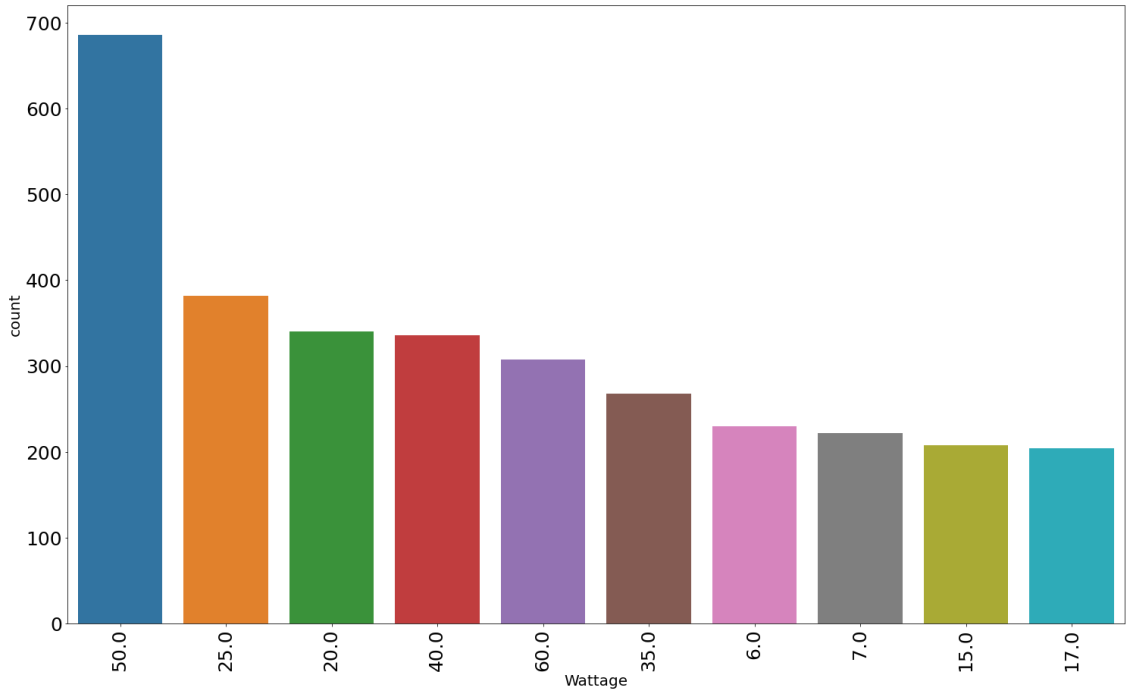
c) What are the top 10 style preferences of customers in the Tools and Home Improvement Category



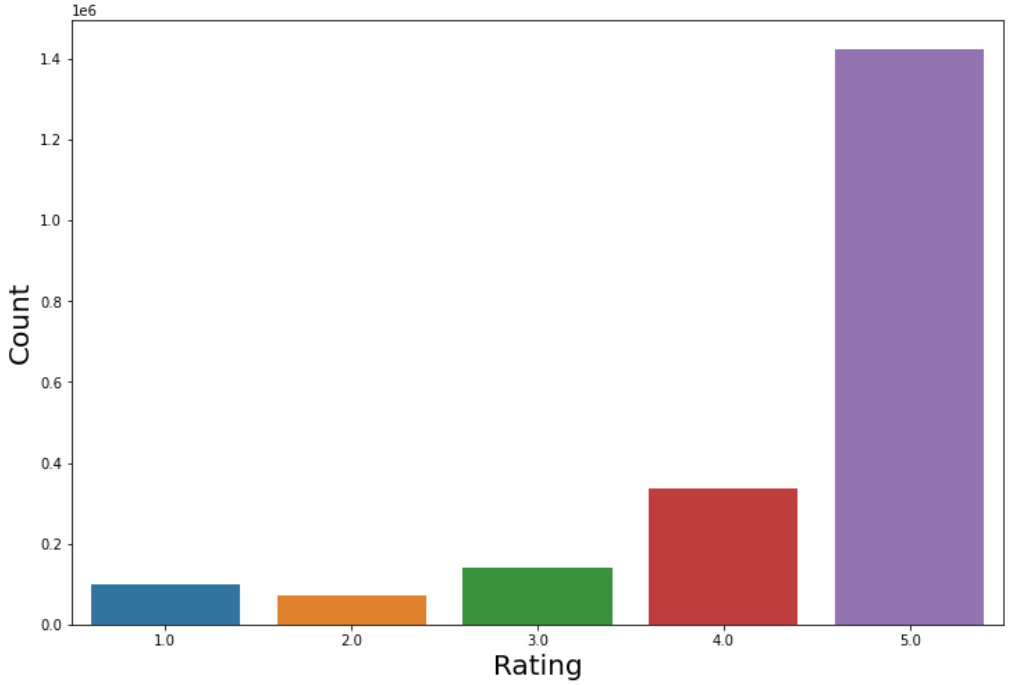
d) What are the top 10 Size preferences of customers in the Tools and Home Improvement Category



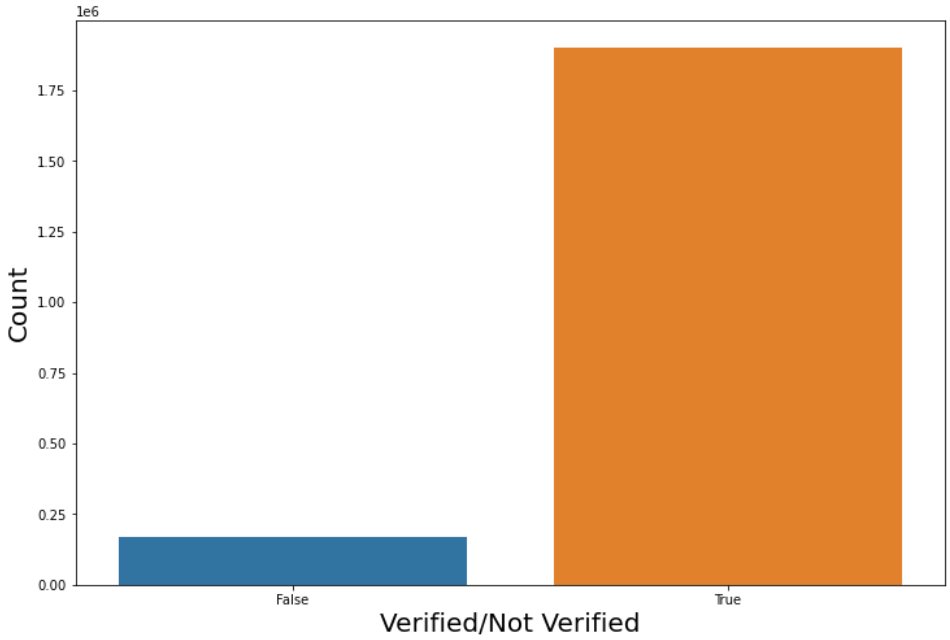
e) What are the top 10 wattage preferences of customers in the Tools and Home Improvement Category



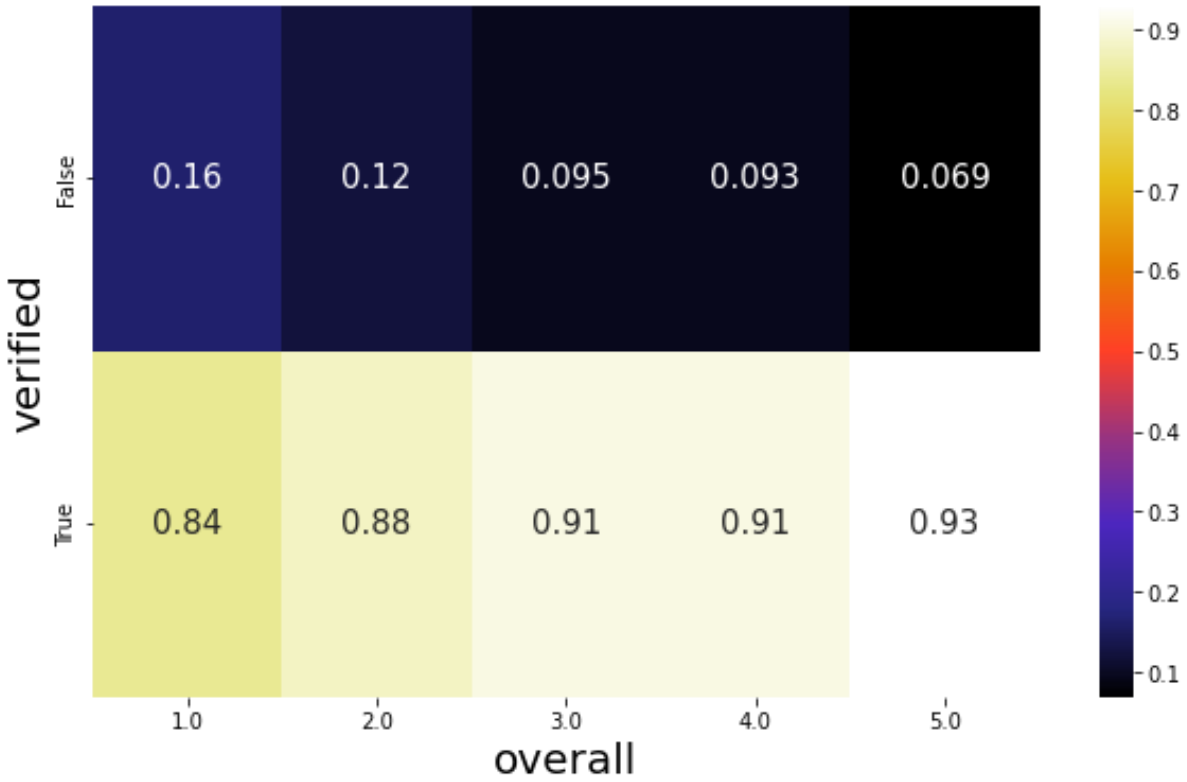
2) How is the count distribution across different ratings in the Tools and Home Improvement Category



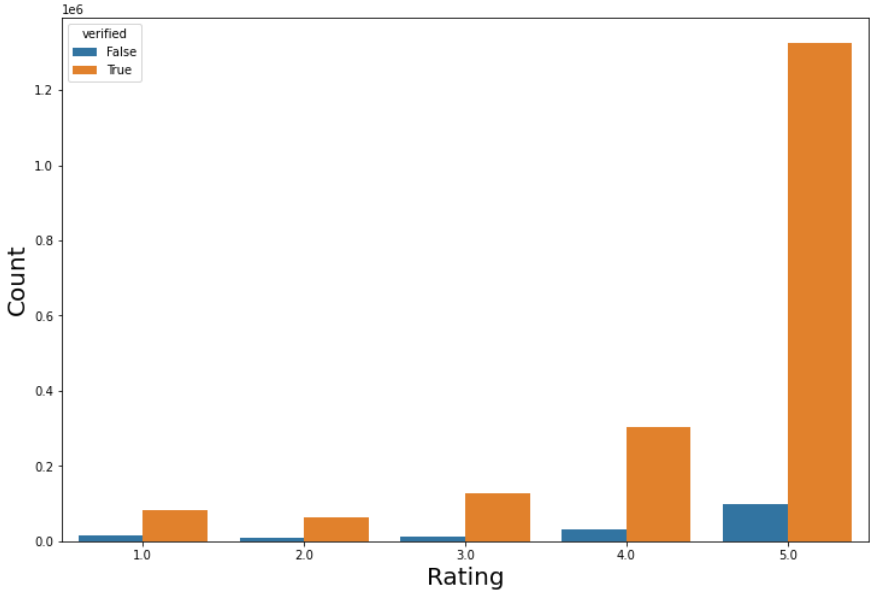
3) How is the count distribution across Verified/Not Verified customers in the Tools and Home Improvement Category



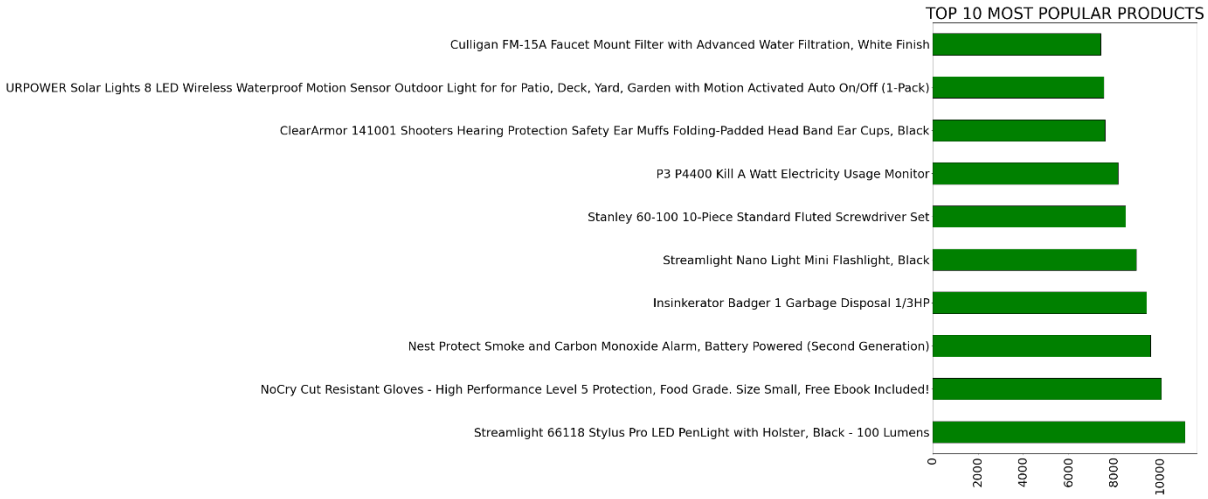
4) How is the percentage distribution of Verified/Not Verified customers across different ratings in the Tools and Home Improvement Category



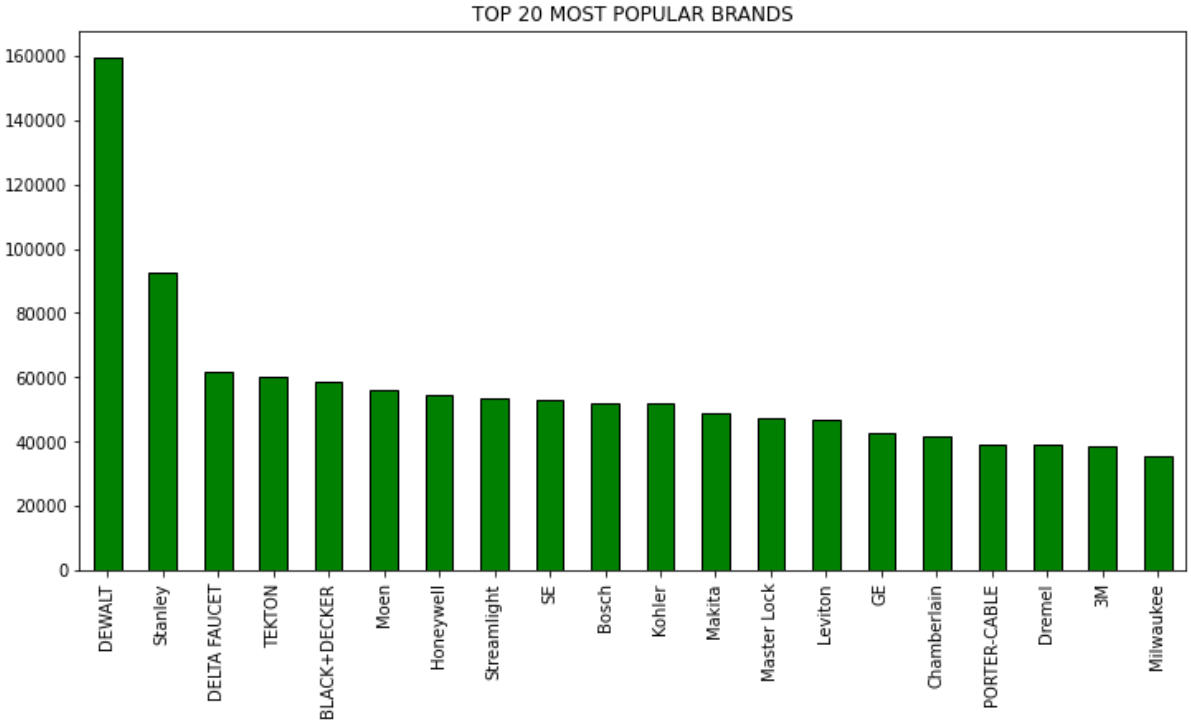
5) How is the count distribution of Verified/Not Verified customers with respect to ratings in the Tools and Home Improvement Category



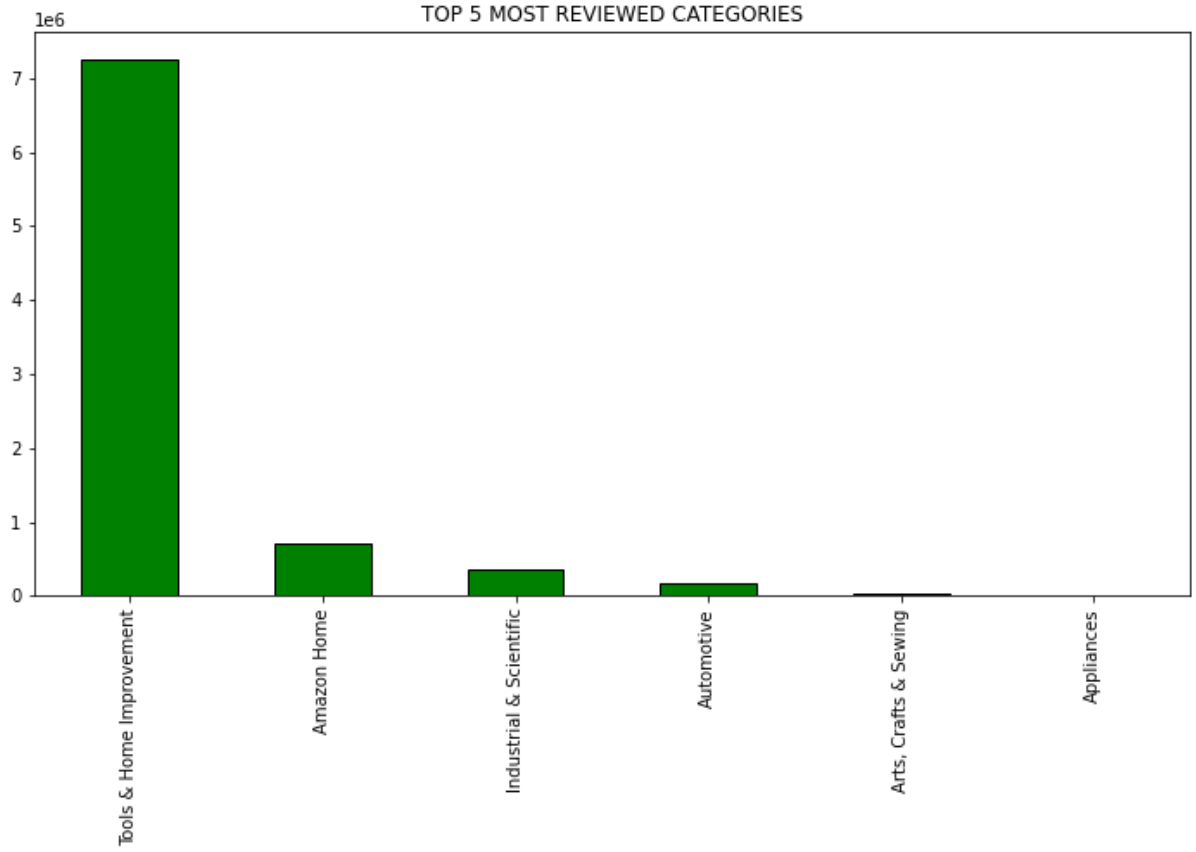
6) What are the top 10 most popular Products among the category of Tools and Home Improvement?



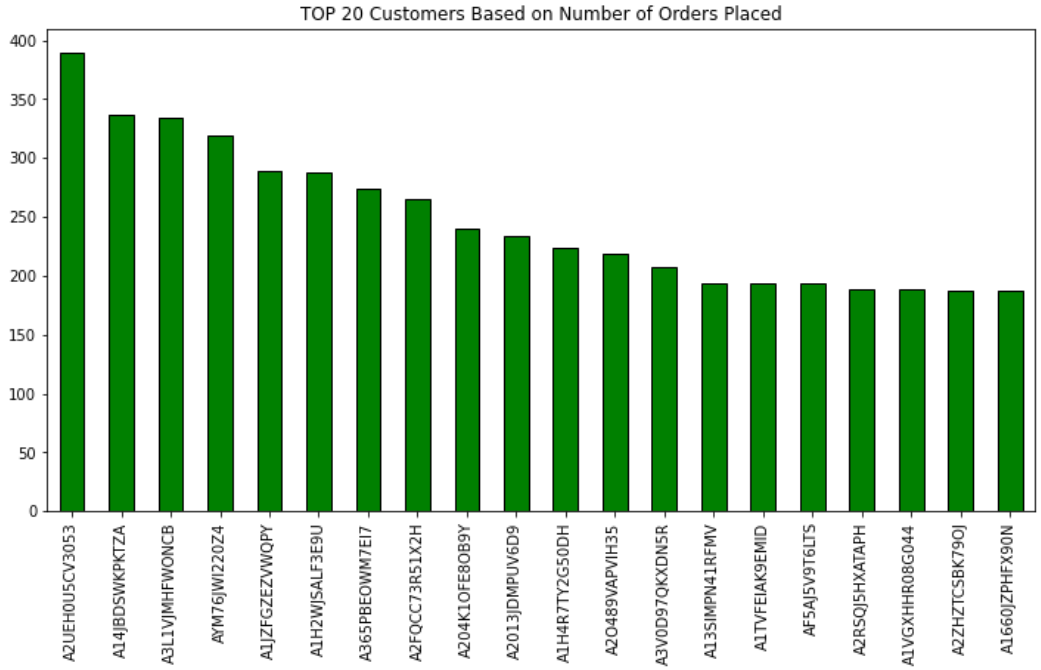
7) What are the top 20 most popular Brands among the category of Tools and Home Improvement?



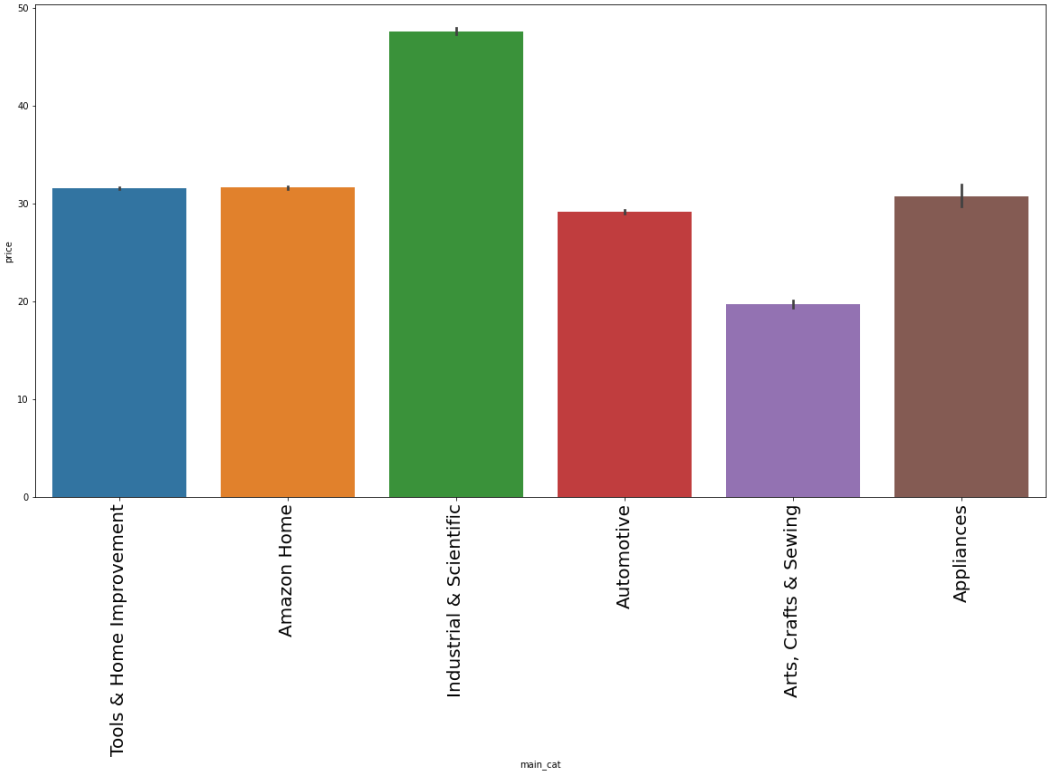
8) Identify the Categories in decreasing order of review count



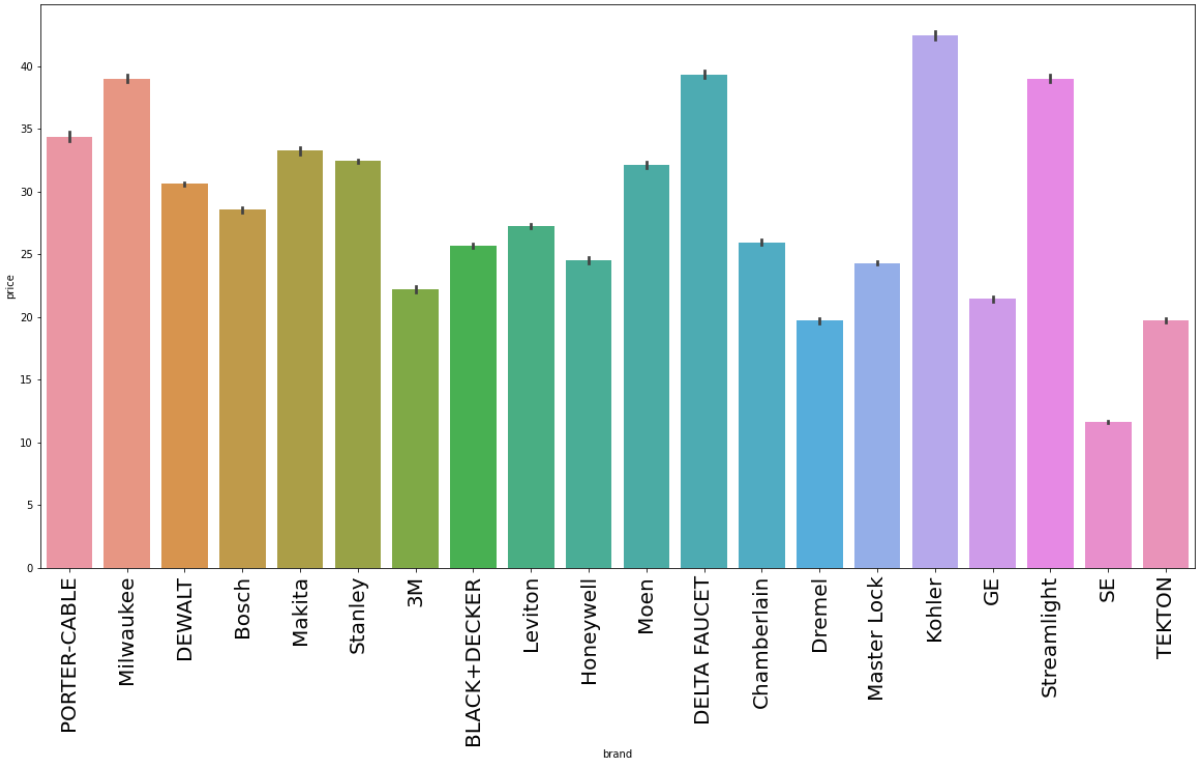
9) Identify the customers who have provided the maximum reviews in the Tools & Home Improvement Category



10) Represent the Spread of Price with respect to the main categories for Tools and Home Improvement



11) Represent the Spread of Price with respect to the top 20 most Popular Brands



Now, we shall conduct some EDA based on sales with respect to time.
In order to do that we must first convert our unixtime to proper date format.

In [68]:
```
import datetime
```

In [69]:
```
def date_time(epoch_time):
    date_time=datetime.datetime.fromtimestamp(epoch_time)
    return date_time
```

In [70]:
```
df_tools_improvement_time['date_time']=df_tools_improvement_time['date_time'].apply(date_time)
```

In [71]:
```
df_tools_improvement_time.dtypes
```

Out[71]:
```
title                  object
brand                  object
main_cat               object
price                 float64
product_id             object
Reviewer_id            object
rating                float64
date_time      datetime64[ns]
dtype: object
```

In [72]:
```
df_tools_improvement_time['Date']=pd.to_datetime(df_tools_improvement_time['date_time']).dt.date
```

In [73]:
```
df_tools_improvement_time.head()
```

Out[73]:

	title	brand	main_cat	price	product_id	Reviewer_id	rating	date_time	Date
0	Diary of a Wimpy Kid Book Light	Barnes & Noble	Tools & Home Improvement	45.902274	0594510384	ASRI9DDKDTPG5	5.0	2016-08-10 05:30:00	2016-08-10
1	Diary of a Wimpy Kid Book Light	Barnes & Noble	Tools & Home Improvement	45.902274	0594510384	A8PZYSNLD2OJ6	5.0	2016-01-18 05:30:00	2016-01-18
2	Rob Cosman "Hand-Cut Dovetails" DVD	ROB COSMAN	Tools & Home Improvement	45.902274	0980941210	AY1J16313XE0Z	1.0	2015-11-12 05:30:00	2015-11-12
3	Rob Cosman Hand-Cut Dovetails 2.0 DVD	RC Woodworking Hand Tools Inc.	Tools & Home Improvement	45.902274	0980966078	A2IZH42OH108W4	5.0	2015-06-20 05:30:00	2015-06-20
4	Rob Cosman Hand-Cut Dovetails 2.0 DVD	RC Woodworking Hand Tools Inc.	Tools & Home Improvement	45.902274	0980966078	A1ANA3WGU5R7NA	4.0	2013-07-30 05:30:00	2013-07-30

In [74]:
```
# Converting Date column to datetime dtype
df_tools_improvement_time['Date']=pd.to_datetime(df_tools_improvement_time['Date'])
```

In [76]:
```
# Sorting values in terms of date
df_tools_improvement_time.sort_values(by='Date',inplace=True)
```

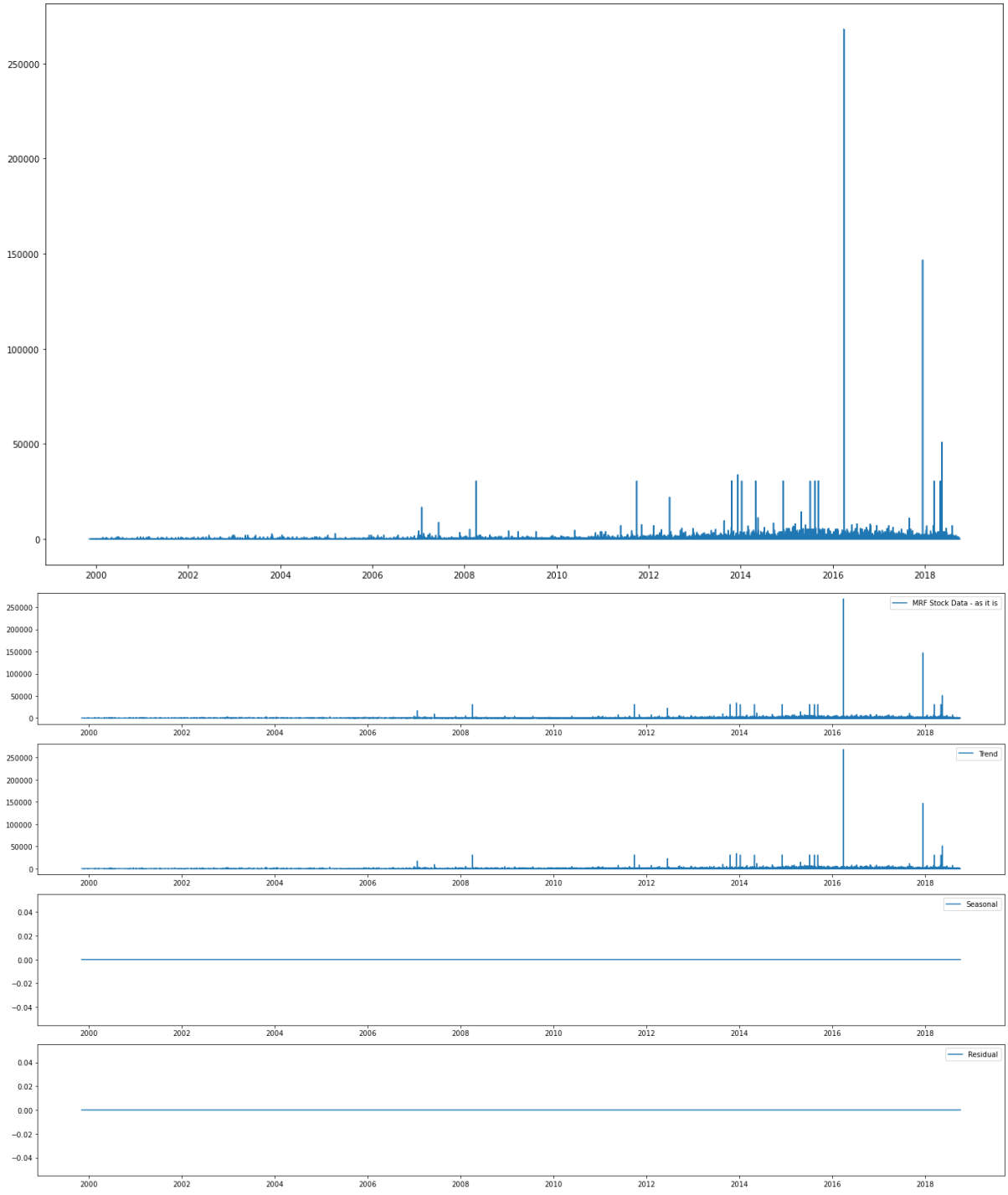
In [77]:
```
# Setting date as index
df_tools_improvement_time=df_tools_improvement_time.set_index('Date')
df_tools_improvement_time
```

