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

In [15]:
```
df_cust = merged[merged['label_y'] == 1]
```

In [16]:
```
df_cust.shape
```

Out[16]:
```
(14450, 62)
```

In [17]:
```
df_cust
```

Out[17]:

	log_id	label_x	user_id	age	gender	residence	city	city_rank	series_dev	series_group
18	583728.0	0.0	100108	2.0	3.0	21.0	200.0	4.0	30.0	3.0
21	364370.0	0.0	100127	7.0	2.0	17.0	343.0	5.0	16.0	5.0
23	588242.0	0.0	100149	8.0	2.0	16.0	425.0	2.0	34.0	7.0
27	679513.0	0.0	100158	6.0	4.0	33.0	319.0	3.0	27.0	2.0
29	1084910.0	0.0	100166	5.0	2.0	30.0	113.0	5.0	16.0	5.0
...	...	...	...	...	...	...	...	...	...	...
179951	NaN	NaN	131907	NaN	NaN	NaN	NaN	NaN	NaN	NaN
179959	NaN	NaN	123724	NaN	NaN	NaN	NaN	NaN	NaN	NaN
179996	NaN	NaN	215157	NaN	NaN	NaN	NaN	NaN	NaN	NaN
179997	NaN	NaN	107610	NaN	NaN	NaN	NaN	NaN	NaN	NaN
180013	NaN	NaN	246671	NaN	NaN	NaN	NaN	NaN	NaN	NaN

14450 rows × 62 columns

In [23]:
```
import plotly.express as px
fig = px.pie(df_cust, values='age', names='age', title = "Potential Customer Age Distri
fig.show()
```

In [24]:

```
fig = px.box(df_cust, x="age", title = "Potential Customer Age Distribution (Boxplot)")
fig.show()
```

In [25]:

```
import plotly.express as px
fig = px.pie(df_cust, values='residence', names='residence', title = "Potential Custome
fig.show()
```

In [26]:

```
import plotly.express as px
fig = px.pie(df_cust, values='city', names='city')
fig.show()
```

In [27]:
```
merged['e_section'].value_counts()
```

Out[27]:
```
1    109898
0     70225
Name: e_section, dtype: int64
```

In [28]:
```
df_cust['e_section'].value_counts()
```

Out[28]:
```
0    8685
1    5765
Name: e_section, dtype: int64
```

In [29]:
```
value_counts = df_cust['e_section'].value_counts()
count_0 = value_counts.get(0, 0)
count_1 = value_counts.get(1, 0)

labels = ['0', '1']
values = [count_0, count_1]

fig = px.pie(values=values, names=labels, title='Distribution of content preferences am
fig.show()
```

In [30]:
```
df_cust.columns
```

Out[30]:
```
Index(['log_id', 'label_x', 'user_id', 'age', 'gender', 'residence', 'city',
       'city_rank', 'series_dev', 'series_group', 'emui_dev', 'device_name',
       'device_size', 'net_type', 'task_id', 'adv_id', 'creat_type_cd',
       'adv_prim_id', 'inter_type_cd', 'slot_id', 'site_id', 'spread_app_id',
       'hispace_app_tags', 'app_second_class', 'app_score',
       'ad_click_list_v001', 'ad_click_list_v002', 'ad_click_list_v003',
       'ad_close_list_v001', 'ad_close_list_v002', 'ad_close_list_v003',
       'pt_d', 'u_newsCatInterestsST_x', 'u_refreshTimes_x',
       'u_feedLifeCycle_x', 'u_phonePrice', 'u_browserLifeCycle',
       'u_browserMode', 'u_feedLifeCycle_y', 'u_refreshTimes_y',
       'u_newsCatInterests', 'u_newsCatDislike', 'u_newsCatInterestsST_y',
       'u_click_ca2_news', 'i_docId', 'i_s_sourceId', 'i_regionEntity',
       'i_cat', 'i_entities', 'i_dislikeTimes', 'i_upTimes', 'i_dtype', 'e_ch',
       'e_m', 'e_po', 'e_pl', 'e_rn', 'e_section', 'e_et', 'label_y',
       'cillabel', 'pro'],
      dtype='object')
```

In [31]:
```
import plotly.express as px
fig = px.pie(df_cust, values='series_dev', names='series_dev', title = "Potential Custo
fig.show()
```

In [32]:

```
import plotly.express as px
fig = px.pie(df_cust, values='series_group', names='series_group', title = "Potential C
fig.show()
```

In [33]:

```
import plotly.express as px
fig = px.pie(df_cust, values='emui_dev', names='emui_dev', title = "Potential Customer
fig.show()
```

In [34]:

```
import plotly.express as px
fig = px.pie(df_cust, values='device_name', names='device_name', title = "Potential Cus
fig.show()
```

In [35]:
```
import plotly.express as px
fig = px.pie(df_cust, values='device_size', names='device_size', title = "Potential Cus
fig.show()
```

In [36]:
```
df_cust.columns
```

Out[36]:
```
Index(['log_id', 'label_x', 'user_id', 'age', 'gender', 'residence', 'city',
       'city_rank', 'series_dev', 'series_group', 'emui_dev', 'device_name',
       'device_size', 'net_type', 'task_id', 'adv_id', 'creat_type_cd',
       'adv_prim_id', 'inter_type_cd', 'slot_id', 'site_id', 'spread_app_id',
       'hispace_app_tags', 'app_second_class', 'app_score',
       'ad_click_list_v001', 'ad_click_list_v002', 'ad_click_list_v003',
       'ad_close_list_v001', 'ad_close_list_v002', 'ad_close_list_v003',
       'pt_d', 'u_newsCatInterestsST_x', 'u_refreshTimes_x',
       'u_feedLifeCycle_x', 'u_phonePrice', 'u_browserLifeCycle',
       'u_browserMode', 'u_feedLifeCycle_y', 'u_refreshTimes_y',
       'u_newsCatInterests', 'u_newsCatDislike', 'u_newsCatInterestsST_y',
       'u_click_ca2_news', 'i_docId', 'i_s_sourceId', 'i_regionEntity',
       'i_cat', 'i_entities', 'i_dislikeTimes', 'i_upTimes', 'i_dtype', 'e_ch',
       'e_m', 'e_po', 'e_pl', 'e_rn', 'e_section', 'e_et', 'label_y',
```

```
       'cillabel', 'pro'],
      dtype='object')
```

In [37]:
```
df_cust['pt_d'] = pd.to_datetime(df_cust['pt_d'], format='%Y%m%d%H%M')
df_cust['e_et'] = pd.to_datetime(df_cust['e_et'], format='%Y%m%d%H%M')
```

```
C:\Users\anime\anaconda3\envs\tensor\lib\site-packages\ipykernel_launcher.py:1: SettingW
ithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy

C:\Users\anime\anaconda3\envs\tensor\lib\site-packages\ipykernel_launcher.py:2: SettingW
ithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
```

In [38]:
```
df_cust['ads_hour'] = df_cust['pt_d'].dt.hour
df_cust['feeds_hour'] = df_cust['e_et'].dt.hour
```

```
C:\Users\anime\anaconda3\envs\tensor\lib\site-packages\ipykernel_launcher.py:1: SettingW
ithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy

C:\Users\anime\anaconda3\envs\tensor\lib\site-packages\ipykernel_launcher.py:2: SettingW
ithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
```

In [39]:
```
df_cust['ads_day'] = df_cust['pt_d'].dt.dayofweek
df_cust['feeds_day'] = df_cust['e_et'].dt.dayofweek
```

```
C:\Users\anime\anaconda3\envs\tensor\lib\site-packages\ipykernel_launcher.py:1: SettingW
ithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy

C:\Users\anime\anaconda3\envs\tensor\lib\site-packages\ipykernel_launcher.py:2: SettingW
ithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
```

In [40]:
```
df_cust['ads_dayname'] = df_cust['pt_d'].dt.day_name()
df_cust['feeds_dayname'] = df_cust['e_et'].dt.day_name()
```

```
C:\Users\anime\anaconda3\envs\tensor\lib\site-packages\ipykernel_launcher.py:1: SettingW
ithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy

C:\Users\anime\anaconda3\envs\tensor\lib\site-packages\ipykernel_launcher.py:2: SettingW
ithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
```

In [41]:
```
df_cust.columns
```

Out[41]:
```
Index(['log_id', 'label_x', 'user_id', 'age', 'gender', 'residence', 'city',
       'city_rank', 'series_dev', 'series_group', 'emui_dev', 'device_name',
       'device_size', 'net_type', 'task_id', 'adv_id', 'creat_type_cd',
       'adv_prim_id', 'inter_type_cd', 'slot_id', 'site_id', 'spread_app_id',
       'hispace_app_tags', 'app_second_class', 'app_score',
       'ad_click_list_v001', 'ad_click_list_v002', 'ad_click_list_v003',
       'ad_close_list_v001', 'ad_close_list_v002', 'ad_close_list_v003',
       'pt_d', 'u_newsCatInterestsST_x', 'u_refreshTimes_x',
       'u_feedLifeCycle_x', 'u_phonePrice', 'u_browserLifeCycle',
       'u_browserMode', 'u_feedLifeCycle_y', 'u_refreshTimes_y',
       'u_newsCatInterests', 'u_newsCatDislike', 'u_newsCatInterestsST_y',
       'u_click_ca2_news', 'i_docId', 'i_s_sourceId', 'i_regionEntity',
       'i_cat', 'i_entities', 'i_dislikeTimes', 'i_upTimes', 'i_dtype', 'e_ch',
       'e_m', 'e_po', 'e_pl', 'e_rn', 'e_section', 'e_et', 'label_y',
       'cillabel', 'pro', 'ads_hour', 'feeds_hour', 'ads_day', 'feeds_day',
       'ads_dayname', 'feeds_dayname'],
      dtype='object')
```

In [42]:
```
import plotly.express as px
fig = px.pie(df_cust, values='ads_hour', names='ads_hour', title = "Potential Customer
```

```
fig.show()
```

In [43]:

```
import plotly.express as px
fig = px.pie(df_cust, values= df_cust['feeds_hour'].value_counts().values, names=df_cus
fig.show()
```

In [44]:

```
import plotly.express as px
fig = px.pie(df_cust, values= df_cust['ads_day'].value_counts().values, names=df_cust['
fig.show()
```

In [45]:

```
value_counts = df_cust['e_section'].value_counts()
count_0 = value_counts.get(0, 0)
count_1 = value_counts.get(1, 0)

labels = ['0', '1']
values = [count_0, count_1]
```

```
fig = px.pie(values=values, names=labels, title='Distribution of Content Preferences am
fig.show()
```

In [46]:
```
df_cust['feeds_day']
```

Out[46]:
```
18        4
21        2
23        2
27        4
29        4
         ..
179951    6
179959    6
179996    6
179997    6
180013    6
Name: feeds_day, Length: 14450, dtype: int64
```

In [47]:
```
import plotly.express as px
fig = px.pie(df_cust, values= df_cust['feeds_day'].value_counts().values, names=df_cust
fig.show()
```

In [48]:
```
df_noncust = merged[merged['label_y'] == -1.0]
```

In [49]:
```
merged.to_csv("merged_dataframe.csv")
```

In [50]:
```
import plotly.express as px
fig = px.pie(df_noncust, values='age', names='age', title = "Non-Potential Customer Age
fig.show()
```

In [51]:

```
import plotly.express as px
fig = px.pie(df_noncust, values='residence', names='residence', title = "Non-Potential
fig.show()
```

In [52]:

```
value_counts = df_noncust['e_section'].value_counts()
count_0 = value_counts.get(0, 0)
count_1 = value_counts.get(1, 0)

labels = ['0', '1']
values = [count_0, count_1]
```

```
fig = px.pie(values=values, names=labels, title='Distribution of content preferences am
fig.show()
```

In [53]:

```
import plotly.express as px
fig = px.pie(df_noncust, values='series_group', names='series_group', title = "Non-Pote
fig.show()
```

In [54]:

```
import plotly.express as px
fig = px.pie(df_noncust, values='series_dev', names='series_dev', title = "Non-Potentia
fig.show()
```

In [55]:

```
import plotly.express as px
fig = px.pie(df_noncust, values='emui_dev', names='emui_dev', title = "Non-Potential Cu
fig.show()
```

In [56]:
```
df_noncust['pt_d'] = pd.to_datetime(df_noncust['pt_d'], format='%Y%m%d%H%M')
df_noncust['e_et'] = pd.to_datetime(df_noncust['e_et'], format='%Y%m%d%H%M')
```

```
C:\Users\anime\anaconda3\envs\tensor\lib\site-packages\ipykernel_launcher.py:1: SettingW
ithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy

C:\Users\anime\anaconda3\envs\tensor\lib\site-packages\ipykernel_launcher.py:2: SettingW
ithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
```

In [57]:
```
df_noncust['ads_hour'] = df_noncust['pt_d'].dt.hour
df_noncust['feeds_hour'] = df_noncust['e_et'].dt.hour
```

```
C:\Users\anime\anaconda3\envs\tensor\lib\site-packages\ipykernel_launcher.py:1: SettingW
ithCopyWarning:
```

```
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy

C:\Users\anime\anaconda3\envs\tensor\lib\site-packages\ipykernel_launcher.py:2: SettingW
ithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
```

In [58]:

```
df_noncust['ads_day'] = df_noncust['pt_d'].dt.dayofweek
df_noncust['feeds_day'] = df_noncust['e_et'].dt.dayofweek
```

```
C:\Users\anime\anaconda3\envs\tensor\lib\site-packages\ipykernel_launcher.py:1: SettingW
ithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy

C:\Users\anime\anaconda3\envs\tensor\lib\site-packages\ipykernel_launcher.py:2: SettingW
ithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
```

In [59]:

```
df_noncust['ads_dayname'] = df_noncust['pt_d'].dt.day_name()
df_noncust['feeds_dayname'] = df_noncust['e_et'].dt.day_name()
```

```
C:\Users\anime\anaconda3\envs\tensor\lib\site-packages\ipykernel_launcher.py:1: SettingW
ithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy

C:\Users\anime\anaconda3\envs\tensor\lib\site-packages\ipykernel_launcher.py:2: SettingW
ithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_
guide/indexing.html#returning-a-view-versus-a-copy
```

In [60]:

```
import plotly.express as px
fig = px.pie(df_noncust, values='ads_hour', names='ads_hour', title = "Non-Potential Cu
fig.show()
```

In [61]:

```
import plotly.express as px
fig = px.pie(df_noncust, values= df_noncust['feeds_hour'].value_counts().values, names=
fig.show()
```

In [62]:

```
import plotly.express as px
fig = px.pie(df_noncust, values= df_noncust['ads_day'].value_counts().values, names=df_
fig.show()
```

In [63]:
```
import plotly.express as px
fig = px.pie(df_noncust, values= df_noncust['feeds_day'].value_counts().values, names=d
fig.show()
```

In [64]:
```
df_cust.to_csv("customer_df.csv")
```

In [65]:
```
df_cust.columns
```

Out[65]:
```
Index(['log_id', 'label_x', 'user_id', 'age', 'gender', 'residence', 'city',
       'city_rank', 'series_dev', 'series_group', 'emui_dev', 'device_name',
       'device_size', 'net_type', 'task_id', 'adv_id', 'creat_type_cd',
       'adv_prim_id', 'inter_type_cd', 'slot_id', 'site_id', 'spread_app_id',
       'hispace_app_tags', 'app_second_class', 'app_score',
       'ad_click_list_v001', 'ad_click_list_v002', 'ad_click_list_v003',
       'ad_close_list_v001', 'ad_close_list_v002', 'ad_close_list_v003',
       'pt_d', 'u_newsCatInterestsST_x', 'u_refreshTimes_x',
       'u_feedLifeCycle_x', 'u_phonePrice', 'u_browserLifeCycle',
       'u_browserMode', 'u_feedLifeCycle_y', 'u_refreshTimes_y',
       'u_newsCatInterests', 'u_newsCatDislike', 'u_newsCatInterestsST_y',
       'u_click_ca2_news', 'i_docId', 'i_s_sourceId', 'i_regionEntity',
       'i_cat', 'i_entities', 'i_dislikeTimes', 'i_upTimes', 'i_dtype', 'e_ch',
       'e_m', 'e_po', 'e_pl', 'e_rn', 'e_section', 'e_et', 'label_y',
       'cillabel', 'pro', 'ads_hour', 'feeds_hour', 'ads_day', 'feeds_day',
```

```
       'ads_dayname', 'feeds_dayname'],
      dtype='object')
```

In [66]:
```
df_noncust.to_csv("noncustomer_df.csv")
```

In [ ]:
```

```

In [67]:
```
import plotly.graph_objects as go


fig = go.Figure(data=[
    go.Bar(name='Potential Customers', x=df_cust['age'].value_counts().index, y=df_cust
    go.Bar(name='Non Customers', x=df_noncust['age'].value_counts().index, y=df_noncust
])
# Change the bar mode
fig.update_layout(barmode='group', title = "Grouped Barchart to Visualize Age Distribut
fig.show()
```

In [68]:
```
fig = px.pie(merged, values=merged['label_y'].value_counts().values, names=merged['labe
fig.show()
```

In [69]:
```
merged.shape
```

Out[69]:
```
(180123, 62)
```

In [70]:
```
merged['label_y'].value_counts()
```

Out[70]:
```
-1    165673
 1     14450
Name: label_y, dtype: int64
```

In [71]:
```
merged.columns
```

Out[71]:
```
Index(['log_id', 'label_x', 'user_id', 'age', 'gender', 'residence', 'city',
       'city_rank', 'series_dev', 'series_group', 'emui_dev', 'device_name',
       'device_size', 'net_type', 'task_id', 'adv_id', 'creat_type_cd',
       'adv_prim_id', 'inter_type_cd', 'slot_id', 'site_id', 'spread_app_id',
       'hispace_app_tags', 'app_second_class', 'app_score',
       'ad_click_list_v001', 'ad_click_list_v002', 'ad_click_list_v003',
       'ad_close_list_v001', 'ad_close_list_v002', 'ad_close_list_v003',
       'pt_d', 'u_newsCatInterestsST_x', 'u_refreshTimes_x',
       'u_feedLifeCycle_x', 'u_phonePrice', 'u_browserLifeCycle',
       'u_browserMode', 'u_feedLifeCycle_y', 'u_refreshTimes_y',
       'u_newsCatInterests', 'u_newsCatDislike', 'u_newsCatInterestsST_y',
       'u_click_ca2_news', 'i_docId', 'i_s_sourceId', 'i_regionEntity',
       'i_cat', 'i_entities', 'i_dislikeTimes', 'i_upTimes', 'i_dtype', 'e_ch',
       'e_m', 'e_po', 'e_pl', 'e_rn', 'e_section', 'e_et', 'label_y',
```

```
       'cillabel', 'pro'],
      dtype='object')
```

In [72]:
```
ads.columns
```

Out[72]:
```
Index(['log_id', 'label', 'user_id', 'age', 'gender', 'residence', 'city',
       'city_rank', 'series_dev', 'series_group', 'emui_dev', 'device_name',
       'device_size', 'net_type', 'task_id', 'adv_id', 'creat_type_cd',
       'adv_prim_id', 'inter_type_cd', 'slot_id', 'site_id', 'spread_app_id',
       'hispace_app_tags', 'app_second_class', 'app_score',
       'ad_click_list_v001', 'ad_click_list_v002', 'ad_click_list_v003',
       'ad_close_list_v001', 'ad_close_list_v002', 'ad_close_list_v003',
       'pt_d', 'u_newsCatInterestsST', 'u_refreshTimes', 'u_feedLifeCycle'],
      dtype='object')
```

In [73]:
```
feeds.columns
```

Out[73]:
```
Index(['u_phonePrice', 'u_browserLifeCycle', 'u_browserMode',
       'u_feedLifeCycle', 'u_refreshTimes', 'u_newsCatInterests',
       'u_newsCatDislike', 'u_newsCatInterestsST', 'u_click_ca2_news',
       'i_docId', 'i_s_sourceId', 'i_regionEntity', 'i_cat', 'i_entities',
       'i_dislikeTimes', 'i_upTimes', 'i_dtype', 'e_ch', 'e_m', 'e_po', 'e_pl',
       'e_rn', 'e_section', 'e_et', 'label', 'cillabel', 'pro', 'user_id'],
      dtype='object')
```

In [74]:
```
feeds.shape
```

Out[74]:
```
(180123, 28)
```

In [75]:
```
feeds['cillabel'].value_counts()
```

Out[75]:
```
-1    180046
 1        77
Name: cillabel, dtype: int64
```

In [76]:
```
import plotly.express as px
fig = px.pie(df_cust, values='gender', names='gender', title = "Potential Customer Gend
fig.show()
```

In [77]:

```
import plotly.express as px
fig = px.pie(df_noncust, values='gender', names='gender', title = "Non Customer Gender
fig.show()
```

In [78]:

```
df_cust['gender'].unique()
```

Out[78]:

```
array([ 3.,  2.,  4., nan])
```

In [79]:

```
import plotly.graph_objects as go


fig = go.Figure(data=[
    go.Bar(name='Potential Customers', x=df_cust['gender'].value_counts().index, y=df_c
    go.Bar(name='Non Customers', x=df_noncust['gender'].value_counts().index, y=df_nonc
])
# Change the bar mode
fig.update_layout(barmode='group', title = "Grouped Barchart to Visualize Age Distribut
fig.show()
```

In [80]:

```
import plotly.graph_objects as go


fig = go.Figure(data=[
    go.Bar(name='Potential Customers', x=df_cust['residence'].value_counts().index, y=d
    go.Bar(name='Non Customers', x=df_noncust['residence'].value_counts().index, y=df_n
])
# Change the bar mode
fig.update_layout(barmode='group', title = "Grouped Barchart to Visualize Residence Dis
fig.show()
```

In [81]:

```
import plotly.graph_objects as go


fig = go.Figure(data=[
    go.Bar(name='Potential Customers', x=df_cust['e_section'].value_counts().index, y=d
    go.Bar(name='Non Customers', x=df_noncust['e_section'].value_counts().index, y=df_n
])
# Change the bar mode
fig.update_layout(barmode='group', title = "Grouped Barchart to Visualize Content Prefe
fig.show()
```

In [82]:
```
df_cust['e_section'].value_counts()
```

Out[82]:
```
0    8685
1    5765
Name: e_section, dtype: int64
```

In [83]:
```
df_noncust['e_section'].value_counts()
```

Out[83]:
```
1    104133
0     61540
Name: e_section, dtype: int64
```

In [84]:
```
df_cust.columns
```

Out[84]:
```
Index(['log_id', 'label_x', 'user_id', 'age', 'gender', 'residence', 'city',
       'city_rank', 'series_dev', 'series_group', 'emui_dev', 'device_name',
       'device_size', 'net_type', 'task_id', 'adv_id', 'creat_type_cd',
       'adv_prim_id', 'inter_type_cd', 'slot_id', 'site_id', 'spread_app_id',
       'hispace_app_tags', 'app_second_class', 'app_score',
       'ad_click_list_v001', 'ad_click_list_v002', 'ad_click_list_v003',
       'ad_close_list_v001', 'ad_close_list_v002', 'ad_close_list_v003',
       'pt_d', 'u_newsCatInterestsST_x', 'u_refreshTimes_x',
       'u_feedLifeCycle_x', 'u_phonePrice', 'u_browserLifeCycle',
       'u_browserMode', 'u_feedLifeCycle_y', 'u_refreshTimes_y',
       'u_newsCatInterests', 'u_newsCatDislike', 'u_newsCatInterestsST_y',
       'u_click_ca2_news', 'i_docId', 'i_s_sourceId', 'i_regionEntity',
       'i_cat', 'i_entities', 'i_dislikeTimes', 'i_upTimes', 'i_dtype', 'e_ch',
       'e_m', 'e_po', 'e_pl', 'e_rn', 'e_section', 'e_et', 'label_y',
       'cillabel', 'pro', 'ads_hour', 'feeds_hour', 'ads_day', 'feeds_day',
       'ads_dayname', 'feeds_dayname'],
      dtype='object')
```

In [85]:
```
import plotly.graph_objects as go


fig = go.Figure(data=[
    go.Bar(name='Potential Customers', x=df_cust['series_dev'].value_counts().index, y=
    go.Bar(name='Non Customers', x=df_noncust['series_dev'].value_counts().index, y=df_
])
# Change the bar mode
fig.update_layout(barmode='group', title = "Grouped Barchart to Visualize Device Series
fig.show()
```

In [86]:

```
import plotly.graph_objects as go


fig = go.Figure(data=[
    go.Bar(name='Potential Customers', x=df_cust['series_group'].value_counts().index, 
    go.Bar(name='Non Customers', x=df_noncust['series_group'].value_counts().index, y=d
])
# Change the bar mode
fig.update_layout(barmode='group', title = "Grouped Barchart to Visualize Device Series
fig.show()
```

In [87]:

```
import plotly.graph_objects as go


fig = go.Figure(data=[
    go.Bar(name='Potential Customers', x=df_cust['emui_dev'].value_counts().index, y=df
    go.Bar(name='Non Customers', x=df_noncust['emui_dev'].value_counts().index, y=df_no
])
# Change the bar mode
fig.update_layout(barmode='group', title = "Grouped Barchart to Visualize Device EMUI D
fig.show()
```

In [88]:

```
import plotly.graph_objects as go


fig = go.Figure(data=[
    go.Bar(name='Potential Customers', x=df_cust['ads_hour'].value_counts().index, y=df
    go.Bar(name='Non Customers', x=df_noncust['ads_hour'].value_counts().index, y=df_no
])
# Change the bar mode
fig.update_layout(barmode='group', title = "Grouped Barchart to Visualize Ad Hour Viewe
fig.show()
```

In [89]:

```
import plotly.graph_objects as go


fig = go.Figure(data=[
    go.Bar(name='Potential Customers', x=df_cust['feeds_hour'].value_counts().index, y=
    go.Bar(name='Non Customers', x=df_noncust['feeds_hour'].value_counts().index, y=df_
])
# Change the bar mode
fig.update_layout(barmode='group', title = "Grouped Barchart to Visualize Feed Hour Vie
fig.show()
```

In [90]:

```
import plotly.graph_objects as go


fig = go.Figure(data=[
    go.Bar(name='Potential Customers', x=df_cust['feeds_day'].value_counts().index, y=d
    go.Bar(name='Non Customers', x=df_noncust['feeds_day'].value_counts().index, y=df_n
])
# Change the bar mode
fig.update_layout(barmode='group', title = "Grouped Barchart to Visualize Feed Day View
fig.show()
```

In [91]:

```
import plotly.graph_objects as go


fig = go.Figure(data=[
    go.Bar(name='Potential Customers', x=df_cust['ads_day'].value_counts().index, y=df_
    go.Bar(name='Non Customers', x=df_noncust['ads_day'].value_counts().index, y=df_non
])
# Change the bar mode
fig.update_layout(barmode='group', title = "Grouped Barchart to Visualize Ad Day Viewed
fig.show()
```

In [92]:
```
df_cust.columns
```

Out[92]:
```
Index(['log_id', 'label_x', 'user_id', 'age', 'gender', 'residence', 'city',
       'city_rank', 'series_dev', 'series_group', 'emui_dev', 'device_name',
       'device_size', 'net_type', 'task_id', 'adv_id', 'creat_type_cd',
       'adv_prim_id', 'inter_type_cd', 'slot_id', 'site_id', 'spread_app_id',
       'hispace_app_tags', 'app_second_class', 'app_score',
       'ad_click_list_v001', 'ad_click_list_v002', 'ad_click_list_v003',
       'ad_close_list_v001', 'ad_close_list_v002', 'ad_close_list_v003',
       'pt_d', 'u_newsCatInterestsST_x', 'u_refreshTimes_x',
       'u_feedLifeCycle_x', 'u_phonePrice', 'u_browserLifeCycle',
       'u_browserMode', 'u_feedLifeCycle_y', 'u_refreshTimes_y',
       'u_newsCatInterests', 'u_newsCatDislike', 'u_newsCatInterestsST_y',
       'u_click_ca2_news', 'i_docId', 'i_s_sourceId', 'i_regionEntity',
       'i_cat', 'i_entities', 'i_dislikeTimes', 'i_upTimes', 'i_dtype', 'e_ch',
       'e_m', 'e_po', 'e_pl', 'e_rn', 'e_section', 'e_et', 'label_y',
       'cillabel', 'pro', 'ads_hour', 'feeds_hour', 'ads_day', 'feeds_day',
       'ads_dayname', 'feeds_dayname'],
      dtype='object')
```

In [93]:
```
import plotly.express as px
fig = px.pie(df_cust, values= df_cust['i_upTimes'].value_counts().values, names=df_cust
fig.show()
```

In [94]:

```
import plotly.express as px
fig = px.pie(df_noncust, values= df_noncust['i_upTimes'].value_counts().values, names=d
fig.show()
```

In [95]:

```
import plotly.express as px
fig = px.pie(df_cust, values= df_cust['i_dislikeTimes'].value_counts().values, names=df
fig.show()
```

In [96]:

```
import plotly.express as px
fig = px.pie(df_noncust, values= df_noncust['i_dislikeTimes'].value_counts().values, na
fig.show()
```

In [97]:

```
import plotly.express as px
fig = px.pie(df_cust, values= df_cust['pro'].value_counts().values, names=df_cust['pro'
```

```
fig.show()
```

In [98]:
```
df_noncust['pro'].value_counts()
```

Out[98]:
```
0    165673
Name: pro, dtype: int64
```

In [99]:
```
import plotly.express as px
fig = px.pie(df_noncust, values= df_noncust['pro'].value_counts().values, names=df_nonc
fig.show()
```

In [100…

```
import plotly.graph_objects as go


fig = go.Figure(data=[
    go.Bar(name='Potential Customers', x=df_cust['i_upTimes'].value_counts().index, y=d
    go.Bar(name='Non Customers', x=df_noncust['i_upTimes'].value_counts().index, y=df_n
])
# Change the bar mode
fig.update_layout(barmode='group', title = "Grouped Barchart to Visualize Article Liked
fig.show()
```

In [101…

```
import plotly.graph_objects as go


fig = go.Figure(data=[
    go.Bar(name='Potential Customers', x=df_cust['i_dislikeTimes'].value_counts().index
    go.Bar(name='Non Customers', x=df_noncust['i_dislikeTimes'].value_counts().index, y
])
# Change the bar mode
fig.update_layout(barmode='group', title = "Grouped Barchart to Visualize Article Disli
fig.show()
```

In [102…

```
import plotly.graph_objects as go


fig = go.Figure(data=[
    go.Bar(name='Potential Customers', x=df_cust['pro'].value_counts().index, y=df_cust
    go.Bar(name='Non Customers', x=df_noncust['pro'].value_counts().index, y=df_noncust
])
# Change the bar mode
fig.update_layout(barmode='group', title = "Grouped Barchart to Visualize Article Progr
fig.show()
```

In [ ]: