

supply chain analysis

October 20, 2024

1 importing the necessary Python libraries and the dataset:

```
[190]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import linear_model
import pylab as pl
from sklearn.metrics import r2_score
```

```
[223]: df = pd.read_csv('C:\\Users\\AQ\\Downloads\\supply_chain_data.csv')
```

```
[72]: df.head()
```

```
[72]:
```

	Product type	SKU	Price	Availability	Number of products sold	\
0	haircare	SKU0	69.808006	55	802	
1	skincare	SKU1	14.843523	95	736	
2	haircare	SKU2	11.319683	34	8	
3	skincare	SKU3	61.163343	68	83	
4	skincare	SKU4	4.805496	26	871	

	Revenue generated	Customer demographics	Stock levels	Lead times	\
0	8661.996792	Non-binary	58	7	
1	7460.900065	Female	53	30	
2	9577.749626	Unknown	1	10	
3	7766.836426	Non-binary	23	13	
4	2686.505152	Non-binary	5	3	

	Order quantities	...	Location	Lead time	Production volumes	\
0	96	...	Mumbai	29	215	
1	37	...	Mumbai	23	517	
2	88	...	Mumbai	12	971	
3	59	...	Kolkata	24	937	
4	56	...	Delhi	5	414	

	Manufacturing lead time	Manufacturing costs	Inspection results	\
0	29	46.279879	Pending	

1	30	33.616769	Pending
2	27	30.688019	Pending
3	18	35.624741	Fail
4	3	92.065161	Fail

	Defect rates	Transportation modes	Routes	Costs
0	0.226410	Road	Route B	187.752075
1	4.854068	Road	Route B	503.065579
2	4.580593	Air	Route C	141.920282
3	4.746649	Rail	Route A	254.776159
4	3.145580	Air	Route A	923.440632

[5 rows x 24 columns]

[]:

2 Statistics for Numerical Columns

[73]: `df.describe().T`

```
[73]:
```

	count	mean	std	min	\
Price	100.0	49.462461	31.168193	1.699976	
Availability	100.0	48.400000	30.743317	1.000000	
Number of products sold	100.0	460.990000	303.780074	8.000000	
Revenue generated	100.0	5776.048187	2732.841744	1061.618523	
Stock levels	100.0	47.770000	31.369372	0.000000	
Lead times	100.0	15.960000	8.785801	1.000000	
Order quantities	100.0	49.220000	26.784429	1.000000	
Shipping times	100.0	5.750000	2.724283	1.000000	
Shipping costs	100.0	5.548149	2.651376	1.013487	
Lead time	100.0	17.080000	8.846251	1.000000	
Production volumes	100.0	567.840000	263.046861	104.000000	
Manufacturing lead time	100.0	14.770000	8.912430	1.000000	
Manufacturing costs	100.0	47.266693	28.982841	1.085069	
Defect rates	100.0	2.277158	1.461366	0.018608	
Costs	100.0	529.245782	258.301696	103.916248	

	25%	50%	75%	max
Price	19.597823	51.239830	77.198228	99.171329
Availability	22.750000	43.500000	75.000000	100.000000
Number of products sold	184.250000	392.500000	704.250000	996.000000
Revenue generated	2812.847151	6006.352023	8253.976920	9866.465458
Stock levels	16.750000	47.500000	73.000000	100.000000
Lead times	8.000000	17.000000	24.000000	30.000000
Order quantities	26.000000	52.000000	71.250000	96.000000
Shipping times	3.750000	6.000000	8.000000	10.000000
Shipping costs	3.540248	5.320534	7.601695	9.929816

Lead time	10.000000	18.000000	25.000000	30.000000
Production volumes	352.000000	568.500000	797.000000	985.000000
Manufacturing lead time	7.000000	14.000000	23.000000	30.000000
Manufacturing costs	22.983299	45.905622	68.621026	99.466109
Defect rates	1.009650	2.141863	3.563995	4.939255
Costs	318.778455	520.430444	763.078231	997.413450

3 Check for missing values in each column

```
[188]: categorical_cols = [
        'Product type', 'Customer demographics', 'Shipping carriers',
        'Supplier name', 'Location', 'Inspection results', 'Transportation modes',
        'Routes'
    ]
    for col in categorical_cols:
        print(f"\nDistribution of {col}:\n", df[col].value_counts())
```

Distribution of Product type:

```
Product type
skincare      40
haircare      34
cosmetics     26
Name: count, dtype: int64
```

Distribution of Customer demographics:

```
Customer demographics
Unknown       31
Female        25
Non-binary    23
Male          21
Name: count, dtype: int64
```

Distribution of Shipping carriers:

```
Shipping carriers
Carrier B     43
Carrier C     29
Carrier A     28
Name: count, dtype: int64
```

Distribution of Supplier name:

```
Supplier name
Supplier 1    27
Supplier 2    22
Supplier 5    18
Supplier 4    18
Supplier 3    15
```

Name: count, dtype: int64

Distribution of Location:

Location
Kolkata 25
Mumbai 22
Chennai 20
Bangalore 18
Delhi 15

Name: count, dtype: int64

Distribution of Inspection results:

Inspection results
Pending 41
Fail 36
Pass 23

Name: count, dtype: int64

Distribution of Transportation modes:

Transportation modes
Road 29
Rail 28
Air 26
Sea 17

Name: count, dtype: int64

Distribution of Routes:

Routes
Route A 43
Route B 37
Route C 20

Name: count, dtype: int64

```
[78]: print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 100 entries, 0 to 99
```

```
Data columns (total 24 columns):
```

#	Column	Non-Null Count	Dtype
0	Product type	100 non-null	object
1	SKU	100 non-null	object
2	Price	100 non-null	float64
3	Availability	100 non-null	int64
4	Number of products sold	100 non-null	int64
5	Revenue generated	100 non-null	float64
6	Customer demographics	100 non-null	object
7	Stock levels	100 non-null	int64

8	Lead times	100 non-null	int64
9	Order quantities	100 non-null	int64
10	Shipping times	100 non-null	int64
11	Shipping carriers	100 non-null	object
12	Shipping costs	100 non-null	float64
13	Supplier name	100 non-null	object
14	Location	100 non-null	object
15	Lead time	100 non-null	int64
16	Production volumes	100 non-null	int64
17	Manufacturing lead time	100 non-null	int64
18	Manufacturing costs	100 non-null	float64
19	Inspection results	100 non-null	object
20	Defect rates	100 non-null	float64
21	Transportation modes	100 non-null	object
22	Routes	100 non-null	object
23	Costs	100 non-null	float64

dtypes: float64(6), int64(9), object(9)
memory usage: 18.9+ KB
None

```
[74]: df.isnull().sum()
```

```
[74]: Product type      0
SKU                  0
Price                0
Availability          0
Number of products sold  0
Revenue generated     0
Customer demographics  0
Stock levels          0
Lead times            0
Order quantities       0
Shipping times         0
Shipping carriers       0
Shipping costs         0
Supplier name          0
Location               0
Lead time              0
Production volumes      0
Manufacturing lead time  0
Manufacturing costs     0
Inspection results      0
Defect rates            0
Transportation modes    0
Routes                  0
Costs                   0
dtype: int64
```

```
[80]: print(df.drop_duplicates(inplace=True))
```