Out[77]:

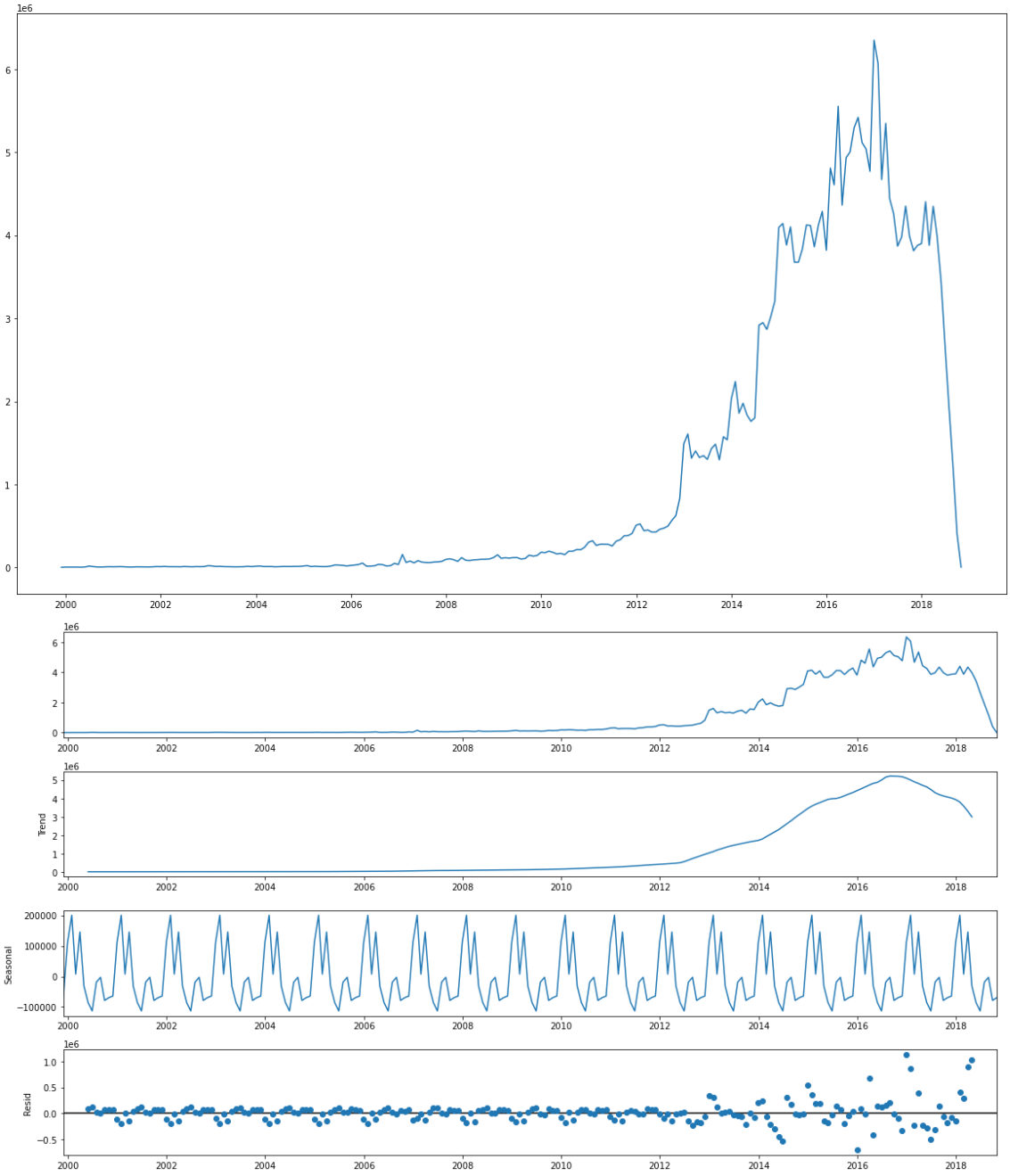
Date	title	brand	main_cat	price	product_id	Reviewer_id	rating	date_time
1999-11-08	Moen 87540 One-Handle Kitchen Faucet	Moen	Tools & Home Improvement	45.902274	B00002N6NB	A1JSO7PPEA0W72	5.0	1999-11-08 05:30:00
1999-11-08	Black & Decker 9074CTN 3.6-Volt Cordless S...	BLACK+DECKER	Tools & Home Improvement	45.902274	B0000302UJ	A1JSO7PPEA0W72	5.0	1999-11-08 05:30:00
1999-11-08	Stanley 15-087 20-Inch, 8-Point Contractor Gra...	Stanley	Tools & Home Improvement	23.610000	B00002X20X	A1B3GNO9C8YX0N	5.0	1999-11-08 05:30:00
1999-11-10	Makita 6233DWBLE 3/8-Inch 14.4 Volt Drill/Driv...	Makita	Tools & Home Improvement	45.902274	B00002269X	A2DGRJC2K15CG8	5.0	1999-11-10 05:30:00
1999-11-10	Moen 87540 One-Handle Kitchen Faucet	Moen	Tools & Home Improvement	45.902274	B00002N6NB	A3DY54YYU3X9HX	4.0	1999-11-10 05:30:00
...	...	...	...	...	...	...	...	...
2018-10-04	Ekena Millwork MIC03X03BI-CASE-4 3-1/8" P...	Ekena Millwork	Tools & Home Improvement	30.000000	B01HI9FU50	A2ONRSPKBWPKHH	5.0	2018-10-04 05:30:00
2018-10-04	Franklin Brass W35074-CPS-C Classic Lace Switc...	Franklin Brass	Tools & Home Improvement	8.660000	B01HGNPS64	A3PN1PTVXRUE4G	5.0	2018-10-04 05:30:00
2018-10-04	Hotis Modern Spring 360 degree Swivel Pull Out...	HOTIS HOME	Tools & Home Improvement	4.960000	B01HJ5TTE6	A1FSJYXAIJEXIB	5.0	2018-10-04 05:30:00
2018-10-04	Portfolio 6.25-in Amber Single Textured Glass ...	Portfolio	Tools & Home Improvement	2.050000	B01HIIYDTU	A26HSGFDFYDGRL	5.0	2018-10-04 05:30:00
2018-10-05	Hotis Modern Spring 360 degree Swivel Pull Out...	HOTIS HOME	Tools & Home Improvement	4.960000	B01HJ5TTE6	A2W1EDB3RU2FJ2	4.0	2018-10-05 05:30:00

7877236 rows × 8 columns

12) Represent the Sales in the Tools and Home Improvement category with respect to time. Also identify if there is any trend or seasonality in the data.



13) Represent the Sales in the Tools and Home Improvement category with respect to time (Monthly). Also identify if there is any trend or seasonality in the data




```
In [287]: time_date_tools_monthly[time_date_tools_monthly['price']==time_date_tools_monthly['price'].max()]
```

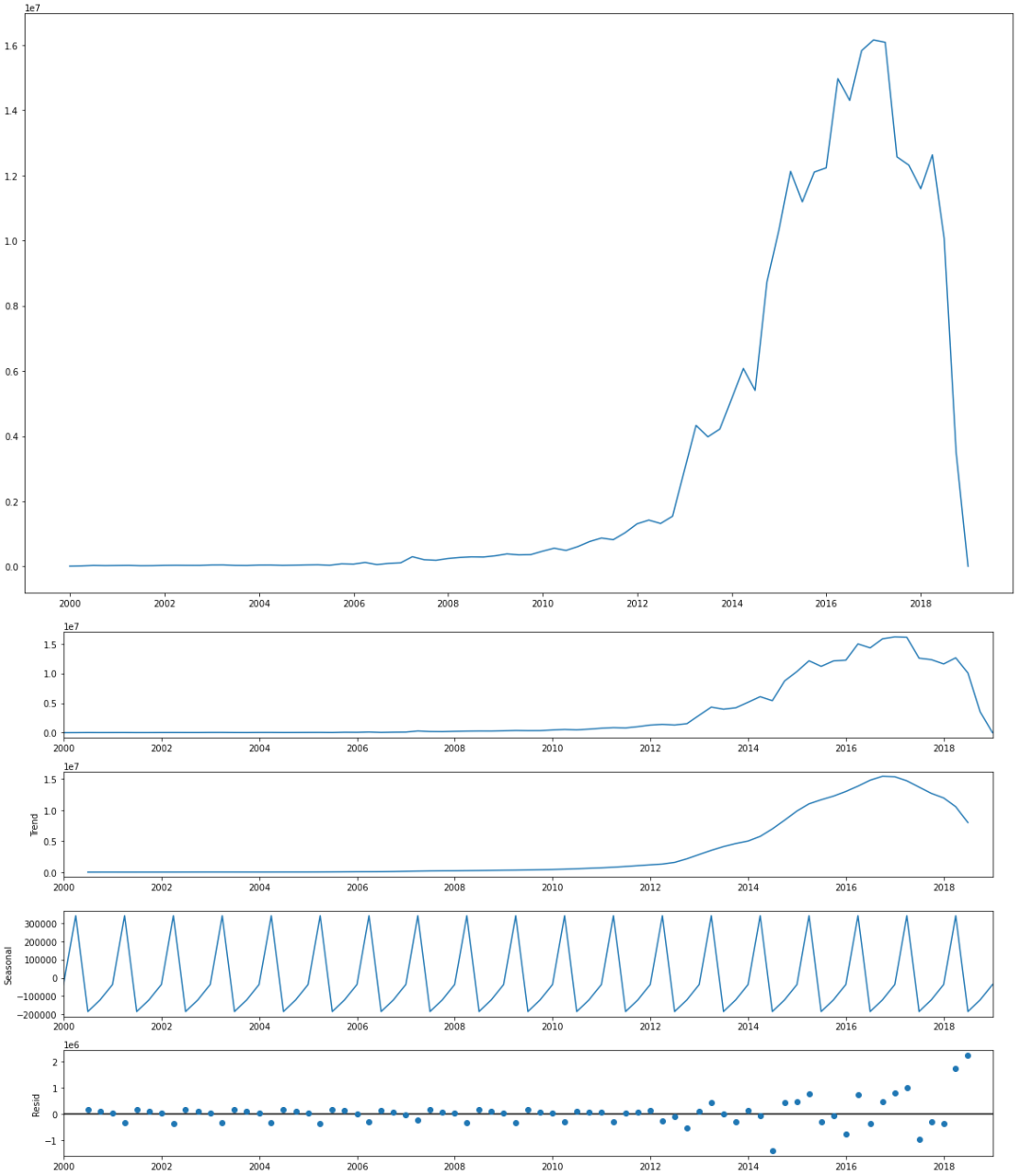
Out[287]:

Date	price
2016-12-31	6.350941e+06

We observe that December of 2016 had the maximum sales

(month having highest sales)

14) Represent the Sales in the Tools and Home Improvement category with respect to time (Quarterly). Also identify if there is any trend or seasonality in the data



Business Question 15.b) Identify the quarter having the maximum sales

```
In [288]: time_date_tools_quarterly[time_date_tools_quarterly['price']==time_date_tools_quarterly['price'].max()]
```

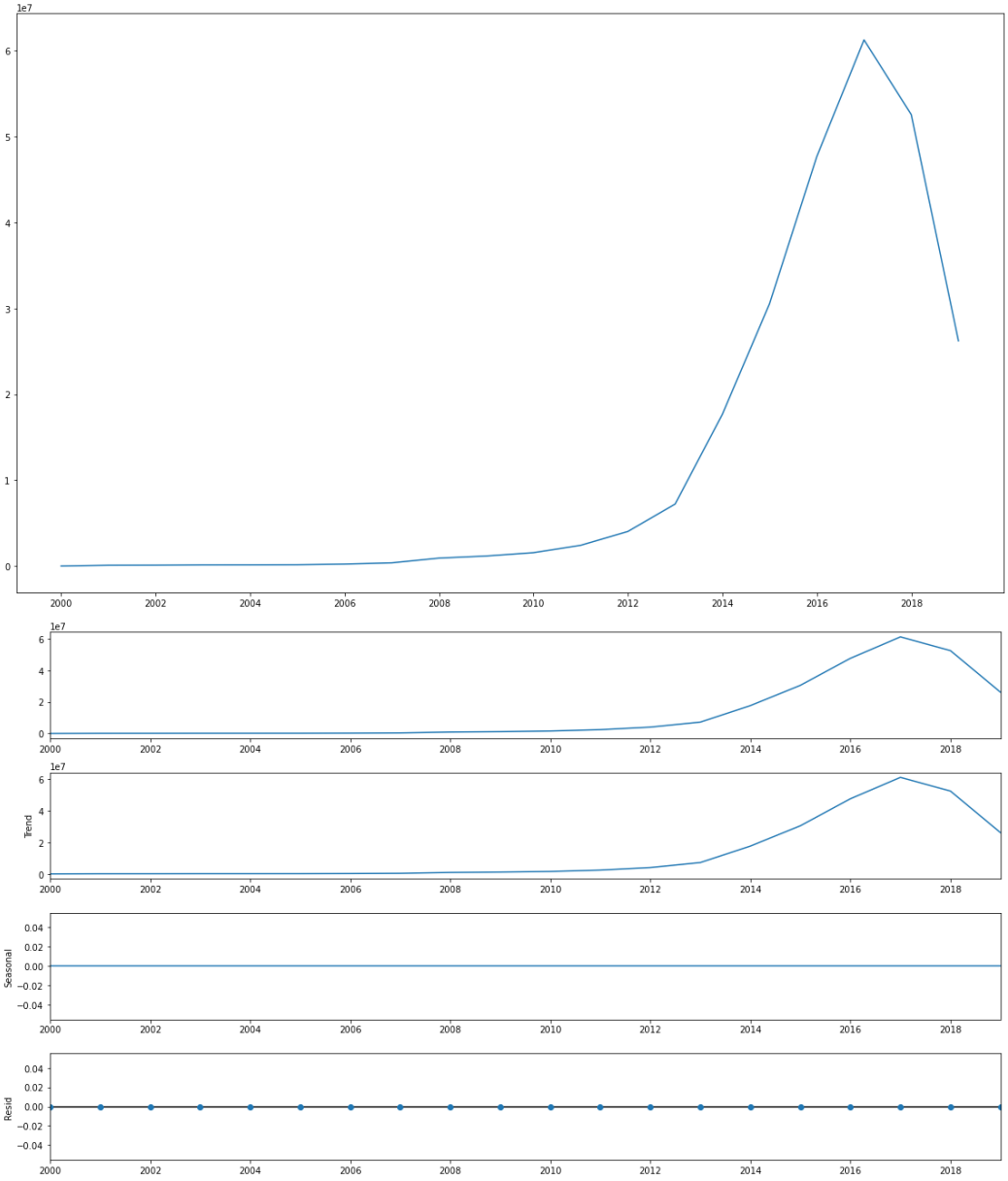
Out[288]:

Date	price
2016-12-31	1.616184e+07

We observe that the final quarter of 2016 had the maximum sales

(Quarter having the highest sales)

15) Represent the Sales in the Tools and Home Improvement category with respect to time (Yearly). Also identify if there is any trend or seasonality in the data



Business Question 16.b) Identify the year having the maximum sales

```
In [289]: time_date_tools_yearly[time_date_tools_yearly['price']==time_date_tools_yearly['price'].max()]
```

Out[289]:

Date	price
2016-12-31	6.127080e+07

We observe that 2016 was the year with maximum sales

(Year giving the maximum sales)

B) Patio Lawn and Garden

1) Analysis on style column Even though we observe that the style column had plenty of missing values, EDA was conducted to observe trends among the existing values.

```
In [163]: # Dropping null values to perform analysis on the style column
          df_style_eda_patio=patio_lawn_and_garden.dropna()

In [164]: df_style_eda_patio['style']

Out[164]: 7                                          {'Size:': ' 20'}
          8                                           {'Size:': ' 3'}
          9                                         {'Size:': ' C001'}
          10                                          {'Size:': ' 19'}
          11                                          {'Size:': ' 19'}
                                              ...
          798402                                   {'Style:': ' Grill'}
          798409                              {'Size:': ' M 58*24*48'}
          798410                              {'Size:': ' L 64*24*48'}
          798411                              {'Size:': ' L 64*24*48'}
          798412    {'Color:': ' Espresso Brown', 'Style:': ' Unity'}
          Name: style, Length: 396445, dtype: object

In [165]: # Checking type of data in style column
          for i in df_style_eda_patio['style'][:3]:
              print(type(i))

          <class 'str'>
          <class 'str'>
          <class 'str'>
```

We observe that our style column is in a dictionary format but is of string type. We must therefore, first convert it to dictionary type first by using the ast module.
It was converted by following the same procedure as was followed in Tools and Home Improvement category.
For our analysis we will be conducting EDA on only those style parameters which were occurring frequently in our data. The following parameters were taken into consideration:

i) Style
ii) Format
iii) Color
iv) Size
v) Material

The details of the styles were segregated and placed in a data frame as such:

```
In [185]: for i in style_dict:
              for j,k in i.items():
                  if (j=='Color:')|(j=='Color Name:'):
                      color_count.append(k)
                  else:
                      color_count.append(np.nan)

                  if j=='Format:':
                      format_count.append(k)
                  else:
                      format_count.append(np.nan)

                  if (j=='Style:')|(j=='Style Name:')|(j=='style name:'):
                      style_count.append(k)
                  else:
                      style_count.append(np.nan)

                  if (j=='Size:')|(j=='Size Name:'):
                      size_count.append(k)
                  else:
                      size_count.append(np.nan)

                  if (j=='Material:')|(j=='Material Type:'):
                      material_count.append(k)
                  else:
                      material_count.append(np.nan)
```

On, viewing our columns in the df_style data frame we observe that there is need to perform some noise treatment in our color and size columns.

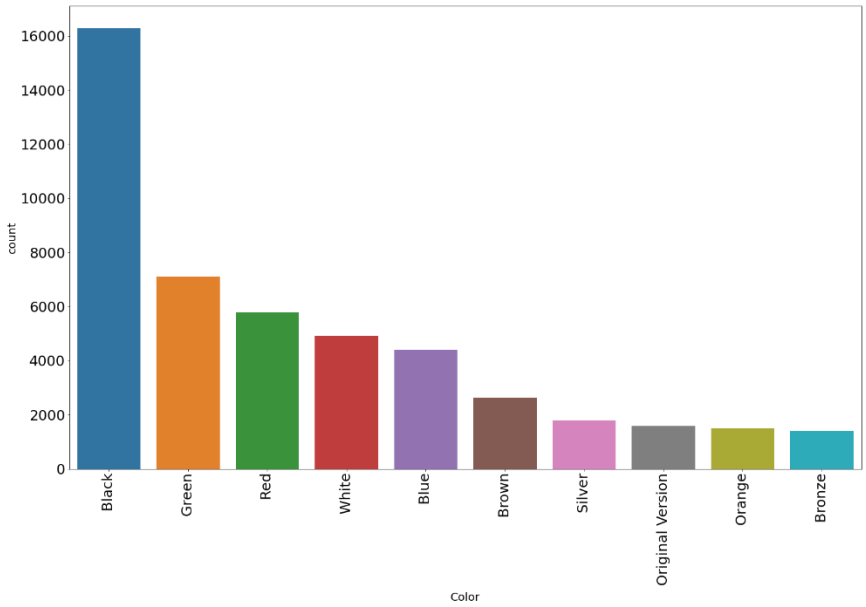
i) Our color column contained numbers for example: 293, 294, 11-D.

ii) There was no defined format in which data existed in our size column. For example: ‘1-pack’ being represented as '1 Pack','1-(Pack)',' Pack of 1',' 1'.

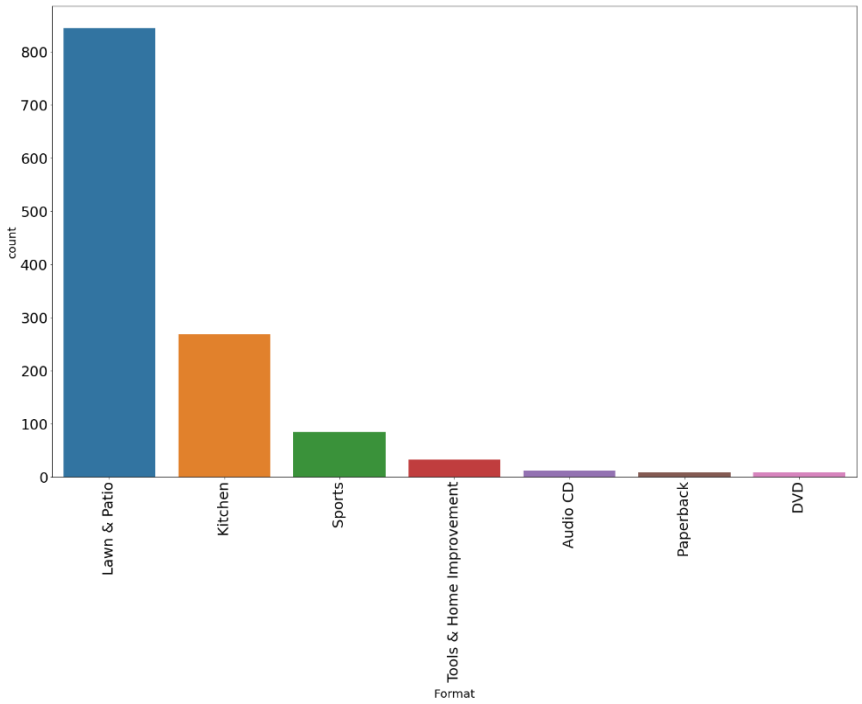
We thereby, perform treatment to these columns in the same way as was done on Tools and Home Improvement.

Now, that all noise is removed from our style column, we shall solve some business questions with respect to our style column.

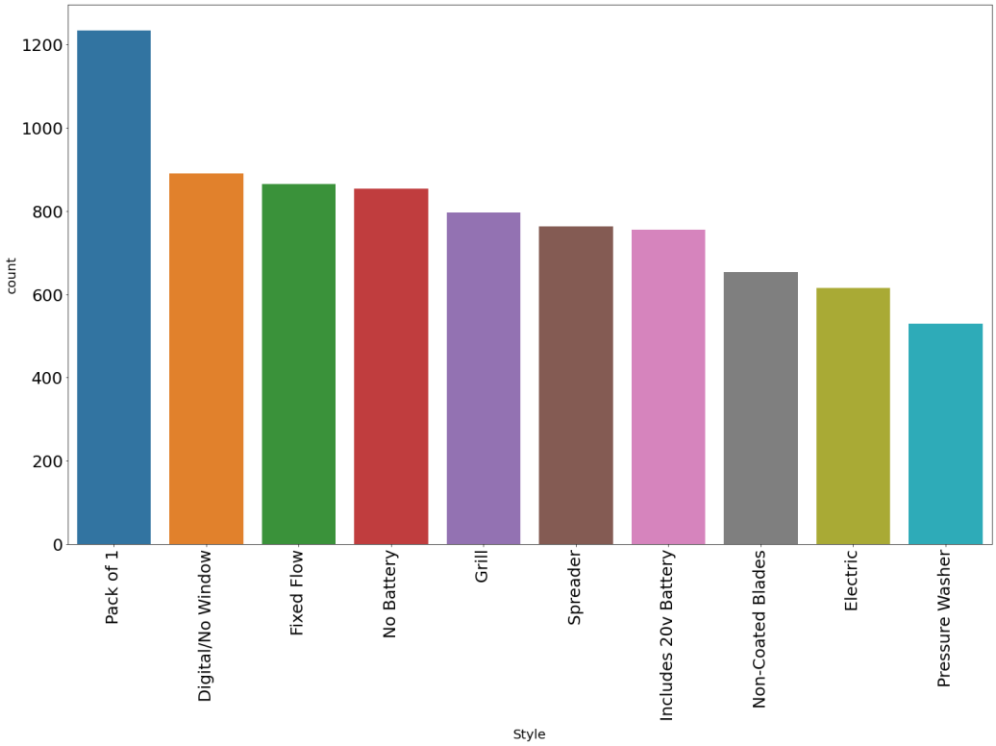
a) What are the top 10 color preferences of customers in the Patio Lawn and Garden Category



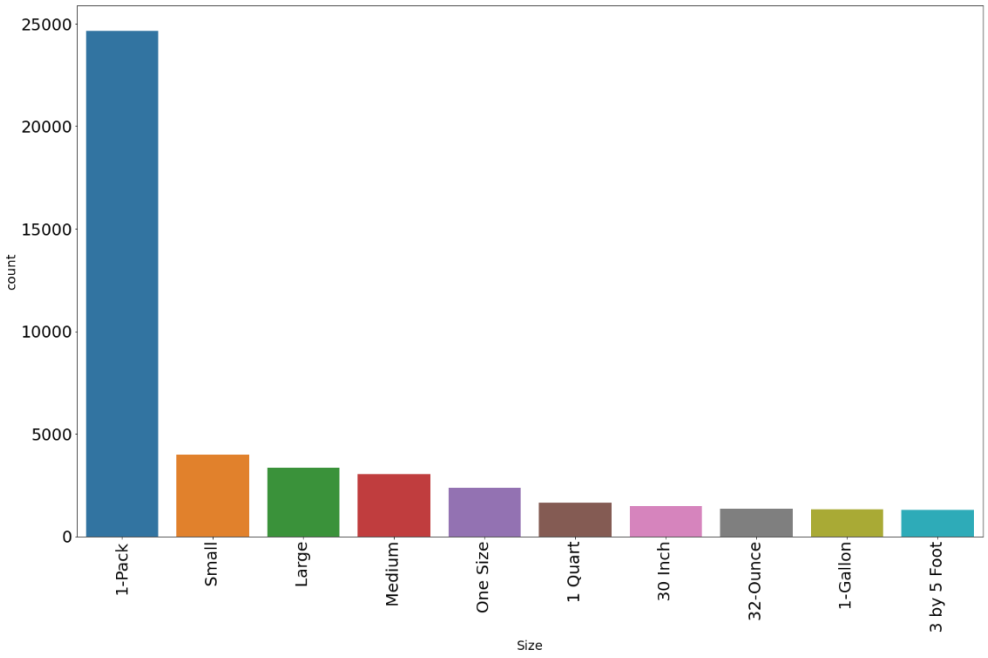
b) What are the top format preferences of customers in the Patio Lawn and Garden Category



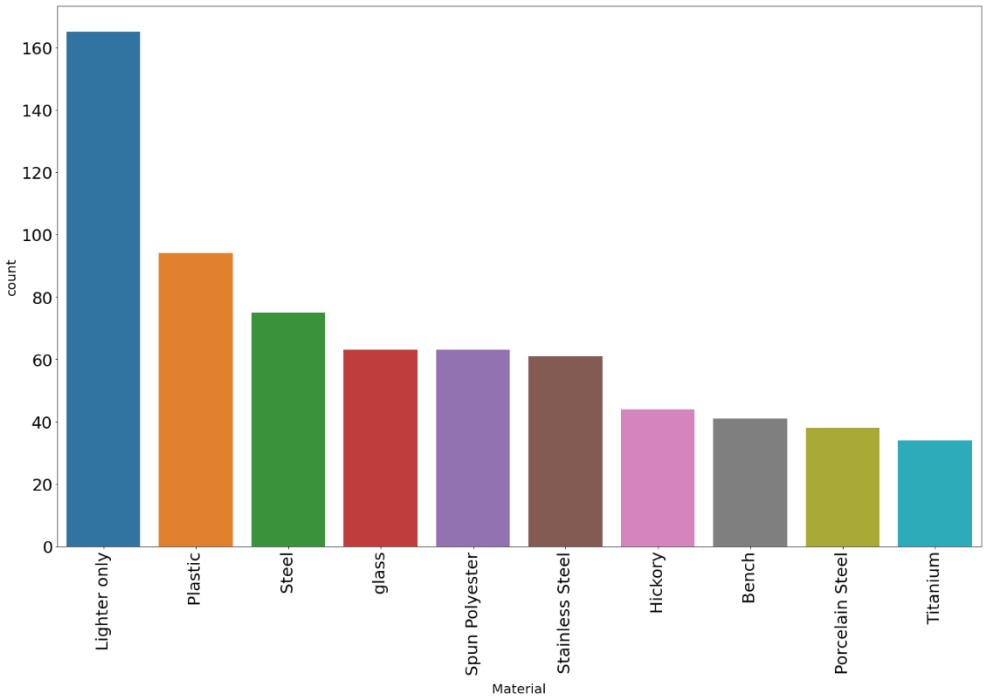
c) What are the top 10 style preferences of customers in the Patio Lawn and Garden Category



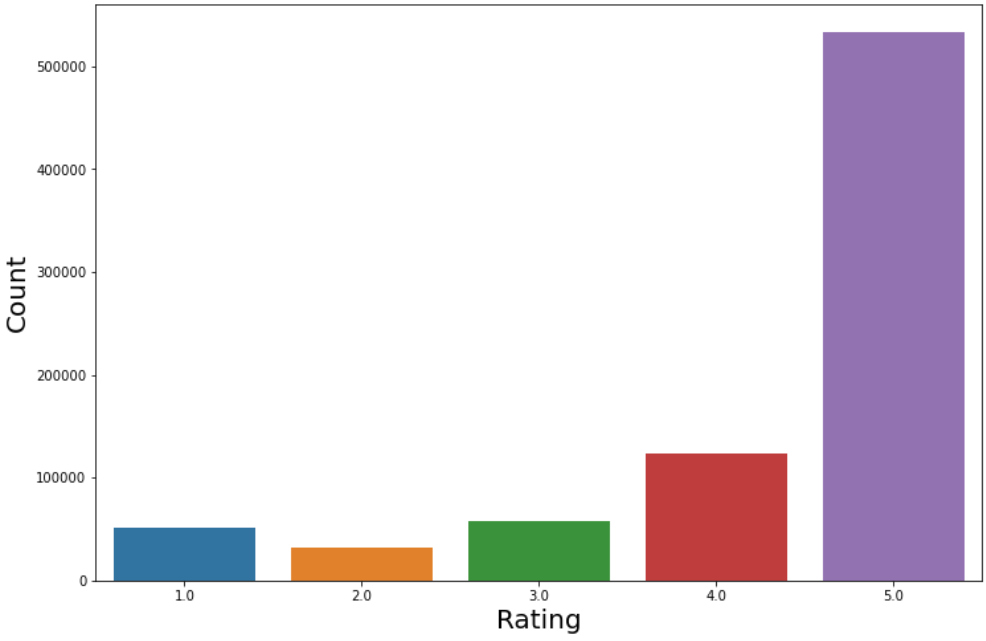
d) What are the top 10 size preferences of customers in the Patio Lawn and Garden Category



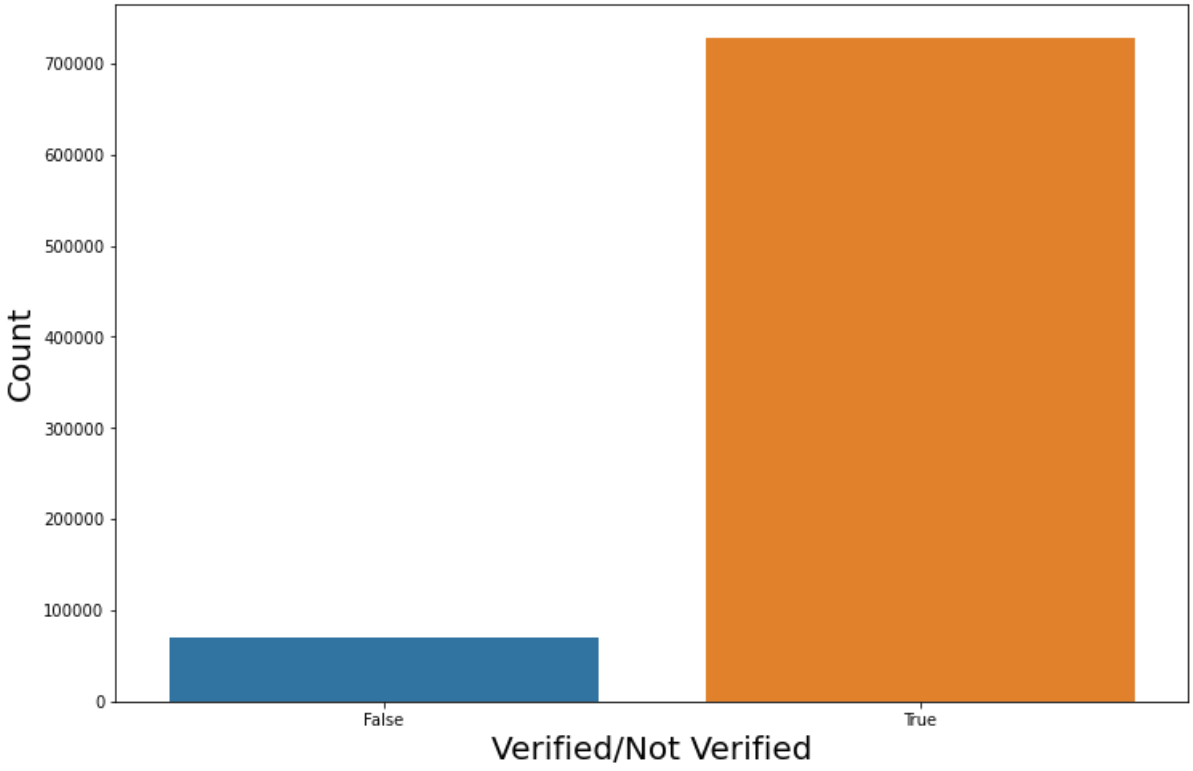
e) What are the top 10 material preferences of customers in the Patio Lawn and Garden Category



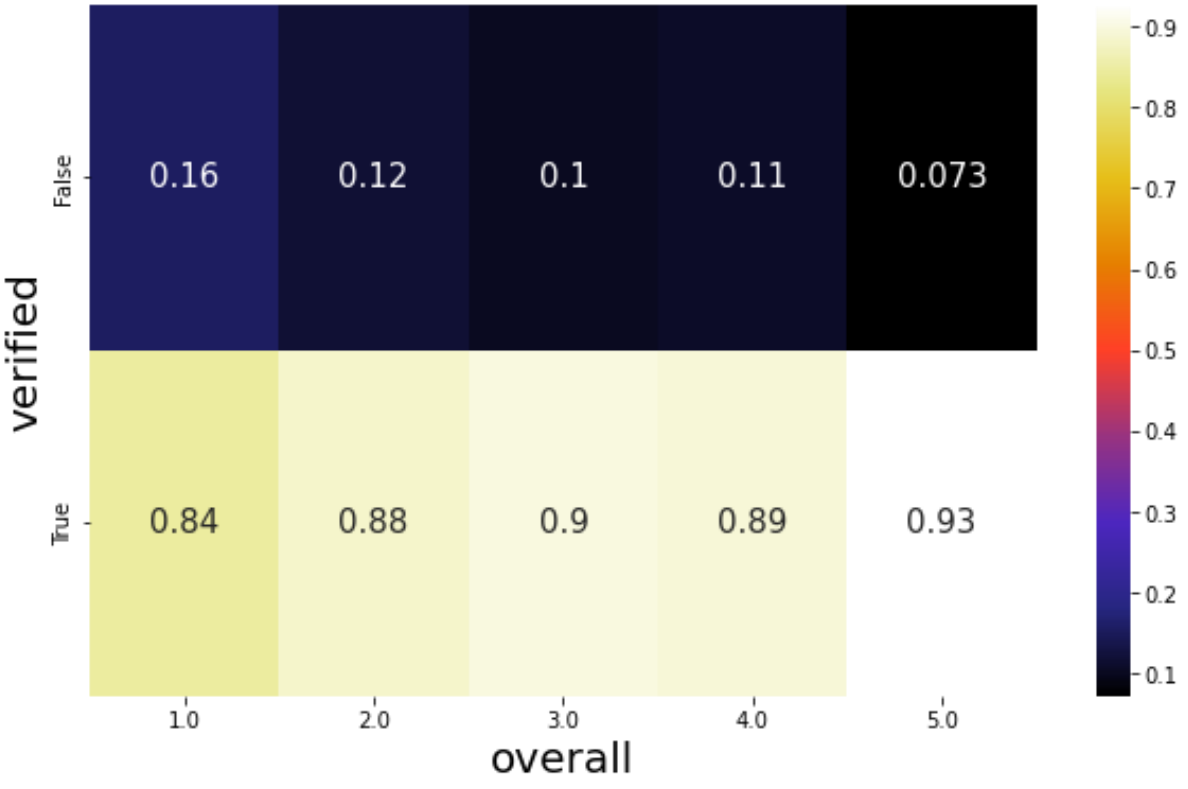
2) How is the count distribution across different ratings in the Patio and Lawn Garden Category



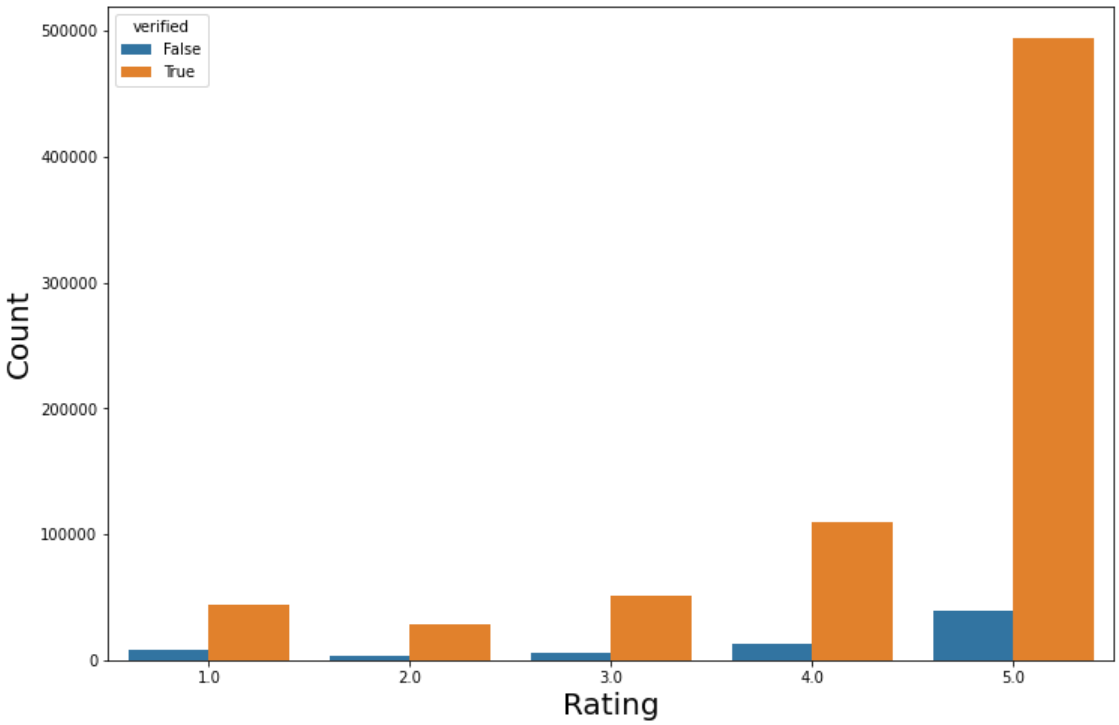
3) How is the count distribution across Verified/Not Verified customers in the Patio Lawn and Garden Category



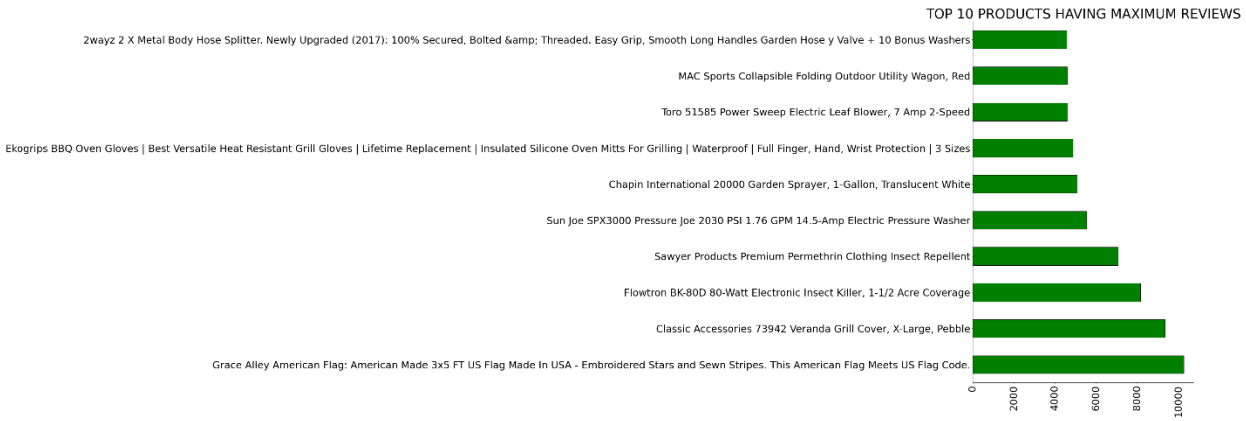
4) How is the percentage distribution of Verified/Not Verified customers across different ratings in the Patio Lawn and Garden Category



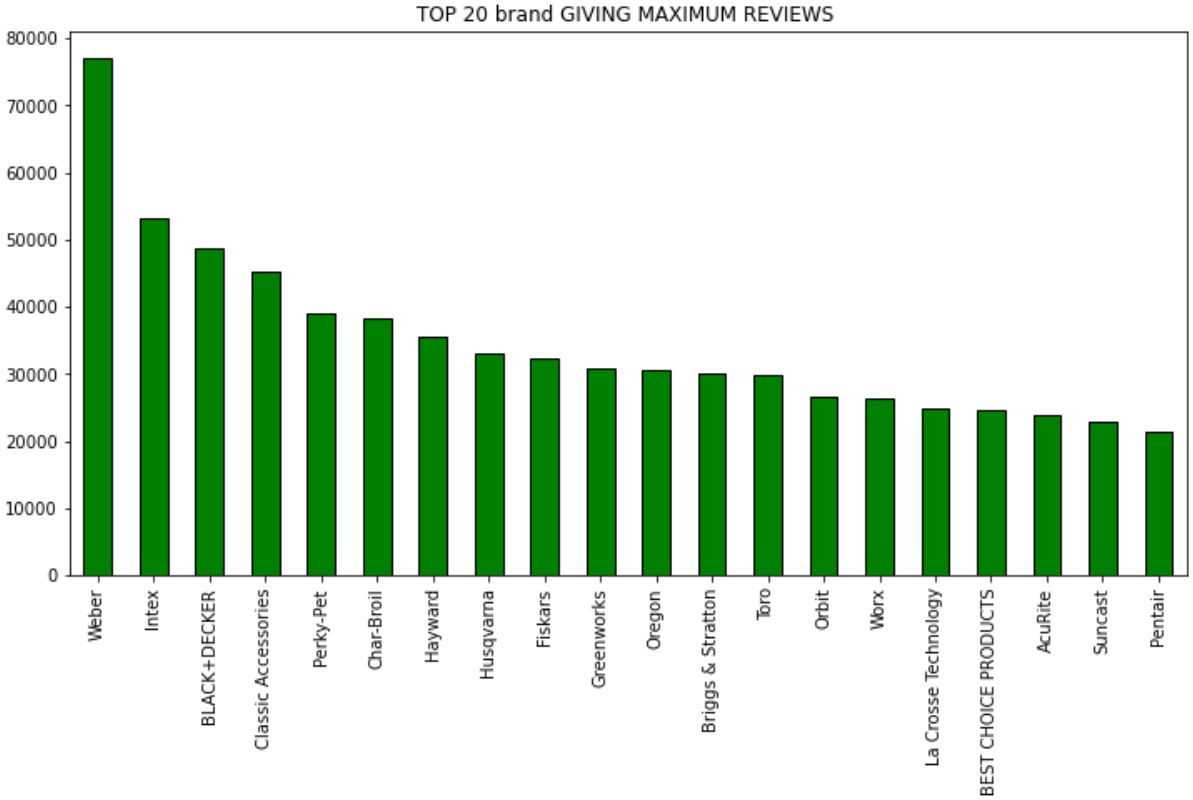
5) How is the count distribution of Verified/Not Verified customers with respect to ratings in the Tools and Home Improvement Category



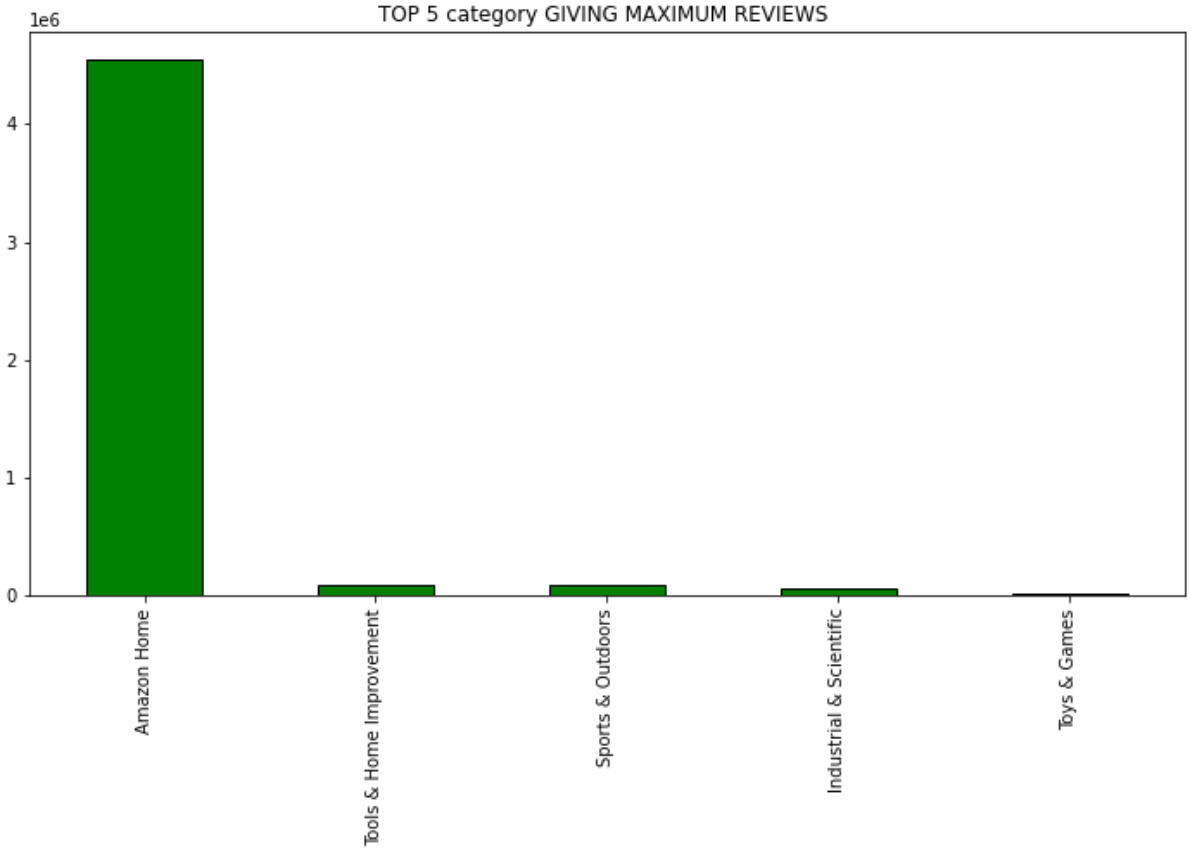
6) What are the top 10 most popular Products in the category of Patio Lawn and Garden?



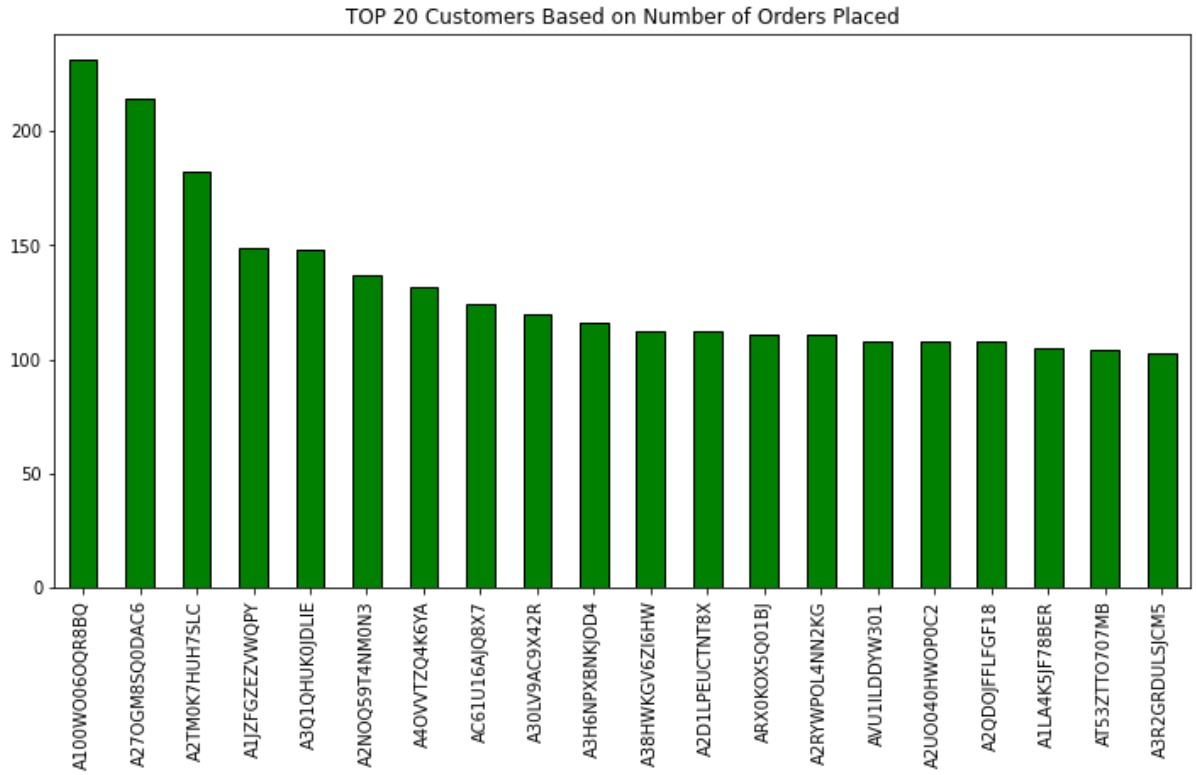
7) What are the top 20 most popular Brands in the category of Patio Lawn and Garden?



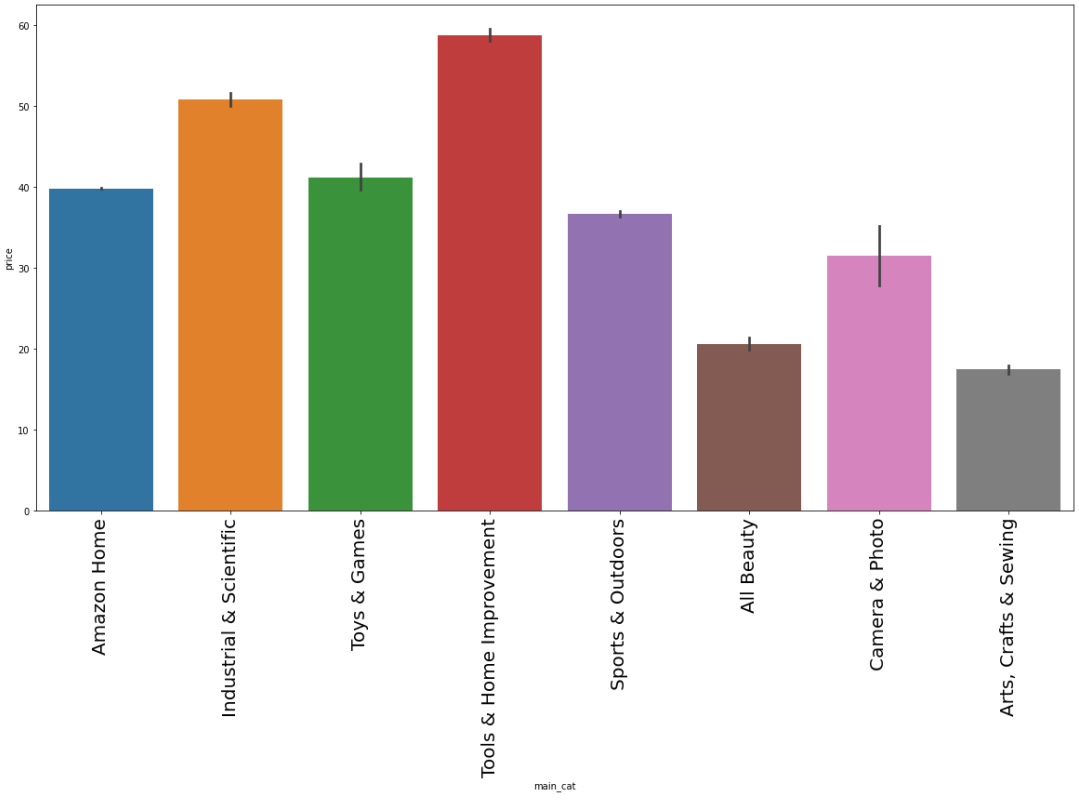
8) Identify the Categories in decreasing order of review count



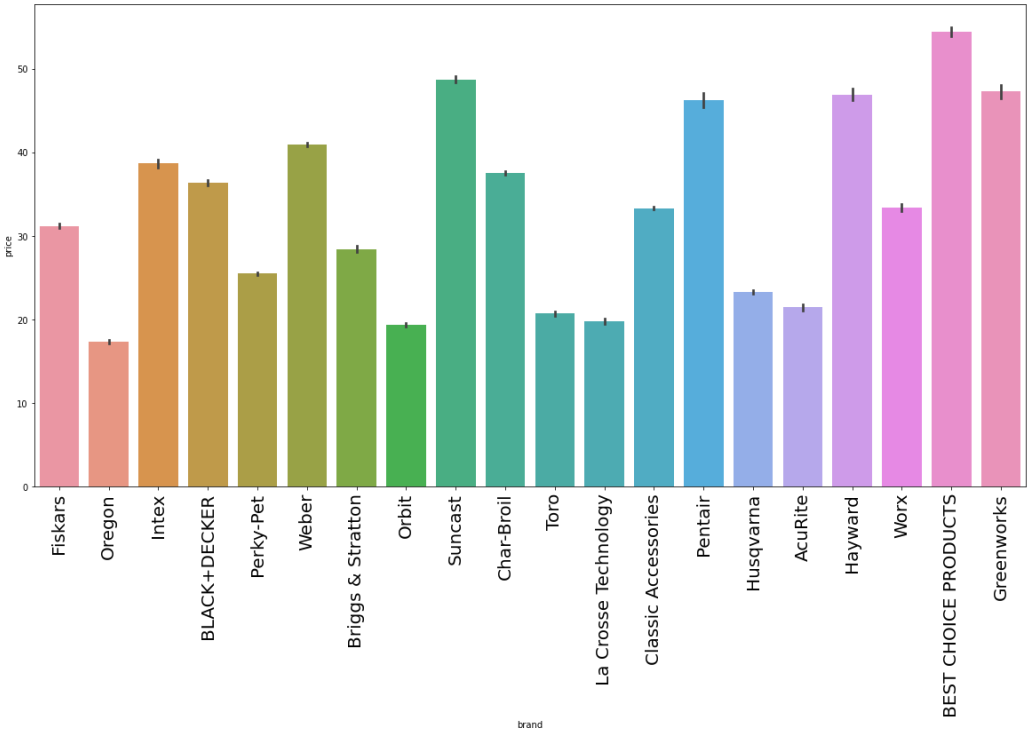
9) Identify the customers who have provided the maximum reviews in the Patio Lawn and Garden Category



10) Represent the Spread of Price with respect to the main categories for Patio Lawn and Garden



11) Represent the Spread of Price with respect to the top 20 most Popular Brands



Now, we shall conduct some EDA based on sales with respect to time. In order to do that we must first convert our unixtime to proper date format

```
In [330]: import datetime

In [331]: def date_time(epoch_time):
              date_time = datetime.datetime.fromtimestamp( epoch_time )
              return date_time

In [332]: df_patio_time['date_time'] = df_patio_time['date_time'].apply(date_time)

In [333]: df_patio_time.dtypes

Out[333]: title                  object
          brand                  object
          main_cat               object
          price                 float64
          product_id             object
          Reviewer_id            object
          rating                float64
          date_time      datetime64[ns]
          dtype: object

In [334]: df_patio_time['Date'] = pd.to_datetime(df_patio_time['date_time']).dt.date

In [335]: df_patio_time.head()
```

Out[335]:

	title	brand	main_cat	price	product_id	Reviewer_id	rating	date_time	Date
0	Audio CD-I Was On His Mind	Harrison House Publishing	Amazon Home	16.990000	0881149659	A7BBJK5J2GJAN	5.0	2016-08-03 05:30:00	2016-08-03
1	My Baby Compass, Four to Seven Years (An Easy ...	JALL	Amazon Home	52.730167	0984408525	AXBO8IITQB0FH	5.0	2012-04-15 05:30:00	2012-04-15
2	My Baby Compass, Four to Seven Years (An Easy ...	JALL	Amazon Home	52.730167	0984408525	ASJ3N62R9QUEG	5.0	2011-11-17 05:30:00	2011-11-17
3	My Baby Compass, Four to Seven Years (An Easy ...	JALL	Amazon Home	52.730167	0984408525	A1X34K98YZP7AX	5.0	2011-11-10 05:30:00	2011-11-10
4	My Baby Compass, Four to Seven Years (An Easy ...	JALL	Amazon Home	52.730167	0984408525	AX7MOATV5NE5B	5.0	2011-11-08 05:30:00	2011-11-08

```
In [336]: # Converting to datetime dtype
          df_patio_time['Date']=pd.to_datetime(df_patio_time['Date'])
```

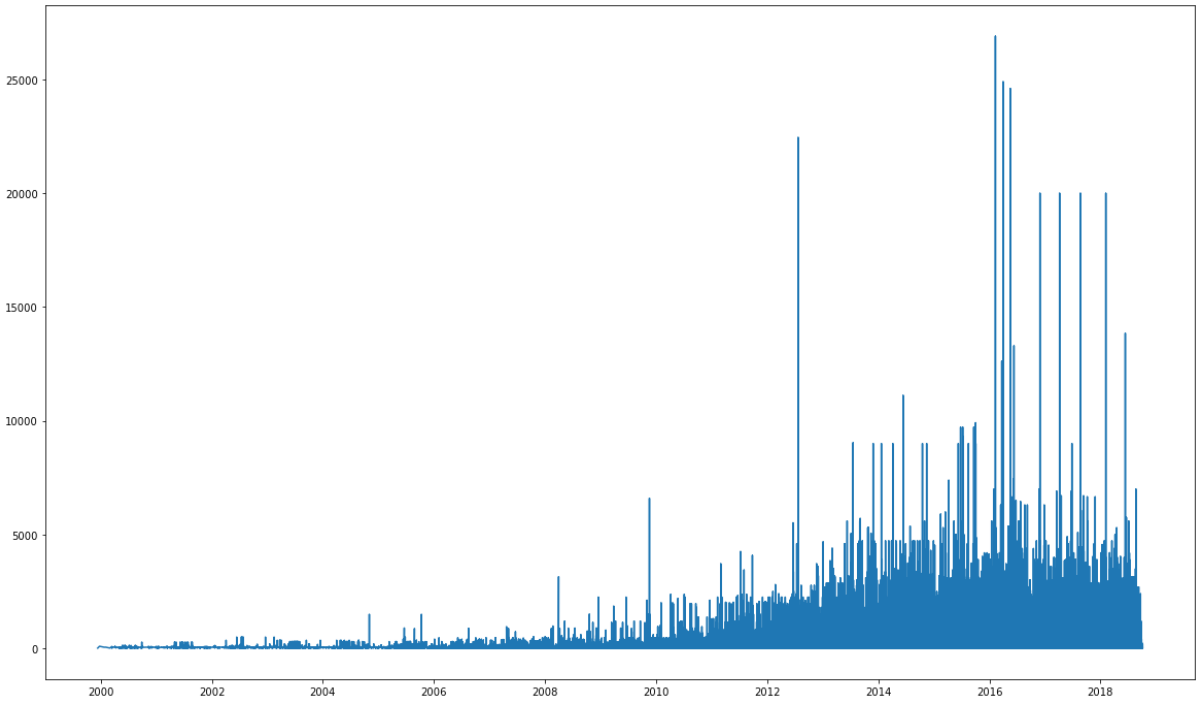
```
In [338]: # Sorting values by date
          df_patio_time.sort_values(by='Date',inplace=True)

In [339]: df_patio_time=df_patio_time.set_index('Date')
          df_patio_time
```

Out[339]:

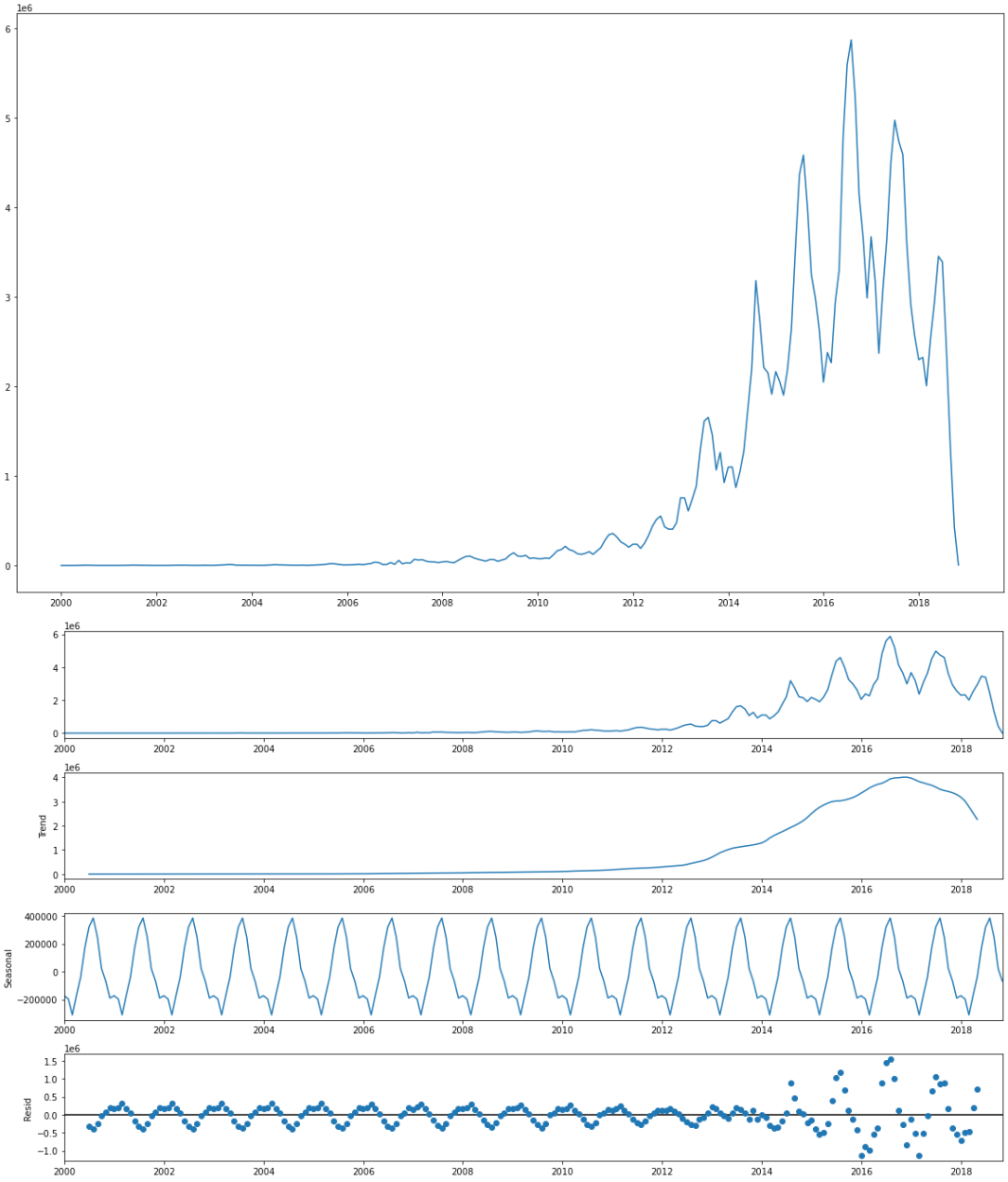
Date	title	brand	main_cat	price	product_id	Reviewer_id	rating	date_time
1999-12-13	Fiskars 91095935J Steel Bypass Pruning Shears	Fiskars	Amazon Home	11.980000	B00002N66H	A3GC0EL0VHT1UP	3.0	1999-12-13 05:30:00
1999-12-25	Center Point 972510 25 Tape CenterPoint	CenterPoint	Tools & Home Improvement	94.586797	B00002254U	A3IUHJF24XNQOP	4.0	1999-12-25 05:30:00
2000-01-24	Safer Brand Victor M500 Poison-Free Indoor Fly...	Safer Brand	Amazon Home	52.730167	B00002N8SM	A32DW342WBJ6BX	5.0	2000-01-24 05:30:00
2000-02-05	Hefner Plastics 414 12-Ounce Hummingbird Feeder	Gardener's Choice	Amazon Home	52.730167	B00002N8FP	A3G3MWKYMJAWS2	5.0	2000-02-05 05:30:00
2000-03-02	Nelson 5200 Automatic Lawn/Garden Water Timer	Nelson	Amazon Home	15.000000	B00002N6AG	A11XDU6CI6OT4I	4.0	2000-03-02 05:30:00
...	...	...	...	...	...	...	...	...
2018-10-04	Homitt Gas Grill Cover, 58-inch 3-4 Burner 600...	Homitt	Amazon Home	18.990000	B01HGFBNL6	A19HGGROP4PRCK	5.0	2018-10-04 05:30:00
2018-10-04	Keter Unity XL Indoor Outdoor Entertainment BB...	Keter	Amazon Home	155.640000	B01HJATY32	A2H5CVJXAXPRIX	5.0	2018-10-04 05:30:00
2018-10-05	Char-Broil Classic 405 4-Burner Liquid Propane...	Char-Broil	Amazon Home	10.420000	B01HITNEE4	A2FEI0VQYMDAW6	4.0	2018-10-05 05:30:00
2018-10-05	Char-Broil Performance 300 2-Burner Cabinet Li...	Char-Broil	Amazon Home	10.590000	B01HID4Y7Q	A2N6AM022CSIXF	1.0	2018-10-05 05:30:00
2018-10-05	Keter Unity XL Indoor Outdoor Entertainment BB...	Keter	Amazon Home	155.640000	B01HJATY32	A3M0T6U6SBT88A	5.0	2018-10-05 05:30:00

