In [ ]:
```
pip install plotly
```

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import KMeans
import warnings
import os
import plotly as py
import plotly.graph_objs as go
import plotly.figure_factory as ff
import plotly.graph_objects as go
import plotly.express as px
from plotly.subplots import make_subplots
```

In [2]:
```
df = pd.read_csv('sirclo_final.csv')
df_raw = df.copy()
```

In [3]:
```
df
```

Out[3]:

	site_id	latest_plan	plan_duration	num_activeproducts	lastproduct_duration	num_orders
0	cv-alam-sakti-perkasa	0	149	6	149	0
1	naufal-digistore	0	354	6	354	0
2	ibby	0	231	6	231	0
3	jackbiz	0	200	6	200	0
4	hantaranadiba	0	42	6	42	0
...	...	...	...	...	...	...
50098	angid	0	219	6	219	0
50099	the-perfume-boutique	0	126	6	126	0
50100	miftahulhudaalfurqon	0	100	6	100	0
50101	momoril	0	145	6	145	0
50102	lilbroandsist	1	146	25	5	773

50103 rows × 8 columns

In [4]:
```
df.describe().T
```

Out[4]:

	count	mean	std	min	25%	50%	75%	max
latest_plan	50103.0	4.209329e-02	2.008040e-01	0.0	0.0	0.0	0.0	1.000000e+00
plan_duration	50103.0	1.497810e+02	1.431464e+02	0.0	67.0	124.0	177.0	3.067000e+03
num_activeproducts	50103.0	1.867559e+01	6.527439e+02	0.0	6.0	6.0	6.0	9.961400e+04
lastproduct_duration	50103.0	1.558831e+02	1.439335e+02	0.0	65.0	123.0	178.0	7.000000e+02

	count	mean	std	min	25%	50%	75%	max
num_orders	50103.0	1.088977e+02	2.867127e+03	0.0	0.0	0.0	0.0	3.830830e+05
total_sales	50103.0	2.748903e+07	6.438988e+08	0.0	0.0	0.0	0.0	6.506127e+10
order_source	50103.0	4.779993e+00	8.621143e-01	0.0	5.0	5.0	5.0	5.000000e+00

1. Data Preparation - SMOTE

In [5]:
```
!pip install imblearn
from imblearn.over_sampling import SMOTE
```

```
Requirement already satisfied: imblearn in /Users/shanneysuhendra/Desktop/cs135/miniconda
3/lib/python3.9/site-packages (0.0)
Requirement already satisfied: imbalanced-learn in /Users/shanneysuhendra/Desktop/cs135/mi
niconda3/lib/python3.9/site-packages (from imblearn) (0.9.0)
Requirement already satisfied: joblib>=0.11 in /Users/shanneysuhendra/Desktop/cs135/minico
nda3/lib/python3.9/site-packages (from imbalanced-learn->imblearn) (1.1.0)
Requirement already satisfied: scikit-learn>=1.0.1 in /Users/shanneysuhendra/Desktop/cs13
5/miniconda3/lib/python3.9/site-packages (from imbalanced-learn->imblearn) (1.0.2)
Requirement already satisfied: numpy>=1.14.6 in /Users/shanneysuhendra/Desktop/cs135/minic
onda3/lib/python3.9/site-packages (from imbalanced-learn->imblearn) (1.21.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in /Users/shanneysuhendra/Desktop/cs13
5/miniconda3/lib/python3.9/site-packages (from imbalanced-learn->imblearn) (3.1.0)
Requirement already satisfied: scipy>=1.1.0 in /Users/shanneysuhendra/Desktop/cs135/minico
nda3/lib/python3.9/site-packages (from imbalanced-learn->imblearn) (1.8.0)
```

In [6]:
```
from collections import Counter
from sklearn.datasets import make_classification
from matplotlib import pyplot
from numpy import where
df = df.drop(['site_id'],axis=1)
df = df.drop(['order_source'], axis=1)
X = df.drop(['latest_plan'],axis=1)
X['total_sales'] = X['total_sales']/65061272983*100
y = df['latest_plan']
counter = Counter(y)
print(counter)

oversample = SMOTE()
X, y = oversample.fit_resample(X, y)
counter = Counter(y)
print(counter)
```

```
Counter({0: 47994, 1: 2109})
Counter({0: 47994, 1: 47994})
```

In [7]:
```
data = X.copy()