None

3.0.1 A new column titled ‘Inventory Turnover’ has been added to the dataset to assess inventory management effectiveness.

```
[87]: df['Inventory Turnover'] = df['Number of products sold'] / df['Stock levels']
print(df)
```

	Product type	SKU	Price	Availability	Number of products sold \
0	haircare	SKU0	69.808006	55	802
1	skincare	SKU1	14.843523	95	736
2	haircare	SKU2	11.319683	34	8
3	skincare	SKU3	61.163343	68	83
4	skincare	SKU4	4.805496	26	871
..
95	haircare	SKU95	77.903927	65	672
96	cosmetics	SKU96	24.423131	29	324
97	haircare	SKU97	3.526111	56	62
98	skincare	SKU98	19.754605	43	913
99	haircare	SKU99	68.517833	17	627

	Revenue generated	Customer demographics	Stock levels	Lead times \
0	8661.996792	Non-binary	58	7
1	7460.900065	Female	53	30
2	9577.749626	Unknown	1	10
3	7766.836426	Non-binary	23	13
4	2686.505152	Non-binary	5	3
..
95	7386.363944	Unknown	15	14
96	7698.424766	Non-binary	67	2
97	4370.916580	Male	46	19
98	8525.952560	Female	53	1
99	9185.185829	Unknown	55	8

	Order quantities	...	Lead time	Production volumes \
0	96	...	29	215
1	37	...	23	517
2	88	...	12	971
3	59	...	24	937
4	56	...	5	414
..
95	26	...	18	450
96	32	...	28	648
97	4	...	10	535
98	27	...	28	581
99	59	...	29	921

	Manufacturing lead time	Manufacturing costs	Inspection results	\
0	29	46.279879	Pending	
1	30	33.616769	Pending	
2	27	30.688019	Pending	
3	18	35.624741	Fail	
4	3	92.065161	Fail	
..	
95	26	58.890686	Pending	
96	28	17.803756	Pending	
97	13	65.765156	Fail	
98	9	5.604691	Pending	
99	2	38.072899	Fail	

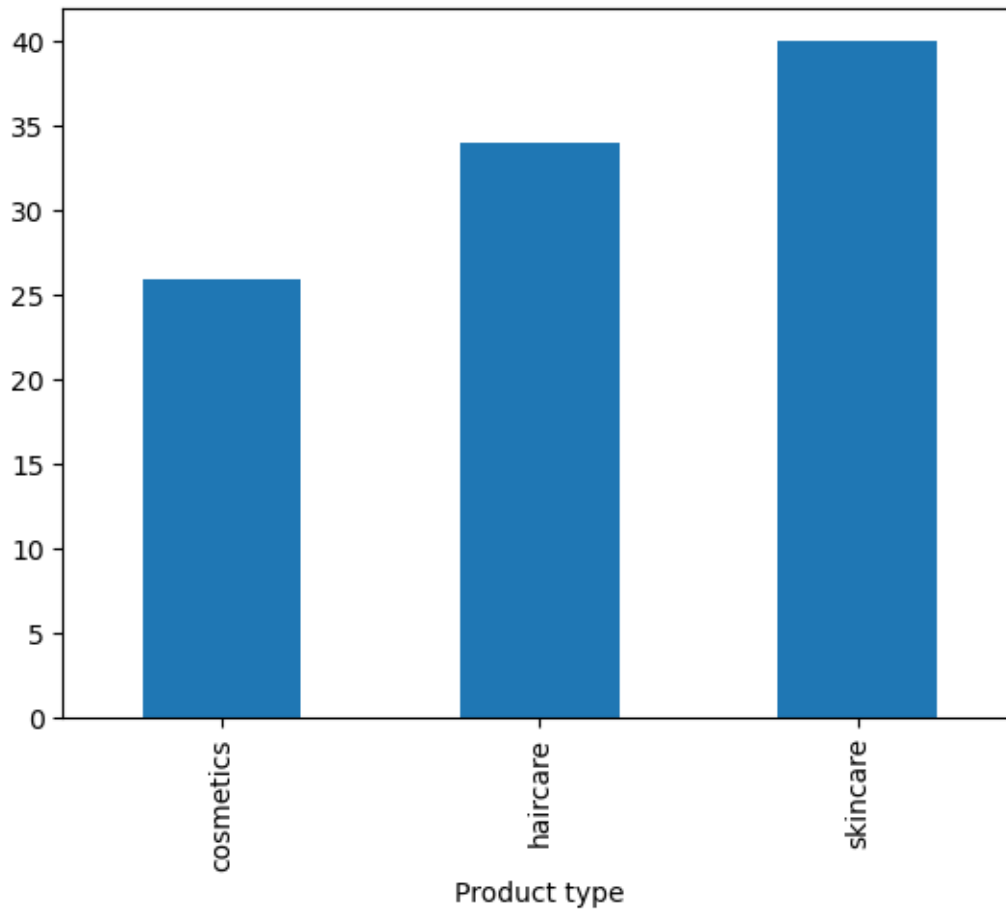
	Defect rates	Transportation modes	Routes	Costs	Inventory Turnover
0	0.226410	Road	Route B	187.752075	13.827586
1	4.854068	Road	Route B	503.065579	13.886792
2	4.580593	Air	Route C	141.920282	8.000000
3	4.746649	Rail	Route A	254.776159	3.608696
4	3.145580	Air	Route A	923.440632	174.200000
..
95	1.210882	Air	Route A	778.864241	44.800000
96	3.872048	Road	Route A	188.742141	4.835821
97	3.376238	Road	Route A	540.132423	1.347826
98	2.908122	Rail	Route A	882.198864	17.226415
99	0.346027	Rail	Route B	210.743009	11.400000

[100 rows x 25 columns]

4 Deep Exploratory Analysis

```
[90]: df.groupby(['Product type'])['Product type'].count().plot(kind='bar')
print(df.groupby(['Product type'])['Product type'].count())
```

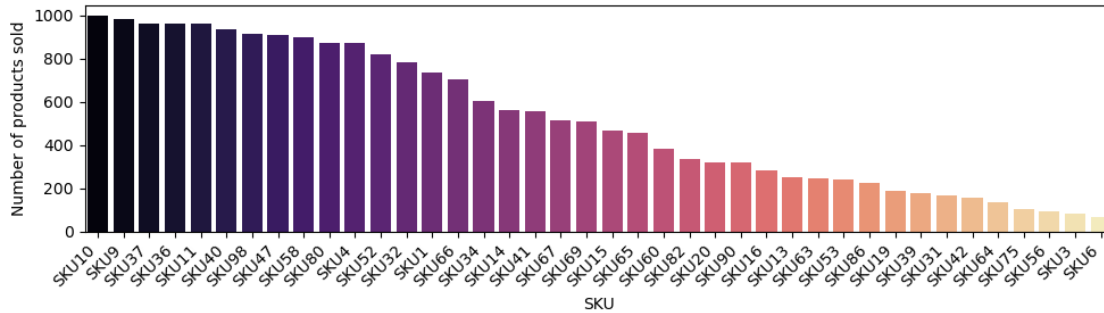
```
Product type
cosmetics    26
haircare     34
skincare     40
Name: Product type, dtype: int64
```



5 Analyzing SKU

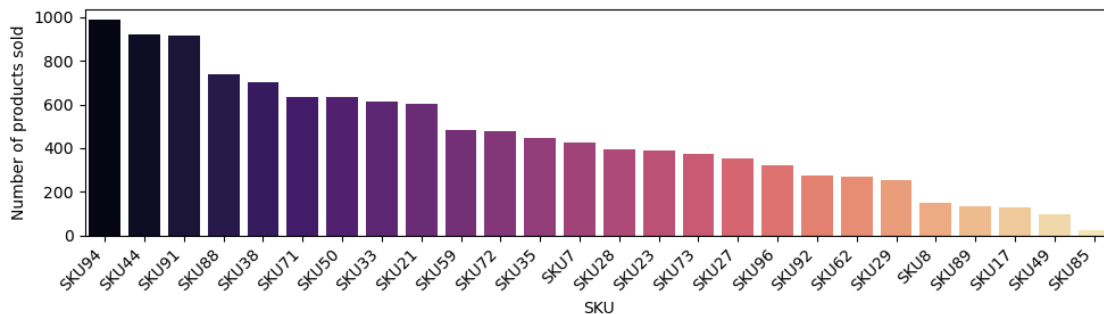
5.1 Number of sold Skincare products by SKU

```
[92]: df_skincare = df[df['Product type'] == 'skincare'].sort_values(by=['Number of_
    ↳products sold'], ascending=False)
fig, ax = plt.subplots(figsize=(10, 3))
colors = sns.color_palette("magma", n_colors=len(df_skincare))
sns.barplot(data=df_skincare, x='SKU', y='Number of products sold', ax=ax,
    ↳hue='SKU', palette=colors, dodge=False)
ax.set_xticks(range(len(df_skincare['SKU'])))
ax.set_xticklabels(df_skincare['SKU'], rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

5.2 Number of sold cosmetics by SKU

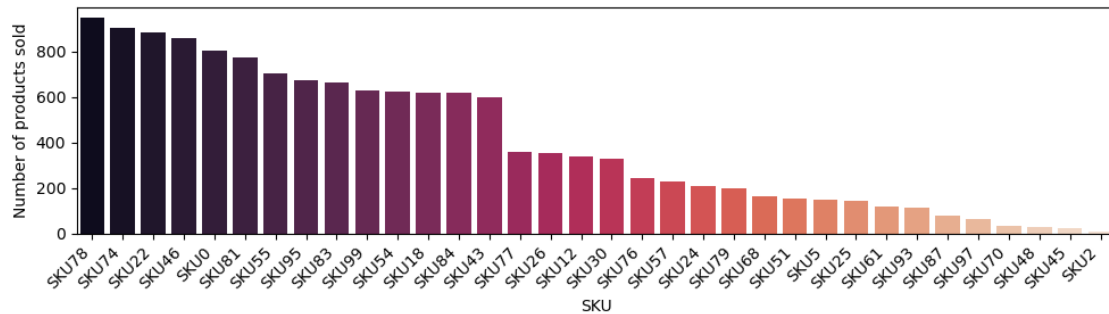
```
[95]: df_cosmetics = df[df['Product type'] == 'cosmetics'].sort_values(by=['Number of
    ↳products sold'], ascending=False)
fig, ax = plt.subplots(figsize=(10, 3))
colors = sns.color_palette("magma", n_colors=len(df_cosmetics))
sns.barplot(data=df_cosmetics, x='SKU', y='Number of products sold', ax=ax,
    ↳hue='SKU', palette=colors, dodge=False)
ax.set_xticks(range(len(df_cosmetics['SKU'])))
ax.set_xticklabels(df_cosmetics['SKU'], rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



5.3 Number of sold Haircare products by SKU

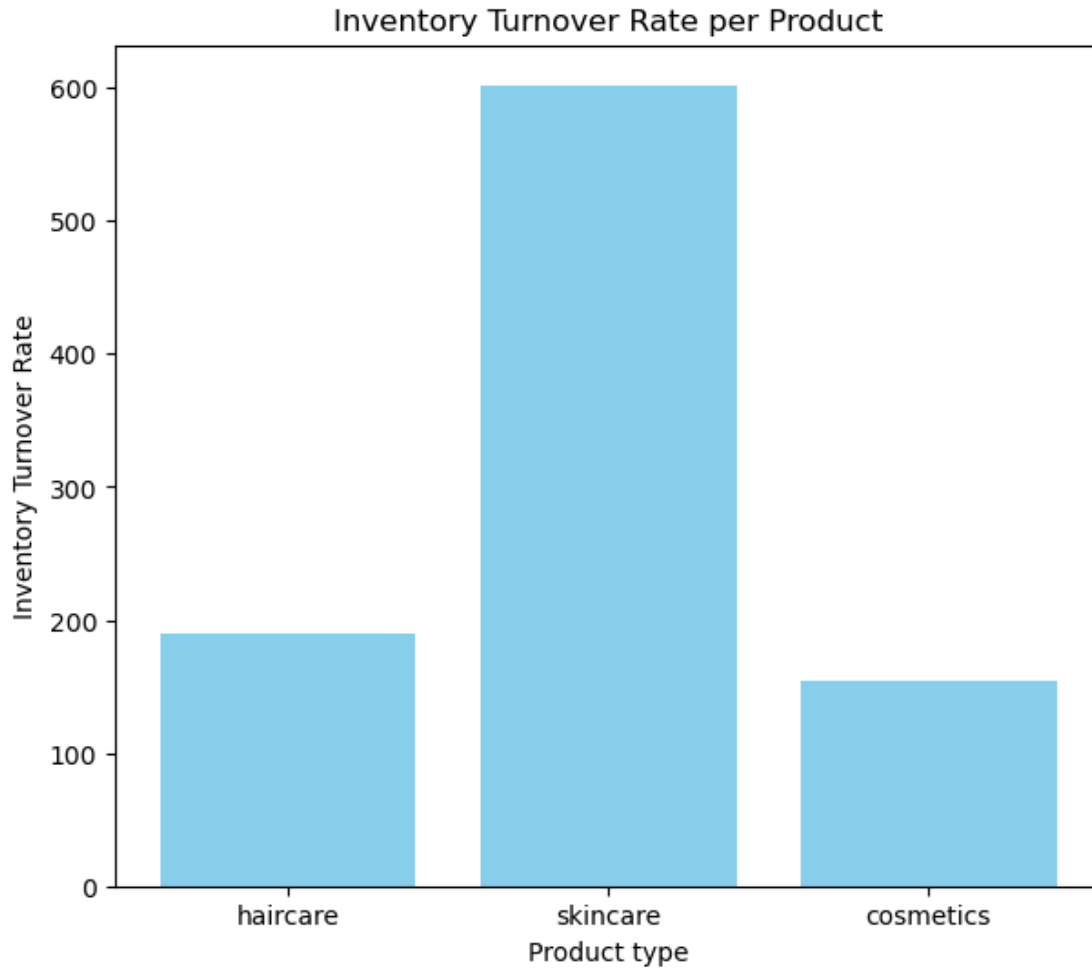
```
[97]: df_haircare = df[df['Product type'] == 'haircare'].sort_values(by=['Number of
    ↳products sold'], ascending=False)
fig, ax = plt.subplots(figsize=(10, 3))
colors = sns.color_palette("rocket", n_colors=len(df_haircare))
sns.barplot(data=df_haircare, x='SKU', y='Number of products sold', ax=ax,
    ↳hue='SKU', palette=colors, dodge=False)
ax.set_xticks(range(len(df_haircare['SKU'])))
```

```
ax.set_xticklabels(df_haircare['SKU'], rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



5.4 Inventory Turnover Rate Per Product

```
[99]: plt.figure(figsize=(7, 6))
plt.bar(df['Product type'], df['Inventory Turnover'], color='skyblue')
plt.title('Inventory Turnover Rate per Product')
plt.xlabel('Product type')
plt.ylabel('Inventory Turnover Rate')
plt.show()
```

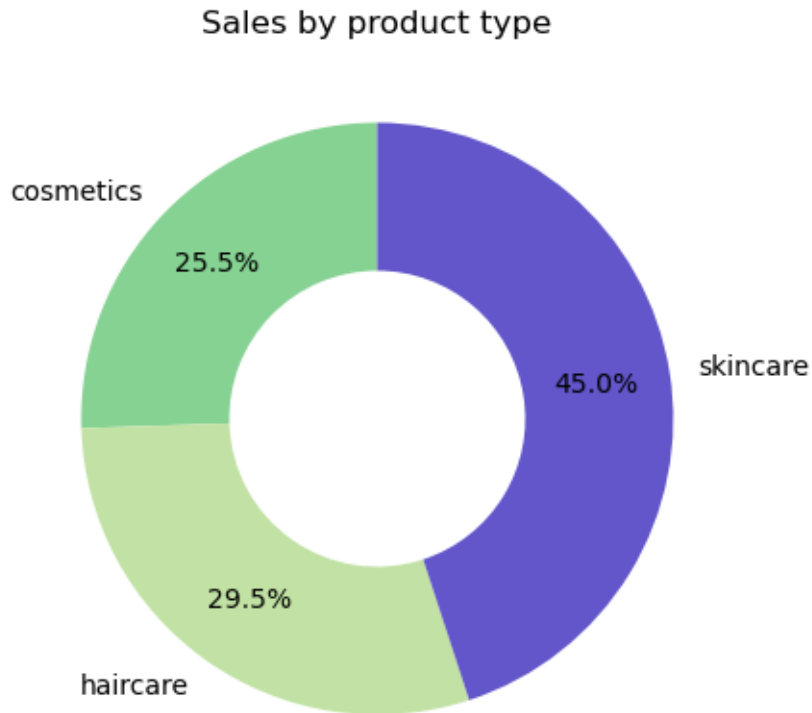


5.4.1 The analysis indicates that skincare products have the highest inventory levels, reflecting strong demand and consumer interest in skincare solutions.

5.5 Sales by product type

```
[102]: colors=['#86D293', '#C1E2A4', '#6256CA']
product_sales=df.groupby('Product type')['Number of products sold'].sum().
    ↪reset_index()
plt.figure(figsize=(8, 5))
plt.pie(product_sales['Number of products sold'],
        labels=product_sales['Product type'],
        autopct='%1.1f%%',startangle=90, colors=colors,
        ↪,wedgeprops=dict(width=0.5),pctdistance=0.75 )
plt.title('Sales by product type')
plt.show
```

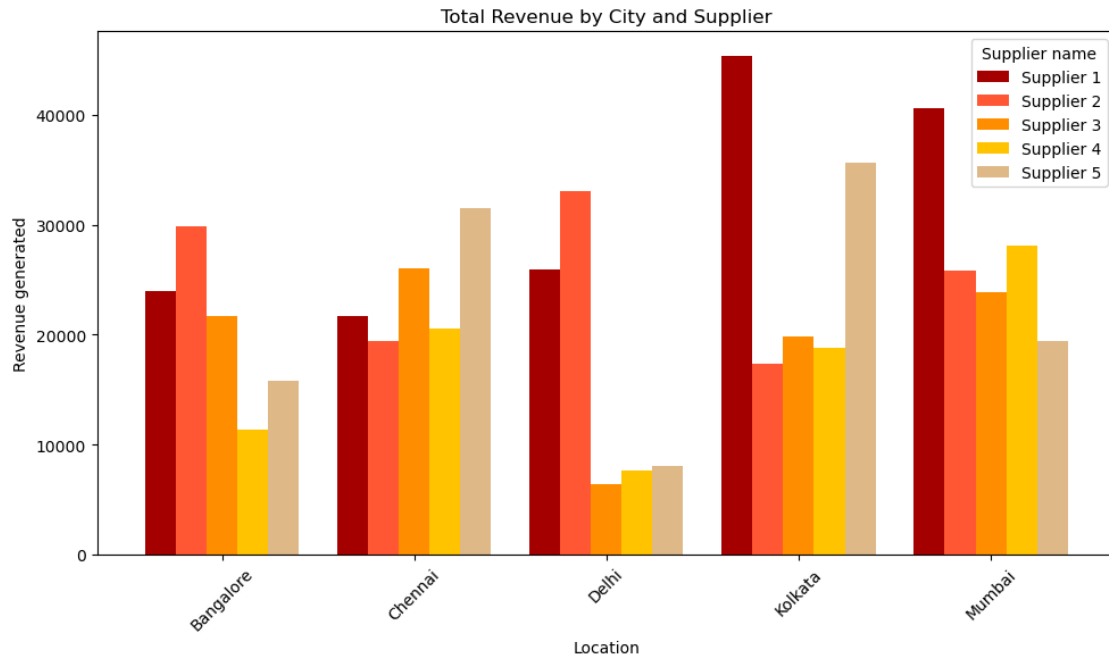
```
[102]: <function matplotlib.pyplot.show(close=None, block=None)>
```



5.5.1 Skincare products are the most preferred products as they are the highest in sales.

5.6 Total Revenue by City

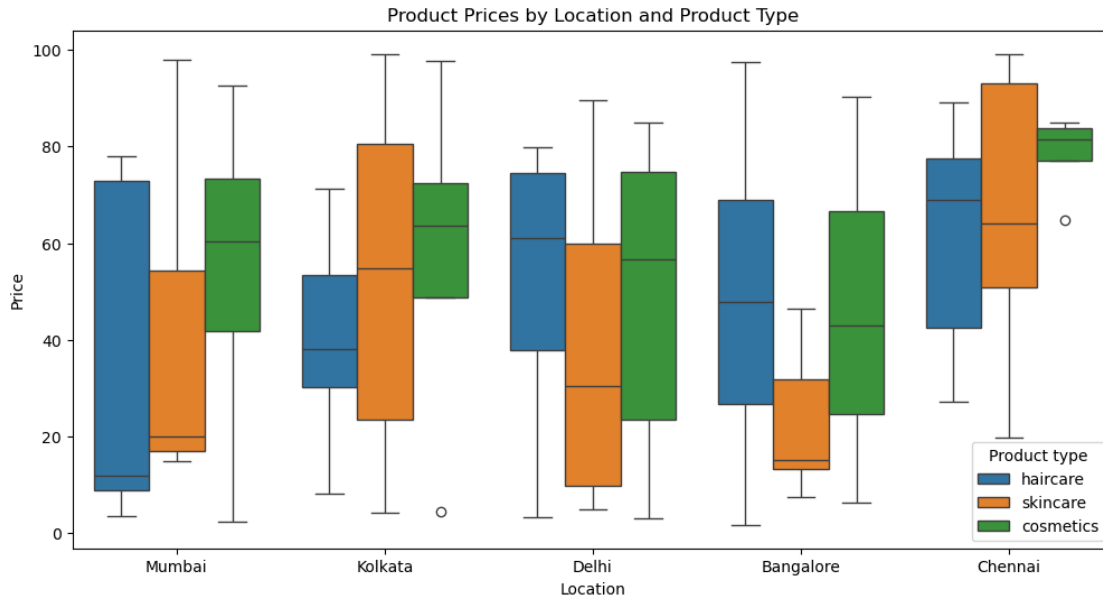
```
[109]: city_supplier_sales = df.groupby(['Location', 'Supplier name'])['Revenue_↵
generated'].sum().unstack().fillna(0)
colors = ['#A50000', '#FF5733', '#FF8D00', '#FFC300', '#DEB887']
city_supplier_sales.plot(kind='bar', color=colors, figsize=(10,6), width=0.8)
plt.xlabel('Location')
plt.ylabel('Revenue generated')
plt.title('Total Revenue by City and Supplier')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



5.6.1 Supplier 1 generates the highest revenue in the city of Kolkata.

5.7 Distribution of Product Prices by Location and Type

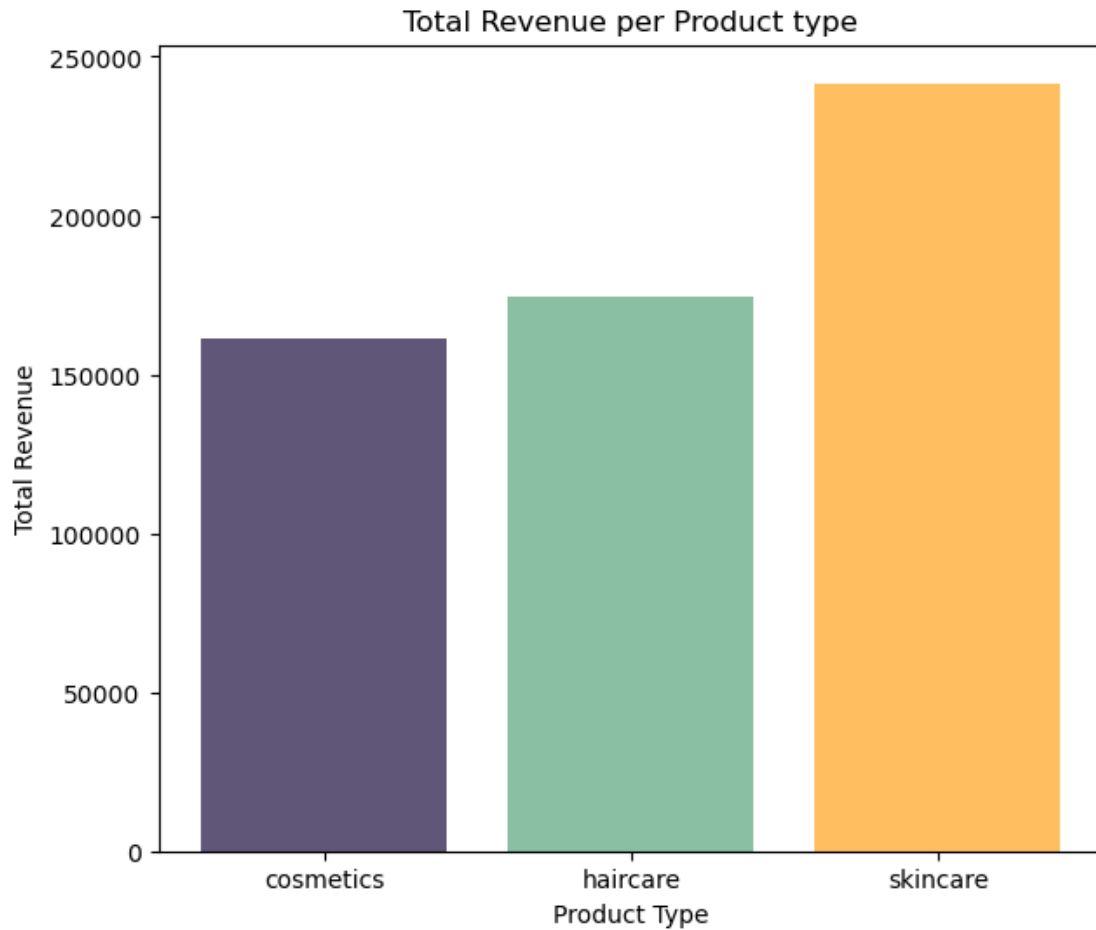
```
[113]: plt.figure(figsize=(12, 6))
sns.boxplot(data=df, x='Location', y='Price', hue='Product type')
plt.title('Product Prices by Location and Product Type')
plt.xlabel('Location')
plt.ylabel('Price')
plt.show()
```



5.7.1 When it comes to prices, Chennai leads in all the categories.

5.8 Total Revenue by product type

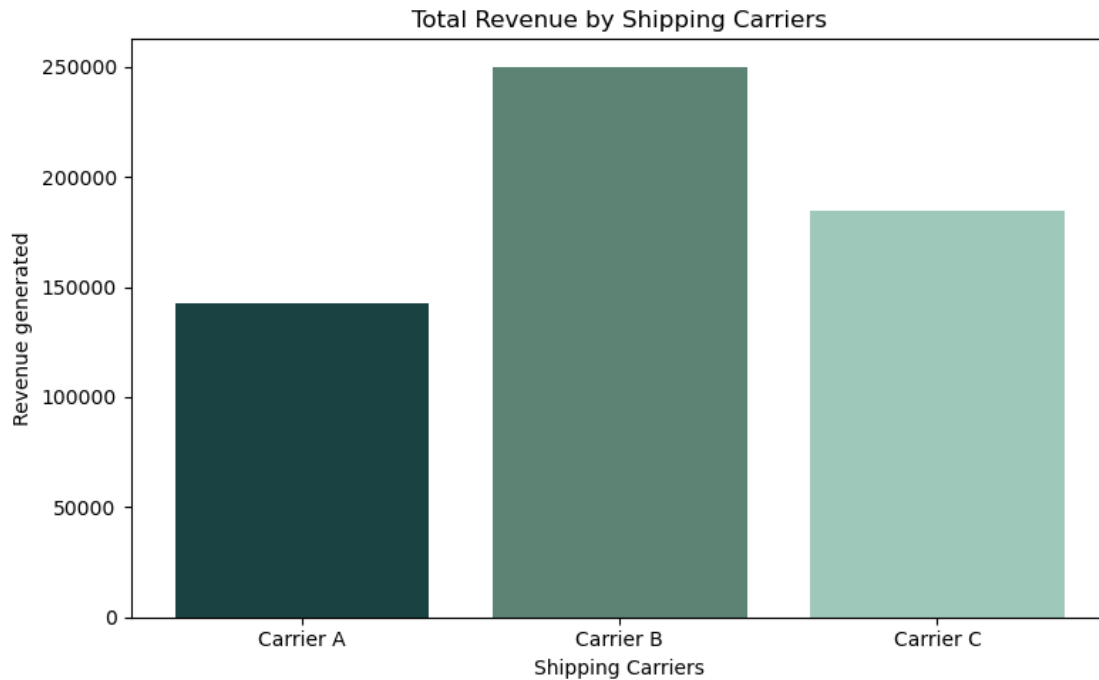
```
[119]: colors=['#605678', '#8ABFA3', '#FFBF61']
revenue_per_product = df.groupby('Product type')['Revenue generated'].sum().
    ↪reset_index()
plt.figure(figsize=(7, 6))
plt.bar(revenue_per_product['Product type'], revenue_per_product['Revenue_
    ↪generated'], color=colors)
plt.title('Total Revenue per Product type')
plt.xlabel('Product Type')
plt.ylabel('Total Revenue')
plt.show()
```



5.8.1 Skincare products generate the highest revenue.

5.9 Total Revenue by shipping carrier

```
[122]: colors=['#1B4242','#5C8374','#9EC8B9']
carrier_revenue=df.groupby('Shipping carriers')['Revenue generated'].sum().
    ↪reset_index()
plt.figure(figsize=(8,5))
plt.bar(carrier_revenue['Shipping carriers'],carrier_revenue['Revenue_
    ↪generated'], color=colors)
plt.xlabel('Shipping Carriers')
plt.ylabel('Revenue generated')
plt.title('Total Revenue by Shipping Carriers')
plt.tight_layout()
plt.show()
```

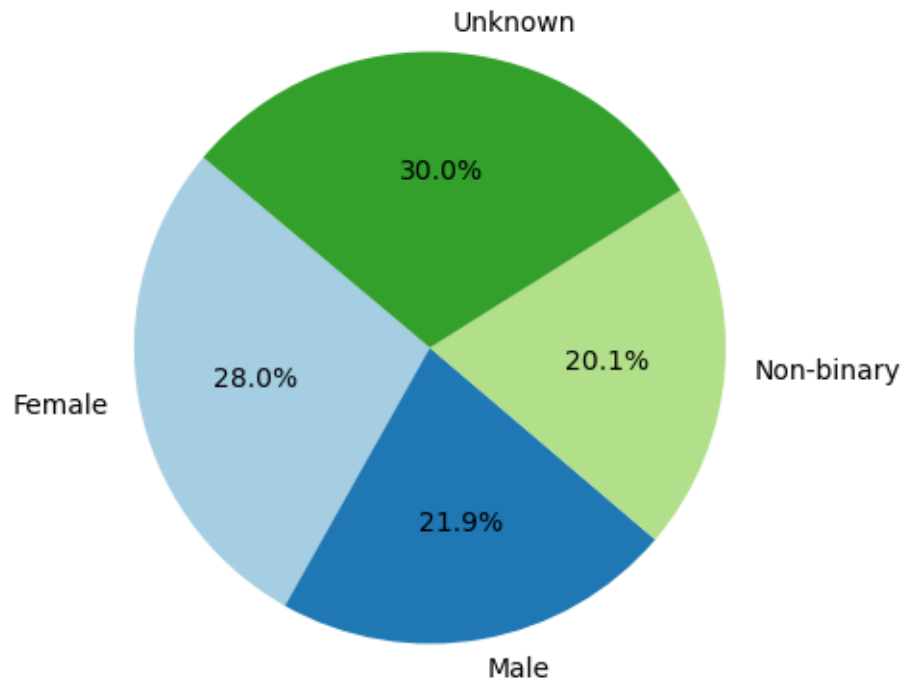


5.9.1 The company use 3 Carriers for transportation and carrier B helps the company in generating more revenue.

5.10 Revenue distribution by customers demographics

```
[126]: revenue_per_demographics = df.groupby('Customer demographics')['Revenue_
        ↪generated'].sum().reset_index()
plt.figure(figsize=(5, 5))
plt.pie(revenue_per_demographics['Revenue generated'],
        ↪labels=revenue_per_demographics['Customer demographics'], autopct='%1.1f%%',
        ↪startangle=140, colors=plt.cm.Paired.colors)
plt.title('Total Revenue per Customer Demographics')
plt.show()
```


Total Revenue per Customer Demographics



5.10.1 so 28% of total revenue comes from Femals & 21% comes from Males.

5.11 Analyzing Lead Time and Manufacturing cost

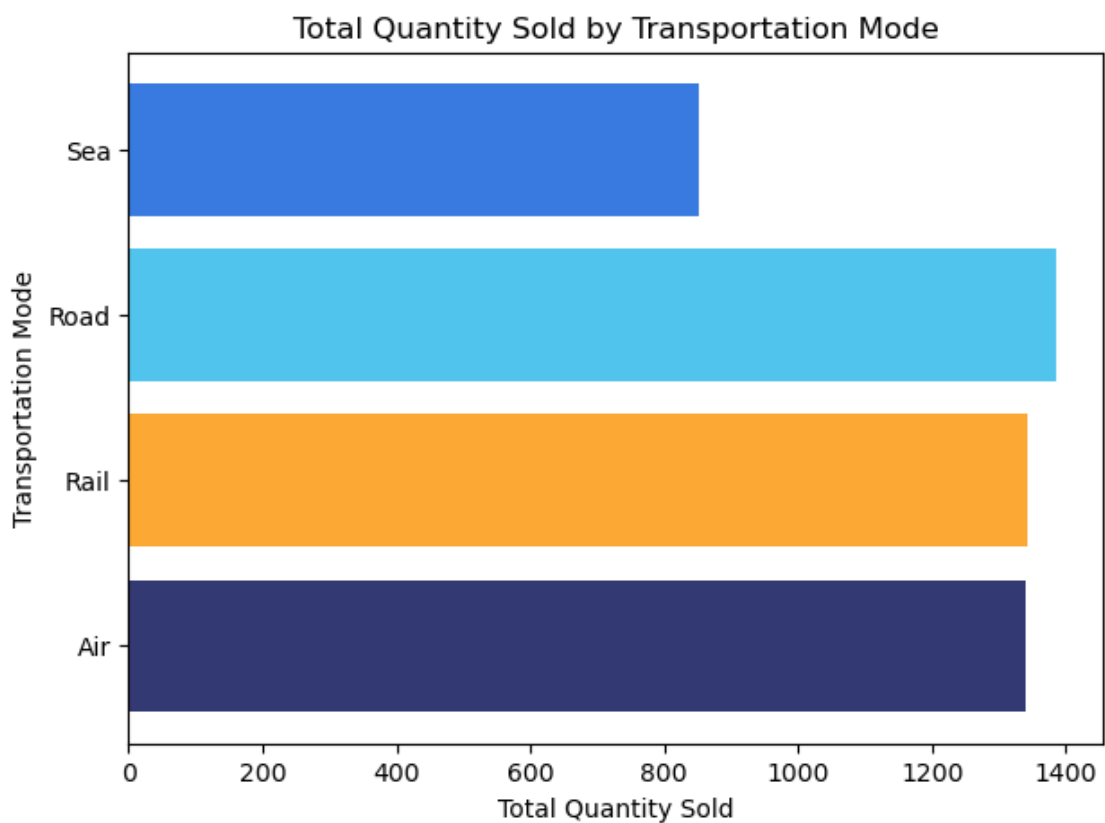
5.11.1 Average lead time and Average Manufacturing Cost for all the products

```
[129]: av_time=df.groupby('Product type')['Lead time'].mean().reset_index()
av_manufacturecost=df.groupby('Product type')['Manufacturing costs'].mean().
        ↪reset_index()
comb=pd.merge(av_time,av_manufacturecost ,on='Product type')
comb.rename(columns={'Lead time': 'Average Lead Time', 'Manufacturing costs':
        ↪'Average Manufacturing Costs'}, inplace=True)
comb
```

```
[129]:  Product type  Average Lead Time  Average Manufacturing Costs
0    cosmetics          13.538462          43.052740
1     haircare          18.705882          48.457993
2     skincare          18.000000          48.993157
```

5.12 Total Quantity Sold by Transportation Mode

```
[132]: colors=['#333A73', '#FBA834', '#50C4ED', '#387ADF']
usage_per_transportation = df.groupby('Transportation modes')['Order_
↳quantities'].sum().reset_index()
plt.figure(figsize=(7, 5))
plt.barh(usage_per_transportation['Transportation modes'],
↳usage_per_transportation['Order quantities'],color=colors )
plt.title('Total Quantity Sold by Transportation Mode')
plt.xlabel('Total Quantity Sold')
plt.ylabel('Transportation Mode')
plt.show()
```

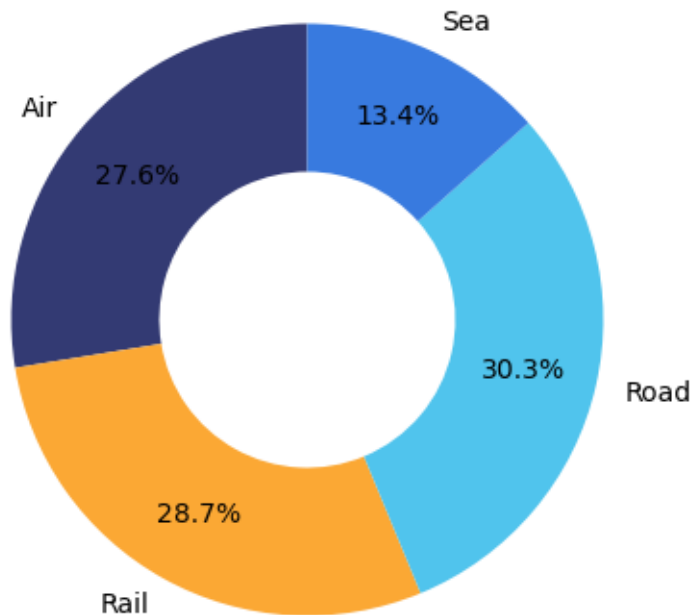


5.13 Cost Distribution by Transportation Modes

```
[135]: colors=['#333A73', '#FBA834', '#50C4ED', '#387ADF']
Tcosts=df.groupby('Transportation modes')['Costs'].sum().reset_index()
plt.figure(figsize=(8, 5))
plt.pie(Tcosts['Costs'],labels=Tcosts['Transportation modes'],
↳autopct='%1.1f%%',startangle=90 ,colors=colors ,wedgeprops=dict(width=0.
↳5),pctdistance=0.75 )
```

```
plt.title('Cost Distribution by Transportation Modes')
plt.show()
```

Cost Distribution by Transportation Modes

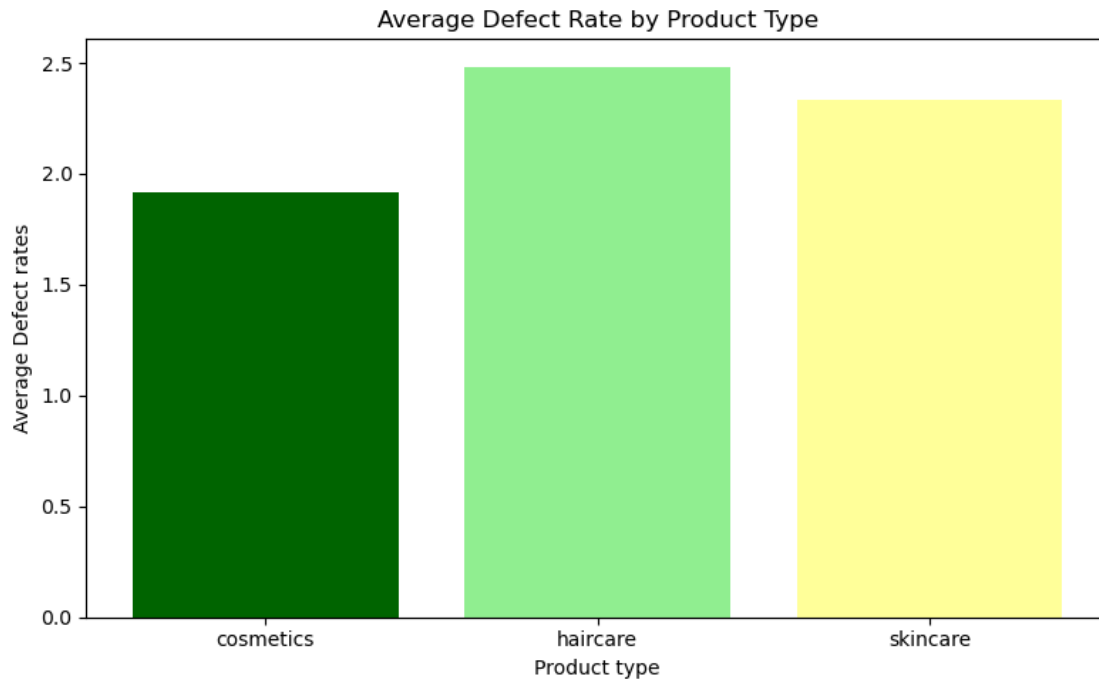


5.13.1 Road and Rail transportation methods are more expensive for the company compared to other forms of transport.

5.14 Analyzing Defect Rate

5.14.1 Average Defect Rate by Product Type

```
[142]: colors=['#006400','#90EE90','#FFFF99']
avg_defect=df.groupby('Product type')['Defect rates'].mean().reset_index()
plt.figure(figsize=(8,5))
plt.bar(avg_defect['Product type'],avg_defect['Defect rates'], color=colors)
plt.xlabel('Product type')
plt.ylabel('Average Defect rates')
plt.title('Average Defect Rate by Product Type')
plt.tight_layout()
plt.show()
```

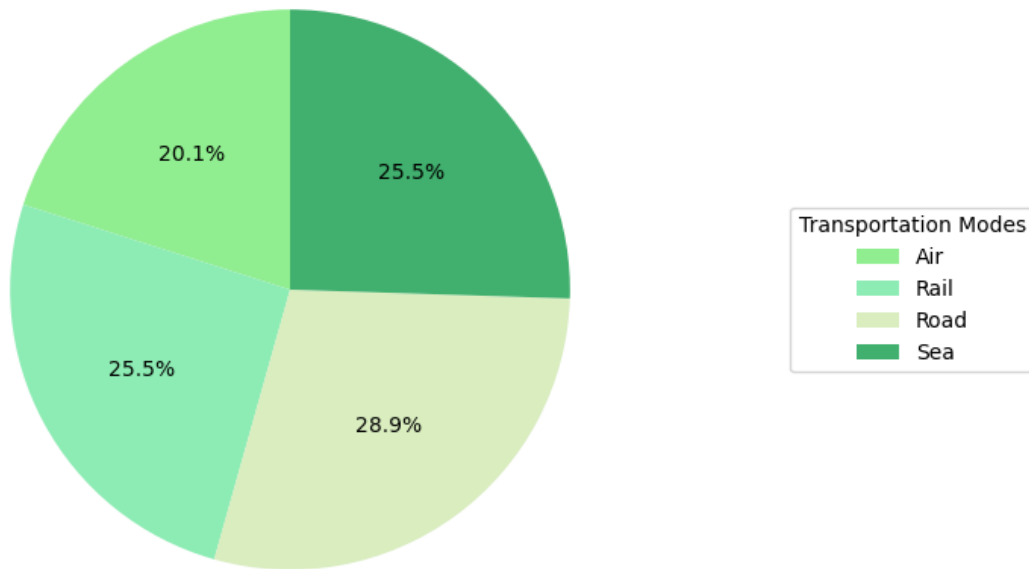


5.14.2 The defect rate for haircare products is the highest one.

5.14.3 Defect Rate by Transportation Mode

```
[147]: colors=['#90EE90','#8DECB4','#D9EDBF','#41B06E']
T_defect=df.groupby('Transportation modes')['Defect rates'].mean().reset_index()
plt.figure(figsize=(8, 6))
plt.pie(T_defect['Defect rates'],
        labels=None,
        autopct='%1.1f%%',startangle=90, colors=colors,labeldistance=1.1)
plt.title('Defect Rate by Transportation Mode')
plt.legend(T_defect['Transportation modes'], title="Transportation Modes",
           loc="center left",bbox_to_anchor=(1.2, 0.5))
plt.show()
```

Defect Rate by Transportation Mode



5.14.4 Products delivered by road have the highest chance of being defective, while products delivered by air have the lowest rate of defects.

5.14.5 “Revenue vs. Manufacturing Scatter Plot”

```
[169]: regr = linear_model.LinearRegression()
train_x = np.asanyarray(df[['Revenue generated']])
train_y = np.asanyarray(df[['Manufacturing costs']])
regr.fit(train_x, train_y)

print ('Coefficients: ', regr.coef_)
print ('Intercept: ',regr.intercept_)

y_pred = regr.predict(train_x)
r2 = r2_score(train_y, y_pred)

print("R-squared:", r2)

plt.scatter(df[['Revenue generated']], df[['Manufacturing costs']], 
            color='blue')
plt.plot(train_x, regr.coef_[0][0]*train_x + regr.intercept_[0], '-r')
plt.xlabel('Revenue generated')
plt.ylabel("Manufacturing costs")
```

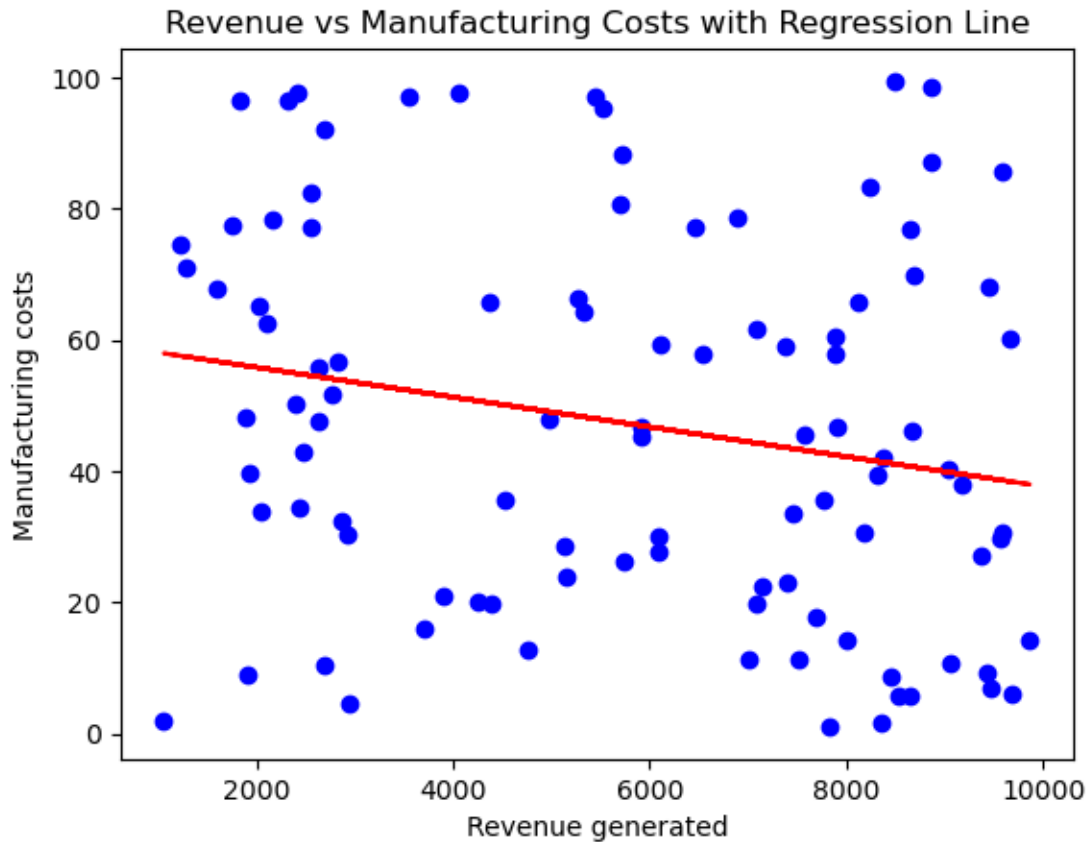
```
plt.title('Revenue vs Manufacturing Costs with Regression Line')
```

```
Coefficients: [[-0.00226982]]
```

```
Intercept: [60.37727428]
```

```
R-squared: 0.0458067325432433
```

```
[169]: Text(0.5, 1.0, 'Revenue vs Manufacturing Costs with Regression Line')
```



5.14.6 Based on the linear regression analysis shown in the image, the R-squared value of 0.0458 indicates that the relationship between Manufacturing and Revenue is very weak. It also showed that all the tested data gave the same result..