In [1]:
```
#importing required libraries
import os
import numpy as np # linear algebra
import pandas as pd # for data processing, CSV file I/O (e.g. pd.read_csv)
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
# assigning path of eda csv file to variable eda_data_path
eda_data_path = 'D:\Amazon_Project\exploratory_data_analysis_file.csv'
```

In [3]:
```
#Reading csv file
eda_data = pd.read_csv(eda_data_path)
```

In [4]:
```
# display top 5 rows
eda_data.head()
```

Out[4]:

	order_id	customer_id	date	nearest_warehouse	shopping_cart	order_price	delivery_charges	customer_lat	customer_long	coupon_discount	order_total	season	is_expedited_delivery	dis
0	ORD162484	ID6197311990	22-09-2019	Arizona	[('Lucent 330S', 1), ('Thunder line', 1), ('Orbiter', 2)]	12200	79.89	-37.815105	144.952843	10	11059.89	Winter	True	
1	ORD195510	ID3928250845	29-11-2019	Arizona	[('Thunder line', 1), ('Orbiter Note', 2)]	9080	62.71	-37.802736	144.951118	0	9142.71	Summer	False	
2	ORD149479	ID8579391881	03-01-2019	Arkansas	[('Thunder line', 1), ('Jewel TV', 2)]	10670	65.87	-37.821302	144.957561	10	9668.87	Autumn	False	
3	ORD193324	ID4544561904	12-04-2019	Arkansas	[('Universe Note', 1), ('Alvion EVO 3000', 1), ('Orbiter', 2)]	24800	57.61	-37.814416	144.973073	15	23137.61	Summer	False	
4	ORD104532	ID6231006329	15-09-2019	Arkansas	[('Universe Note', 1), ('Orbiter x460', 1), ('...'	9145	75.54	-37.823859	144.968892	25	6934.29	Spring	False	

In [5]:
```
#Displaying statistical info of eda data
eda_data.describe()
```

Out[5]:

	order_price	delivery_charges	customer_lat	customer_long	coupon_discount	order_total	distance_to_nearest_warehouse
count	500.000000	500.000000	500.000000	500.000000	500.000000	5.000000e+02	500.000000
mean	25522.210000	76.656250	-35.826294	144.960404	10.890000	3.920967e+04	2.204224
std	86333.729168	14.461465	12.040393	0.022720	8.648134	2.741948e+05	8.812416
min	585.000000	46.350000	-37.927122	144.924967	0.000000	6.360906e+02	0.107980
25%	7050.000000	65.962500	-37.816222	144.953488	5.000000	6.404758e+03	0.751425
50%	12807.500000	76.310000	-37.812165	144.965357	10.000000	1.129366e+04	1.030100
75%	37890.000000	82.550000	-37.805364	144.983885	15.000000	1.811919e+04	1.409825
max	947591.000000	114.040000	-37.626339	145.019837	25.000000	5.686278e+06	94.973400

In [6]:
```
print("Shape of eda data is:{eda_data.shape}")
```

```
Shape of eda data is: (500, 16)
```

In [7]:
```
#whether any columns contains null value or not
print(eda_data.isnull().sum())
```

```
order_id                       0
customer_id                    0
date                           0
nearest_warehouse              0
shopping_cart                  0
order_price                    0
delivery_charges               0
customer_lat                   0
customer_long                  0
coupon_discount                0
order_total                    0
season                         0
is_expedited_delivery          0
distance_to_nearest_warehouse  0
latest_customer_review         0
is_happy_customer              0
dtype: int64
```

In [8]:
```
#Analyzing datatype of each column
print(eda_data.dtypes)
```

```
order_id                       object
customer_id                    object
date                           object
nearest_warehouse              object
shopping_cart                  object
order_price                     int64
delivery_charges               float64
customer_lat                   float64
customer_long                  float64
coupon_discount                 int64
order_total                    float64
season                         object
is_expedited_delivery          object
distance_to_nearest_warehouse  float64
latest_customer_review         object
is_happy_customer               bool
dtype: object
```

In [9]:
```
#data type of date column is object instead of datetime, So fixing this problem
eda_data.date = pd.to_datetime(eda_data['date'])
```

In [10]:
```
print(eda_data.date.dtypes)
```

```
datetime64[ns]
```

In [11]:
```
print(eda_data.dtypes)  # Now data in date column is in datetime64 form
```

```
order_id                       object
customer_id                    object
date                   datetime64[ns]
nearest_warehouse              object
shopping_cart                  object
order_price                     int64
delivery_charges               float64
customer_lat                   float64
customer_long                  float64
coupon_discount                 int64
order_total                    float64
season                         object
is_expedited_delivery          object
distance_to_nearest_warehouse  float64
latest_customer_review         object
is_happy_customer               bool
dtype: object
```

In [12]:
```
#Analyzing order_id
#Checking whether order_id is unique or not
if eda_data['order_id'].nunique():
    print("Data in order_id is unique")          # execute when 'if' block will true
else:
    print("Data in order_id is not unique")      # execute when 'else' block will true
```

```
Data in order_id is unique
```

In [13]:
```
#Analyzing nearest_warehouse
eda_data['nearest_warehouse'].unique()
```

Out[13]:
```
array(['Arizona', 'Arkansas', 'California'], dtype=object)
```

In [14]:
```
#Arizona and arizona are same but here they both are different so make it same¶
#Similarly California and california are same
#To solve this problem, convert data in nearest_warehouse in title case
eda_data.nearest_warehouse = eda_data.nearest_warehouse.str.title()
```

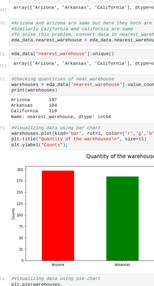
In [15]:
```
eda_data['nearest_warehouse'].unique()
```

Out[15]:
```
array(['Arizona', 'Arkansas', 'California'], dtype=object)
```

In [16]:
```
#Checking quantities of near_warehouse
warehouses = eda_data['nearest_warehouse'].value_counts()
print(warehouses)
```

```
Arizona      197
Arkansas     184
California   119
Name: nearest_warehouse, dtype: int64
```

In [17]:
```
#Visualizing data using bar chart
warehouses.plot(kind='bar', rot=0, color=['r','g','b'], figsize=(10,5))
plt.title("Quantity of the warehouses", size=15)
plt.ylabel("Counts");
```



In [18]:
```
#Visualizing data using pie chart
plt.pie(warehouses,
        labels=['Arizona','Arkansas','California'],
        autopct="%d.1f%%",
        explode=(0,.01,0.1),
        startangle=45,radius=1)
plt.title("Quantity of warehouses", size=15)
plt.show()
```

In [19]:
```
#it seems that Arizona is nearest warehouse accessible to mostly customer
```

In [20]:
```
#Analyzing order_price and order_total
#order_price: Price before any discounts and delivery charges
#order_total: Price after all discounts and/or delivery charges are applied

#Displaying Record whose order price is maximum
eda_data[eda_data.order_price == eda_data.order_price.max()]
```

Out[20]:

	order_id	customer_id	date	nearest_warehouse	shopping_cart	order_price	delivery_charges	customer_lat	customer_long	coupon_discount	order_total	season	is_expedited_delivery	d
335	ORD115640	ID6325546517	2019-12-31	Arizona	[('Lucent 330S', 1), ('Alvion Lite', 1), ('Ca...	947591	90.82	-37.809368	144.945542	10	9266.32	Summer	True	

In [21]:
```
#Displaying Record whose order price is minimum
```

In [22]:
```
#Now checking whether the order_total(after all discount and delivery charges are applied) is correct or there is some problem in it
eda_data.loc[:,['order_price','delivery_charges','coupon_discount','order_total']].head(10)
```

Out[22]:

	order_price	delivery_charges	coupon_discount	order_total
0	12200	79.89	10	11059.89
1	9080	62.71	0	9142.71
2	10670	65.87	10	9668.87
3	24800	57.61	15	23137.61
4	9145	75.54	25	6934.29
5	7810	71.22	10	7100.22
6	13700	74.84	5	13008.84
7	7960	52.28	5	10788.79
8	25390	107.58	10	22956.58
9	13320	62.26	15	11384.26

In [1]:

In [23]:
```
#It looks like that there is an error in order_total feature of eda data, So fixing this¶
order_final_list = list()
```

In [24]:
```
#Computing and appending order_final in 'order_final_list'
for i in range(0,len(eda_data['order_price'])):
    if (eda_data['coupon_discount'][i] != 0):
        dis_price = (eda_data['order_price'][i]) * (eda_data['coupon_discount'][i] / 100)   # apply coupon to original price
        price = dis_price + eda_data['delivery_charges'][i]                                  # adding delivery charges
        order_final = np.round(price,2)
        order_final_list.append(order_final)                                                 # appending final_price to order_final_list
    else:
        final_price = eda_data['order_price'][i] + eda_data['delivery_charges'][i]           # adding order_price and delivery_charges
        order_final = np.round(final_price,2)                                                # round off upto 2nd decimal place
        order_final_list.append(order_final)                                                 # appending final_price to order_final_list
```

In [25]:
```
print("Items (Price) in list is: {len(order_final_list)}")
```

```
Items (Price) in list is: 500
```

In [26]:
```
#Converting list to Series and set it in order_total
eda_data['order_total'] = pd.Series(order_final_list)
```

In [27]:
```
#Removing outlier
#Using IQR
#Checking for Outlier in order_total
Q1_order_total = eda_data['order_total'].quantile(0.25)     # Lower Quartile
Q3_order_total = eda_data['order_total'].quantile(0.75)     # upper Quartile
```

In [28]:
```
print("Lower Quartile is: {Q1_order_total}")
print("Upper Quartile is: {Q3_order_total}")
```

```
Lower Quartile is: 860.375
Upper Quartile is: 5020.8
```

In [29]:
```
#Finding IQR
# upper quartile - lower quartile

IQR_order_total = Q3_order_total - Q1_order_total
```

In [30]:
```
print("Value of IQR_order_total is: {IQR_order_total}")
```

```
Value of IQR_order_total is: 4159.625
```

In [31]:
```
#Finding lower and upper limit:
lower_limit_order_total = Q1_order_total - 1.5 * IQR_order_total
upper_limit_order_total = Q3_order_total + 1.5 * IQR_order_total

print("Lower limit of order_total is: {lower_limit_order_total}")
print("Upper limit of order_total is: {upper_limit_order_total}")

Lower limit of order_total is: -5379.8625
Upper limit of order_total is: 11359.4375
```

In [32]:
```
#Any value less then lower limit and greater then upper limit is outlier
#Displaying all outlier values
outlier_in_eda_data = eda_data[eda_data['order_total'] < lower_limit_order_total] | (eda_data['order_total'] > upper_limit_order_total)]

outlier_in_eda_data
```

Out[32]:

	order_id	customer_id	date	nearest_warehouse	shopping_cart	order_price	delivery_charges	customer_lat	customer_long	coupon_discount	order_total	season	is_expedited_delivery	d
18	ORD154680	ID4283906179	2019-03-06	California	[('Alvion 10', 2), ('JetaTV', 2), ('Gilmore', 1...]	31895	78.25	-37.811334	145.008678	0	31973.25	Winter	True	
12	ORD124395	ID8702352504	2019-01-02	Arizona	[('Alvion 10', 1), ('Universe Note', 1), ('Jewel ...]	19010	94.75	-37.805433	144.941333	0	19104.75	Summer	True	
13	ORD105642	ID3085952531	2019-12-24	Arkansas	[('Alvion Liner 2', 1), ('Alvion 10', 1), ('JetaT...]	19710	75.64	-37.816166	144.975278	0	19785.64	Summer	True	
14	ORD496722	ID3988448620	2019-04-04	Arkansas	[('JetaTV', 2), ('Gilmore', 1), ('Universe 3000', ...]	31900	79.78	-37.809403	144.972367	0	31979.78	Autumn	True	
20	ORD178890	ID4167440329	2019-11-20	Arizona	[('Candle Infuser', 1), ('JetaTV', 2), ('Universe ...]	13710	77.61	-37.809311	145.018595	0	13787.61	Spring	False	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
483	ORD103311	ID3777768861	2019-05-17	Arizona	[('Orbiter x460', 1), ('JetaTV', 2), ('Universe ...]	20745	67.67	-37.806155	144.958304	0	20812.67	Autumn	False	
487	ORD124473	ID4127674011	2019-04-07	Arkansas	[('Candle Infuser', 1), ('Thunder 750', 2), ('...]	33870	77.06	-37.802392	144.998158	0	33947.06	Winter	True	
496	ORD470810	ID4127417092	2019-04-29	Arkansas	[('Alvion 10', 2), ('Alvion Liner 2', 1), ('...]	38677	59.70	-37.817662	144.970405	10	38677.60	Winter	True	
496	ORD606900	ID6130015090	2019-03-28	Arkansas	[('JetaTV', 2), ('Universe Note', 2), ('Thun...]	22120	62.29	-37.815449	144.960083	0	22182.29	Autumn	False	
497	ORD657020	ID3707985642	2019-10-16	Arkansas	[('JetaTV', 1), ('Alvion 10', 1), ('Universe 3000...]	16490	106.99	-37.805979	144.961031	0	16596.99	Spring	True	

72 rows × 16 columns

In [33]:
```
#above records is outlier, So removing it from original data
print("Total outlier records in eda data is:{outlier_in_eda_data.shape[0]}")
```

```
Total outlier records in eda data is: 72
```

In [34]:
```
#Any data which is greater than lower limit and less then upper limit is Not an Outlier
data_without_outlier = eda_data[(eda_data['order_total'] > lower_limit_order_total) & (eda_data['order_total'] < upper_limit_order_total)]
```

In [35]:
```
print("Total numbers of records without outliers is:{data_without_outlier.shape[0]}")
```

```
Total numbers of records without outliers is: 428
```

In [36]:
```
#Displaying top 5 records of data without outliers
data_without_outlier.head()
```

Out[36]:

	order_id	customer_id	date	nearest_warehouse	shopping_cart	order_price	delivery_charges	customer_lat	customer_long	coupon_discount	order_total	season	is_expedited_delivery	dis
0	ORD162484	ID6197311990	2019-09-22	Arizona	[('Lucent 330S', 1), ('Thunder line', 1), ('Orbiter', 2)]	12200	79.89	-37.815105	144.952843	10	1220.00	Winter	True	
1	ORD195510	ID3928250845	2019-12-29	Arizona	[('Thunder line', 1), ('Orbiter Note', 2)]	9080	62.71	-37.802736	144.951118	0	9142.71	Summer	False	
2	ORD149479	ID8579391881	2019-01-03	Arkansas	[('Thunder line', 1), ('Jewel TV', 2)]	10670	65.87	-37.821302	144.957561	10	1067.00	Autumn	False	
3	ORD193324	ID4544561904	2019-12-04	Arkansas	[('Universe Note', 1), ('Alvion EVO 3000', 1), ('Orbiter', 2)]	24800	57.61	-37.814416	144.973073	15	3720.00	Summer	False	
4	ORD104532	ID6231006329	2019-11-28	Arkansas	[('Universe Note', 1), ('Orbiter x460', 1), ('...'	9145	75.54	-37.823859	144.968892	25	2286.25	Spring	False	

In [37]:
```
#Analyzing how many customers are happy with service
data_without_outlier['is_happy_customer'].unique()
```

Out[37]:
```
array([ True, False])
```

In [38]:
```
satisfaction = data_without_outlier['is_happy_customer'].value_counts()
print(satisfaction)
```

```
True    301
False   127
Name: is_happy_customer, dtype: int64
```

In [39]:
```
#Visualizing Data
plt.figure(figsize=(15,5))

plt.subplot(121)
satisfaction.plot(kind='bar', color=['g','r'], rot=1)
plt.xticks([0,1], labels=['Satisfy', 'Not Satisfy'])
plt.ylabel("Counts")
plt.title("Customer satisfaction Vs Not satisfaction", size=15)

plt.subplot(122)
plt.pie(satisfaction, labels=['Satisfy','Not Satisfy'], autopct='%.1f%%',explode=(0,0.1),shadow=True)
plt.title("Percentage of Customer satisfaction Vs Not satisfaction", size=15)

plt.show()
```

In [40]:
```
#70.3% customers are satisfy with service
```