data['latest_plan'] = y
data
```

Out[7]:

	plan_duration	num_activeproducts	lastproduct_duration	num_orders	total_sales	latest_plan
0	149	6	149	0	0.000000	0
1	354	6	354	0	0.000000	0
2	231	6	231	0	0.000000	0
3	200	6	200	0	0.000000	0

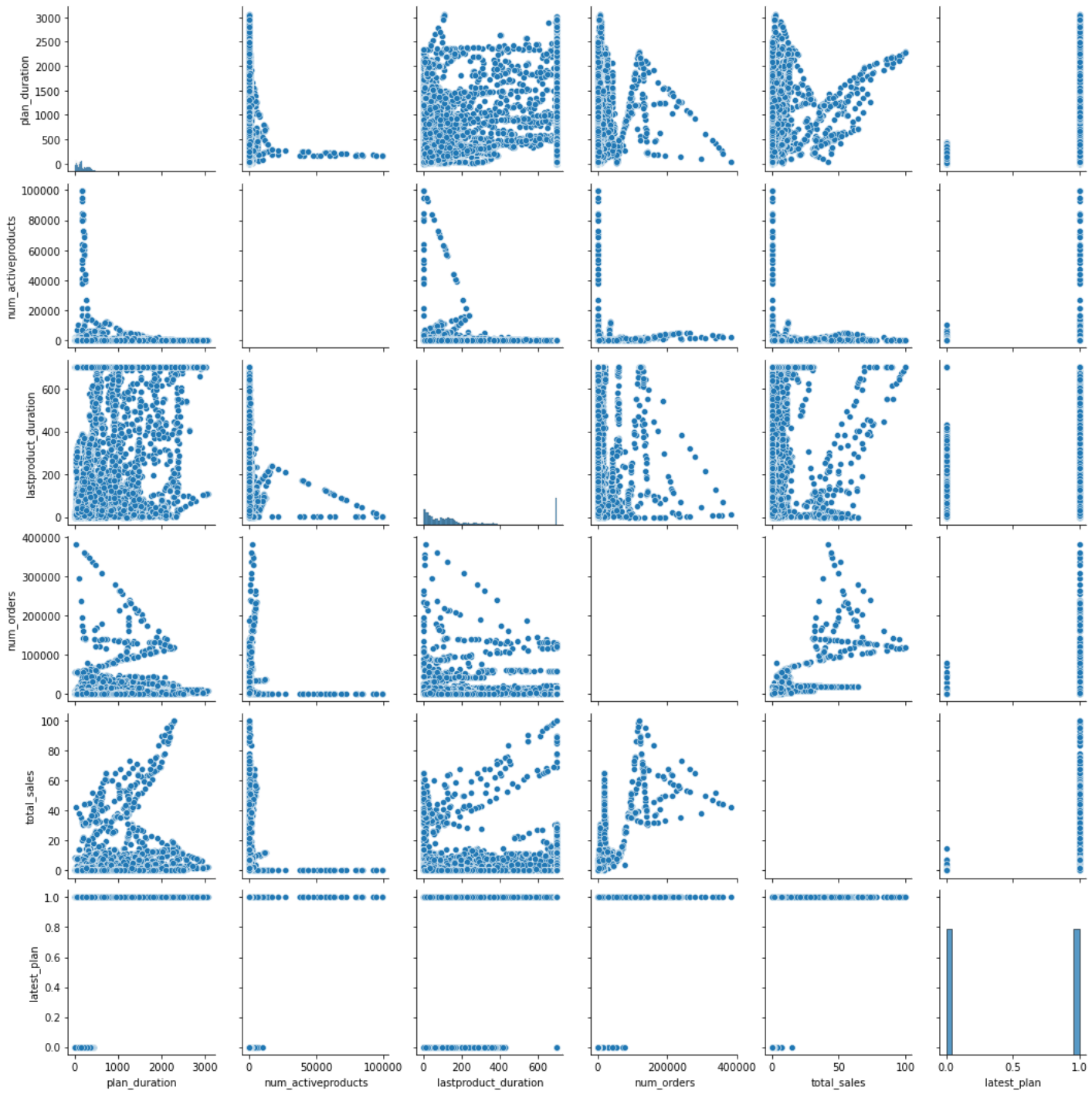
	plan_duration	num_activeproducts	lastproduct_duration	num_orders	total_sales	latest_plan
4	42	6	42	0	0.000000	0
...	...	...	...	...	...	...
95983	165	336	43	250	0.105273	1
95984	339	31	19	33	0.021784	1
95985	173	9	127	1	0.000712	1
95986	390	0	700	8	0.002524	1
95987	254	132	34	3134	1.382491	1

95988 rows × 6 columns

EDA

In [8]:

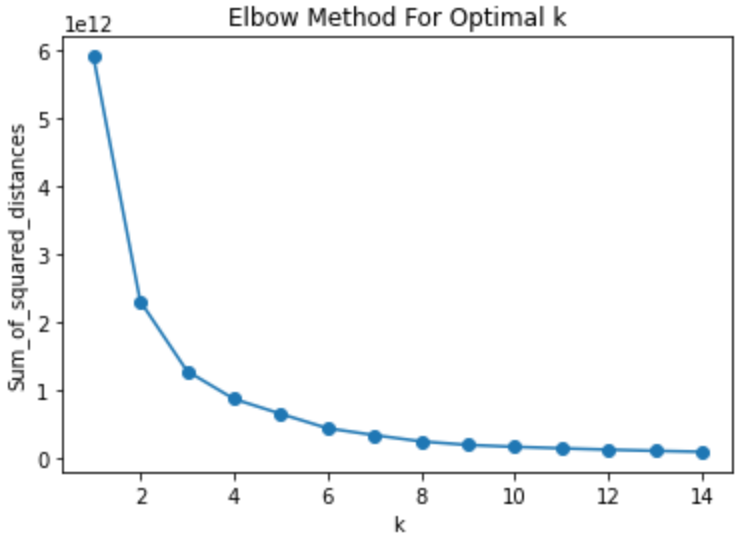
```
sns.pairplot(data)
plt.show()
```



Method 1: K-means

In [9]:

```
Sum_of_squared_distances = []
K = range(1,15)

for k in K:
    km = KMeans(n_clusters=k, init='k-means++')
    km = km.fit(data)
    Sum_of_squared_distances.append(km.inertia_)

# Plot Results
plt.plot(K, Sum_of_squared_distances, marker='o')
plt.xlabel('k')
plt.ylabel('Sum_of_squared_distances')
plt.title('Elbow Method For Optimal k')
plt.show()
```



In [10]:
```
from sklearn.metrics import silhouette_score

sill = []

for n_cluster in range(2, 11):
    kmeans = KMeans(n_clusters=n_cluster).fit(data)
    label = kmeans.labels_
    sil_coeff = silhouette_score(data, label, metric='euclidean')
    sill.append(sil_coeff)
    print("For n_clusters={}, The Silhouette Coefficient is {}".format(n_cluster, sil_coef
```

```
For n_clusters=2, The Silhouette Coefficient is 0.9859863787195863
For n_clusters=3, The Silhouette Coefficient is 0.9663757890275877
For n_clusters=4, The Silhouette Coefficient is 0.9626577341209976
For n_clusters=5, The Silhouette Coefficient is 0.8831352863566014
For n_clusters=6, The Silhouette Coefficient is 0.8993950259952919
For n_clusters=7, The Silhouette Coefficient is 0.8609174738606845
For n_clusters=8, The Silhouette Coefficient is 0.8609906150559536
For n_clusters=9, The Silhouette Coefficient is 0.861092604770891
For n_clusters=10, The Silhouette Coefficient is 0.8526230145251246
```

In [11]:
```
km = KMeans(n_clusters=6, init='random', max_iter=100, random_state=0)
km = km.fit(data)
```

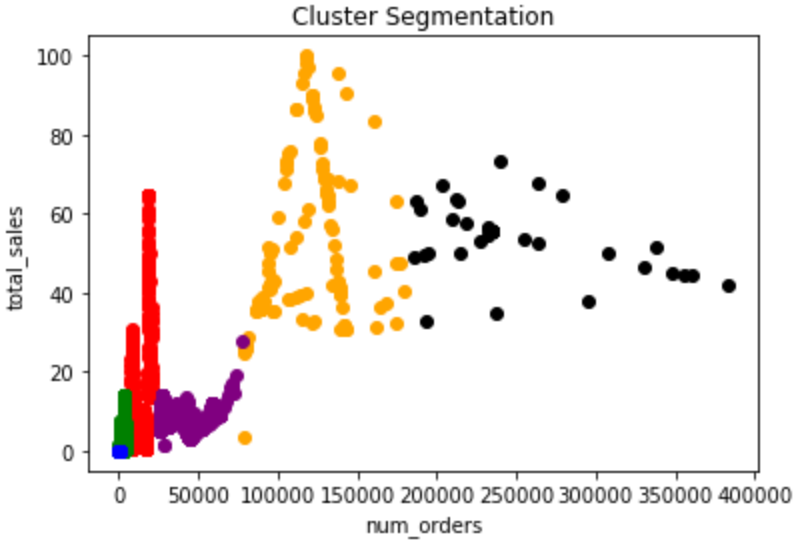
In [12]:
```
label = km.predict(data)
data1 = X.copy()
data1['latest_plan'] = y
data1['Label'] = label
```

In [13]:
```
# Showing cluster labels
data1['Label'] = [i+1 for i in label]
np.unique(data1['Label'])
```

Out[13]:
```
array([1, 2, 3, 4, 5, 6])
```

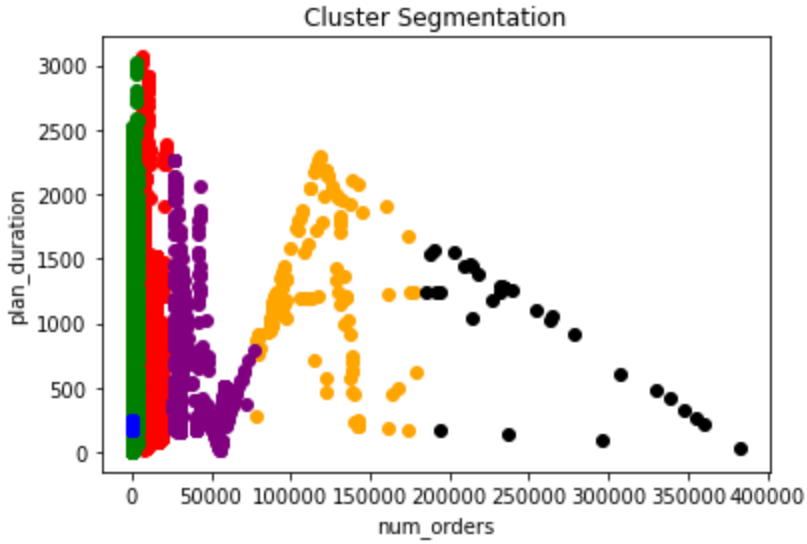
In [14]:
```
filtered_label1 = data1[data1['Label'] == 1]
filtered_label2 = data1[data1['Label'] == 2]
filtered_label3 = data1[data1['Label'] == 3]
filtered_label4 = data1[data1['Label'] == 4]
filtered_label5 = data1[data1['Label'] == 5]
filtered_label6 = data1[data1['Label'] == 6]
clust_cent = km.cluster_centers_
```

```
plt.scatter(filtered_label1['num_orders'] , filtered_label1['total_sales'] , color = 'red'
plt.scatter(filtered_label2['num_orders'] , filtered_label2['total_sales'] , color = 'gree
plt.scatter(filtered_label3['num_orders'] , filtered_label3['total_sales'] , color = 'blue
plt.scatter(filtered_label4['num_orders'] , filtered_label4['total_sales'] , color = 'oran
plt.scatter(filtered_label5['num_orders'] , filtered_label5['total_sales'] , color = 'purp
plt.scatter(filtered_label6['num_orders'] , filtered_label6['total_sales'] , color = 'blac

plt.title('Cluster Segmentation')
plt.xlabel('num_orders')
plt.ylabel('total_sales')
plt.show()
```



In [15]:

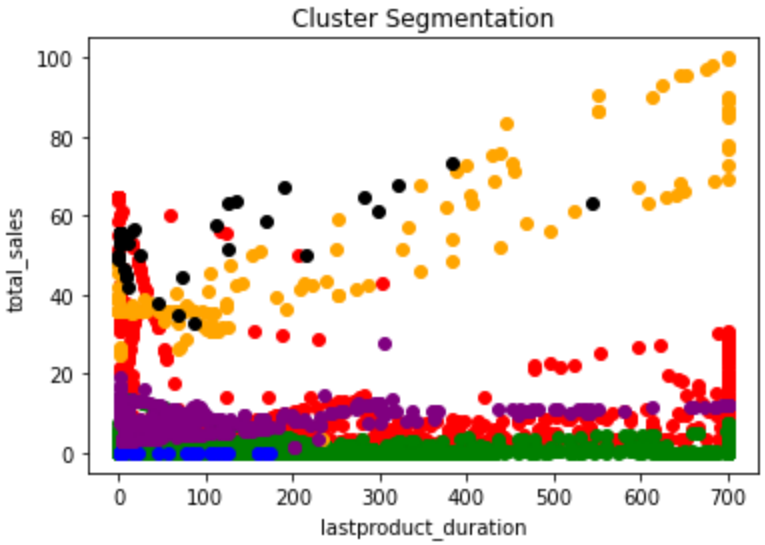
```
plt.scatter(filtered_label1['num_orders'] , filtered_label1['plan_duration'] , color = 're
plt.scatter(filtered_label2['num_orders'] , filtered_label2['plan_duration'] , color = 'gr
plt.scatter(filtered_label3['num_orders'] , filtered_label3['plan_duration'] , color = 'bl
plt.scatter(filtered_label4['num_orders'] , filtered_label4['plan_duration'] , color = 'or
plt.scatter(filtered_label5['num_orders'] , filtered_label5['plan_duration'] , color = 'pu
plt.scatter(filtered_label6['num_orders'] , filtered_label6['plan_duration'] , color = 'bl