4838956 rows × 8 columns

12) Represent the Sales in the Patio Lawn and Garden category with respect to time. Also identify if there is any trend or seasonality in the data



13) Represent the Sales in the Patio Lawn and Garden category with respect to time (Monthly). Also identify if there is any trend or seasonality in the data



Business Question 30.b) Identify the months having the maximum sales

```
In [358]: time_data_patio_monthly[time_data_patio_monthly['price']==time_data_patio_monthly['price'].max()]
```

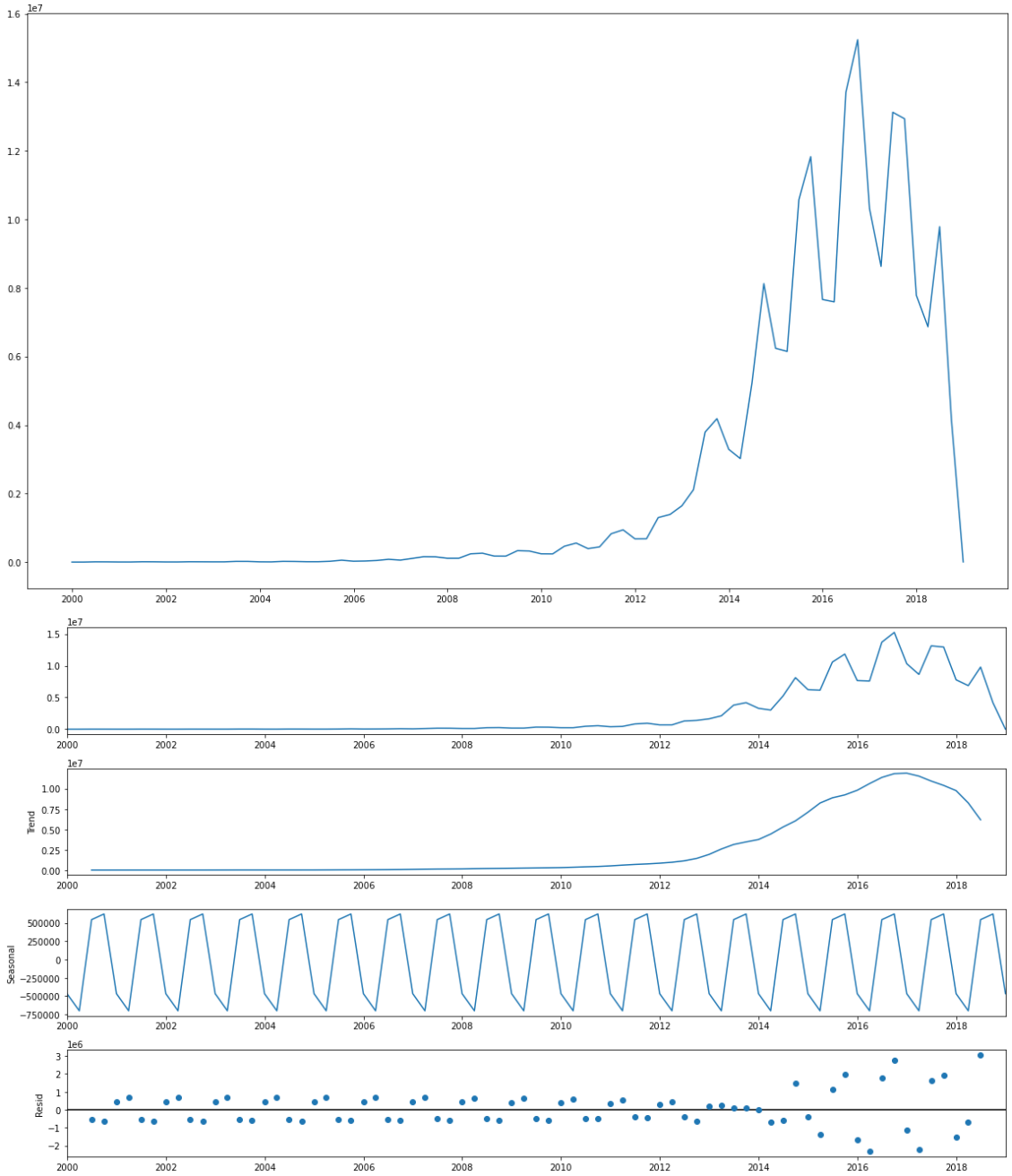
Out[358]:

Date	price
2016-07-31	5.874558e+06

We observe that July of 2016 was the month noticed the maximum sales

(Month giving maximum sales)

14) Represent the Sales in the Patio Lawn and Garden category with respect to time (Quarterly). Also identify if there is any trend or seasonality in the data



Business Question 31.b) Identify the quarter having the maximum sales

```
In [359]: time_data_patio_quarterly[time_data_patio_quarterly['price']==time_data_patio_quarterly['price'].max()]
```

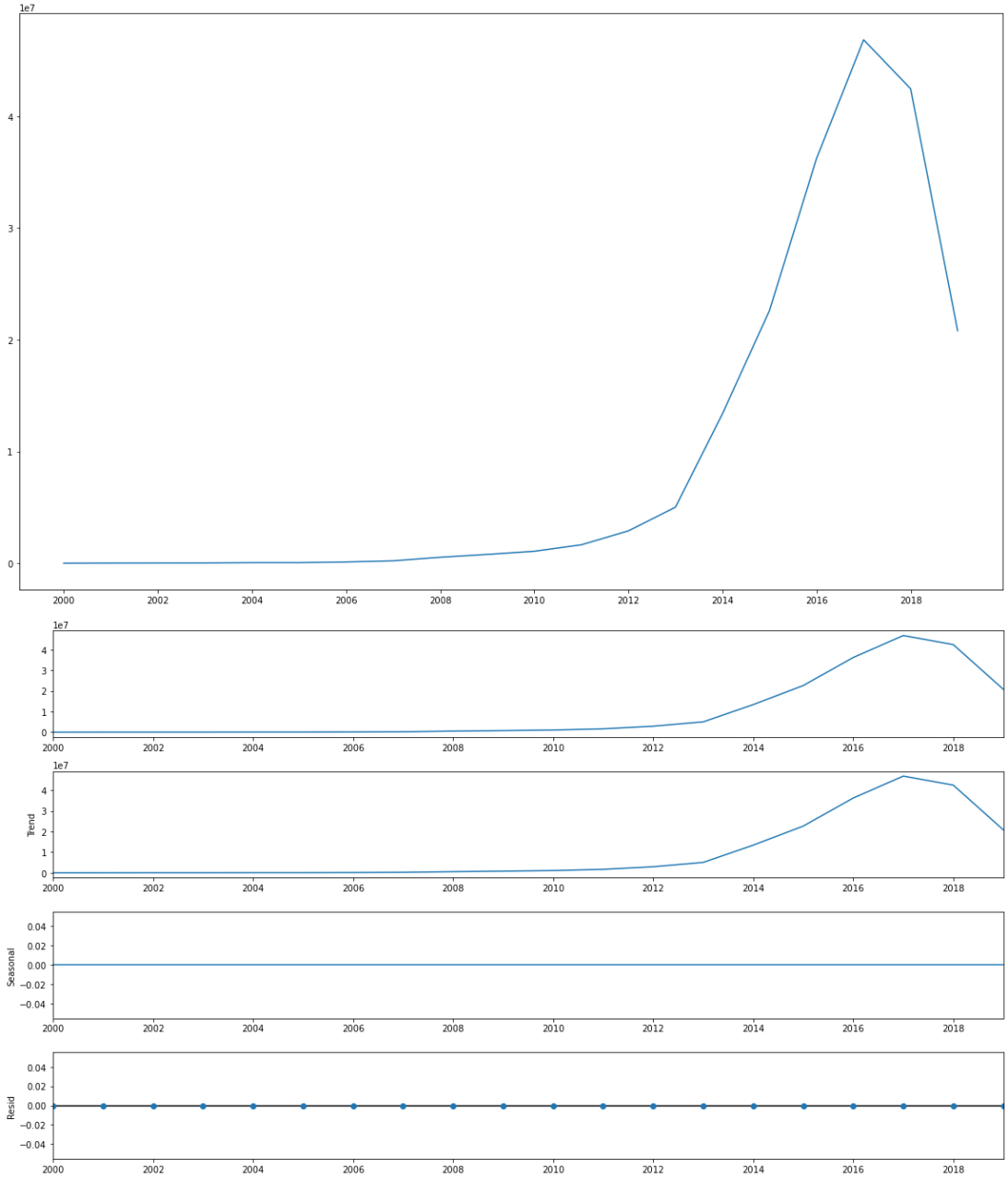
Out[359]:

Date	price
2016-09-30	1.524150e+07

We notice that the 3rd quarter of 2016 noticed the maximum sales

(Quarter with maximum sales)

15) Represent the Sales in the Patio Lawn and Garden category with respect to time (Yearly). Also identify if there is any trend or seasonality in the data



Business Question 32.b) Identify the year having the maximum sales

```
In [360]: time_data_patio_yearly[time_data_patio_yearly['price']==time_data_patio_yearly['price'].max()]
```

Out[360]:

Date	price
2016-12-31	4.685739e+07

We notice that 2016 was the year that noticed the maximum sales

(Year with maximum sales)

SENTIMENT ANALYSIS ON USER REVIEWS

What is sentiment analysis and why is it necessary?

Sentiment analysis (or opinion mining) is a natural language processing (NLP) technique used to determine whether data is positive, negative or neutral. Sentiment analysis is often performed on textual data to help businesses monitor brand and product sentiment in customer feedback, and understand customer needs.

For our analysis we have conducted three text classification techniques

i) One vs Rest Modelling (Logistic Regression)
ii) Multinominal Naïve Bayes Modelling
iii) Fasttext Modelling

In our Logistic regression model and our naïve bayes model, we have performed different types of text vectorizations. Like:

i) TFIDF Vectorizer
ii) Bag of Words Vectorizer
iii) Ngram Vectorization

Depending on the accuracy scores given by different models with their different vectorization techniques, the best model was chosen to perform text classification on user reviews.

A) Tools and Home Improvement

1) One Vs Rest Classification (Logistic Regression)

i) We begin by importing necessary libraries

ii) We now perform text pre-processing on our text which involves: removing numbers from the text, making uppercase alphabets as lowercase, removal of extra spaces from text and removal of stop words (words occurring very frequently in our text like ‘a’, ’the’, ’is’ etc. They can also be defined as connector words or articles, prepositions, pronouns, conjunctions etc.)

```
def clean_text(text):
    # Split the text into individual words
    words = text.split()

    # Remove any punctuation or non-alphabetic characters
    words = [word for word in words if word.isalpha()]

    # Convert all words to lowercase
    words = [word.lower() for word in words]

    # Remove any stop words
    stop_words = stopwords=('english')
    words = [word for word in words if word not in stop_words]

    # Join the words back into a single string
    return " ".join(words)


# Clean the "text" column using the clean_text() function
Text_Tools["reviewText"] = Text_Tools["reviewText"].apply(clean_text)
```

iii) Set sentiment based on ratings. If rating is greater than 3, set sentiment as positive. If rating is less than 3, set sentiment as negative. If rating equals 3, then set sentiment as neutral.

```
sentiment=[]
for i in df_tools['rating']:
    if (i>3):
        sentiment.append('Positive')
    elif(i<3):
        sentiment.append('Negative')
    else:
        sentiment.append('Neutral')
```

```
df_tools['ReviewSentiment']=sentiment
```

iv) Set X as the review text. Set y as sentiment

```
X=[]
y=[]
for i in Text_Tools['reviewText']:
    X.append(i)
for j in Text_Tools['ReviewSentiment']:
    y.append(j)
```

v) Perform train test split in 80:20 ratio

```
# Splitting the data in 80:20 ratio for train and test respectively
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.20,random_state=9)
```

```
len(X_train)
```

```
1454220
```

```
len(X_test)
```

```
363555
```

vi) Perform vectorization using TFIDF Vectorizor. We must perform tfidfvectorizor() with fit_transform on train set and use fit for the test set.

What is vectorization and why is it needed?

Word Embeddings or Word vectorization is a methodology in NLP to map words or phrases from vocabulary to a corresponding vector of real numbers which used to find word predictions, word similarities/semantics. The process of converting words into numbers are called Vectorization.

```
tfidf_vectorizer = TfidfVectorizer()
X_train = tfidf_vectorizer.fit_transform(X_train)
X_test = tfidf_vectorizer.transform(X_test)
```

```
print("X_train",X_train.shape)
print("X_test",X_test.shape)
```

```
X_train (1454220, 109944)
X_test (363555, 109944)
```

vii) Perform modelling using one vs rest strategy using logistic regression.

```
lr = LogisticRegression(max_iter=200)
ovr = OneVsRestClassifier(lr)
```

```
# fit the model on the training data
ovr.fit(X_train, y_train)
y_pred = ovr.predict(X_test)
```

viii) Perform evaluation metrics by observing classification report and confusion matrix.

classification report

```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

    Negative       0.67      0.45      0.54     29993
     Neutral       0.44      0.04      0.07     23977
    Positive       0.89      0.99      0.94    309585

    accuracy                           0.88    363555
   macro avg       0.67      0.49      0.52    363555
weighted avg       0.85      0.88      0.85    363555
```

confusion matrix

```
confusion_matrix(y_test,y_pred)
```

```
array([[ 13382,    474,  16137],
       [  3014,    937,  20026],
       [  3466,    729, 305390]], dtype=int64)
```

```
print('accuracy:', accuracy_score(y_test, y_pred))
print('recall:', recall_score(y_test, y_pred, average='weighted'))
print('f1-score:', f1_score(y_test, y_pred, average='weighted'))
print('precision:', precision_score(y_test, y_pred,average='weighted'))
```

```
accuracy: 0.8793965149702246
recall: 0.8793965149702246
f1-score: 0.8477905491603605
precision: 0.845849467477149
```

ix) Now, we shall perform tfidf vectorization using hyperparameters to observe if we can obtain a better score. We shall try with parameters max_df=0.5 and min_df=1

```
# Splitting the data in 80:20 ratio for train and test respectively
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.20,random_state=9)
```

```
tfidf_vectorizer = TfidfVectorizer(sublinear_tf=True,
                           max_df=0.5,
                           min_df=1,
                           use_idf=True,smooth_idf=True)
X_train = tfidf_vectorizer.fit_transform(X_train)
X_test = tfidf_vectorizer.transform(X_test)
```

```
print("X_train",X_train.shape)
print("X_test",X_test.shape)
```

```
X_train (1454220, 109944)
X_test (363555, 109944)
```

x) Perform model training

xi) Observe performance metrics

```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

    Negative       0.67      0.45      0.54     29993
     Neutral       0.44      0.04      0.07     23977
    Positive       0.89      0.99      0.94    309585

    accuracy                           0.88    363555
   macro avg       0.67      0.49      0.52    363555
weighted avg       0.85      0.88      0.85    363555
```

```
confusion_matrix(y_test,y_pred)
```

```
array([[ 13594,    478,  15921],
       [  3113,    949,  19915],
       [  3563,    738, 305284]], dtype=int64)
```

```
print('accuracy:', accuracy_score(y_test, y_pred))
print('recall:', recall_score(y_test, y_pred, average='weighted'))
print('f1-score:', f1_score(y_test, y_pred, average='weighted'))
print('precision:', precision_score(y_test, y_pred,average='weighted'))
```

```
accuracy: 0.879721087593349
recall: 0.879721087593349
f1-score: 0.8484370916319444
precision: 0.8463274617454206
```

xii) Now, we perform bag of words vectorization

```
# Vectorizing the training data using CountVectorizer()
cv = CountVectorizer()
X_train=cv.fit_transform(X_train)


print(X_train.shape)



X_test=cv.transform(X_test)

print(X_test.shape)
```

```
(1454220, 109944)
(363555, 109944)
```

xiii) Train the model

xiv) Observe evaluation metrics

```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

    Negative       0.66      0.40      0.50     29993
     Neutral       0.37      0.06      0.10     23977
    Positive       0.89      0.98      0.94    309585

    accuracy                           0.87    363555
   macro avg       0.64      0.48      0.51    363555
weighted avg       0.84      0.87      0.84    363555
```

```
confusion_matrix(y_test,y_pred)
```

```
array([[ 12100,    890,  17003],
       [  2758,   1384,  19835],
       [  3552,   1469, 304564]], dtype=int64)
```

```
print('accuracy:', accuracy_score(y_test, y_pred))
print('recall:', recall_score(y_test, y_pred, average='weighted'))
print('f1-score:', f1_score(y_test, y_pred, average='weighted'))
print('precision:', precision_score(y_test, y_pred,average='weighted'))
```

```
accuracy: 0.8748277427074308
recall: 0.8748277427074308
f1-score: 0.8446266459930104
precision: 0.8382740618173977
```

xv) Perform ngram (bigram) vectorization

```
# Vectorizing the training data using CountVectorizer()
cv = CountVectorizer(ngram_range=(2,2))
X_train=cv.fit_transform(X_train)


print(X_train.shape)



X_test=cv.transform(X_test)

print(X_test.shape)
```

```
(1454220, 6109630)
(363555, 6109630)
```

xvi) Train the model

xvii) Observe evaluation metrics

```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

    Negative       0.66      0.33      0.44     29993
     Neutral       0.41      0.09      0.14     23977
    Positive       0.89      0.98      0.93    309585

    accuracy                           0.87    363555
   macro avg       0.65      0.47      0.50    363555
weighted avg       0.84      0.87      0.84    363555
```

```
confusion_matrix(y_test,y_pred)
```

```
array([[  9838,   1114,  19041],
       [  2174,   2054,  19749],
       [  2824,   1829, 304932]], dtype=int64)
```

```
print('accuracy:', accuracy_score(y_test, y_pred))
print('recall:', recall_score(y_test, y_pred, average='weighted'))
print('f1-score:', f1_score(y_test, y_pred, average='weighted'))
print('precision:', precision_score(y_test, y_pred,average='weighted'))
```

```
accuracy: 0.8714609893963774
recall: 0.8714609893963774
f1-score: 0.8404844275686301
precision: 0.8372652899511801
```

xviii) View the data frame comparing the accuracy scores of all the vectorization techniques using logistic regression

	MODEL NAME	TEXT EXTRACTION METHOD	ACCURACY	PRECESSION	RECALL	F1_SCORE
0	OneVsRest	TF_IDF	87.94	84.58	87.94	84.78
1	OneVsRest	TF_IDF_HYPERPARAMETER	88.00	85.00	88.00	85.00
2	OneVsRest	BAG OF WORDS(UNIGRAM)	87.00	84.00	87.00	84.00
3	OneVsRest	BAG OF WORDS(BIGRAM)	87.00	84.00	87.00	84.00

2) Multinominal Naïve Bayes

i) Perform the exact steps as that performed in One vs rest using Logistic regression

ii) Data frame to compare the different evaluation metrics:

	MODEL NAME	TEXT EXTRACTION METHOD	ACCURACY	PRECESSION	RECALL	F1_SCORE
0	MultinomialNB (NAIEVE_BAYES)	TF_IDF	86.0	83.0	86.0	79.0
1	MultinomialNB (NAIEVE_BAYES)	TF_IDF_HYPERPARAMETRE	86.0	82.0	86.0	79.0
2	MultinomialNB (NAIEVE_BAYES)	BAG OF WORDS(UNIGRAM)	86.0	83.0	86.0	84.0
3	MultinomialNB (NAIEVE_BAYES)	BAG OF WORDS(BIGRAM)	87.0	85.0	87.0	82.0

3) Fasttext

i) Perform text pre-processing

ii) Make a column in data frame starting with '__label__' with the review text.

```
text_patio['ReviewSentiment'] = '__label__' + text_patio['ReviewSentiment'].astype(str)
text_patio.head(5)
```

	reviewText	ReviewSentiment
0	spend lot flags beat lesser quality figuring r...	__label__Positive
1	super fast processing looking flags look good ...	__label__Positive
2	great recommend recommend	__label__Positive
3	great price love state country	__label__Positive
4	great display flag	__label__Positive

```
text_patio['review_description'] = text_patio['ReviewSentiment'] + ' ' + text_patio['reviewText']
text_patio.head()
```

	reviewText	ReviewSentiment	review_description
0	spend lot flags beat lesser quality figuring r...	__label__Positive	__label__Positive spend lot flags beat lesser ...
1	super fast processing looking flags look good ...	__label__Positive	__label__Positive super fast processing lookin...
2	great recommend recommend	__label__Positive	__label__Positive great recommend recommend
3	great price love state country	__label__Positive	__label__Positive great price love state country
4	great display flag	__label__Positive	__label__Positive great display flag

iii) Perform train-test split in 80:20 ratio

iv) Create a text file for the train and test set

```
from sklearn.model_selection import train_test_split
```

```
patio_train_1,patio_test_1=train_test_split(text_patio,test_size=0.2)
```

```
patio_train_1.shape, patio_test_1.shape
```

```
((581984, 3), (145496, 3))
```

Converting the review_description to txt files

```
patio_train_1.to_csv("PatioTrain1.train", columns=["review_description"], index=False, header=False)
patio_test_1.to_csv("PatioTest1.test", columns=["review_description"], index=False, header=False)
```

v) Perform fasttext modelling on the train dataset

vi) Observe evaluation metrics

```
test_num1, precision1, recall1 = model.test('PatioTest1.test')
f_score1 = (2*precision1*recall1)/(precision1+recall1)
print('Test Samples:',test_num1)
print('Precision Score:',precision1)
print('Recall:',recall1)
print('F1 score:',f_score1)
```

```
Test Samples: 145496
Precision Score: 0.8580923186891736
Recall: 0.8580923186891736
F1 score: 0.8580923186891736
```

vii) Now, we perform fasttext using hyperparameters. It was noticed that training the model with lr=0.4, epoch=2, wordNgrams=2, bucket=20000 gave the better evaluation scores.

viii) Find ideal value of k and threshold.

```
import numpy as np

check = []
for k in np.arange(1,10,1):
  for t in np.arange(0.1,1,0.1):
    test_num, precision, recall = model_hp.test('PatioTest1.test', k=k, threshold=t)
    f_score = (2*precision*recall)/(precision+recall)
    check.append(f_score)

list1 = []
list2 = []

for k in np.arange(1,10,1):
  for t in np.arange(0.1,1,0.1):
    list1.append(k)
    list2.append(t)

mydf = pd.DataFrame([list1,list2,check]).transpose()
mydf.columns = ['K Value','Threshold value','f_score']
mydf.sort_values('f_score',ascending=False,inplace=True)
mydf
```

	K Value	Threshold value	f_score
3	1.0	0.4	0.857742
57	7.0	0.4	0.857640
48	6.0	0.4	0.857640
30	4.0	0.4	0.857640
39	5.0	0.4	0.857640
...	...	...	...
26	3.0	0.9	0.696585
35	4.0	0.9	0.696585
17	2.0	0.9	0.696585
53	6.0	0.9	0.696585
80	9.0	0.9	0.696585

81 rows × 3 columns

We see that k=1 and threshold value=0.4 gave the best F1 score.

ix) Observe the evaluation metrics

```
test_num4, precision4, recall4 = model2_hp.test('ToolsTest1.test',k=1, threshold=0.4)
f_score4 = (2*precision4*recall4)/(precision4+recall4)
print('Test Samples:',test_num4)
print('Precision:',precision4)
print('Recall:',recall4)
print('F-score',f_score4)
```

```
Test Samples: 363555
Precision: 0.8786530709297118
Recall: 0.874126335767628
F-score 0.8763838579664304
```

B) Patio Lawn & Garden

i) Perform the exact same modelling and vectorization techniques for patio lawn & garden category.

ii) Data frame comparison for One vs rest modelling

	MODEL NAME	TEXT EXTRACTION METHOD	ACCURACY	PRECESSION	RECALL	F1_SCORE
0	OneVsRest	TF_IDF	86.08	82.44	86.08	82.69
1	OneVsRest	TF_IDF_HYPERPARAMETER	86.00	83.00	86.00	83.00
2	OneVsRest	BAG OF WORDS(UNIGRAM)	86.00	83.00	86.00	83.00
3	OneVsRest	BAG OF WORDS(BIGRAM)	86.00	83.00	86.00	83.00

iii) Data frame comparison for Multinominal Naïve Bayes modelling

	MODEL NAME	TEXT EXTRACTION METHOD	ACCURACY	PRECESSION	RECALL	F1_SCORE
0	MultinomialNB (NAIVE_BAYES)	TF_IDF	83.0	82.0	83.0	76.0
1	MultinomialNB (NAIVE_BAYES)	TF_IDF_HYPERPARAMETRE	83.0	82.0	83.0	76.0
2	MultinomialNB (NAIVE_BAYES)	BAG OF WORDS(UNIGRAM)	84.0	80.0	84.0	82.0
3	MultinomialNB (NAIVE_BAYES)	BAG OF WORDS(BIGRAM)	84.0	83.0	84.0	79.0

iv) Perform fasttext modelling with hyperparameters and evaluating performance metrics

```
test_num3, precision3, recall3 = model2.test('ToolsTest1.test')
f_score3 = (2*precision3*recall3)/(precision3+recall3)
print('Test Samples:',test_num3)
print('Precision Score:',precision3)
print('Recall:',recall3)
print('F1 score:',f_score3)
```

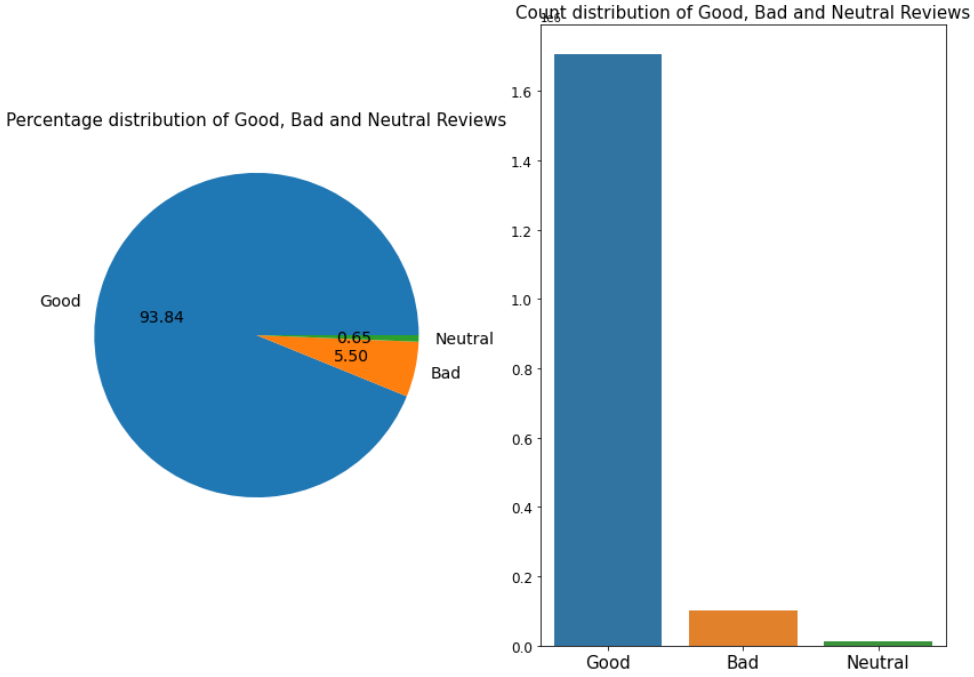
```
Test Samples: 363555
Precision Score: 0.8758784778094099
Recall: 0.8758784778094099
F1 score: 0.8758784778094099
```

Since Fasttext was the best performing model for both categories we shall perform fasttext modelling on the entire dataset and observe trends among sentiments.

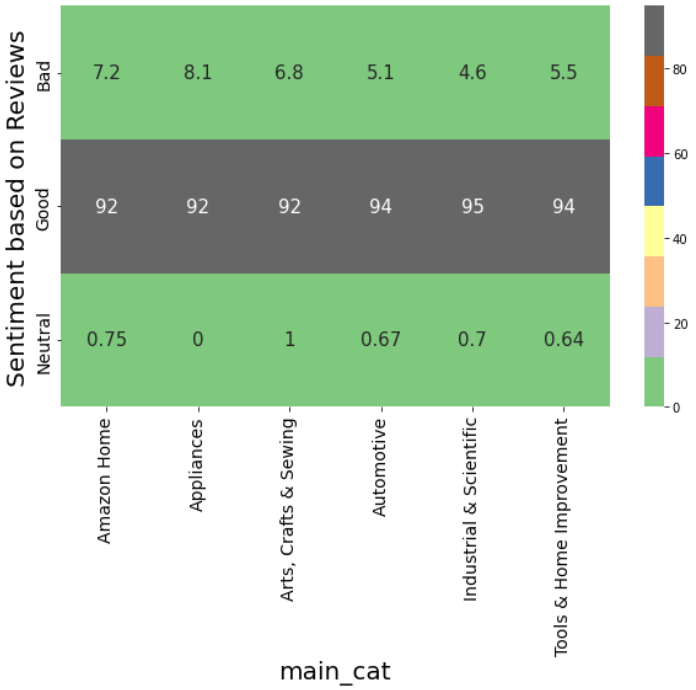
Advantage of using fasttext

The main advantage of FastText embeddings over Word2Vec is to take into account the internal structure of words while learning word representations, which could be very useful for morphologically rich languages, and also for words that occur rarely.

Performing fasttext on Tools & Home Improvement

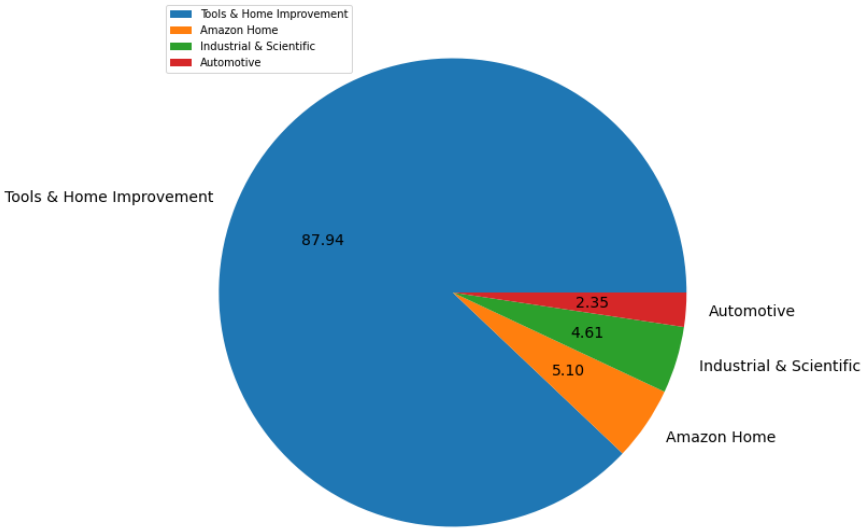


Observing spread of positive, negative and neutral reviews

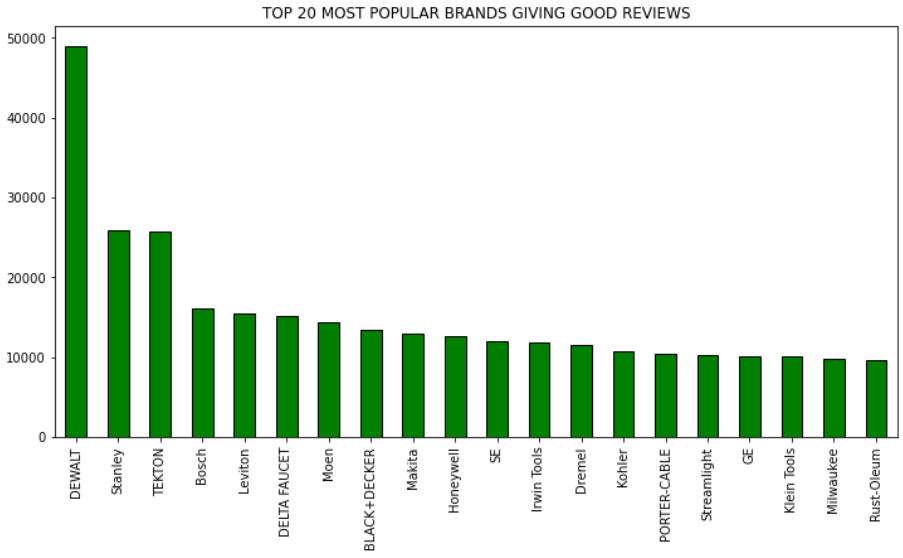


i) Observing trends among positive reviews

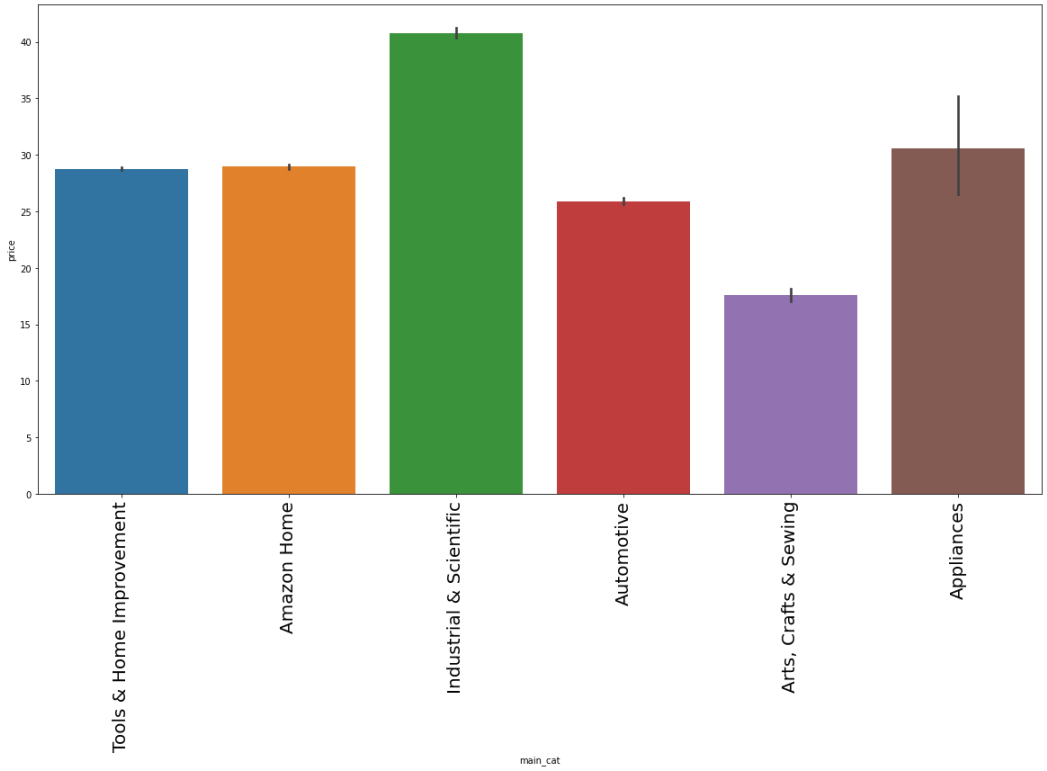
a) Category distribution



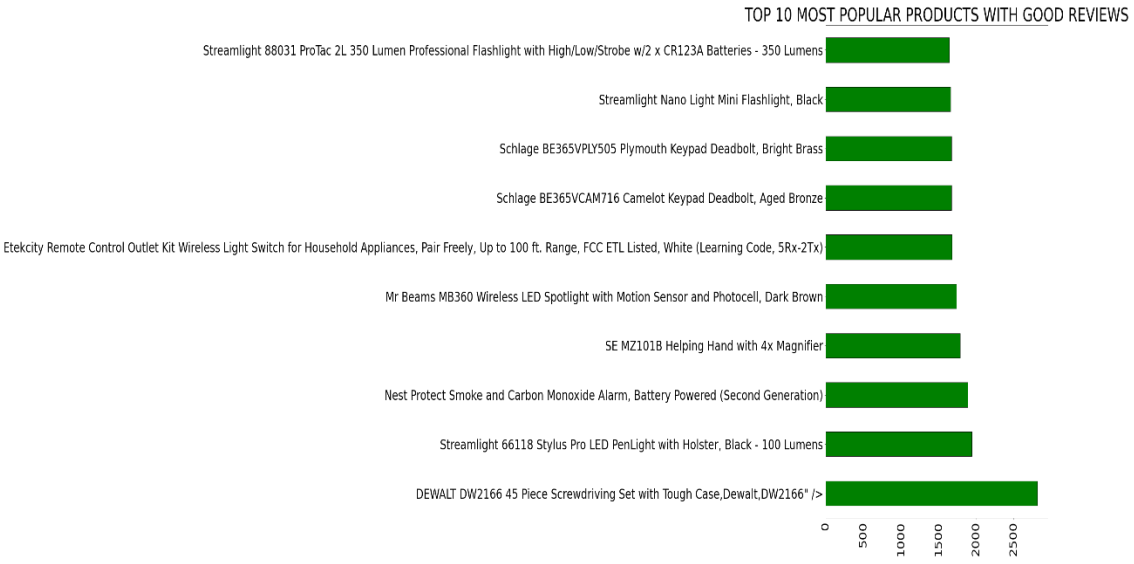
b) Top 20 most popular brands



c) Distribution of price with respect to category

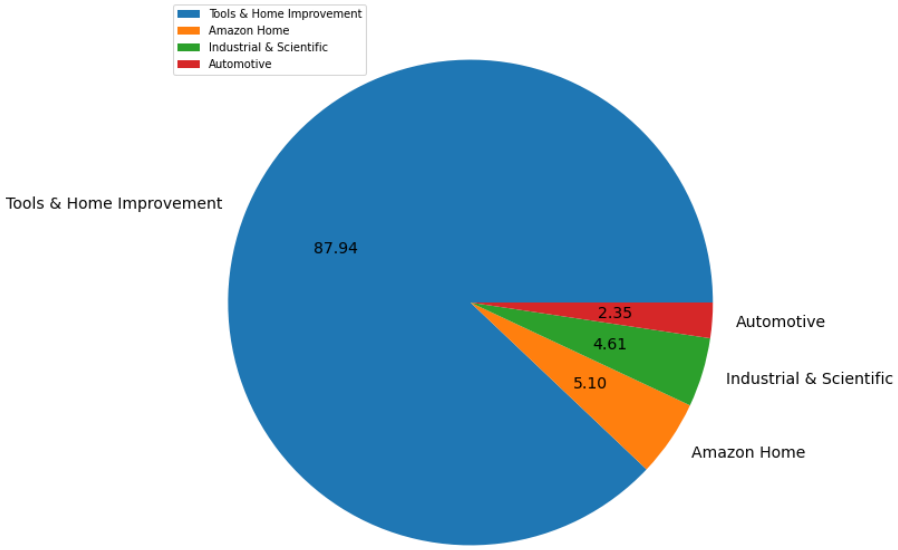


d) Top 10 most popular products

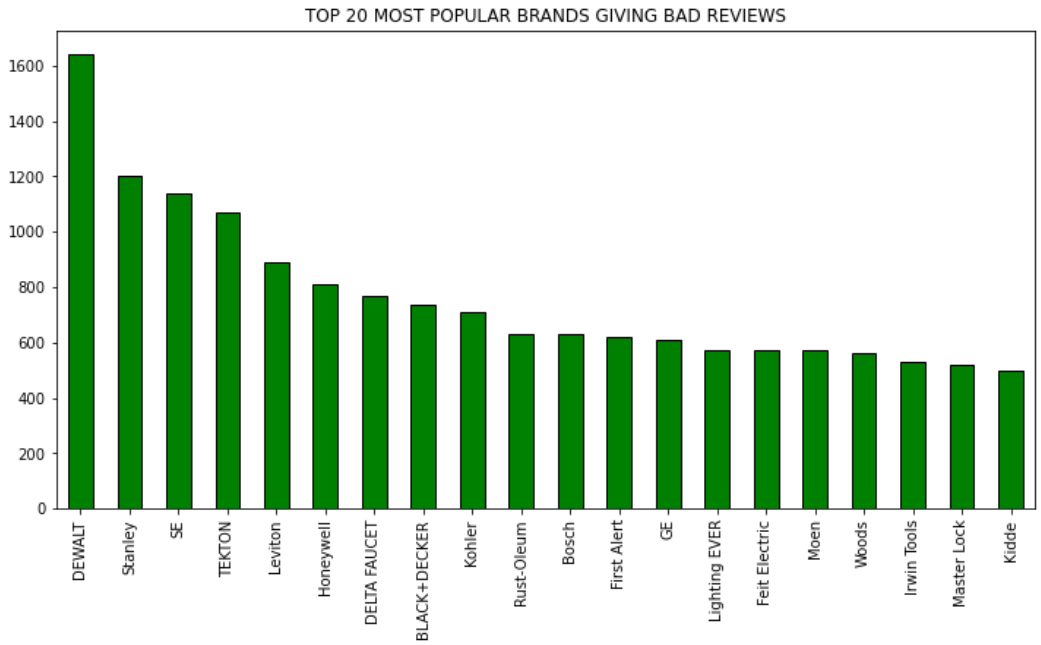


ii) Observing trends among negative reviews

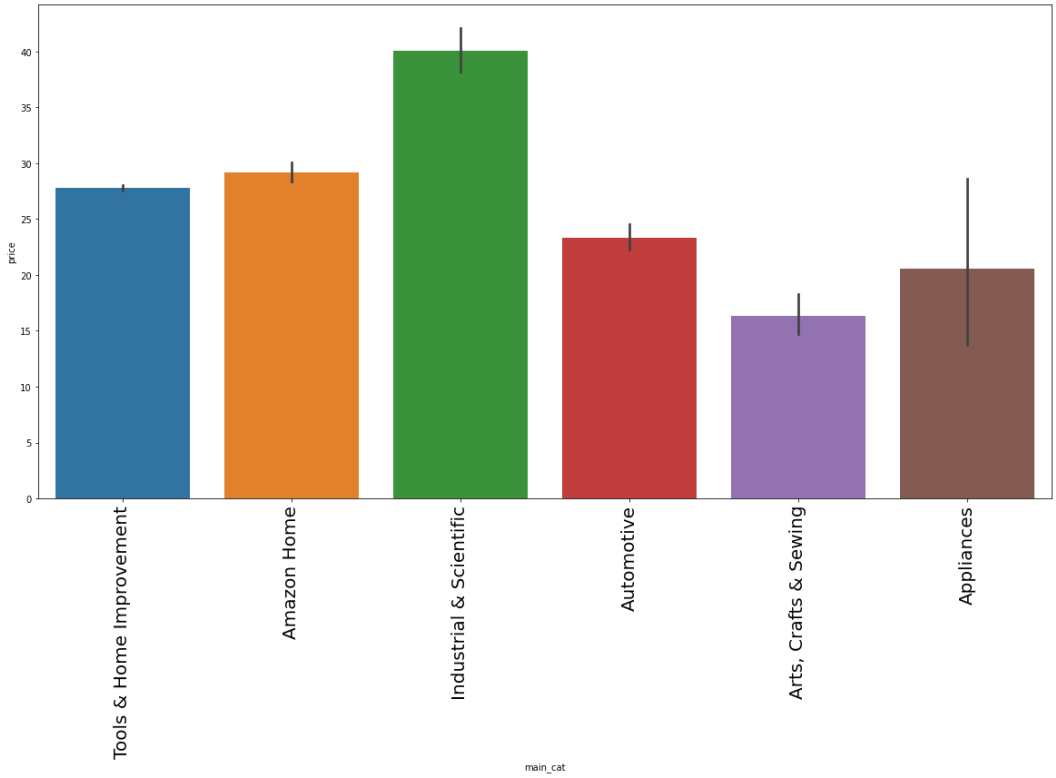
a) Category distribution



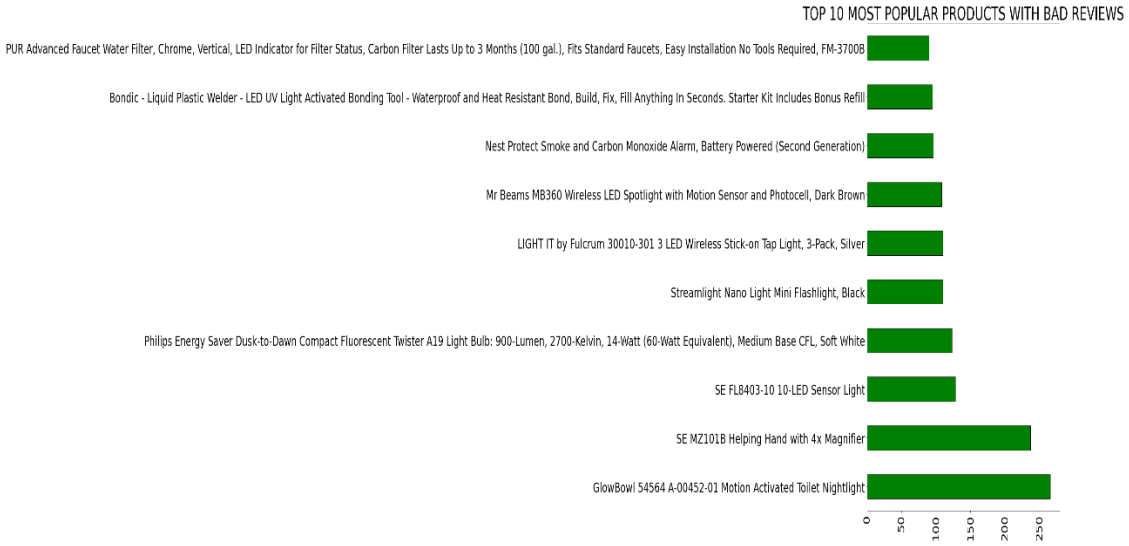
b) Top 20 most sought after brands



c) Distribution of price with respect to category

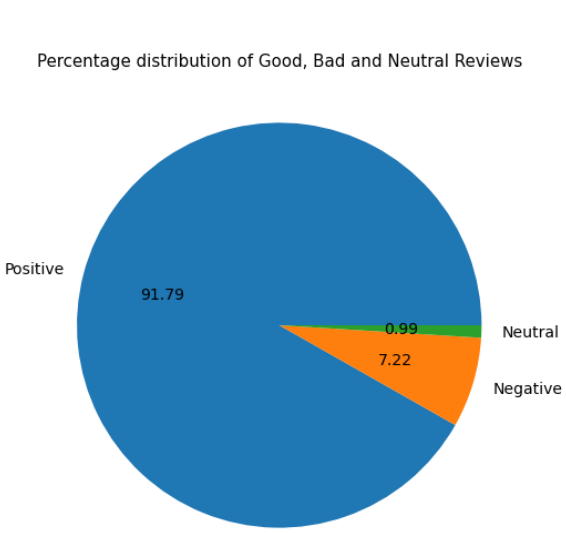


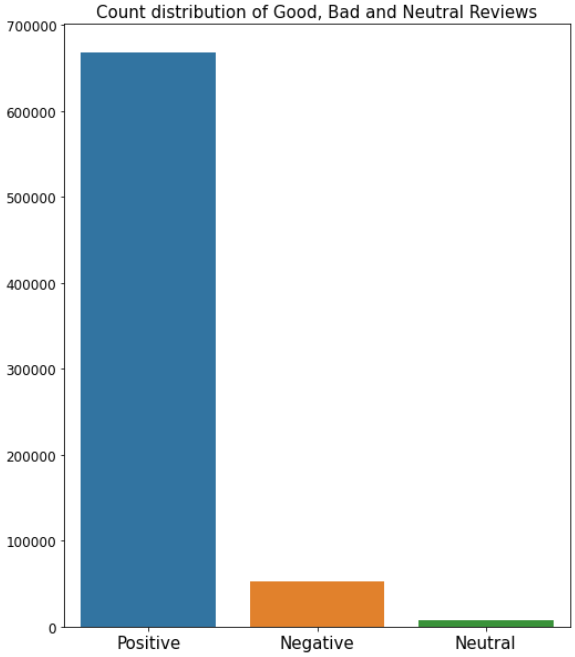
d) Top 10 most sought after products



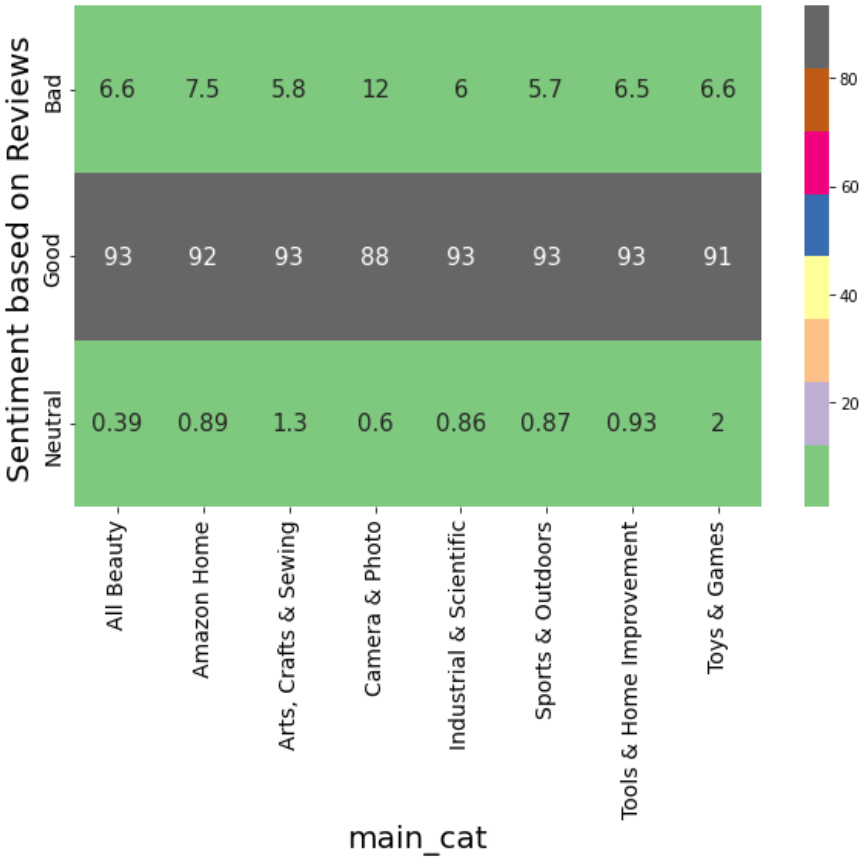
Since, the number of neutral reviews are less than 1%, there is no need to analyse neutral reviews.

A) Patio Lawn & Garden



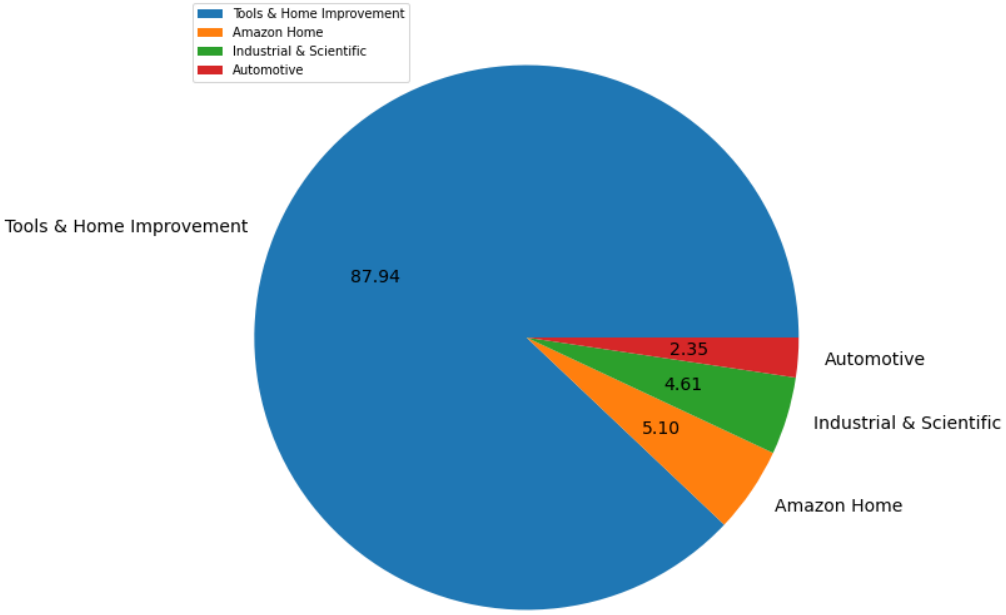


Observing spread of positive, negative and neutral reviews

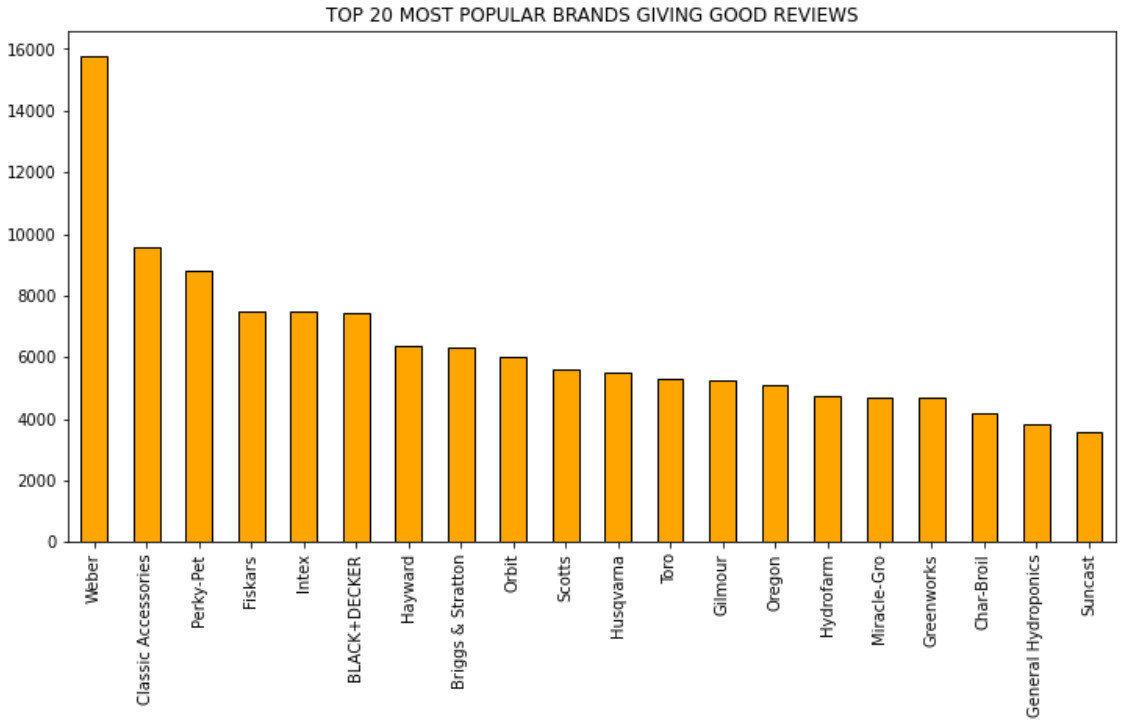


i) Observing trends among positive reviews

a) Category distribution



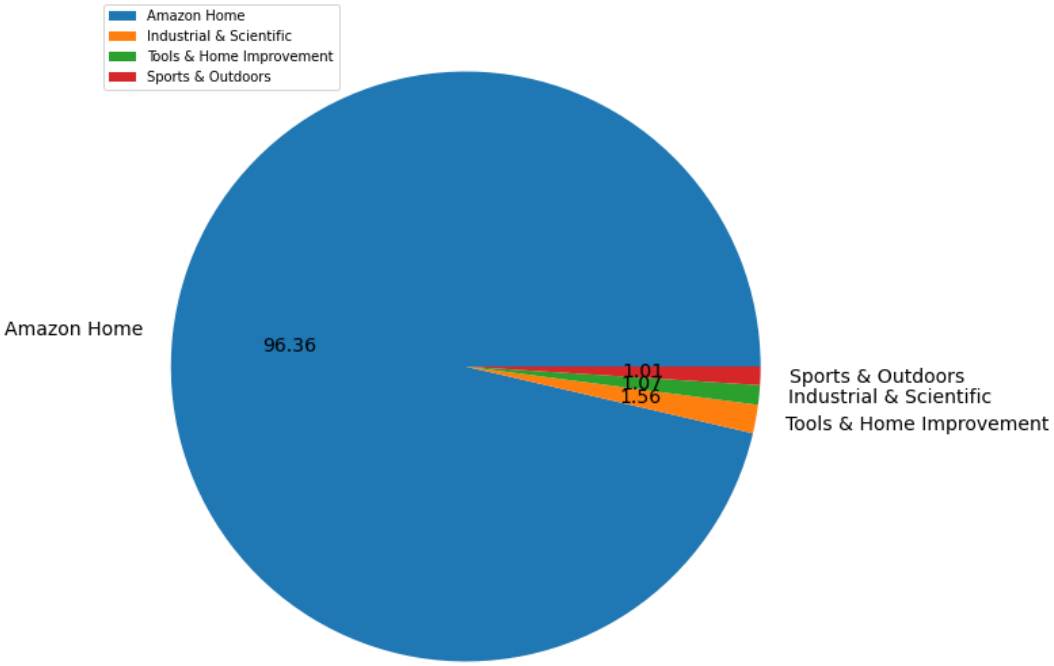
b) Top 20 most popular brands



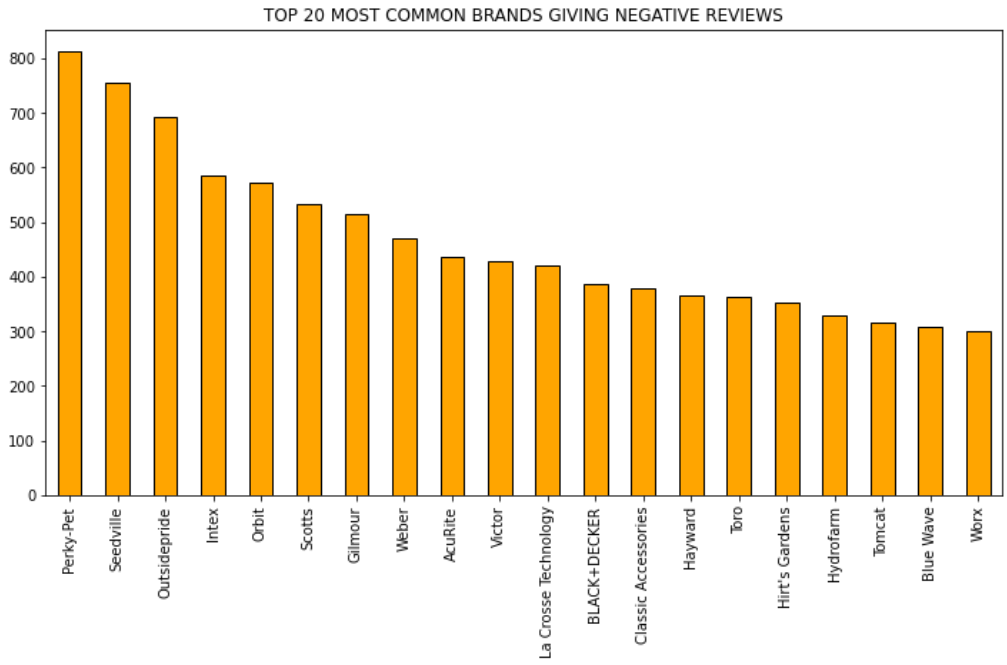
c) Top 10 most popular products

ii) Observing trends among negative reviews

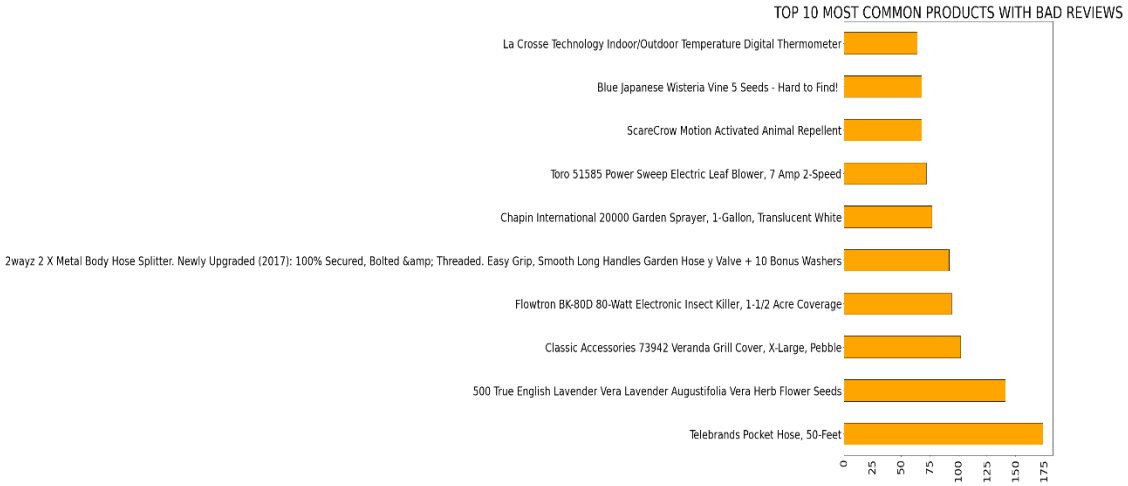
a) Category distribution



b) Top 20 most sought after brands



c) Top 10 most sought after products



Since, neutral reviews account for only 1% of total reviews, there is no need to analyse neutral reviews.

CLUSTERING PRODUCTS ON BASIS OF PRODUCT PRICE AND PRODUCT RATINGS (KMeans Clustering)

A) Tools & Home Improvement

i) We begin by making a pivot table representing product names with their average rating and prices.

```
In [76]: df_pivot=df_tools.pivot_table(index=['title'],values=['rating','price'],aggfunc='mean')
         df_pivot
```

Out[76]:

title	price	rating
18" Garage Door Torsion Spring Winding Bars	16.820000	4.812500
300W Low Voltage Landscape Light Transformer 12V	145.000000	4.900000
360 Siphon RV Fume Extractor - White	38.556766	4.458333
3M 6200 Half Mask for Use With 6000 Series Cartridges, Face Piece	1.960000	4.756757
4 Way Diamond Knife Sharpener Hone Sharpening Stone New	23.980000	4.125000
...	...	...
{New Version 6led} Solar Powered Outdoor Lights -Weatherproof - No Tools Required; Motion Sensor - Detector Activated for Garden, Patio, Stair, Deck, Street, Garage -(Power Saving--no Dim Light Mode)--no Battery Required	199.990000	4.800000
{Upgraded} Powerextra 3000mah Twin Pack Compatible with Lincoln Grease Guns 12V Ni-MH Backup Battery Replace 1201 LIN-1201 218-787	47.990000	4.833333
{Upgraded}Powerextra 12V 3000mAh Replacement Power Tool Battery for MAKITA 5092D, 5092DW, 6011D, 6011DW	1.410000	4.615385
{Upgraded}Powerextra 2 Pack Lincoln 12V 3000mAh Replacement Battery Compatible with LIN-1244 LIN-1242 LIN-1201(with a free voltage tester pen)	48.990000	4.600000
{Upgraded}Powerextra 2500mah 40 Volt MAX Replacement Battery Compatible with Black&Decker LBX2040 LBX36 LBXR36 LBXR2036 40V Lithium Ion Battery	37.990000	4.631579

67549 rows × 2 columns

ii) We now place the resultant pivot table in the form of a data frame.

```
In [77]: df_prod_avg=pd.DataFrame()
         df_prod_avg['title']=list(df_pivot.index)
         df_prod_avg['Avg_Price']=df_pivot[['price']].values
         df_prod_avg['Average Rating']=df_pivot[['rating']].values
```

```
In [78]: df_prod_avg
```

Out[78]:

	title	Avg_Price	Average Rating
0	18" Garage Door Torsion Spring Winding Bars	16.820000	4.812500
1	300W Low Voltage Landscape Light Transformer ...	145.000000	4.900000
2	360 Siphon RV Fume Extractor - White	38.556766	4.458333
3	3M 6200 Half Mask for Use With 6000 Series Ca...	1.960000	4.756757
4	4 Way Diamond Knife Sharpener Hone Sharpening...	23.980000	4.125000
...	...	...	...
67544	{New Version 6led} Solar Powered Outdoor Light...	199.990000	4.800000
67545	{Upgraded} Powerextra 3000mah Twin Pack Compat...	47.990000	4.833333
67546	{Upgraded}Powerextra 12V 3000mAh Replacement P...	1.410000	4.615385
67547	{Upgraded}Powerextra 2 Pack Lincoln 12V 3000mA...	48.990000	4.600000
67548	{Upgraded}Powerextra 2500mah 40 Volt MAX Repla...	37.990000	4.631579

67549 rows × 3 columns

iii) We now perform an inner merge with our original data frame

```
In [79]: tools_df_cluster=pd.merge(df_tools,df_prod_avg,on=('title'),how='inner')

In [80]: tools_df_cluster.head()
```

Out[80]:

	title	brand	main_cat	price	product_id	Reviewer_id	rating	verified	reviewText	DATE	Avg_Price	Average Rating
0	Breeding Organic Vegetables: A Step-by-Step Gu...	SioGreen	Tools & Home Improvement	45.902274	0982085028	AL19QO4XLBQPU	5	True	returned, decided against this product	2018-01-28	45.902274	4.555556
1	Breeding Organic Vegetables: A Step-by-Step Gu...	SioGreen	Tools & Home Improvement	45.902274	0982085028	A1I7CVB7X3T81E	5	True	Awesome heater for the electrical requirements...	2017-11-30	45.902274	4.555556
2	Breeding Organic Vegetables: A Step-by-Step Gu...	SioGreen	Tools & Home Improvement	45.902274	0982085028	A1AQXO4P5U674E	5	True	Keeps the mist of your wood trim and on you. B...	2017-09-12	45.902274	4.555556
3	Breeding Organic Vegetables: A Step-by-Step Gu...	SioGreen	Tools & Home Improvement	45.902274	0982085028	AIRV678P7C4NK	4	True	So far I hooked it up and tested it , filled a...	2017-07-19	45.902274	4.555556
4	Breeding Organic Vegetables: A Step-by-Step Gu...	SioGreen	Tools & Home Improvement	45.902274	0982085028	A22I5QDNTNECDW	1	True	i installed this 10 months ago, instructions w...	2017-05-25	45.902274	4.555556

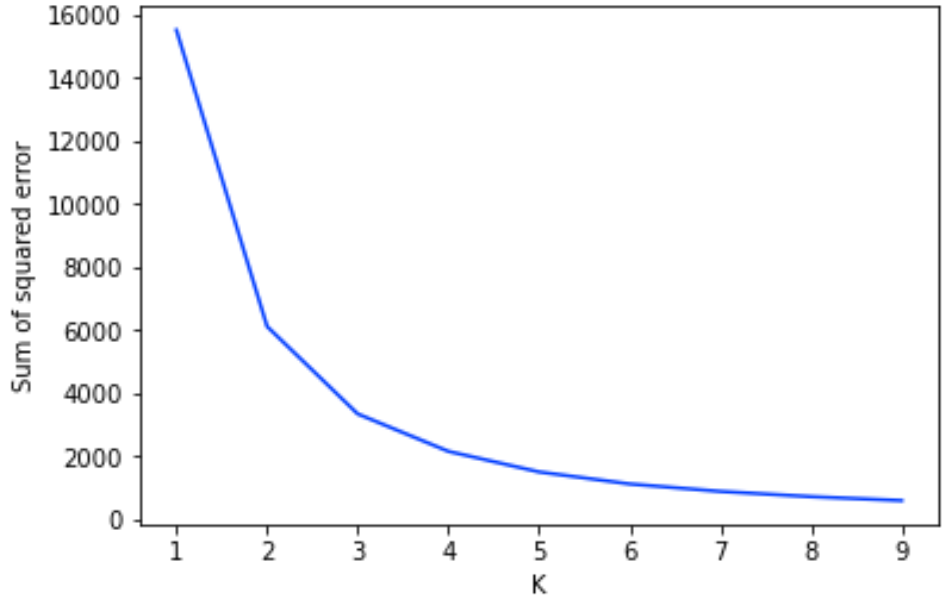
iv) We hall only consider Avg_Price and Average Rating columns for our cluster formation.
The k-Means algorithm is not applicable to categorical data, as categorical variables are discrete and do not have any natural origin. So computing euclidean distance for such as space is not meaningful.

v) Perform min-max scaling on data.

```
In [84]: scaler=MinMaxScaler()

In [85]: df_tools_cluster_scaled=pd.DataFrame(scaler.fit_transform(df_tools_cluster.to_numpy()),columns=df_tools_cluster.columns)
```

vi) We observe our elbow curve plot.

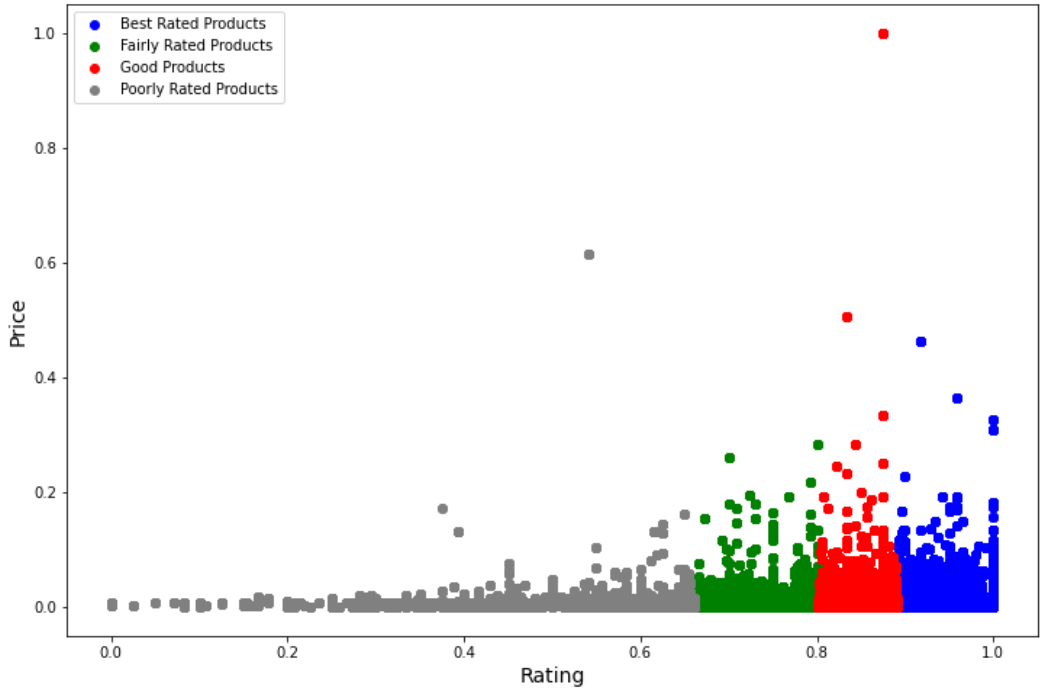
```
In [88]: sse = []
         k_rng = range(1,10)
         for k in k_rng:
             km = KMeans(n_clusters=k,random_state=10)
             km.fit(X)
             sse.append(km.inertia_)                    # sum of squared error
```



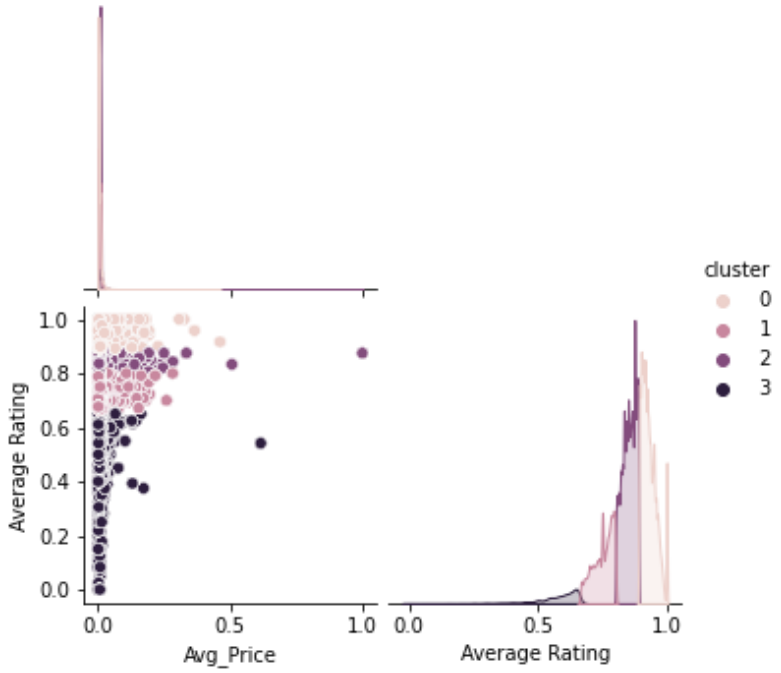
From our plot we can say that the elbow point occurs at point 3 or 4.

vii) We now check for silhouette scores. For k=3, silhouette score was 0.546 and for k=4, silhouette score was 0.565. Therefore, we consider our ideal number of clusters as 4.

viii) Observe the cluster formation and give them labels.



ix) View the pairlot formed.



x) Add predicted clusters to original data frame and segregate based on clusters.

```
In [103]: df_tools['Cluster']=yp
```

```
In [104]: df_tools.head()
```

Out[104]:

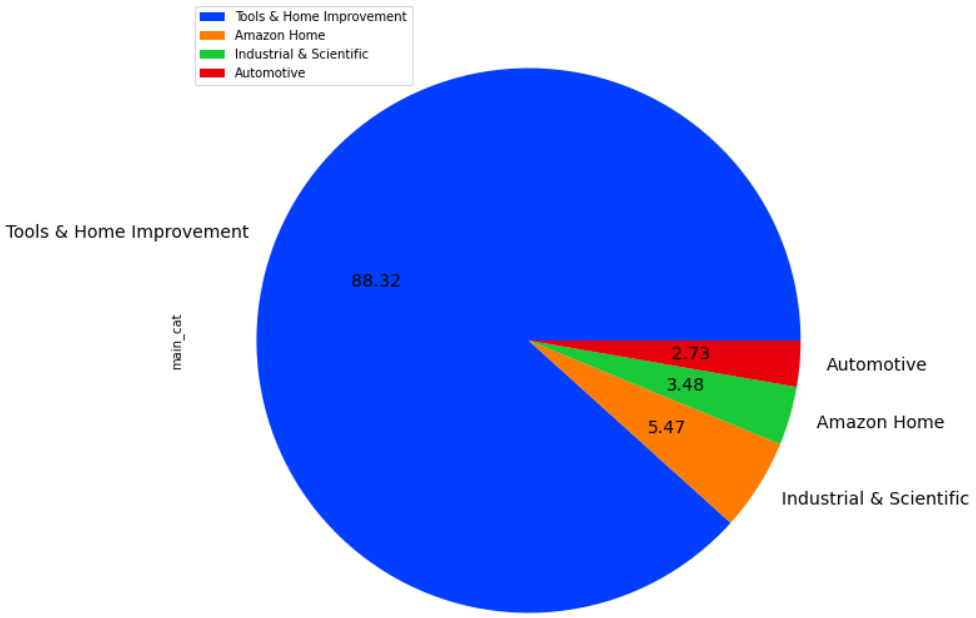
	title	brand	main_cat	price	product_id	Reviewer_id	rating	verified	reviewText	DATE	Cluster
0	Breeding Organic Vegetables: A Step-by-Step Gu...	SioGreen	Tools & Home Improvement	45.902274	0982085028	AL19QO4XLBQPU	5	True	returned, decided against this product	2018-01-28	2
1	Breeding Organic Vegetables: A Step-by-Step Gu...	SioGreen	Tools & Home Improvement	45.902274	0982085028	A1I7CVB7X3T81E	5	True	Awesome heater for the electrical requirements...	2017-11-30	2
2	Breeding Organic Vegetables: A Step-by-Step Gu...	SioGreen	Tools & Home Improvement	45.902274	0982085028	A1AQXO4P5U674E	5	True	Keeps the mist of your wood trim and on you. B...	2017-09-12	2
3	Breeding Organic Vegetables: A Step-by-Step Gu...	SioGreen	Tools & Home Improvement	45.902274	0982085028	AIRV678P7C4NK	4	True	So far I hooked it up and tested it , filled a...	2017-07-19	2
4	Breeding Organic Vegetables: A Step-by-Step Gu...	SioGreen	Tools & Home Improvement	45.902274	0982085028	A22I5QDNTNECDW	1	True	i installed this 10 months ago, instructions w...	2017-05-25	2

```
In [105]: df_best_rated_cluster = df_tools[df_tools['Cluster']==0]
df_fairly_rated_cluster = df_tools[df_tools['Cluster']==1]
df_well_rated_cluster = df_tools[df_tools['Cluster']==2]
df_poorly_rated_cluster = df_tools[df_tools['Cluster']==3]
```

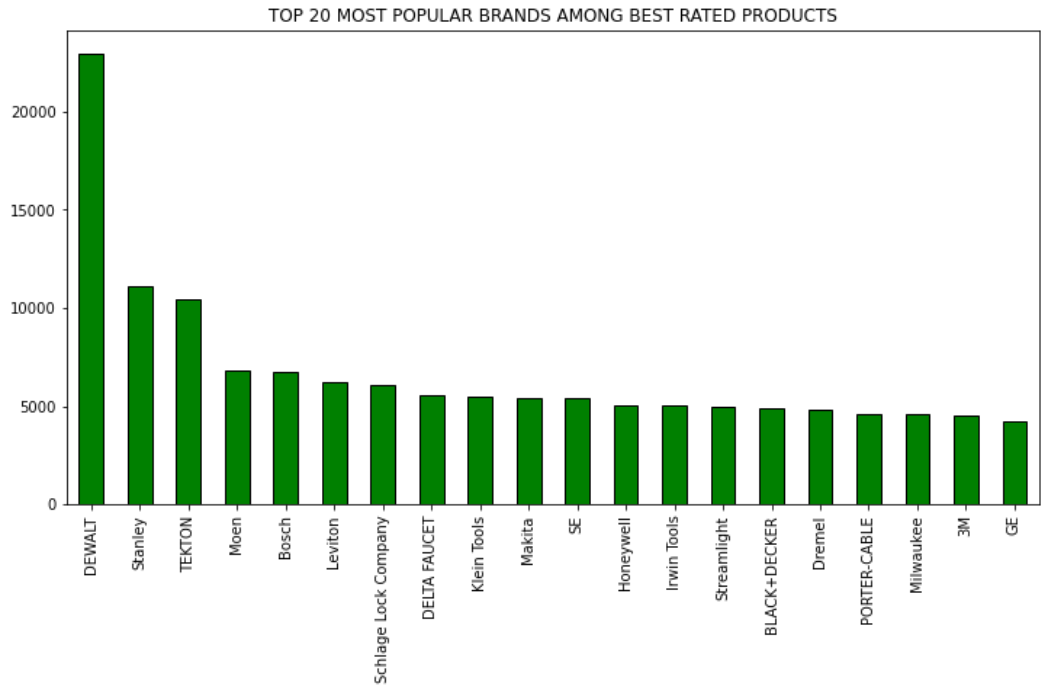
xi) Observe trends among the clusters.

d) For Best Rated Products

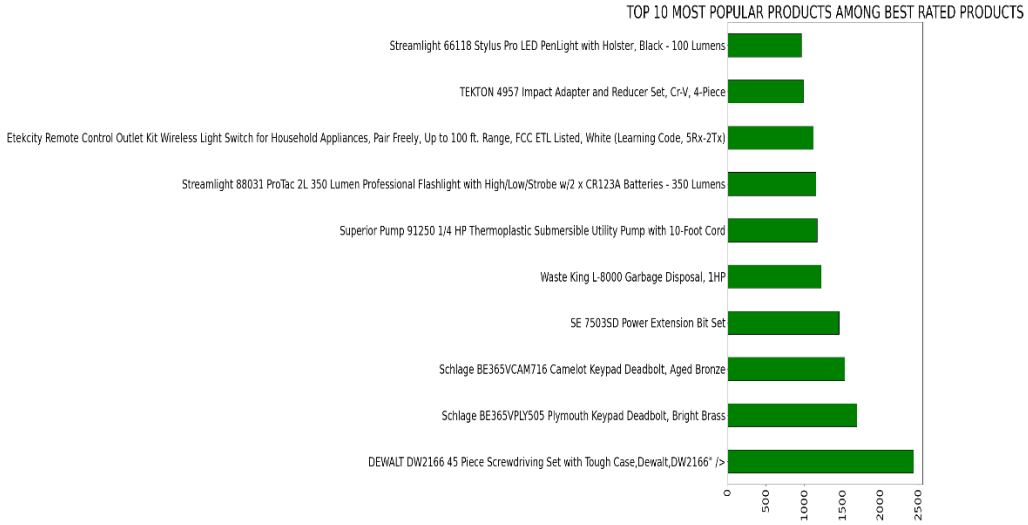
a) Category distribution



b) Top 20 most popular brands

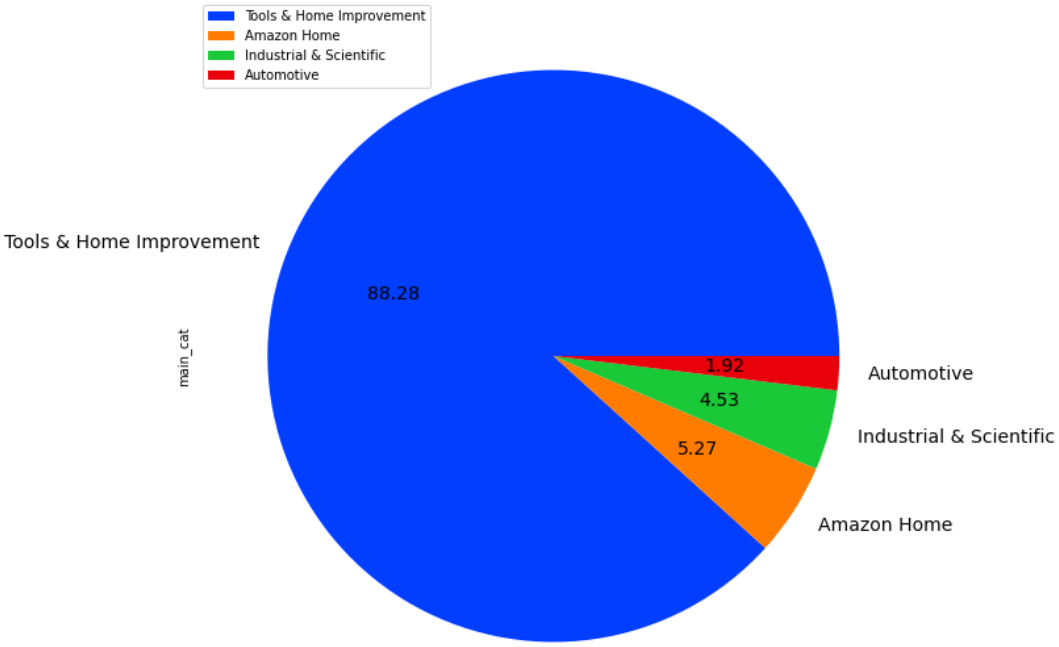


c) Top 10 most popular products

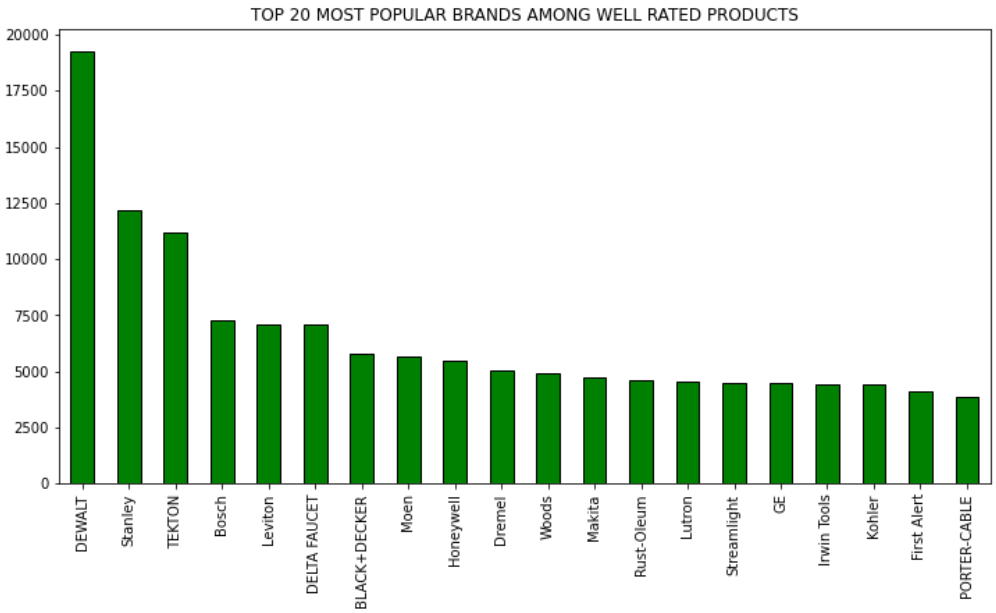


e) For Well Rated Products

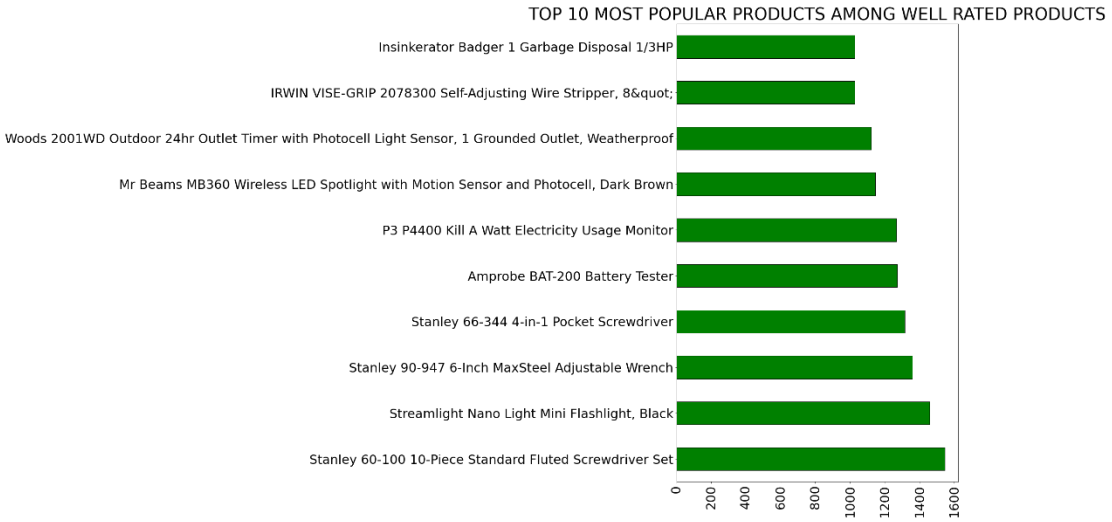
a) Category Distribution



b) Top 20 most popular brands

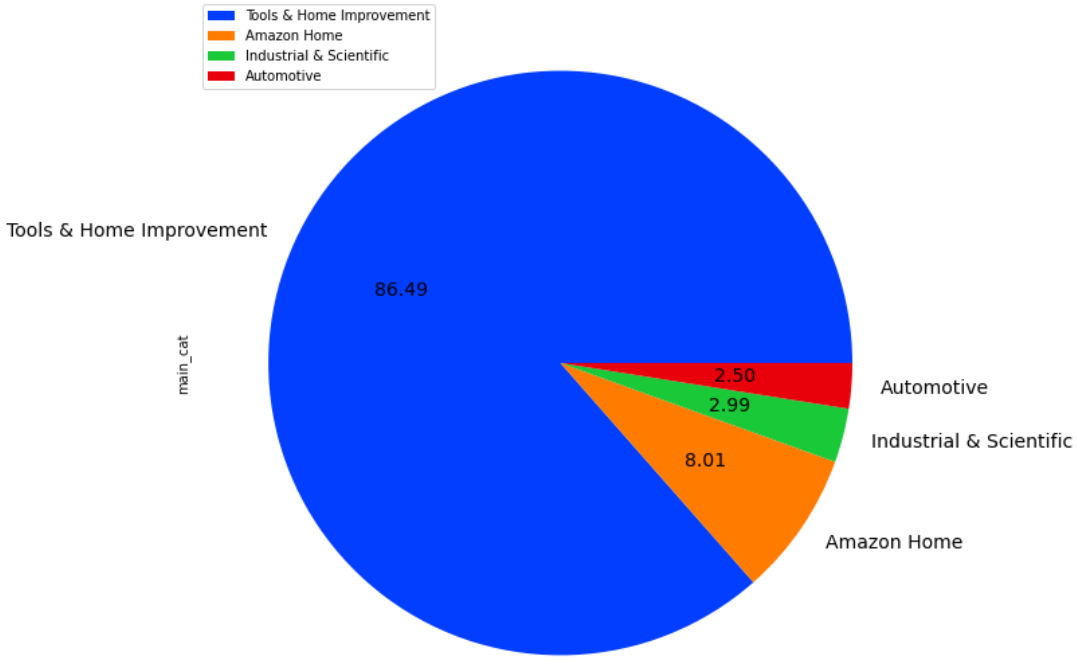


c) Top 10 most popular products

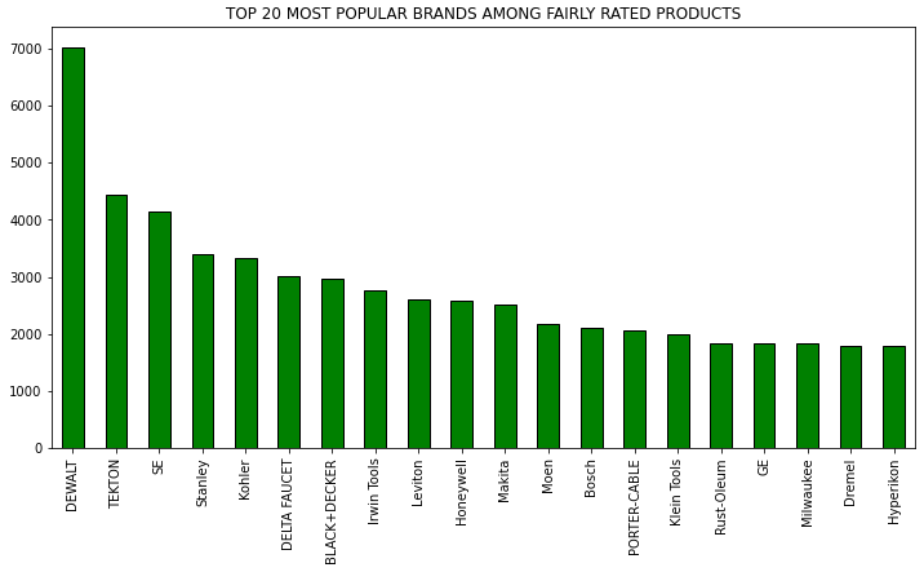


f) For Fairly rated products

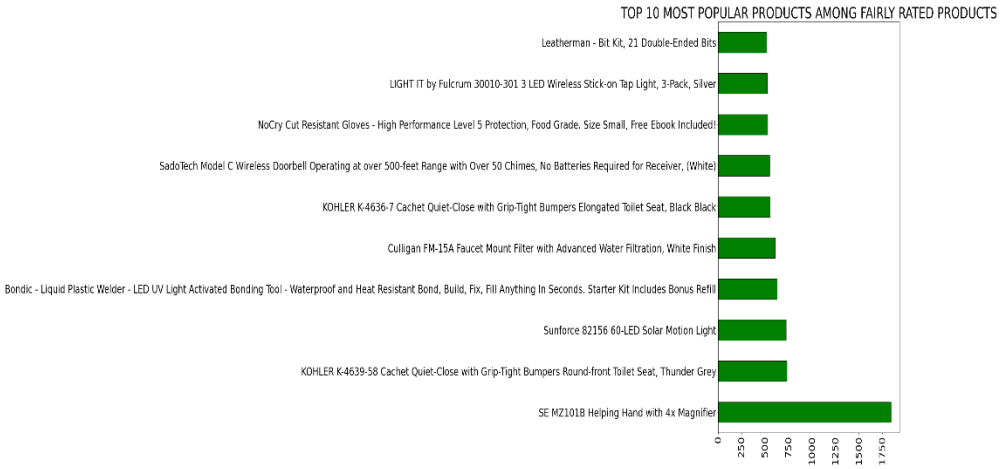
a) Category distribution



b) Top 20 most popular brands

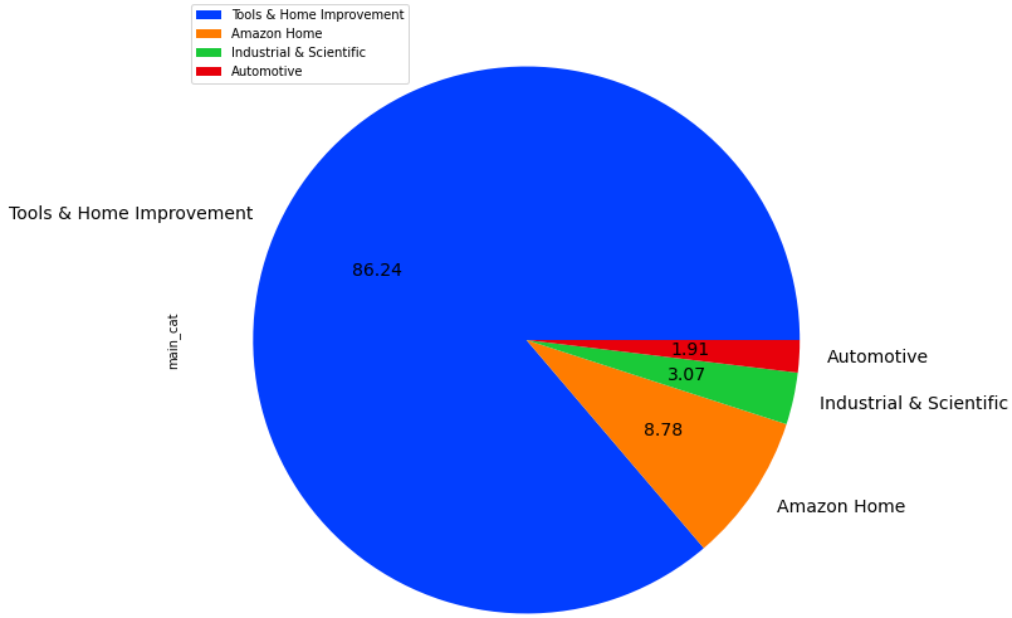


c) Top 10 most popular products

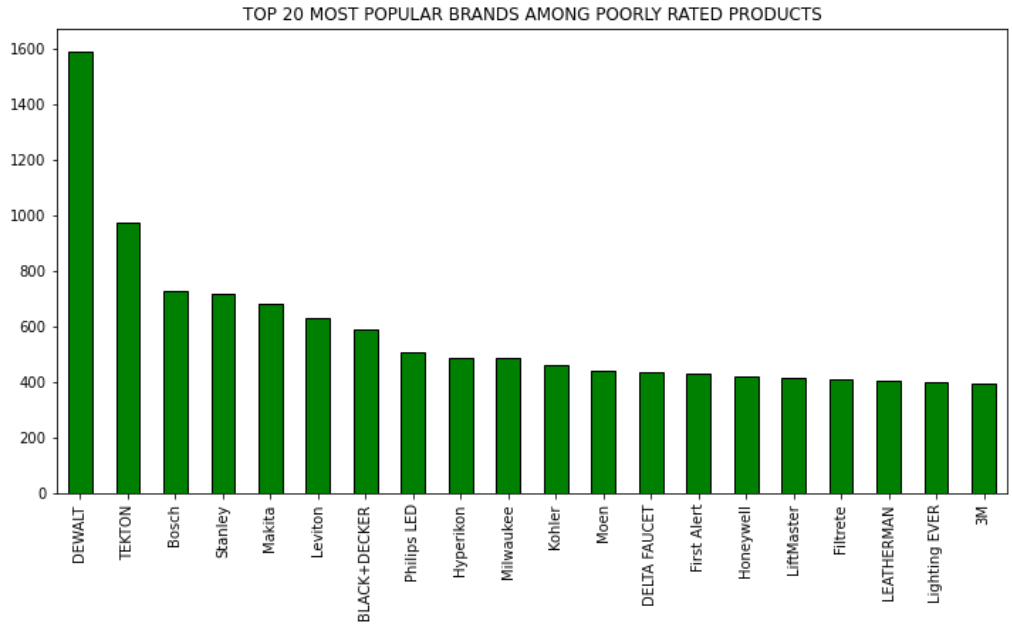


g) For Poorly Rated Products

a) Category Distribution



b) Top 20 most popular brands



c) Top 10 most popular products

B) Patio Lawn & Garden

i) We begin by making a pivot table representing product names with their average rating and prices.

```
df_pivot=df_patio.pivot_table(index=['title'],values=['rating','price'],aggfunc='mean')
df_pivot
```

title	price	rating
1 X Grill Gripper for Ceramic Kamado Grill Such As Big Green Egg, Primo, Grill Dome Etc.	52.730167	4.458333
139573 Replacement belt made to FSP specs. For Craftsman, Poulan, Husqvarna, Wizard, more.	18.870000	4.333333
3, 6, 9, and 12 Volt Solar Panel with 9 Volt and AA Battery Charger	52.730167	3.400000
496914/793281 Briggs and Stratton OEM Coil	19.490000	4.750000
4D Concepts 3-Piece Slate Square Plant Stands with Slate Tops, Metal/Slate	52.730167	4.333333
...	...	...
yueton 20pcs Garden Yard Planter Colorful Whimsical Dragonfly Lawn Stakes Garden Ornaments & Patio Decoration	8.990000	4.428571
yueton Solar Controller 10a 12v/24v Solar Charge Controller Solar Panel Battery Regulator Safe Protection	9.990000	4.300000
zero-G 4001-50 Lightweight, Ultra Flexible, Durable, Kink-Free Garden Hose, 5/8-Inch by 50-Feet	36.980000	4.000000
~ Tibetan Large 12 X 12 Inches Prayer Flags ~ 5 Roll SET ~ Lungta Flag X 105 Prayer Flags, 105 Feet	52.730167	4.571429
~BULK WHOLESALE~ Borage Seed - EDIBLE Blue Flower - Borago officinalis Seeds ~ Improve Tomato Taste - INCREASE SOIL NITRO ~ Blue Flower With Honey Taste ~!! (06000 Seeds - 4 oz)	27.950000	4.888889

31276 rows × 2 columns

ii) We now place the resultant pivot table in the form of a data frame.

```
In [15]: df_prod_avg=pd.DataFrame()
         df_prod_avg['title']=list(df_pivot.index)
         df_prod_avg['Avg_Price']=df_pivot[['price']].values
         df_prod_avg['Average Rating']=df_pivot[['rating']].values

In [16]: df_prod_avg
```

Out[16]:

	title	Avg_Price	Average Rating
0	1 X Grill Gripper for Ceramic Kamado Grill Su...	52.730167	4.458333
1	139573 Replacement belt made to FSP specs. Fo...	18.870000	4.333333
2	3, 6, 9, and 12 Volt Solar Panel with 9 Volt ...	52.730167	3.400000
3	496914/793281 Briggs and Stratton OEM Coil	19.490000	4.750000
4	4D Concepts 3-Piece Slate Square Plant Stands...	52.730167	4.333333
...	...	...	...
31271	yueton 20pcs Garden Yard Planter Colorful Whim...	8.990000	4.428571
31272	yueton Solar Controller 10a 12v/24v Solar Char...	9.990000	4.300000
31273	zero-G 4001-50 Lightweight, Ultra Flexible, Du...	36.980000	4.000000
31274	~ Tibetan Large 12 X 12 Inches Prayer Flags ~ ...	52.730167	4.571429
31275	~BULK WHOLESALE~ Borage Seed - EDIBLE Blue Flo...	27.950000	4.888889

31276 rows × 3 columns

iii) We now perform an inner merge with our original data frame

```
patio_df_cluster=pd.merge(df_patio,df_prod_avg,on=('title'),how='inner')
```

```
patio_df_cluster.head()
```

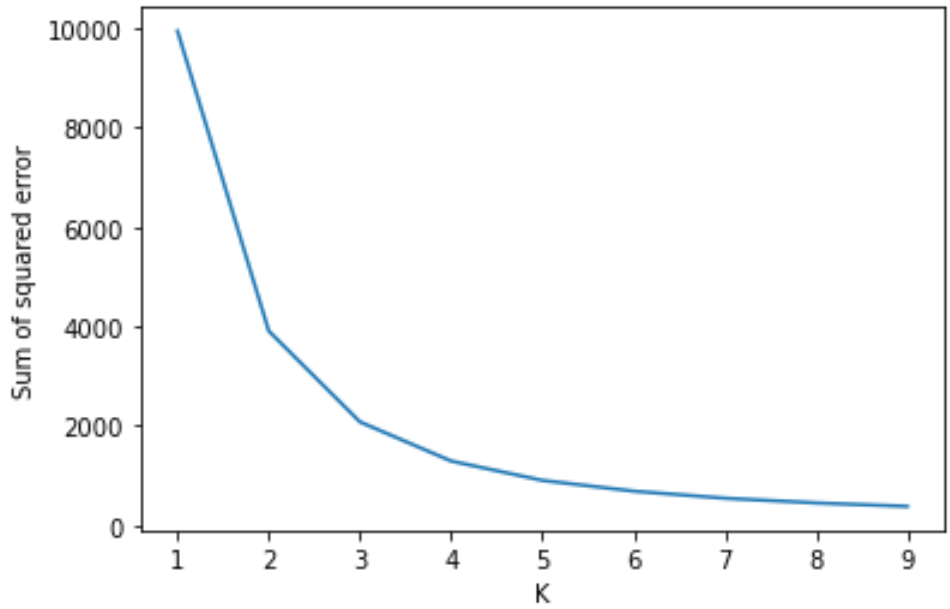
	title	brand	main_cat	price	product_id	Reviewer_id	rating	verified	reviewText	DATE	Avg_Price	Average Rating
0	Colorado US State Flag: 3x5foot poly	3x5fFlags	Amazon Home	52.730167	9539723809	AUVPE2KEXBJVT	5	True	I don't spend a lot on my flags because they r...	2011-12-11	52.730167	4.888889
1	Colorado US State Flag: 3x5foot poly	3x5fFlags	Amazon Home	52.730167	9539723809	A2F6GES1MBOFXS	5	True	Super fast processing and shipping, if you are...	2011-07-30	52.730167	4.888889
2	Colorado US State Flag: 3x5foot poly	3x5fFlags	Amazon Home	52.730167	9539723809	A1SDVD3SZI1BAK	5	True	Great product. I would recommend this product...	2016-05-04	52.730167	4.888889
3	Colorado US State Flag: 3x5foot poly	3x5fFlags	Amazon Home	52.730167	9539723809	A1ZQJ3KCSLUPR3	5	True	GREAT PRICE I LOVE MY STATE AND COUNTRY	2015-11-02	52.730167	4.888889
4	Colorado US State Flag: 3x5foot poly	3x5fFlags	Amazon Home	52.730167	9539723809	ANU8FBZM618M3	5	True	Great display flag for the den.	2015-09-01	52.730167	4.888889

iv) We shall only consider Avg_Price and Average Rating columns for our cluster formation. The k-Means algorithm is not applicable to categorical data, as categorical variables are discrete and do not have any natural origin. So computing euclidean distance for such as space is not meaningful.

v) Perform min-max scaling on data.

```
In [21]: scaler=MinMaxScaler()

In [22]: df_patio_cluster_scaled=pd.DataFrame(scaler.fit_transform(df_patio_cluster.to_numpy()),columns=df_patio_cluster.columns)
```

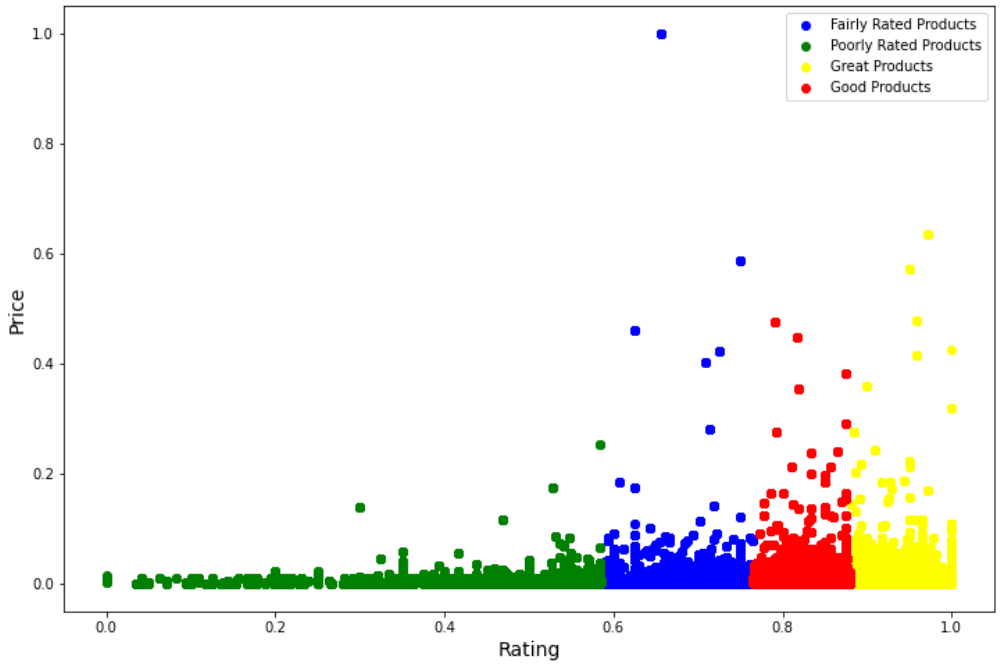
vi) Now, we observe our elbow curve plot.



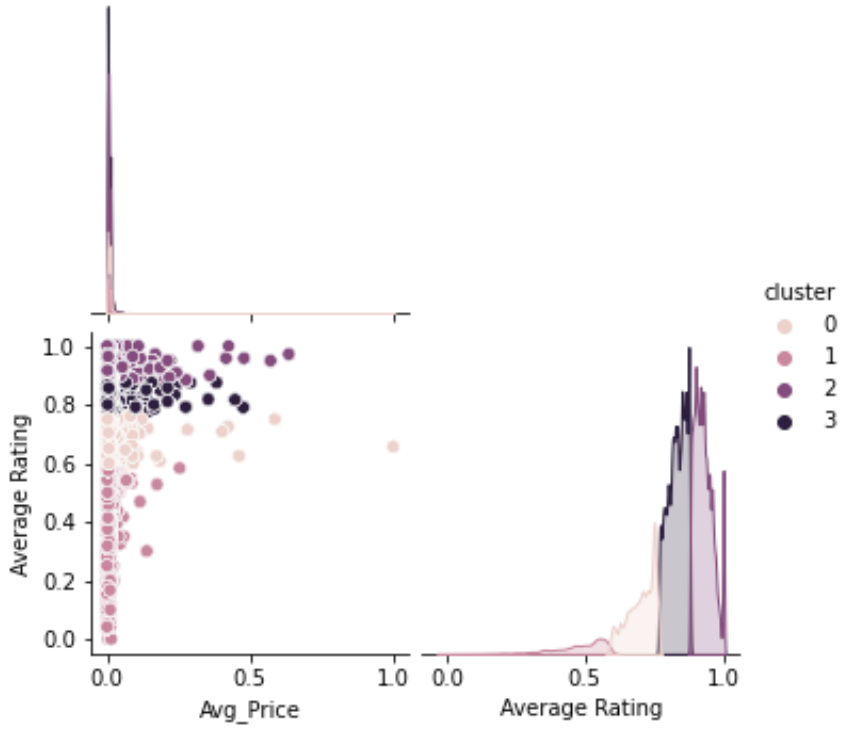
From our plot we can say that the elbow point occurs at point 3 or 4.

vii) We now check for silhouette scores. For k=3, silhouette score was 0.522 and for k=4, silhouette score was 0.531. Therefore, we consider our ideal number of clusters as 4.

viii) Observe the cluster formation and give them labels.



ix) View the pairlot formed.



x) Add predicted clusters to original data frame and segregate based on clusters.

```
df_patio['Cluster']=yp
```

```
df_patio.head()
```

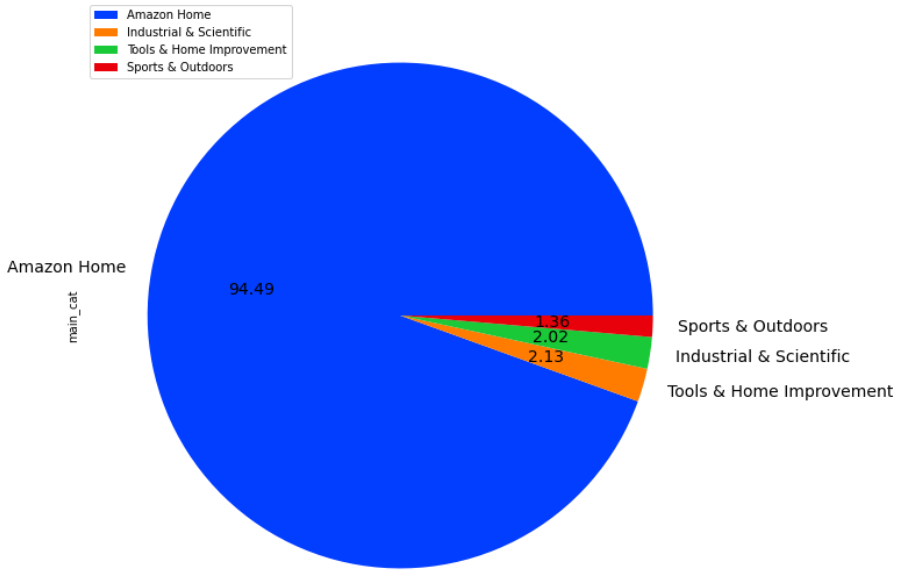
	title	brand	main_cat	price	product_id	Reviewer_id	rating	verified	reviewText	DATE	Cluster
0	Colorado US State Flag: 3x5foot poly	3x5fFlags	Amazon Home	52.730167	9539723809	AUVPE2KEXBJVT	5	True	I don't spend a lot on my flags because they r...	2011-12-11	2
1	Colorado US State Flag: 3x5foot poly	3x5fFlags	Amazon Home	52.730167	9539723809	A2F6GES1MBOFXS	5	True	Super fast processing and shipping, if you are...	2011-07-30	2
2	Colorado US State Flag: 3x5foot poly	3x5fFlags	Amazon Home	52.730167	9539723809	A1SDVD3SZI1BAK	5	True	Great product. I would recommend this product...	2016-05-04	2
3	Colorado US State Flag: 3x5foot poly	3x5fFlags	Amazon Home	52.730167	9539723809	A1ZQJ3KCSLUPR3	5	True	GREAT PRICE I LOVE MY STATE AND COUNTRY	2015-11-02	2
4	Colorado US State Flag: 3x5foot poly	3x5fFlags	Amazon Home	52.730167	9539723809	ANU8FBZM618M3	5	True	Great display flag for the den.	2015-09-01	2

```
df_fairly_rated_cluster = df_patio[df_patio['Cluster']==0]
df_poorly_rated_cluster = df_patio[df_patio['Cluster']==1]
df_best_rated_cluster = df_patio[df_patio['Cluster']==2]
df_well_rated_cluster = df_patio[df_patio['Cluster']==3]
```

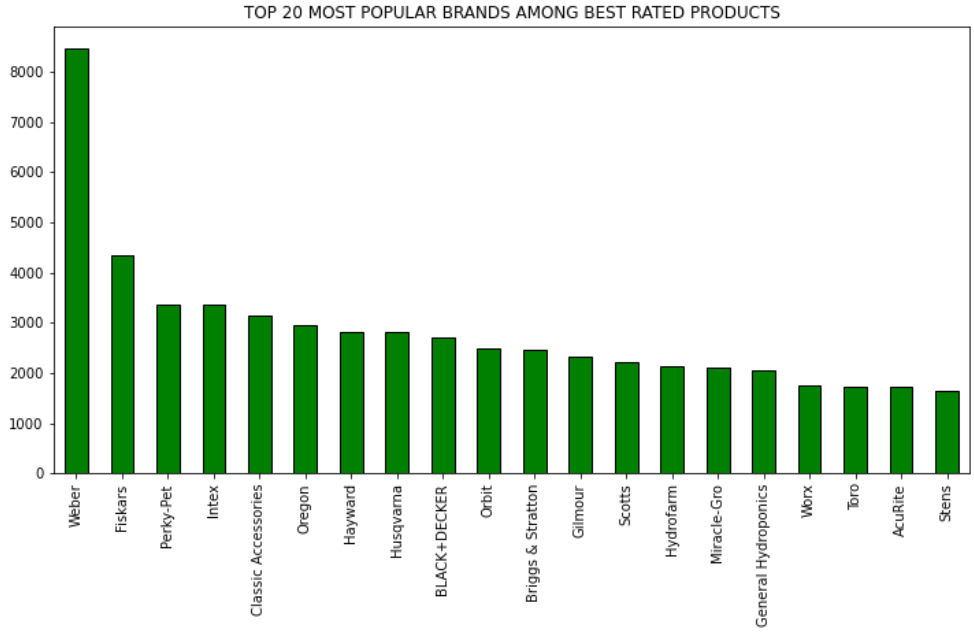
xi) Observe trends among the clusters.

h) For Best Rated Products

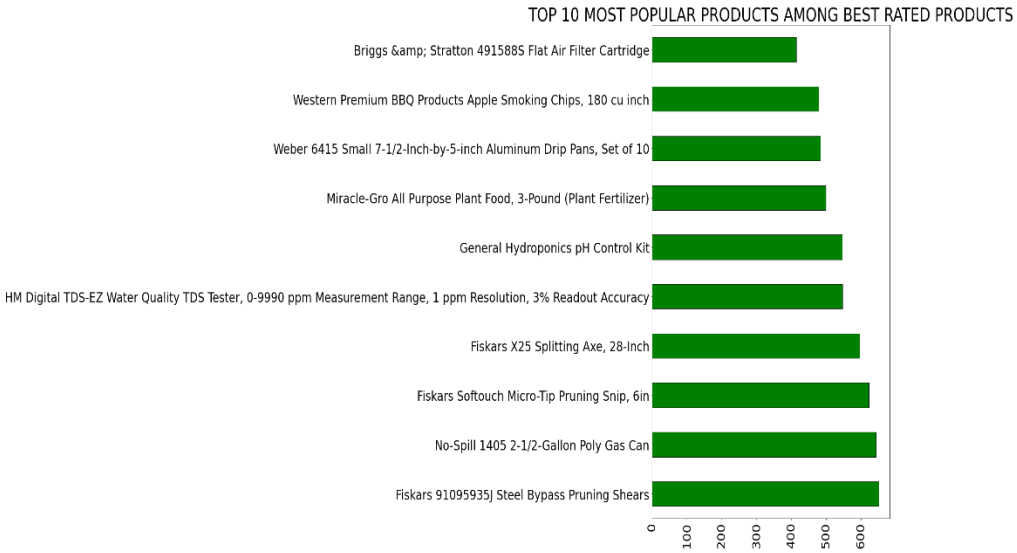
a) Category distribution



b) Top 20 most popular brands

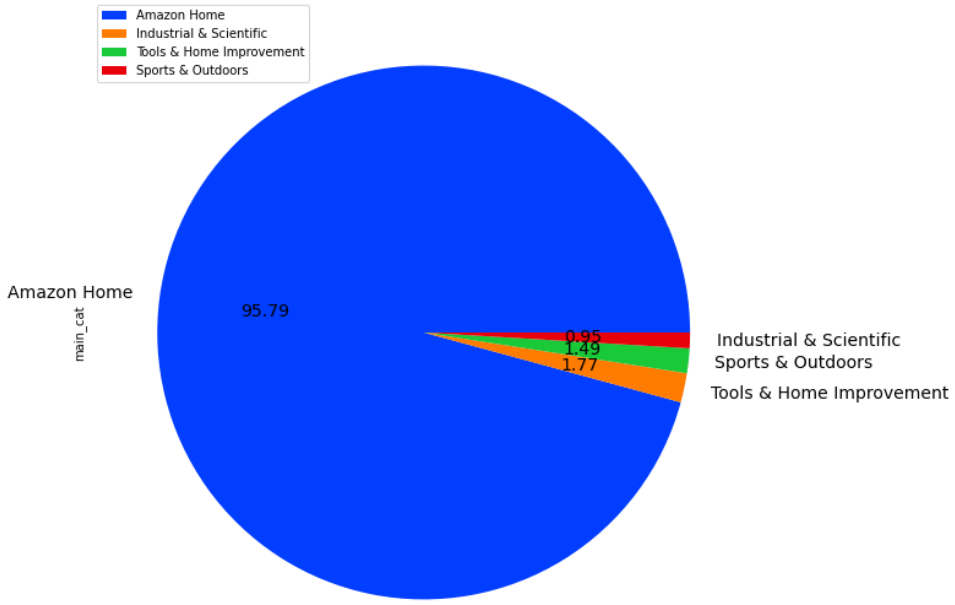


c) Top 10 most popular products

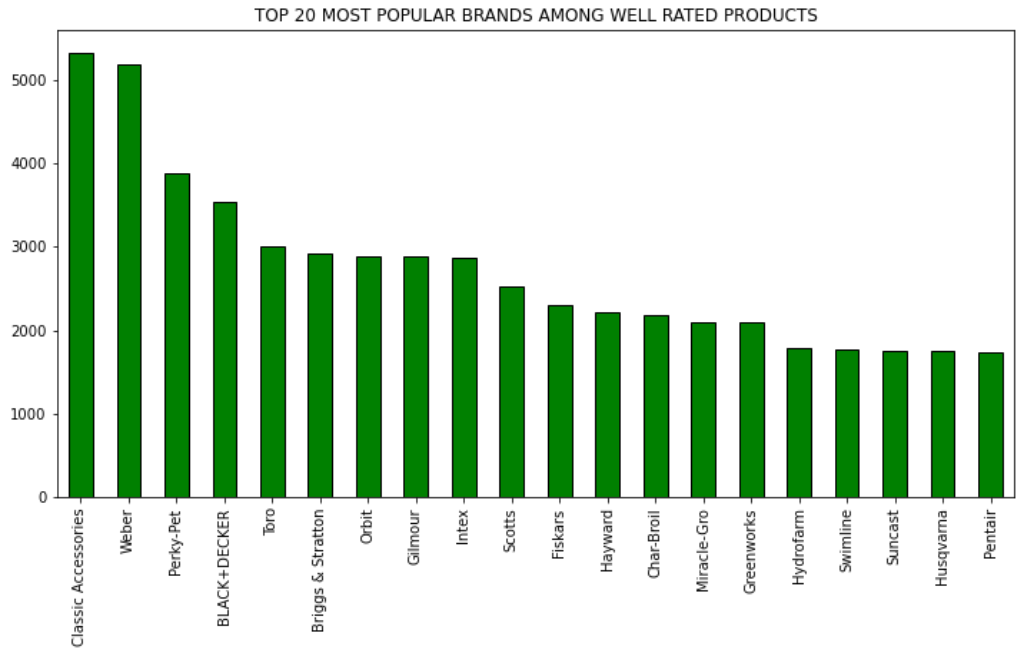


i) For good/well rated products

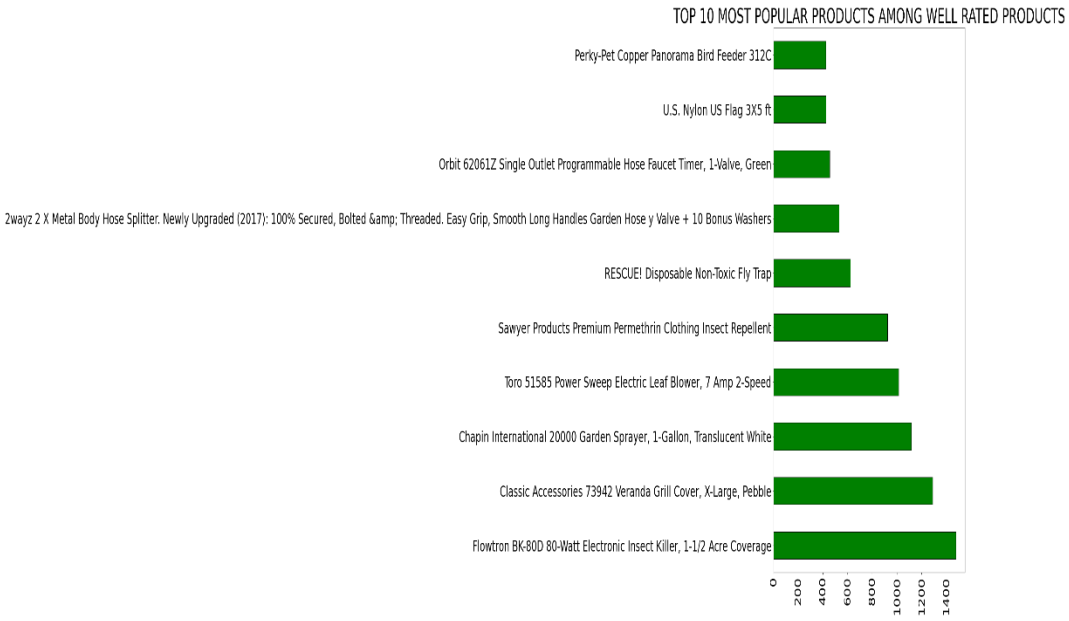
a) Category distribution



b) Top 20 most popular brands

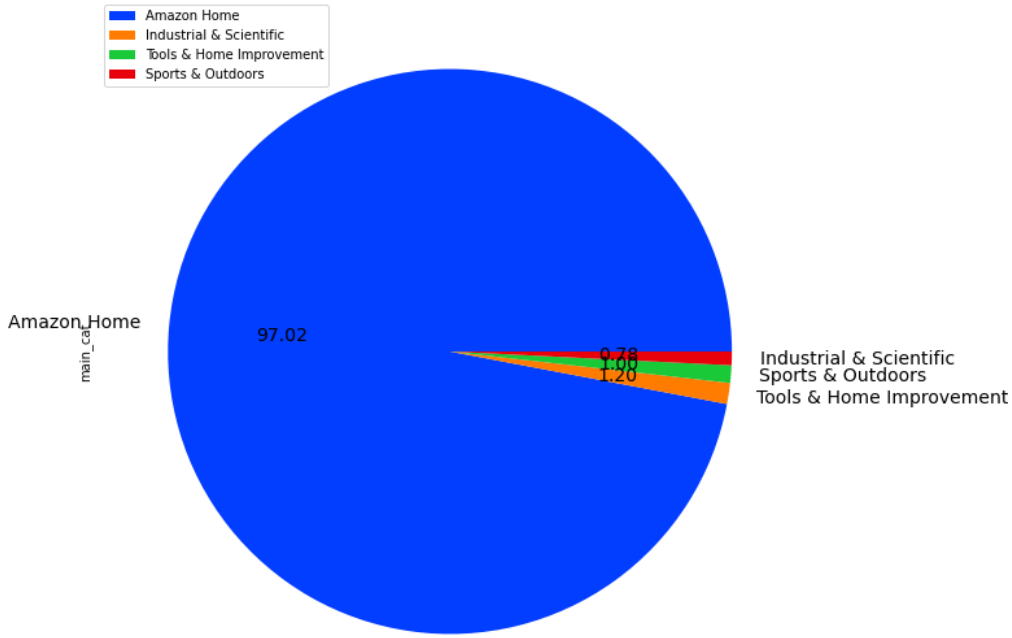


c) Top 10 most popular products

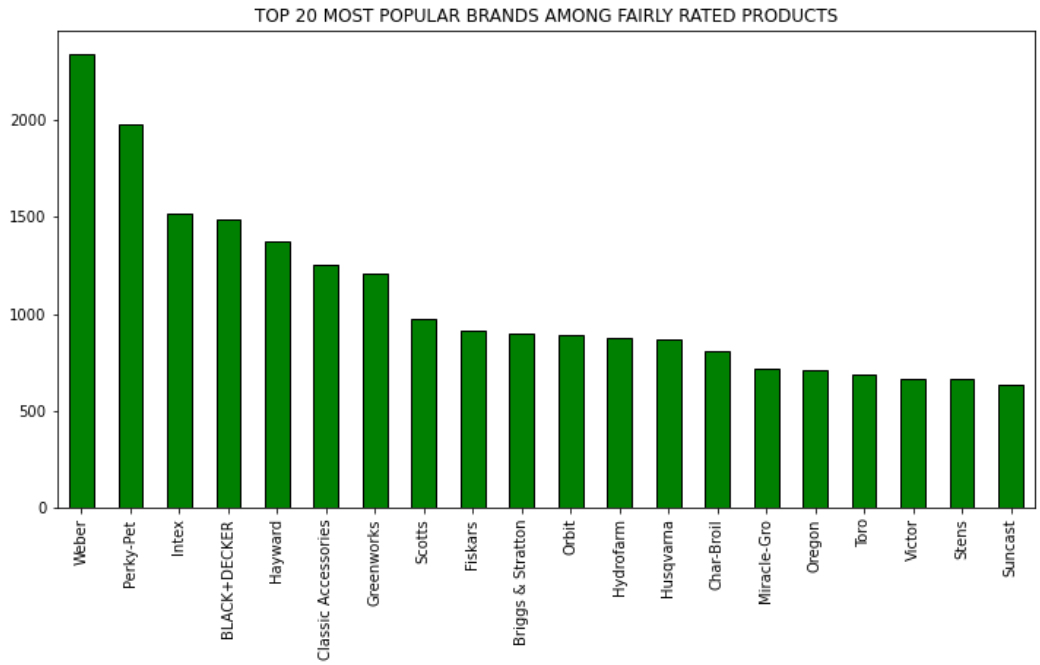


j) For fairly rated products

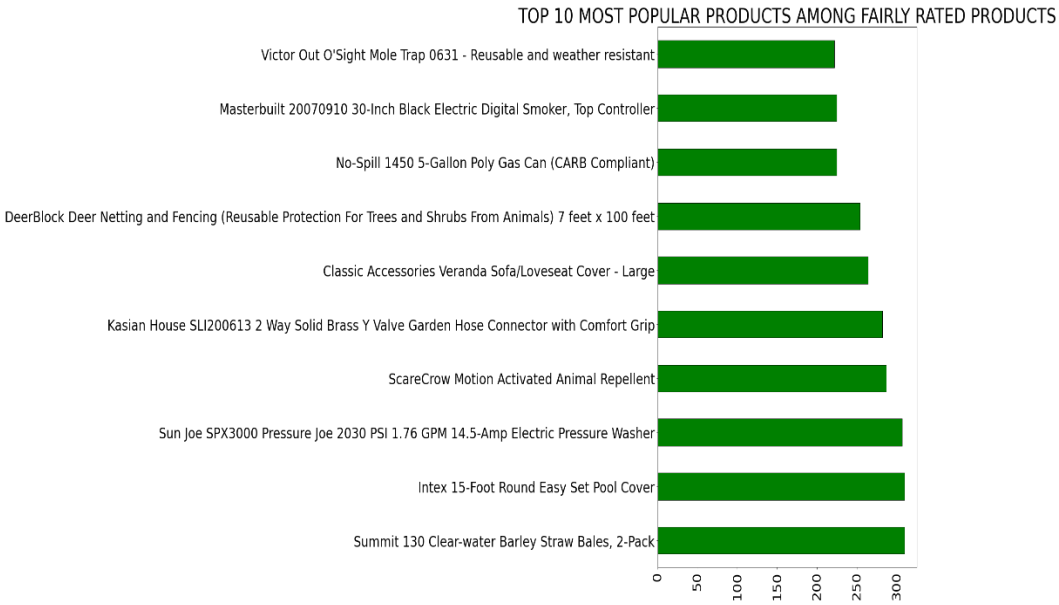
a) Category distribution



b) Top 20 most popular brands

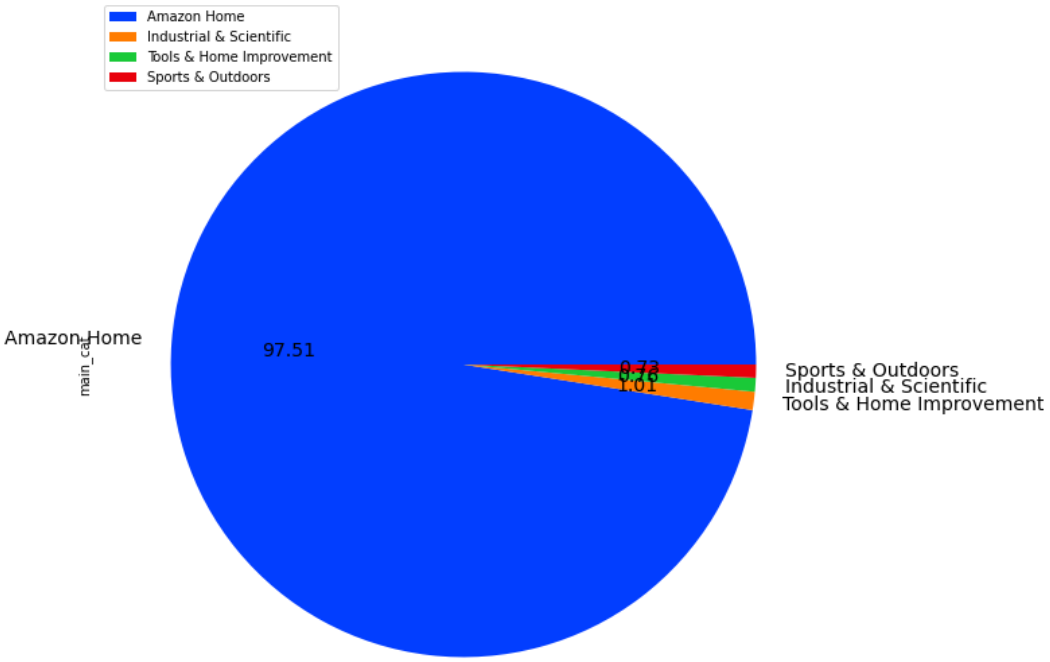


c) Top 10 most popular products

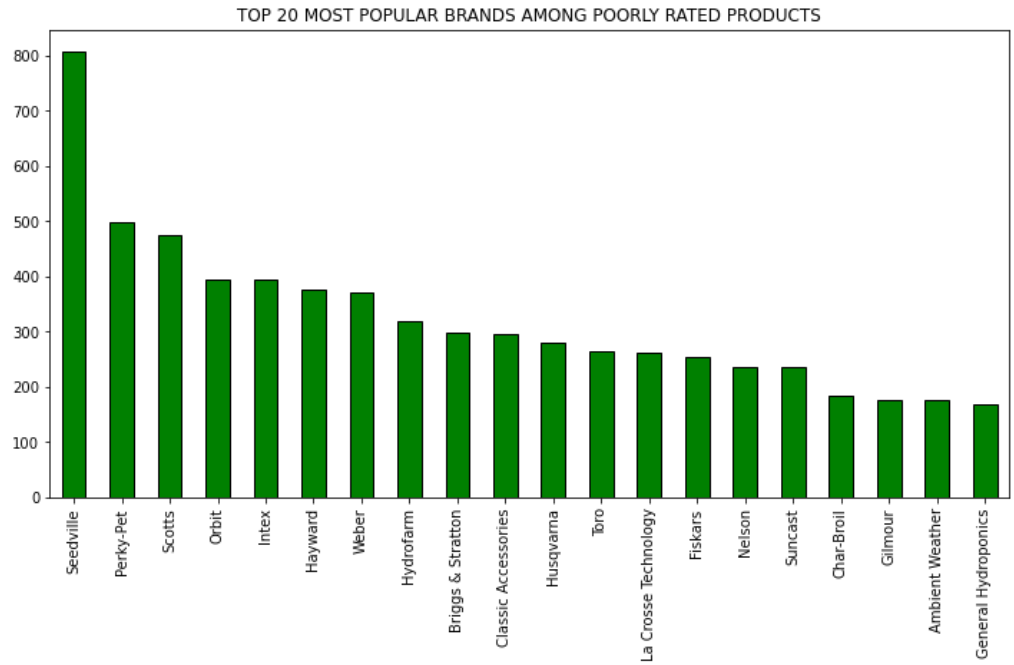


k) For poorly rated products

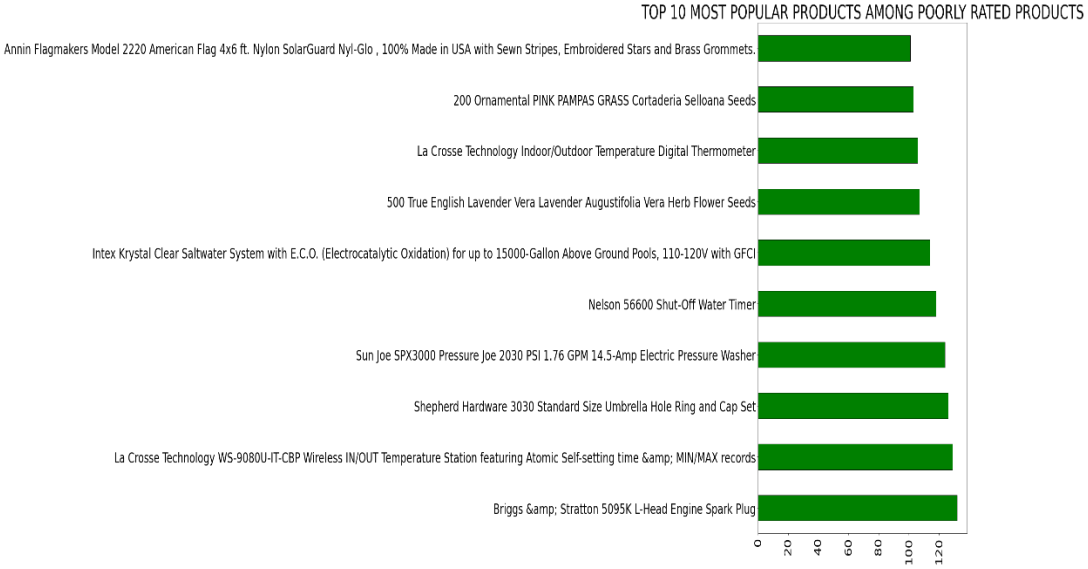
a) Category Distribution



b) Top 20 most sought after brands



c) Top 10 most sought after products



TIME SERIES FORECASTING ON PRODUCT SALES AND PRODUCT DEMAND

What is time series forecasting and why is it needed?

Time series forecasting occurs when you make scientific predictions based on historical time stamped data. It involves building models through historical analysis and using them to make observations and drive future strategic decision-making.

In our project we have used ARIMA model to predict future sales and future demand.

What is ARIMA?

ARIMA is an acronym for “autoregressive integrated moving average.” It’s a model used in statistics and econometrics to measure events that happen over a period of time. The model is used to understand past data or predict future data in a series. It’s used when a metric is recorded in regular intervals, from fractions of a second to daily, weekly or monthly periods. ARIMA is a type of model known as a Box-Jenkins method

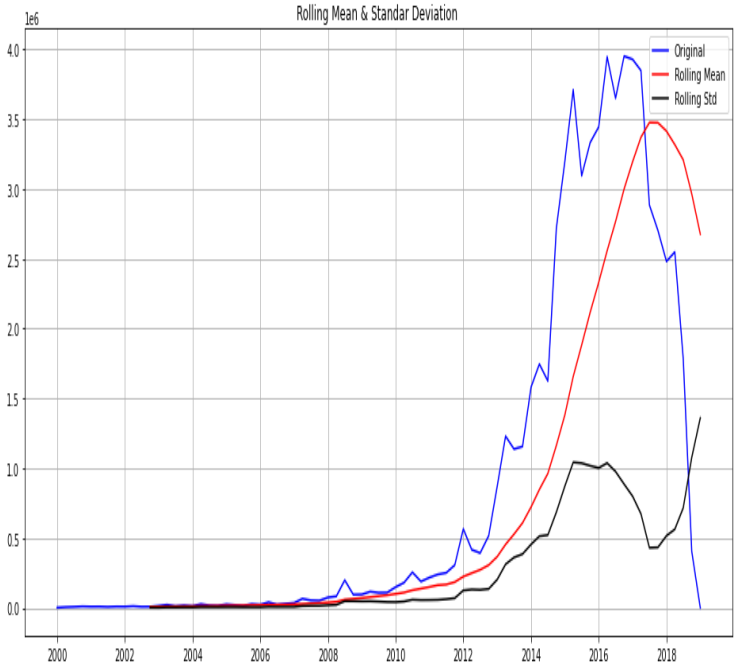
Processes Involved:

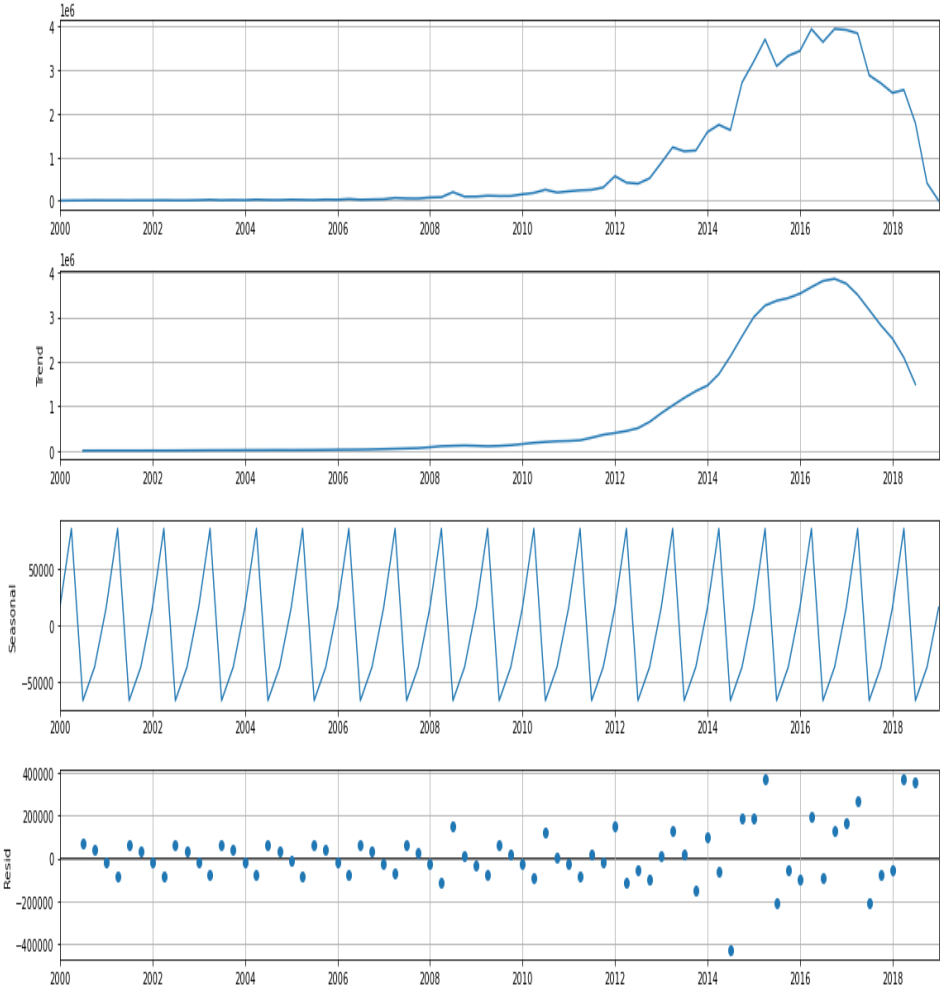
i) Data Collection:
In the field of data science, the most important thing is data. so, the first step is data collection. The first step of our project was data wrangling, where we converted raw data into cleaned and structured data. So, here we will use those files.

ii) Data Preparation:
for doing time series forecasting there are a few prerequisites, data should be time series data so for that firstly we are setting the date column as the date data type because the data frame treats date as an object while loading data then we sorted the date in increasing order along with making index followed by resampling and date -range method. In resampling, we can convert the data month-wise, quarter-wise, and weekly wise so here we are doing quarter wise and with the help of date-range we can extract the data between particular dates.

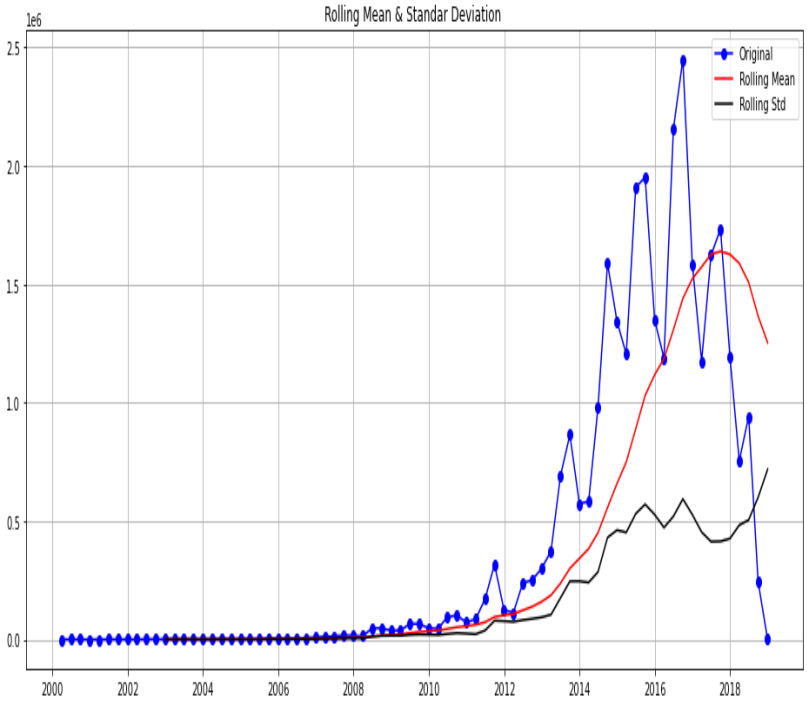
iii) Stationarity check and decomposition:
In this step, firstly we will check whether our data is stationary or not, we can check by using rolling mean and rolling statistics. The stationary data should have constant mean and variance. For checking stationarity the second method is the ADF test which is hypothesis testing, if data is not stationary we are using a differencing method it gives the value of d.

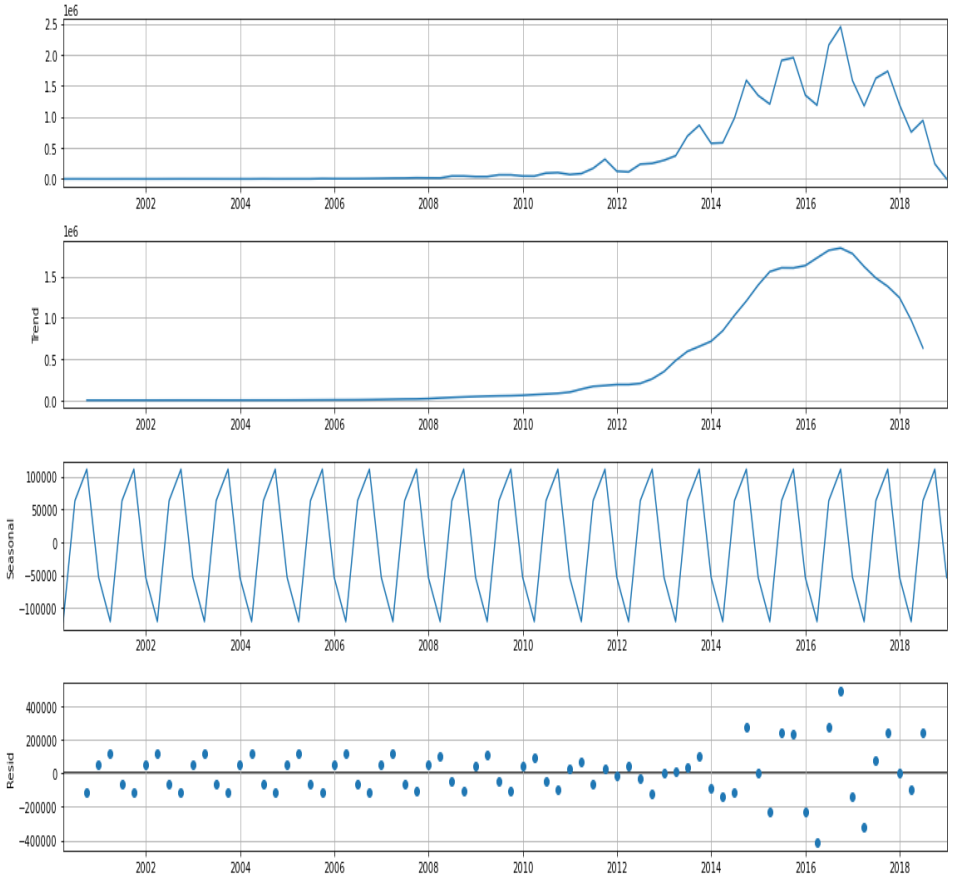
- For Tools & Home Improvement data





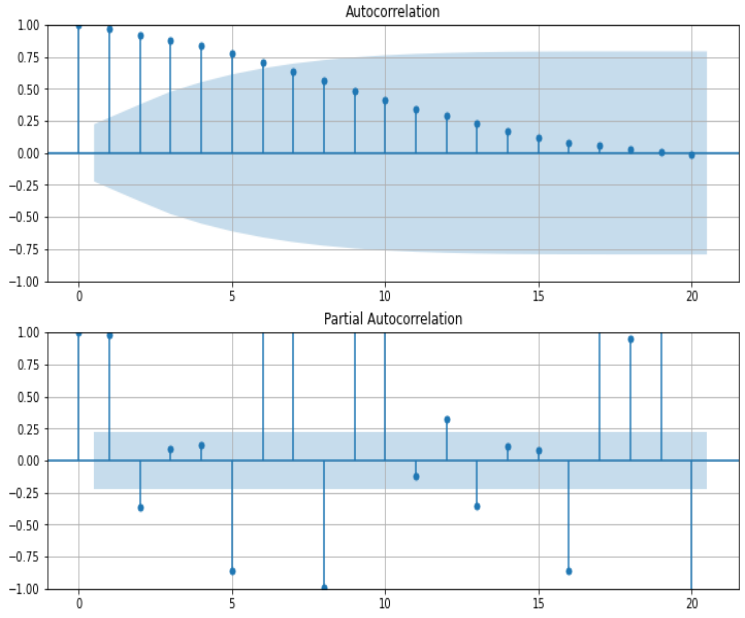
- For Patio Lawn and Garden Data





iv) Finding the best value of lags:

Finding the best value of lags: we can use either ACF and pacf plot or we can use auto arima to get the best values of p,q and d for further analysis. Now our time series data is ready for model building and forecasting.



```
from pmdarima import auto_arima
import warnings
warnings.filterwarnings("ignore")
stepwise_fit = auto_arima(df_tools_quaterly,Trace = True,
                          suppress_warnings=True)
```

```
stepwise_fit.summary()
```

SARIMAX Results

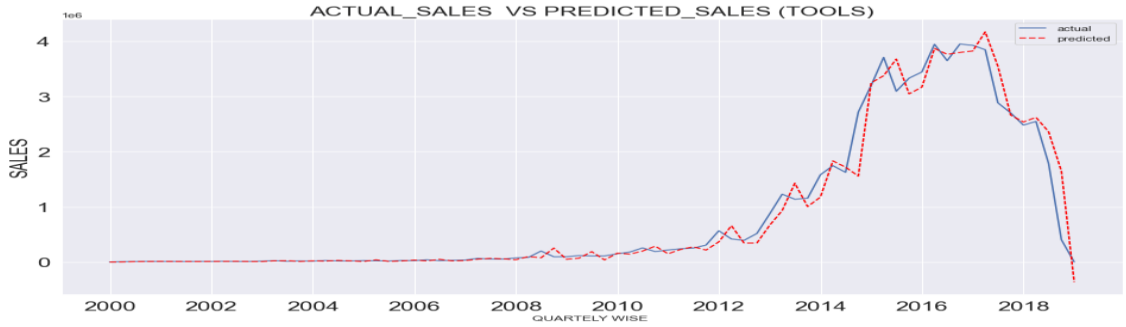
Dep. Variable:	y	**No. Observations:**	77
Model:	SARIMAX(1, 1, 2)	**Log Likelihood**	-1057.737
Date:	Mon, 16 Jan 2023	**AIC**	2123.474
Time:	20:30:19	**BIC**	2132.797
Sample:	12-31-1999	**HQIC**	2127.200
	- 12-31-2018		
Covariance Type:	opg		

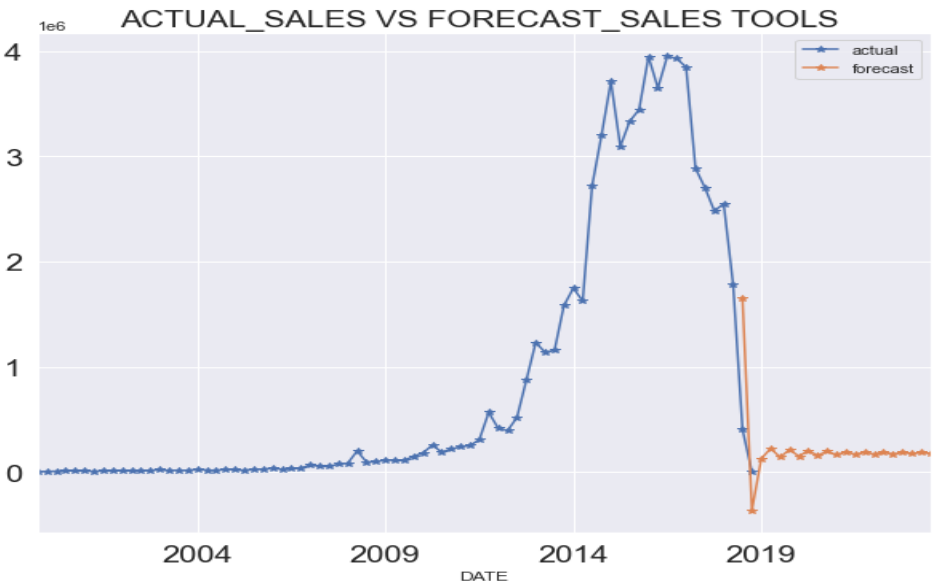
	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	-0.8708	0.097	-8.988	0.000	-1.061	-0.681
ma.L1	1.3214	0.141	9.372	0.000	1.045	1.598
ma.L2	0.5678	0.155	3.670	0.000	0.265	0.871
sigma2	9.154e+10	1.96e-12	4.67e+22	0.000	9.15e+10	9.15e+10

Ljung-Box (L1) (Q):	0.02	**Jarque-Bera (JB):**	336.78
Prob(Q):	0.90	**Prob(JB):**	0.00
Heteroskedasticity (H):	2329.31	**Skew:**	-0.65
Prob(H) (two-sided):	0.00	**Kurtosis:**	13.23

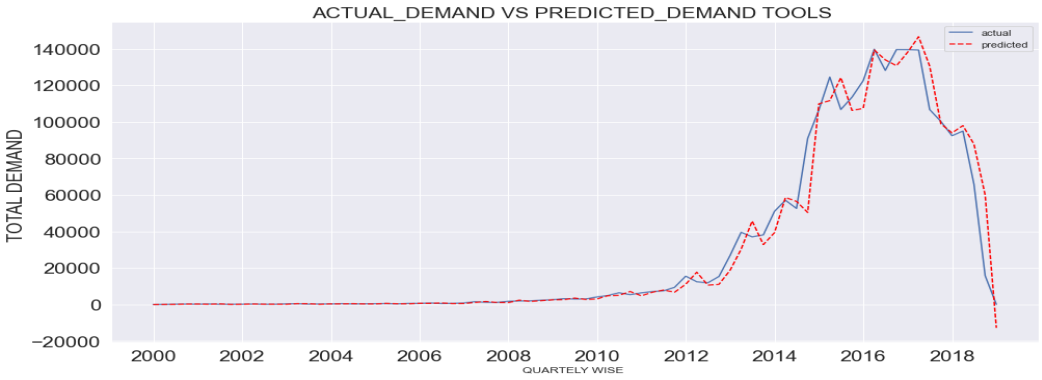
v) Model Building

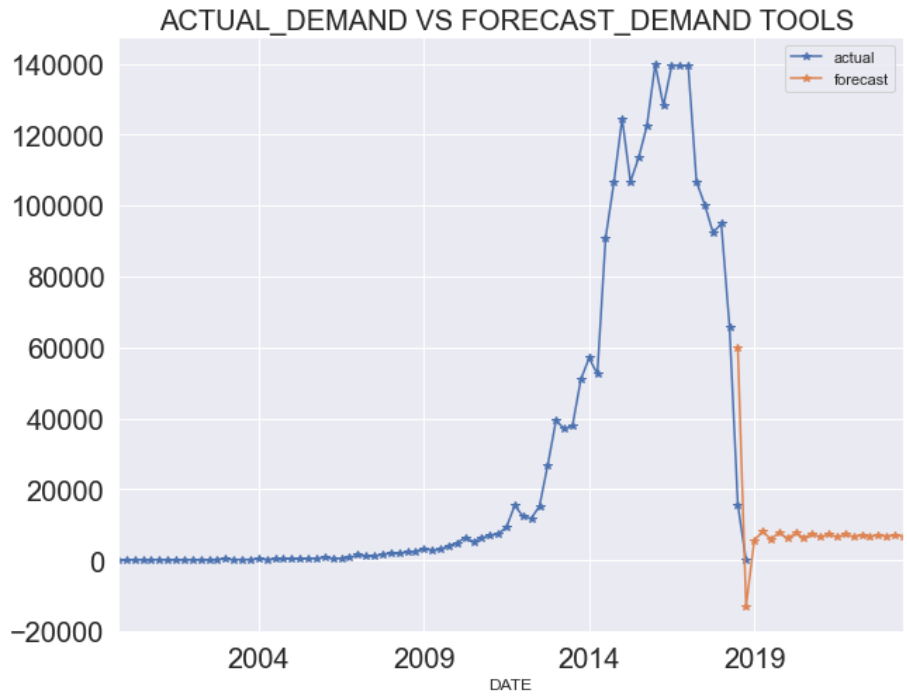
- Tools & Home Improvement (Sales)





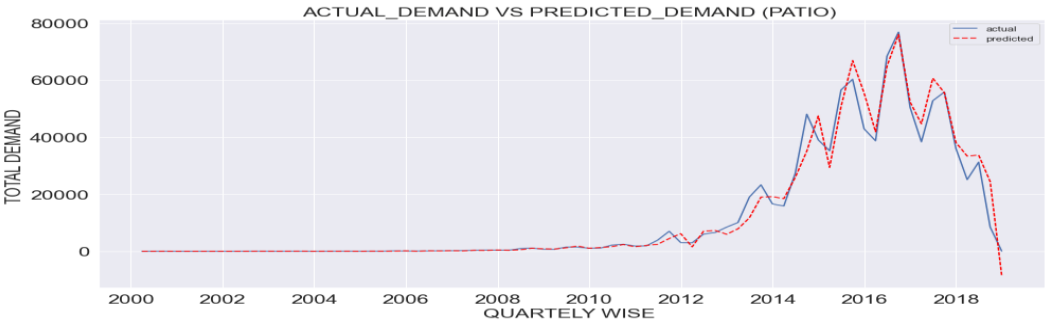
- Tools & Home Improvement (Demand)

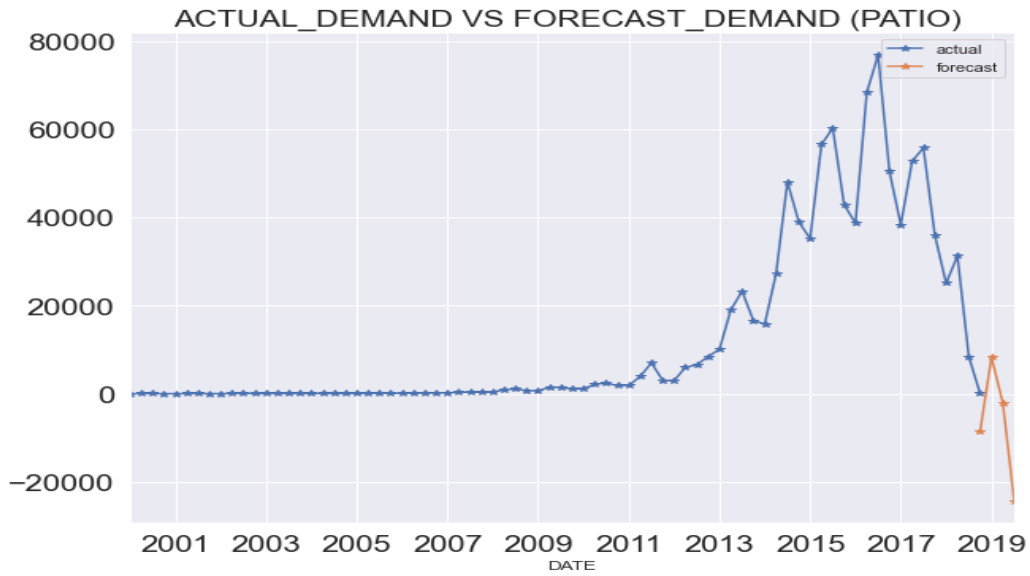




- Patio Lawn and Garden (Sales)

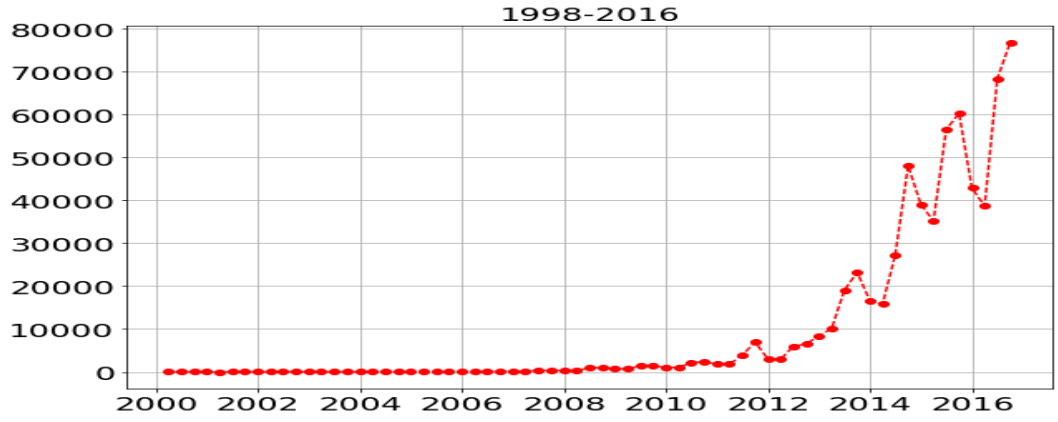
- Patio Lawn and Garden (Demand)

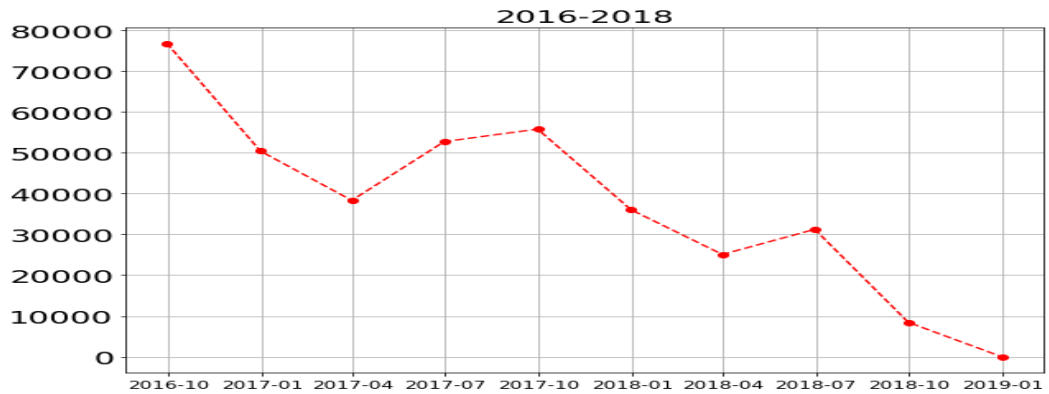




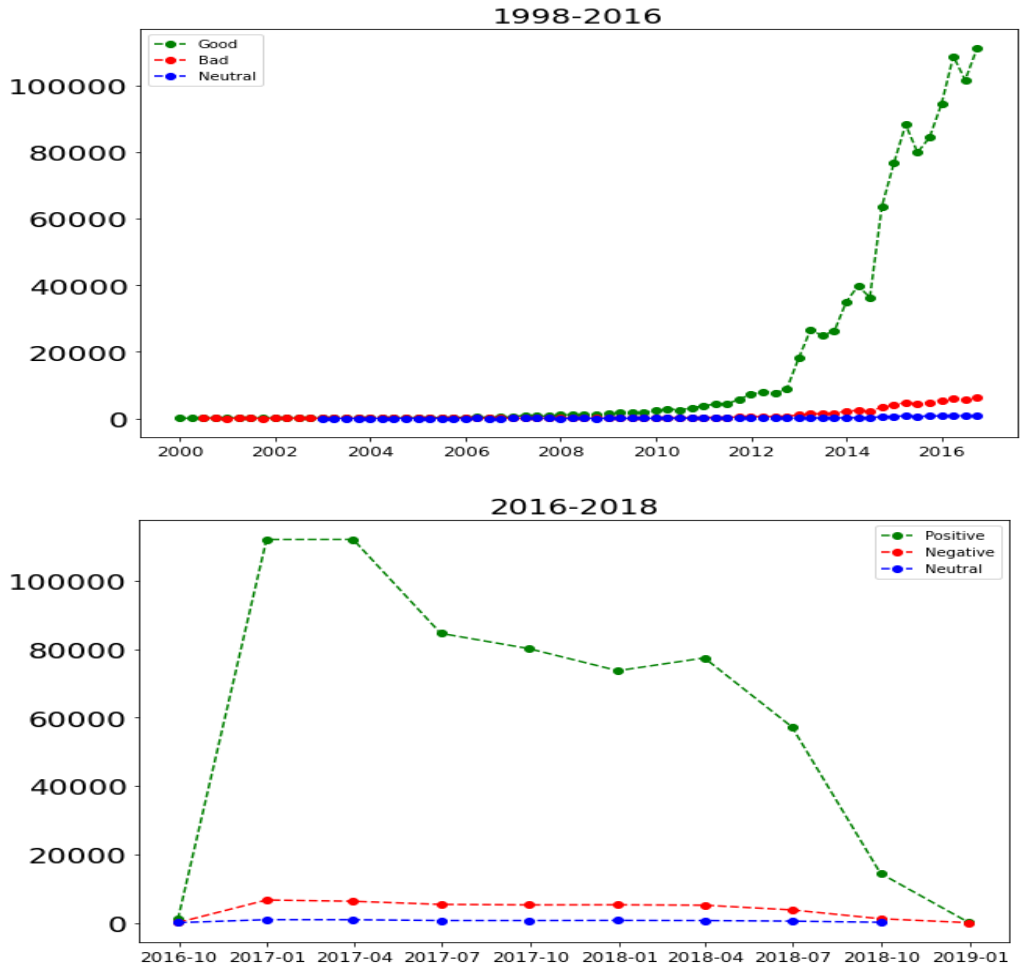
vi) Analysing variation in trend from 1998-2016 and 2016-2018

- Sales and demand





- Sentiments



So here is the variation of sales and demand in two different duration.

in the first chart which contains the data from 1998 to 2016. understandably, our products did well in the past and our demand and sale were at their peak in 2016, during these years the trend is positive and it is increasing gradually. But after 2016 the sales and the demand are falling continuously and it has shown the downward trend for this we have two trends for two different duration we can say the past trend is increasing whereas the recent trend of the product is declining so here we infer that our forecast is correct it is giving more weightage on the recent values.

These two charts which are in the bottom two is showing the sentiment variation in two distinct duration from this we can understand how sentiment is changing over time. Green is for positive reviews, red is for negative and blue is for neutral reviews over the year.

Time series forecasting has given the answer to our question, so the company is suggested to slow down the production of these products first, and do a survey of our top customers and if there are any competitors are providing the same thing at less price and better quality.

CONCLUSIONS

- It is seen that among Tools & Home Improvement the most popular brands among customers are Dewalt, Stanley and Tekton. In Patio Lawn and Garden category the most popular brands were Weber, Fiskars and Gilmour.
- From our sentiment analysis we see that brands like Perky Pet and Weber had mixed result of positive and negative sentiments.
- Brands like Classic Accessories and Fiskars had mostly positive sentiment.
- Analysing our clustering based on product price and rating we observe that brands like Seedville, Perky Pet, Scotts and Orbit were poorly rated.
- 2016 was the most fruitful year for the company both in terms of sales and products sold.
- From our time series forecasting we observe that the future sales and demand for both categories will fall.
- General sales dropped in the years 2017 and 2018 for both categories. This could possibly be because of introduction of other online retailers which had a focused market of selling either Tools & Home Improvement or Patio Lawn & Garden products. It could also mean that there was a drop in product quality.

REFERENCES

- https://www.tableau.com/learn/articles/what-is-data-cleaning
- https://www.ibm.com/in-en/topics/exploratory-data-analysis
- https://docs.python.org/3.7/library/ast.html
- https://monkeylearn.com/sentiment-analysis/#:~:text=Sentiment%20analysis%20(or%20opinion%20mining,feedback%2C%20and%20understand%20customer%20needs.
- https://www.oreilly.com/library/view/deep-learning-essentials/9781785880360/12fe4a55-a5d0-4712-bd68-ac043b87a87e.xhtml#:~:text=The%20main%20advantage%20of%20FastText,for%20words%20that%20occur%20rarely.
- https://www.tableau.com/learn/articles/time-series-forecasting
- https://www.mastersindatascience.org/learning/statistics-data-science/what-is-arima-modeling/
- https://www.nvidia.com/en-us/glossary/data-science/clustering/#:~:text=Clustering%20is%20used%20to%20identify,databases%2C%20among%20many%20other%20places.
- https://towardsdatascience.com/clustering-algorithm-for-data-with-mixed-categorical-and-numerical-features-d4e3a48066a0#:~:text=The%20k%2DMeans%20algorithm%20is,as%20space%20is%20not%20meaningful.