EDA MID EXAM (30 Marks) ¶

In [5]:
```
## Kindly change the below cells from markdown to code and execute it
```

In [1]:
```
import pandas as pd

import csv

with open("data_set.csv","r")as file:

    reader=csv.reader(file)
df=pd.read_csv("data_set.csv")

df.head()
```

Out[1]:

	Style	Price	Rating	Size	Season	Neckline	SleeveLength	WaistLine	Material	Fabri
0	Sexy	Low	4.6	M	Summer	o-neck	sleevless	empire	?	
1	Casual	Low	0.0	L	Summer	o-neck	Petal	natural	microfiber	
2	vintage	High	0.0	L	Automn	o-neck	full	natural	polyster	
3	Brief	Average	4.6	L	Spring	o-neck	full	natural	silk	
4	cute	Low	4.5	M	Summer	o-neck	butterfly	natural	chiffonfabric	

5 rows × 25 columns

In [4]:
```
df.sample(5)
```

Out[4]:

	Style	Price	Rating	Size	Season	Neckline	SleeveLength	WaistLine	Material	Fabric
245	Casual	low	4.6	M	Summer	o-neck	sleevless	natural	cotton	
162	Casual	Average	3.7	free	Automn	boat-neck	full	?	?	
170	Casual	Low	4.5	free	Winter	o-neck	full	natural	mix	wo
489	cute	Average	0.0	L	Summer	o-neck	sleevless	empire	cotton	
221	Casual	Average	4.7	S	Spring	o-neck	halfsleeve	empire	?	cl

5 rows × 25 columns

SECTION A: 5 MARKS

Q1. What are different techniques of Outlier Detection and treatment of outliers? (5 Marks)

In [9]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

OUTLIERS:-

- outier is that point or row in data that have exclusive numbers in the that do not allow column to follow normal distribution that affect our data to study
- famous technique for outlier detection is boxplot that gives us five number summary in the backend it uses IQR technique

IQR= Q3-Q1 min-1.5*IQR it shows lower outlier max+1.5*IQR it shows upper outlier

thats how we detect outliers

Treatment of outliers

- if there are large number of outliers that mostly didn't happen sowe easily remove it

```
Solve Section B questions using these below columns -
1.Style
2.Price
3.Rating
4.Size
5.Season
6.Neckline
7.SleeveLength
8.WaistLine
9.Material
10.FabricType
11.Decoration
12.PatternType
13.Recommendation
```

In [6]:
```
# Code to retrive the data frame for Section B
dresses = df.iloc[:, :13]
dresses.head(1)
```

Out[6]:

	Style	Price	Rating	Size	Season	Neckline	SleeveLength	WaistLine	Material	FabricType
0	Sexy	Low	4.6	M	Summer	o-neck	sleevless	empire	?	chiffon

In [12]:
```
x= ['Style', 'Price', 'Rating', 'Size', 'Season', 'Neckline',
       'SleeveLength', 'WaistLine', 'Material', 'FabricType', 'Decoration',
       'PatternType', 'Recommendation']
```

In [ ]:
```

```

```
Solve Section C questions using these below columns -
1.Product_ID
2.Date
3.Units
4.Product_Name
5.Product_Category
6.Product_Cost
7.Product_Price
8.Store_Name
9.Store_City
10.Store_Location
11.Store_Open_Date
12.Stock_On_Hand
```

In [ ]:
```
Code to retrive the data frame for Section C
Product_sales = df.iloc[:, 13:]
Product_sales.head(1)
```

SECTION B: 15 MARKS

2) Read the dresses dataframe and do the below following questions:

2.A) Separate categorical data and numerical data also each category number of percentages. (3 Marks)

2.B) Replace '?' from the Dataset to NaN values (3 Marks)

2.C) There are some garbage and repetative entries in the columns like Price,Size,Season and sleevlength. use appropriate treatment to convert these columns to have unique and meaningfull classes. (4 Marks)

2.D) Draw a Countplot for Season column and plot piechart for price column. Write down the inference for the same(2 marks)

2.E) Draw boxplot for ratings in every price class. Write down the inference for the same. (3 Marks)

In [13]:
```
dresses = df.iloc[:, :13]
dresses.head(1)
```

Out[13]:

	Style	Price	Rating	Size	Season	Neckline	SleeveLength	WaistLine	Material	FabricType
0	Sexy	Low	4.6	M	Summer	o-neck	sleevless	empire	?	chiffon

2.A) Separate categorical data and numerical data also each category number of percentages.(3 Marks)

In [26]:
```
dresses.head()
```

Out[26]:

	Style	Price	Rating	Size	Season	Neckline	SleeveLength	WaistLine	Material	Fabri
0	Sexy	Low	4.6	M	Summer	o-neck	sleevless	empire	?	
1	Casual	Low	0.0	L	Summer	o-neck	Petal	natural	microfiber	
2	vintage	High	0.0	L	Automn	o-neck	full	natural	polyster	
3	Brief	Average	4.6	L	Spring	o-neck	full	natural	silk	
4	cute	Low	4.5	M	Summer	o-neck	butterfly	natural	chiffonfabric	

In [31]:
```
# categorical columns
dresses.select_dtypes(object).columns
```

Out[31]:
```
Index(['Style', 'Price', 'Size', 'Season', 'Neckline', 'SleeveLength',
       'WaistLine', 'Material', 'FabricType', 'Decoration', 'PatternType'],
      dtype='object')
```

In [33]:
```
# numerical columns
dresses.select_dtypes(np.number).columns
```

Out[33]:
```
Index(['Rating', 'Recommendation'], dtype='object')
```

In [156]:
```
col= ['Style', 'Price', 'Size', 'Season', 'Neckline', 'SleeveLength',
       'WaistLine', 'Material', 'FabricType', 'Decoration', 'PatternType']

for i in col:
    a= dresses[i].value_counts()/df.shape[0]*100
    print(i,a)
```

```
Style Casual      46.4
Sexy        13.8
party       10.2
cute         9.0
vintage      5.0
bohemian     4.8
Brief        3.6
work         3.4
Novelty      1.6
sexy         1.4
Flare        0.4
fashion      0.2
OL           0.2
Name: Style, dtype: float64
Price Average      50.4
Low          34.8
Medium        6.0
very-high     4.2
High          4.2
```

2.B) Find out the percentage of '?' values in each column and Replace '?' from the Dataset to NaN values (3 Marks)

In [70]:
```
for i  in df.columns:
    a= df[i].unique()
    print(i,a)
```

```
Style ['Sexy' 'Casual' 'vintage' 'Brief' 'cute' 'bohemian' 'Novelty' 'Flare'
 'party' 'sexy' 'work' 'OL' 'fashion']
Price ['Low' 'High' 'Average' 'Medium' 'very-high' 'low' 'high' '?']
Rating [4.6 0.  4.5 5.  4.7 4.8 4.3 4.  4.4 4.9 4.1 3.6 3.7 3.5 1.  3. ]
Size ['M' 'L' 'XL' 'free' 'S' 'small' 's']
Season ['Summer' 'Automn' 'Spring' 'Winter' 'spring' 'winter' '?' 'summer'
 'Autumn']
Neckline ['o-neck' 'v-neck' 'boat-neck' 'peterpan-collor' 'ruffled'
 'turndowncollor' 'slash-neck' 'mandarin-collor' 'open' 'sqare-collor'
 'Sweetheart' 'sweetheart' '?' 'Scoop' 'halter' 'backless' 'bowneck']
SleeveLength ['sleevless' 'Petal' 'full' 'butterfly' 'short' 'threequarter'
 'halfsleeve' 'cap-sleeves' 'turndowncollor' 'threequater' 'capsleeves'
 'sleeveless' 'sleeevless' 'half' 'urndowncollor' 'thressqatar' '?'
 'sleveless']
WaistLine ['empire' 'natural' '?' 'princess' 'dropped']
Material ['?' 'microfiber' 'polyster' 'silk' 'chiffonfabric' 'cotton' 'nylon'
 'other' 'milksilk' 'linen' 'rayon' 'lycra' 'mix' 'acrylic' 'spandex'
 'lace' 'modal' 'cashmere' 'viscos' 'knitting' 'sill' 'wool' 'model'
 'shiffon']
FabricType ['chiffon' '?' 'broadcloth' 'jersey' 'other' 'batik' 'satin' 'flan
nael'
 'worsted' 'woolen' 'poplin' 'dobby' 'knitting' 'flannel' 'tulle' 'sattin'
 'organza' 'lace' 'Corduroy' 'wollen' 'knitted' 'shiffon' 'terry']
Decoration ['ruffles' '?' 'embroidary' 'bow' 'lace' 'beading' 'sashes' 'hollo
wout'
 'pockets' 'sequined' 'applique' 'button' 'Tiered' 'rivet' 'feathers'
 'flowers' 'pearls' 'pleat' 'crystal' 'ruched' 'draped' 'tassel' 'plain'
 'none' 'cascading']
PatternType ['animal' 'print' 'dot' 'solid' '?' 'patchwork' 'striped' 'geomet
ric'
 'plaid' 'leopard' 'floral' 'character' 'splice' 'leapord' 'none']
Recommendation [1 0]
Product_ID [ 8 25 15  3 30 12  6  1 34 20  9 19 18 29 10 27 24 21 13 11  7 31
2 33
 14  5 23 22 16 17 26 32  4]
Date ['4/29/2018' '04-08-2018' '2/25/2018' '01-05-2018' '02-08-2018'
 '04-03-2018' '3/26/2018' '4/18/2018' '1/20/2018' '04-01-2018' '3/21/2018'
 '03-10-2018' '1/19/2018' '04-09-2018' '03-03-2018' '4/30/2018'
 '1/31/2018' '03-04-2018' '4/14/2018' '1/14/2018' '3/14/2018' '04-10-2018'
 '3/25/2018' '2/26/2018' '4/15/2018' '02-03-2018' '04-11-2018' '4/24/2018'
 '04-07-2018' '2/22/2018' '3/27/2018' '02-04-2018' '3/31/2018' '3/23/2018'
 '3/18/2018' '1/26/2018' '02-01-2018' '4/16/2018' '2/20/2018' '04-02-2018'
 '04-04-2018' '2/13/2018' '2/21/2018' '1/21/2018' '3/30/2018' '2/16/2018'
 '3/29/2018' '4/13/2018' '03-11-2018' '1/17/2018' '02-10-2018'
 '02-11-2018' '2/14/2018' '01-02-2018' '01-03-2018' '04-12-2018'
 '4/20/2018' '1/23/2018' '3/22/2018' '03-05-2018' '1/28/2018' '2/19/2018'
 '1/15/2018' '4/25/2018' '01-06-2018' '3/13/2018' '3/24/2018' '01-07-2018'
 '4/22/2018' '03-09-2018' '1/13/2018' '3/20/2018' '02-05-2018' '4/21/2018'
 '04-05-2018' '3/17/2018' '03-06-2018' '1/22/2018' '1/30/2018' '3/19/2018'
 '01-04-2018' '1/24/2018' '4/19/2018' '03-07-2018' '01-12-2018'
 '4/27/2018' '01-09-2018' '4/28/2018' '2/24/2018' '4/26/2018' '03-02-2018'
 '2/17/2018' '3/28/2018' '2/18/2018' '02-06-2018' '3/15/2018' '02-02-2018'
 '02-07-2018' '4/17/2018' '01-01-2018' '04-06-2018' '1/27/2018'
 '01-11-2018' '1/29/2018' '2/28/2018' '02-09-2018' '2/15/2018'
 '01-08-2018' '2/23/2018' '3/16/2018' '1/25/2018' '03-08-2018' '1/16/2018'
 '02-12-2018' '03-12-2018' '03-01-2018' '2/27/2018' '4/23/2018']
Units [1 2 3 5 6 4 7]
Product_Name ['Deck Of Cards' 'PlayDoh Can' 'Hot Wheels 5-Pack' "Barrel O' Sl
ime"
 "Rubik's Cube" 'Foam Disk Launcher' 'Colorbuds' 'Action Figure'
 'Toy Robot' 'Mini Basketball Hoop' 'Dino Egg' 'Magic Sand' 'Lego Bricks'
```

```
 'Plush Pony' 'Dinosaur Figures' 'PlayDoh Toolkit' 'Nerf Gun'
 'Mini Ping Pong Set' 'Gamer Headphones' 'Etch A Sketch' 'Dart Gun'
 'Splash Balls' 'Animal Figures' 'Teddy Bear' 'Glass Marbles'
 'Classic Dominoes' 'Mr. Potatohead' 'Monopoly' 'Jenga' 'Kids Makeup Kit'
 'PlayDoh Playset' 'Supersoaker Water Gun' 'Chutes & Ladders']
Product_Category ['Games' 'Art & Crafts' 'Toys' 'Sports & Outdoors' 'Electron
ics']
Product_Cost ['$3.99 ' '$1.99 ' '$17.99 ' '$8.99 ' '$6.99 ' '$9.99 ' '$20.99
'
 '$13.99 ' '$34.99 ' '$10.99 ' '$14.99 ' '$11.99 ' '$7.99 ' '$5.99 '
 '$4.99 ' '$2.99 ']
Product_Price ['$6.99 ' '$2.99 ' '$5.99 ' '$3.99 ' '$19.99 ' '$11.99 ' '$14.9
9 '
 '$15.99 ' '$25.99 ' '$24.99 ' '$10.99 ' '$39.99 ' '$4.99 ' '$9.99 '
 '$20.99 ' '$8.99 ' '$12.99 ']
Store_Name ['Maven Toys Ciudad de Mexico 2' 'Maven Toys Puebla 1'
 'Maven Toys Ciudad Victoria 1' 'Maven Toys Oaxaca 1'
 'Maven Toys Monterrey 1' 'Maven Toys Monterrey 2'
 'Maven Toys Hermosillo 3' 'Maven Toys Hermosillo 1'
 'Maven Toys Aguascalientes 1' 'Maven Toys Puebla 3' 'Maven Toys Merida 1'
 'Maven Toys Chihuahua 1' 'Maven Toys Santiago 1'
 'Maven Toys San Luis Potosi 1' 'Maven Toys Guadalajara 4'
 'Maven Toys Toluca 1' 'Maven Toys Saltillo 1' 'Maven Toys Mexicali 1'
 'Maven Toys Xalapa 2' 'Maven Toys Tuxtla Gutierrez 1'
 'Maven Toys Ciudad de Mexico 4' 'Maven Toys Pachuca 1'
 'Maven Toys Guanajuato 3' 'Maven Toys Chihuahua 2'
 'Maven Toys Ciudad de Mexico 1' 'Maven Toys Mexicali 2'
 'Maven Toys Guadalajara 1' 'Maven Toys Ciudad de Mexico 3'
 'Maven Toys Campeche 2' 'Maven Toys Xalapa 1' 'Maven Toys Campeche 1'
 'Maven Toys Guadalajara 2' 'Maven Toys Saltillo 2' 'Maven Toys Durango 1'
 'Maven Toys La Paz 1' 'Maven Toys Chetumal 1' 'Maven Toys Culiacan 1'
 'Maven Toys Guanajuato 1' 'Maven Toys Cuernavaca 1'
 'Maven Toys Guadalajara 3' 'Maven Toys Guanajuato 2'
 'Maven Toys Morelia 1' 'Maven Toys Monterrey 4' 'Maven Toys Hermosillo 2'
 'Maven Toys Toluca 2' 'Maven Toys Chilpancingo 1' 'Maven Toys Puebla 2'
 'Maven Toys Villahermosa 1' 'Maven Toys Monterrey 3'
 'Maven Toys Zacatecas 1']
Store_City ['Cuidad de Mexico' 'Puebla' 'Ciudad Victoria' 'Oaxaca' 'Monterre
y'
 'Hermosillo' 'Aguascalientes' 'Merida' 'Chihuahua' 'Santiago'
 'San Luis Potosi' 'Guadalajara' 'Toluca' 'Saltillo' 'Mexicali' 'Xalapa'
 'Tuxtla Gutierrez' 'Pachuca' 'Guanajuato' 'Campeche' 'Durango' 'La Paz'
 'Chetumal' 'Culiacan' 'Cuernavaca' 'Morelia' 'Chilpancingo'
 'Villahermosa' 'Zacatecas']
Store_Location ['Airport' 'Commercial' 'Downtown' 'Residential']
Store_Open_Date ['05-04-2012' '12/16/2008' '09-08-2010' '10-02-2010' '4/27/19
95'
 '12/25/2003' '6/27/2014' '8/31/2012' '7/31/2010' '12/27/2014' '8/22/2008'
 '06-12-2010' '11/23/2009' '5/19/2007' '10/31/2015' '12-09-2007'
 '01-01-2000' '12/13/2003' '4/21/2014' '03-05-2007' '6/21/2015'
 '10/14/2004' '5/18/2016' '3/18/2014' '10/15/2004' '8/30/2006' '9/18/1992'
 '11/28/2013' '9/15/2010' '6/21/2011' '1/14/2005' '12/27/1999' '3/23/2016'
 '6/30/2014' '5/31/2001' '05-05-2006' '05-10-2016' '1/31/2007' '4/19/2005'
 '10/20/2011' '3/29/2010' '07-01-2013' '11/21/2015' '06-01-2014'
 '5/27/2014' '06-11-2013' '04-01-2011' '06-07-2013' '3/17/2013'
 '5/29/2009']
Stock_On_Hand [ 84.  18.  13.  55.  50.  38.  29.  65.  11.  19.   8.   2.  2
6.  90.
  57.  71.  14.  36.   7.  17.   3.  46.  22.  25.  53.  10.  47.  23.
  27.   5.  33.  59.   9.  54.   6.  49.  37.  15.  40.  32.  35.  34.
  21.  44.  94.  31.  58.  nan  24.   0.  68.  30.  61.   4.  39.  20.
```

```
43.  16.   1.  12.  41.  66.  63.  45. 129.  28.  56.  64.  74.  76.
73.  48. 125. 100.  70. 110.  42.  86.]
```

In [49]:
```
df[['Price','Season','Neckline','SleeveLength','WaistLine','Material','FabricT
```

Out[49]:

	Price	Season	Neckline	SleeveLength	WaistLine	Material	FabricType	Decoration	PatternTy
0	Low	Summer	o-neck	sleevless	empire	?	chiffon	ruffles	ani

In [97]:
```
dresses['Price'].replace({'?':np.nan},inplace=True)
```

In [96]:
```
for i in ['Season','Neckline','SleeveLength','WaistLine','Material','FabricTyp
    dresses[i].replace({'?':np.nan},inplace=True)
```

In [90]:
```

```

Out[90]: 2

2.C) There are some garbage and repetative entries in the columns like Price,Size,Season and SleeveLength. use appropriate treatment to convert these columns to have unique and meaningfull classes. (4 Marks)

In [125]:
```
dresses['Price'].replace({'high':'High','low':'Low'},inplace=True)
```

In [120]:
```
dresses['Price'].unique()
```

Out[120]:
```
array(['Low', 'High', 'Average', 'Medium', 'very-high', 'low', nan],
      dtype=object)
```

In [121]:
```
dresses['Size'].replace({'s':'S','small':'S'},inplace = True)
```

In [122]:
```
dresses['Size'].unique()
```

Out[122]:
```
array(['M', 'L', 'XL', 'free', 'S'], dtype=object)
```

In [127]:
```
dresses['Season'].replace({'summer':'Summer','Automn':'Autumn','winter':'Winte
```

In [128]:
```
dresses['Season'].unique()
```

Out[128]:
```
array(['Summer', 'Autumn', 'Spring', 'Winter', 'spring', nan],
      dtype=object)
```

In [131]:
```
dresses['SleeveLength'].replace({'sleeveless':'sleevless','sleeevless':'sleevl
```

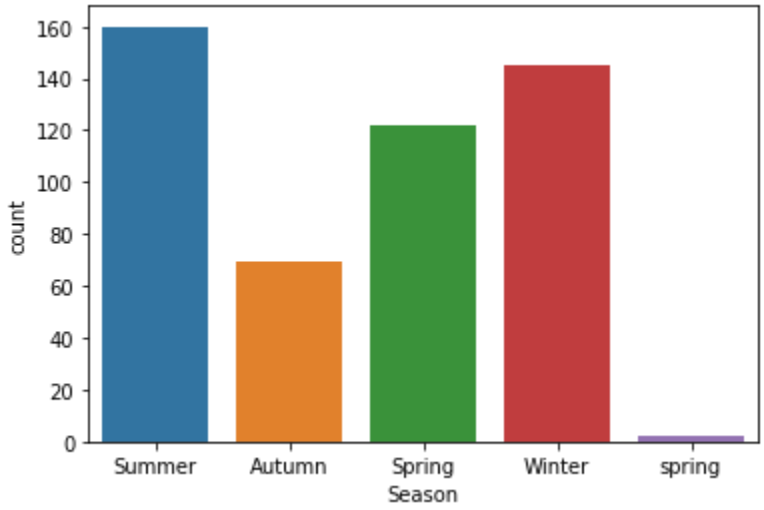
In [133]:
```
dresses['SleeveLength'].unique()
```

Out[133]:
```
array(['sleevless', 'Petal', 'full', 'butterfly', 'short', 'threequarter',
       'halfsleeve', 'cap-sleeves', 'turndowncollor', 'threequater',
       'capsleeves', 'half', 'urndowncollor', nan, 'sleveless'],
      dtype=object)
```

2.D) Draw a Countplot for Season column and plot piechart for price column. Write down the inference for the same(2 marks)

In [134]:
```
sns.countplot(x=dresses['Season'])
```

Out[134]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f44991ef898>
```

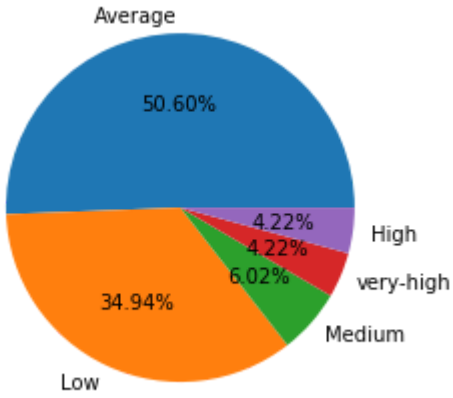


infrences :-

- summer and winter season have more sale than any other

In [144]:
```
a=dresses.Price.value_counts().index
b=dresses.Price.value_counts().values
```

In [150]:
```
plt.pie(x=b,labels= a,autopct='%.2f%%')
plt.show()
```



infrenes :-

- low an low dresses are more in comparison to others
- high and very high prices offcouse less due to mostly people don't buyhigh price dresses

2.E) Calculate the number of dreeses available in different size in every price class. plot the data using appropriate visualisation plot.Write down the inference for the same. (3 Marks)

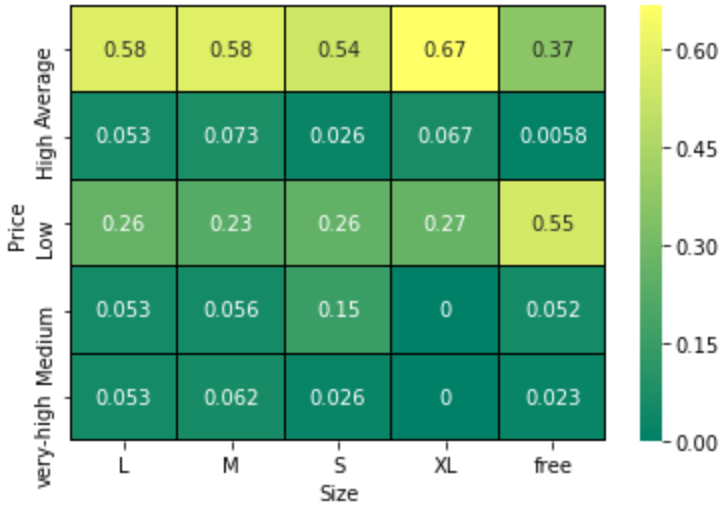
In [152]:
```
dresses.sample(3)
```

Out[152]:

	Style	Price	Rating	Size	Season	Neckline	SleeveLength	WaistLine	Material	FabricTy
33	Brief	Average	4.3	L	Spring	o-neck	sleevless	empire	NaN	N
28	cute	Low	4.3	free	Autumn	o-neck	sleevless	natural	polyster	chif
172	Sexy	Medium	4.6	L	Summer	o-neck	short	natural	spandex	jer

In [211]:
```
sns.heatmap(pd.crosstab(dresses['Price'],dresses['Size'],normalize='columns'),
```

Out[211]: <matplotlib.axes._subplots.AxesSubplot at 0x7f4480b85550>



In [212]:
```
pd.crosstab(dresses['Price'],dresses['Size'])
```

Out[212]:

Size	L	M	S	XL	free
Price					
Average	55	103	21	10	63
High	5	13	1	1	1
Low	25	40	10	4	95
Medium	5	10	6	0	9
very-high	5	11	1	0	4

SECTION C: 10 MARKS

3 Read the Product_sales dataframe answer the questions below

3.A) Create new columns by splitting the `Date` column into Day,Month and Year ?(3 Marks)

3.B) Create new column `Product_profit` from `Product_Price` and `Product_Cost` columns ?Check the distribution of the `Product_profit` . What inferences can be made about distribution of the `Product_profit` ? (3 Marks)

3.C) Use the appropriate plot to display product categories and the average profit in every

In [158]:
```
Product_sales = df.iloc[:, 13:]
Product_sales.head(1)
```

Out[158]:

	Product_ID	Date	Units	Product_Name	Product_Category	Product_Cost	Product_Price
0	8	4/29/2018	1	Deck Of Cards	Games	$3.99	$6.99

3.A) Create new columns by splitting the `Date` column into Day,Month and Year ?(3 Marks)

In [170]:
```
Product_sales['Date']=pd.to_datetime(Product_sales['Date'])
```

In [176]:
```
Product_sales['month']= Product_sales['Date'].dt.month
```

In [177]:
```
Product_sales['day']= Product_sales['Date'].dt.day
```

In [178]:
```
Product_sales['year']= Product_sales['Date'].dt.year
```

In [179]:
```
Product_sales.head()
```

Out[179]:

	Product_ID	Date	Units	Product_Name	Product_Category	Product_Cost	Product_Price	Sto
0	8	2018-04-29	1	Deck Of Cards	Games	$3.99	$6.99	Ma
1	25	2018-04-08	1	PlayDoh Can	Art & Crafts	$1.99	$2.99	Ma
2	15	2018-02-25	1	Hot Wheels 5-Pack	Toys	$3.99	$5.99	Ma
3	3	2018-01-05	1	Barrel O' Slime	Art & Crafts	$1.99	$3.99	Ma
4	30	2018-02-08	1	Rubik's Cube	Games	$17.99	$19.99	Ma Me

3.B) Create new column `Product_profit` from `Product_Price` and `Product_Cost` columns ?

Check the distribution of the `Product_profit` .

What inferences can be made about distribution of the `Product_profit` ? (3 Marks)

In [187]:
```
Product_sales['Product_Cost']=Product_sales.Product_Cost.str.replace('$','')
```

```
/home/gl_jupyter/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:
1: FutureWarning: The default value of regex will change from True to False i
n a future version. In addition, single character regular expressions will*no
t* be treated as literal strings when regex=True.
  """Entry point for launching an IPython kernel.
```

In [205]:
```
Product_sales['Product_Cost'] = Product_sales['Product_Cost'].astype(float)
```

In [196]:
```
Product_sales['Product_Price'] =Product_sales.Product_Price.str.replace('$',''
```

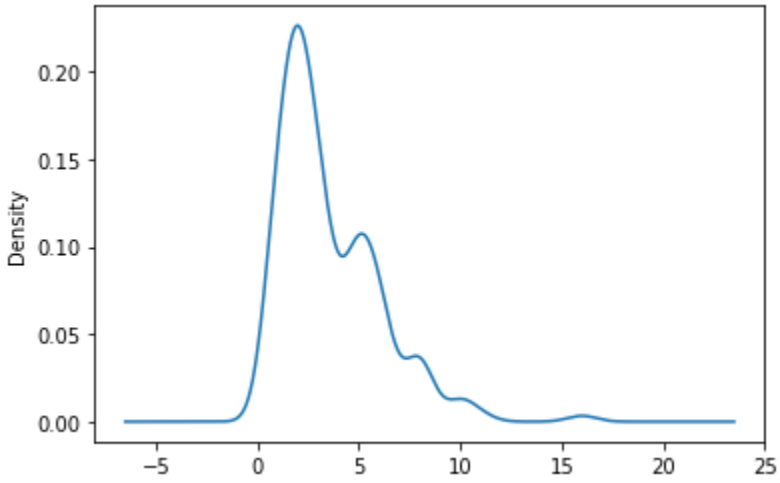
```
/home/gl_jupyter/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:
1: FutureWarning: The default value of regex will change from True to False i
n a future version. In addition, single character regular expressions will*no
t* be treated as literal strings when regex=True.
  """Entry point for launching an IPython kernel.
```

In [203]:
```
Product_sales['Product_Price'] = Product_sales['Product_Price'].astype(float)
```

In [207]:
```
Product_sales['Product_profit']=Product_sales['Product_Price']-Product_sales['
```

In [210]:
```
Product_sales['Product_profit'].plot(kind= 'kde')
```

Out[210]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f4480baafd0>
```



3.C) Use the appropriate plot to display product categories and the average profit in every product category. Display the plot in order of highest to profit to lowest.(4 Marks)

In [214]:
```
Product_sales[['Product_Category','Product_profit']]
```

Out[214]:

	Product_Category	Product_profit
0	Games	3.0
1	Art & Crafts	1.0
2	Toys	2.0
3	Art & Crafts	2.0
4	Games	2.0
...	...	...
495	Toys	5.0
496	Art & Crafts	2.0
497	Toys	4.0
498	Toys	4.0
499	Art & Crafts	1.0

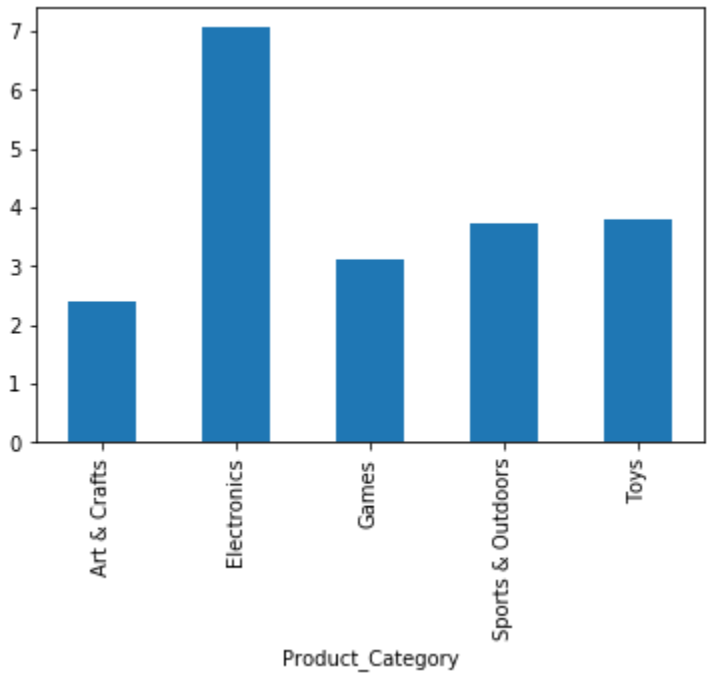
500 rows × 2 columns

In [218]:
```
Product_sales.groupby('Product_Category')['Product_profit'].mean()
```

Out[218]:
```
Product_Category
Art & Crafts          2.385621
Electronics           7.057692
Games                 3.119048
Sports & Outdoors     3.707692
Toys                  3.794521
Name: Product_profit, dtype: float64
```

In [219]:
```
Product_sales.groupby('Product_Category')['Product_profit'].mean().plot(kind=
```

Out[219]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f4480a6c160>
```



infrences:-

- according to the data electronics product have more sales in comparison to others