plt.title('Cluster Segmentation')
plt.xlabel('num_orders')
plt.ylabel('plan_duration')
plt.show()
```



In [16]:

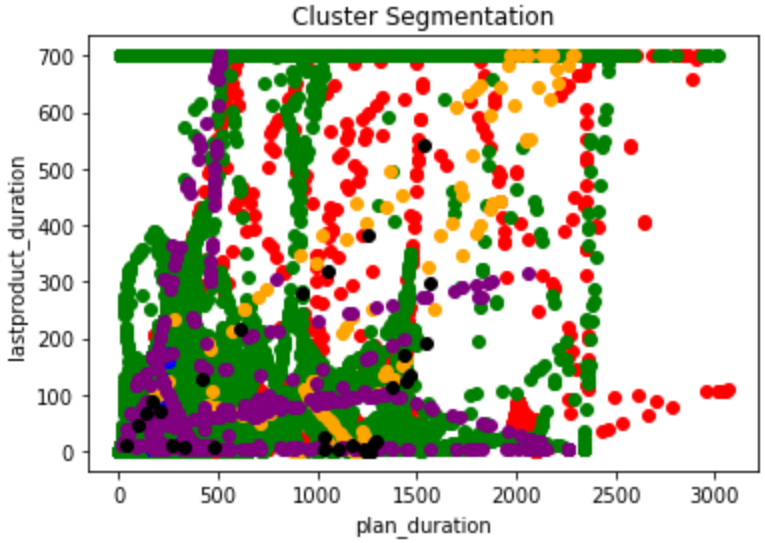
```
plt.scatter(filtered_label1['lastproduct_duration'] , filtered_label1['total_sales'] , col
```

```
plt.scatter(filtered_label2['lastproduct_duration'] , filtered_label2['total_sales'] , col
plt.scatter(filtered_label3['lastproduct_duration'] , filtered_label3['total_sales'] , col
plt.scatter(filtered_label4['lastproduct_duration'] , filtered_label4['total_sales'] , col
plt.scatter(filtered_label5['lastproduct_duration'] , filtered_label5['total_sales'] , col
plt.scatter(filtered_label6['lastproduct_duration'] , filtered_label6['total_sales'] , col

plt.title('Cluster Segmentation')
plt.xlabel('lastproduct_duration')
plt.ylabel('total_sales')
plt.show()
```



In [17]:

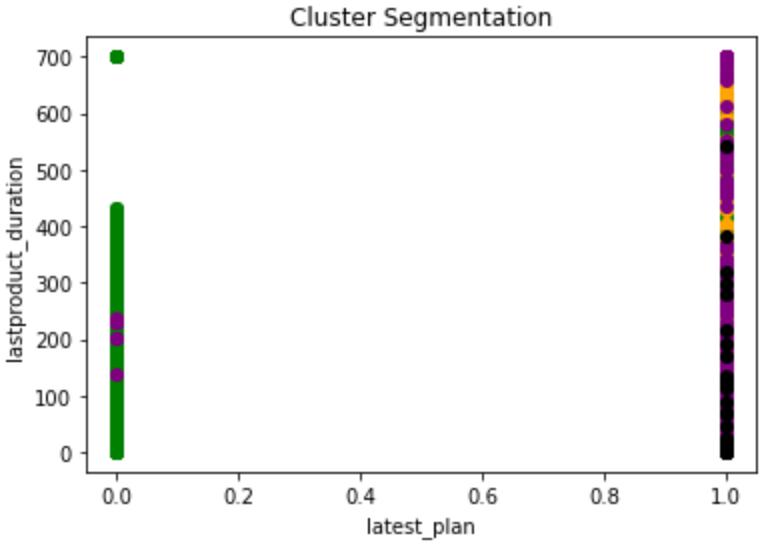
```
plt.scatter(filtered_label1['plan_duration'] , filtered_label1['lastproduct_duration'] , c
plt.scatter(filtered_label2['plan_duration'] , filtered_label2['lastproduct_duration'] , c
plt.scatter(filtered_label3['plan_duration'] , filtered_label3['lastproduct_duration'] , c
plt.scatter(filtered_label4['plan_duration'] , filtered_label4['lastproduct_duration'] , c
plt.scatter(filtered_label5['plan_duration'] , filtered_label5['lastproduct_duration'] , c
plt.scatter(filtered_label6['plan_duration'] , filtered_label6['lastproduct_duration'] , c

plt.title('Cluster Segmentation')
plt.xlabel('plan_duration')
plt.ylabel('lastproduct_duration')
plt.show()
```



In [18]:

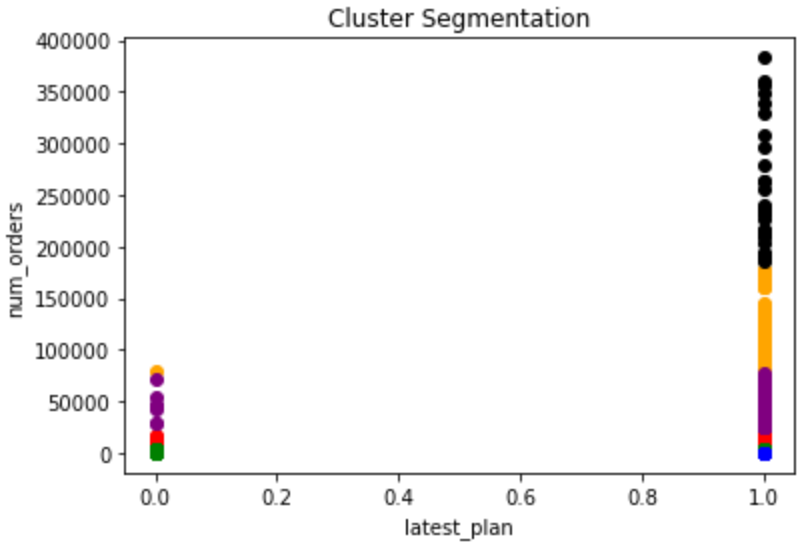
```
plt.scatter(filtered_label1['latest_plan'] , filtered_label1['lastproduct_duration'] , col
plt.scatter(filtered_label2['latest_plan'] , filtered_label2['lastproduct_duration'] , col
plt.scatter(filtered_label3['latest_plan'] , filtered_label3['lastproduct_duration'] , col
```

```
plt.scatter(filtered_label4['latest_plan'] , filtered_label4['lastproduct_duration'] , col
plt.scatter(filtered_label5['latest_plan'] , filtered_label5['lastproduct_duration'] , col
plt.scatter(filtered_label6['latest_plan'] , filtered_label6['lastproduct_duration'] , col

plt.title('Cluster Segmentation')
plt.xlabel('latest_plan')
plt.ylabel('lastproduct_duration')
plt.show()
```



In [45]:

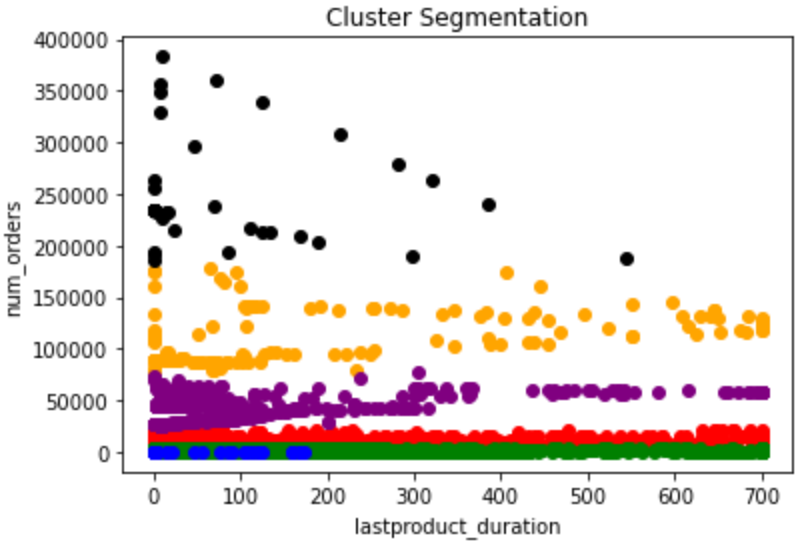
```
plt.scatter(filtered_label1['latest_plan'] , filtered_label1['num_orders'] , color = 'red'
plt.scatter(filtered_label2['latest_plan'] , filtered_label2['num_orders'] , color = 'gree
plt.scatter(filtered_label3['latest_plan'] , filtered_label3['num_orders'] , color = 'blue
plt.scatter(filtered_label4['latest_plan'] , filtered_label4['num_orders'] , color = 'oran
plt.scatter(filtered_label5['latest_plan'] , filtered_label5['num_orders'] , color = 'purp
plt.scatter(filtered_label6['latest_plan'] , filtered_label6['num_orders'] , color = 'blac

plt.title('Cluster Segmentation')
plt.xlabel('latest_plan')
plt.ylabel('num_orders')
plt.show()
```



In [20]:

```
plt.scatter(filtered_label1['lastproduct_duration'] , filtered_label1['num_orders'] , colo
plt.scatter(filtered_label2['lastproduct_duration'] , filtered_label2['num_orders'] , colo
plt.scatter(filtered_label3['lastproduct_duration'] , filtered_label3['num_orders'] , colo
plt.scatter(filtered_label4['lastproduct_duration'] , filtered_label4['num_orders'] , colo
plt.scatter(filtered_label5['lastproduct_duration'] , filtered_label5['num_orders'] , colo
```

```
plt.scatter(filtered_label6['lastproduct_duration'] , filtered_label6['num_orders'] , colo

plt.title('Cluster Segmentation')
plt.xlabel('lastproduct_duration')
plt.ylabel('num_orders')
plt.show()
```



In [21]:

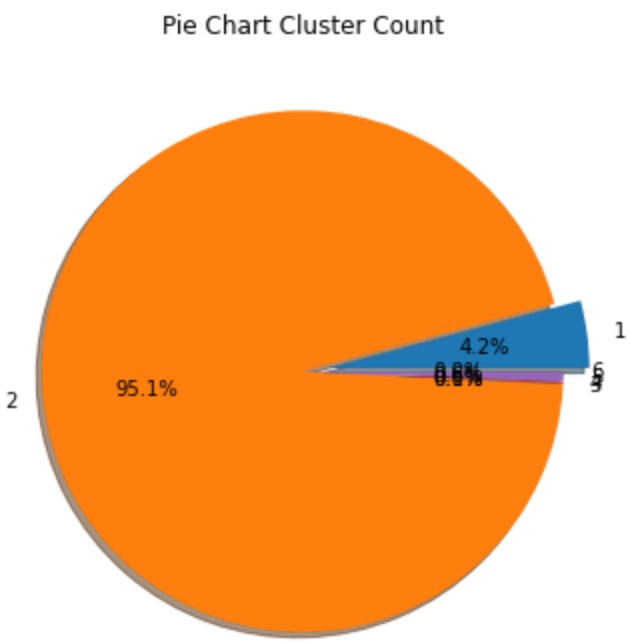
```
data2 = data1.copy()
data2.rename(columns = {"Label":"Cluster"}, inplace = True)
data2.groupby(['Cluster']).agg(len)['latest_plan']
```

Out[21]:

```
Cluster
1     3994
2    91238
3       41
4      136
5      540
6       39
Name: latest_plan, dtype: int64
```

In [23]:

```
df_pie = data1.groupby(['Label']).agg(len).reset_index()


plt.figure(figsize = (6,6))
plt.pie(
    x = df_pie['latest_plan'].tolist(),
    labels = df_pie['Label'],
    autopct='%1.1f%%',
    shadow=True,
    explode = (0.1,0,0,0,0,0)
)
plt.title('Pie Chart Cluster Count')
plt.show()
```



In [24]:
```
data1.groupby(['Label']).agg('mean').reset_index()
```

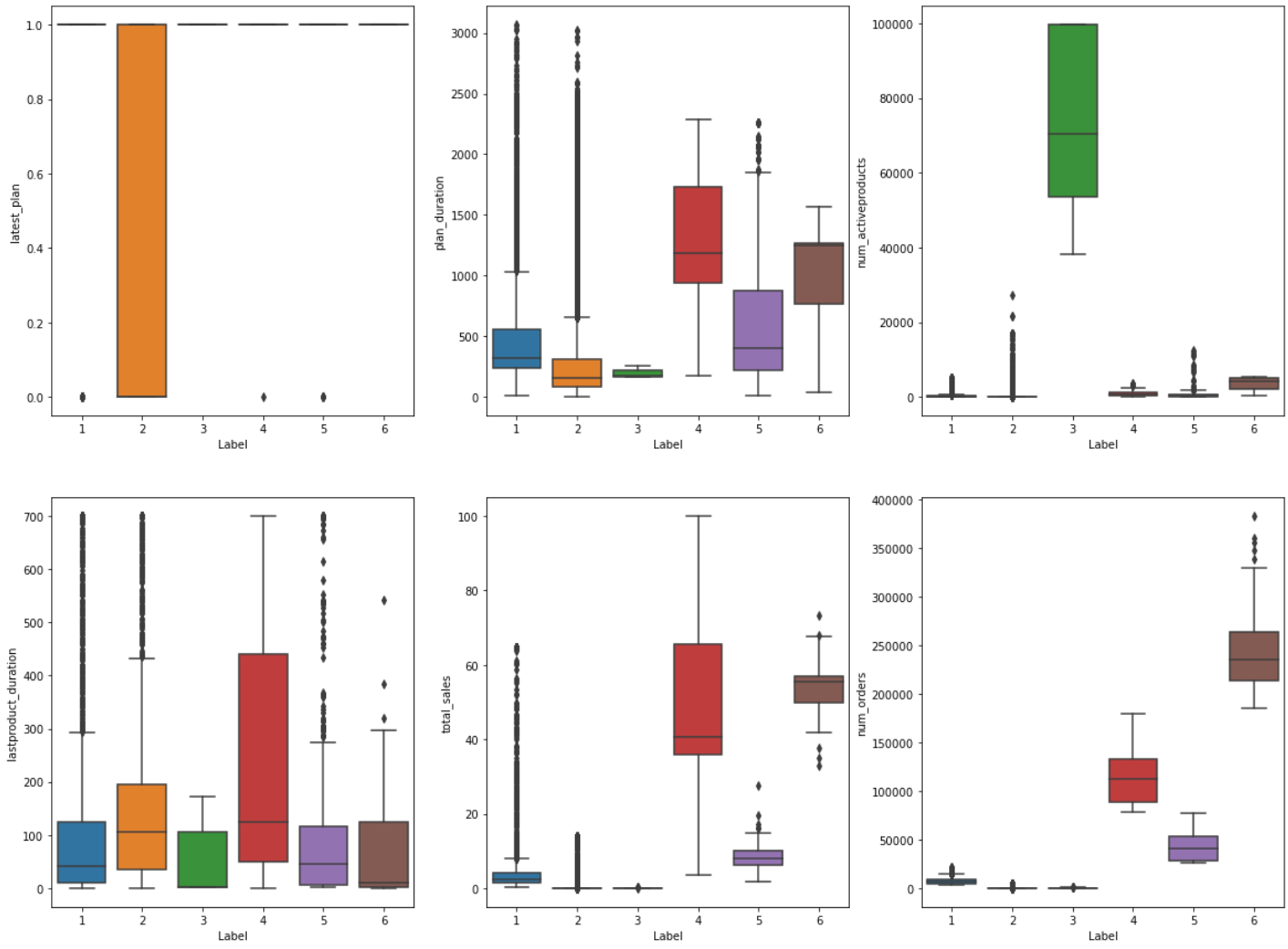
Out[24]:

	Label	plan_duration	num_activeproducts	lastproduct_duration	num_orders	total_sales	latest_plan
0	1	571.331247	353.001252	132.355784	7740.091637	4.507714	0.994742
1	2	249.230452	63.718571	172.549015	211.328240	0.130765	0.474287
2	3	191.536585	72257.365854	48.024390	240.756098	0.047546	1.000000
3	4	1249.205882	813.786765	252.242647	114184.169118	50.319036	0.992647
4	5	626.133333	795.007407	106.851852	41187.714815	8.336485	0.987037
5	6	1016.564103	3647.641026	84.333333	248909.923077	53.740147	1.000000

In [41]:
```
fig, axes  = plt.subplots(2,3, figsize = (20,15))
fig.suptitle('Box Plot for Each Cluster')

sns.boxplot(
    x = 'Label',
    y = 'latest_plan',
    data = data1,
    ax = axes[0][0]
)
sns.boxplot(
    x = 'Label',
    y = 'plan_duration',
    data = data1,
    ax = axes[0][1]
)
sns.boxplot(
    x = 'Label',
    y = 'num_activeproducts',
    data = data1,
    ax = axes[0][2]
)
sns.boxplot(
    x = 'Label',
    y = 'lastproduct_duration',
    data = data1,
```

```
    ax = axes[1][0]
)

sns.boxplot(
    x = 'Label',
    y = 'total_sales',
    data = data1,
    ax = axes[1][1]
)
sns.boxplot(
    x = 'Label',
    y = 'num_orders',
    data = data1,
    ax = axes[1][2]
)

plt.show()
```

Box Plot for Each Cluster



From the table and the boxplot above, the six cluster and their characteristics are as follows:

- Cluster 1:
  - Mix starter+paid merchants (paid > starter)
  - Lowest number of active products
  - largest range
- Cluster 2:

    - Mix starter+paid merchants (paid = starter)
    - Second shortest plan duration
    - Second lowest number of active products
    - Second least recent new products created
    - Second lowest total sales and number of orders
- Cluster 3:
    - All paid merchants
    - Shortest plan duration
    - Highest number of active products
    - Most recent new products created
    - Lowest total sales and number of orders
- Cluster 4:
    - Mix starter+paid merchants (paid > starter)
    - Longest plan duration
    - Least recent new products created
    - Second highest total sales and number of orders
- Cluster 5:
    - Mix starter+paid merchants (paid > starter)
    - Middle
- Cluster 6:
    - All paid merchants
    - Second longest plan duration
    - Second highest number of active products
    - Second most recent new products created
    - Highest total sales and number of orders

- Cluster 1:
- Cluster 2: not so active merchants
- Cluster 3: newly active paid merchants (sales still low)
- Cluster 4: old active merchants (sell same products)
- Cluster 5:
- Cluster 6: old active paid merchants (update products)

In [ ]:

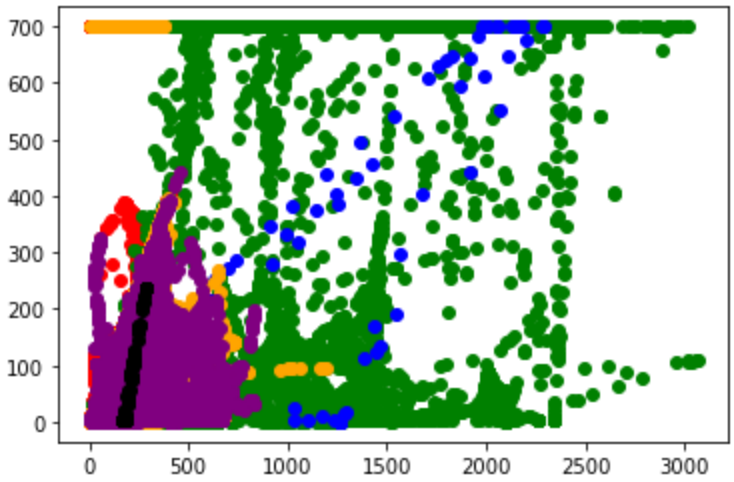
Method 2: Gaussian

In [46]:
```
from sklearn.mixture import GaussianMixture
gmm = GaussianMixture(n_components = 6)
gmm = gmm.fit(data)
```

In [47]:
```
labelsg = gmm.predict(data)
datag = X.copy()
datag['latest_plan'] = y
datag['labelsg']= labelsg

datag['labelsg'] = [i+1 for i in labelsg]
```

```
np.unique(datag['labelsg'])

d0 = datag[datag['labelsg']== 1]
d1 = datag[datag['labelsg']== 2]
d2 = datag[datag['labelsg']== 3]
d3 = datag[datag['labelsg']== 4]
d4 = datag[datag['labelsg']== 5]
d5 = datag[datag['labelsg']== 6]

clust_cent = km.cluster_centers_
# plot three clusters in same plot
plt.scatter(d0['plan_duration'], d0['lastproduct_duration'], c ='r')
plt.scatter(d1['plan_duration'], d1['lastproduct_duration'], c ='green')
plt.scatter(d2['plan_duration'], d2['lastproduct_duration'], c ='blue')
plt.scatter(d3['plan_duration'], d3['lastproduct_duration'], c ='orange')
plt.scatter(d4['plan_duration'], d4['lastproduct_duration'], c ='purple')
plt.scatter(d5['plan_duration'], d5['lastproduct_duration'], c ='black')
```

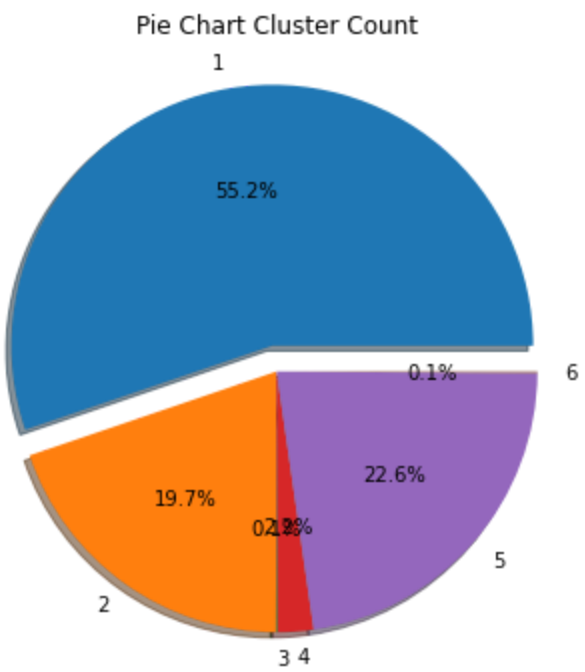
Out[47]:
```
<matplotlib.collections.PathCollection at 0x7fbeb6e61730>
```



In [48]:
```
datag2 = datag.copy()
datag2.rename(columns = {"labelsg":"Cluster"}, inplace = True)
datag2.groupby(['Cluster']).agg(len)['latest_plan']
```

Out[48]:
```
Cluster
1    53003
2    18949
3       88
4     2135
5    21726
6       87
Name: latest_plan, dtype: int64
```

In [49]:
```
dfg_pie = datag.groupby(['labelsg']).agg(len).reset_index()


plt.figure(figsize = (6,6))
plt.pie(
    x = dfg_pie['latest_plan'].tolist(),
    labels = dfg_pie['labelsg'],
    autopct='%1.1f%%',
    shadow=True,
    explode = (0.1,0,0,0,0,0)
)
plt.title('Pie Chart Cluster Count')
plt.show()
```



In [50]:
```
datag.groupby(['labelsg']).agg('mean').reset_index()
```

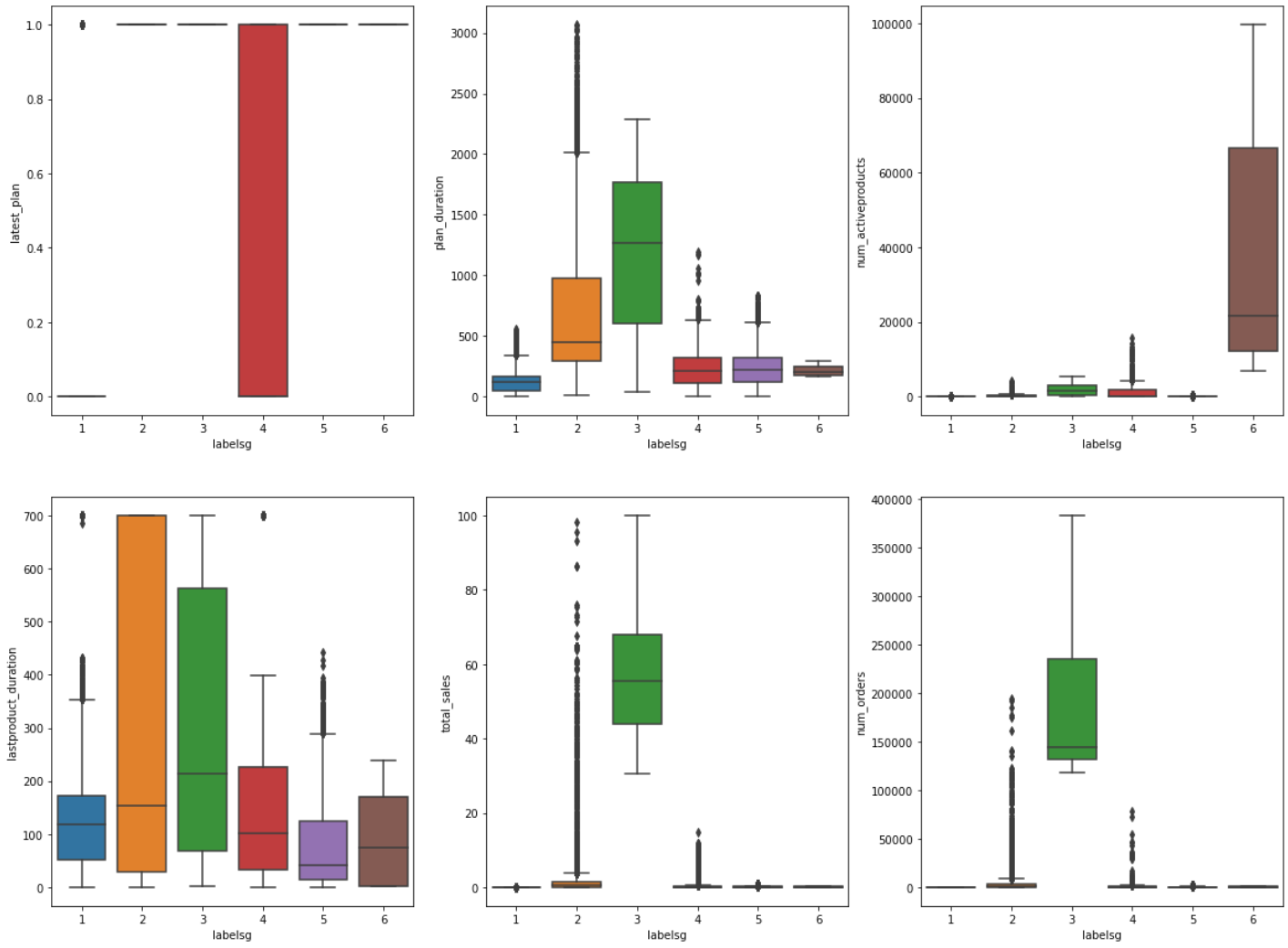
Out[50]:

	labelsg	plan_duration	num_activeproducts	lastproduct_duration	num_orders	total_sales	latest_plan
0	1	133.434957	5.744165	151.855103	0.000000	0.000005	0.123748
1	2	682.989287	169.304977	334.009605	3819.083593	1.824429	1.000000
2	3	1186.681818	1954.511364	294.181818	185012.488636	58.307391	1.000000
3	4	234.281499	1006.996253	147.807026	1351.066511	0.624063	0.274005
4	5	227.325877	67.585382	75.559054	281.479748	0.107543	1.000000
5	6	215.747126	40980.678161	91.298851	390.724138	0.071177	1.000000

In [54]:
```
fig, axes  = plt.subplots(2,3, figsize = (20,15))
fig.suptitle('Box Plot for Each Cluster')

sns.boxplot(
    x = 'labelsg',
    y = 'latest_plan',
    data = datag,
    ax = axes[0][0]
)
sns.boxplot(
    x = 'labelsg',
    y = 'plan_duration',
    data = datag,
    ax = axes[0][1]
)
sns.boxplot(
    x = 'labelsg',
    y = 'num_activeproducts',
    data = datag,
    ax = axes[0][2]
)
sns.boxplot(
    x = 'labelsg',
    y = 'lastproduct_duration',
    data = datag,
```

```
    ax = axes[1][0]
)

sns.boxplot(
    x = 'labelsg',
    y = 'total_sales',
    data = datag,
    ax = axes[1][1]
)
sns.boxplot(
    x = 'labelsg',
    y = 'num_orders',
    data = datag,
    ax = axes[1][2]
)

plt.show()
```

Box Plot for Each Cluster



From the table and the boxplot above, the six cluster and their characteristics are as follows:

- Cluster 1:
  - Mix starter+paid merchants (paid < starter)
  - Shortest plan duration
  - Lowest number of active products
  - Lowest total sales and number of orders

- Cluster 2:
  - All paid merchants
  - Second longest plan duration
  - Largest range
- Cluster 3:
  - All paid merchants
  - Longest plan duration
  - Second highest number of active products
  - Least recent new products created
  - Highest total sales and number of orders
- Cluster 4:
  - Mix starter+paid merchants (paid = starter)
  - Middle
- Cluster 5:
  - All paid merchants
  - Second lowest number of active products
  - Most recent new products created
  - Second lowest total sales and number of orders
- Cluster 6:
  - All paid merchants
  - Highest number of active products
  - Second most recent new products created
  - Thirst lowest total sales and number of orders

- Cluster 1: new non-active starter merchants
- Cluster 2:
- Cluster 3: old active paid merchants (sell same products)
- Cluster 4:
- Cluster 5: fairly new active paid merchants (sales still low)
- Cluster 6: active paid merchants (update products, but sales low)

In [ ]:

In [ ]: