report

July 28, 2024

1 Exercise 3

[28]:
```
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
import numpy as np
```

1.1 Loading the data and casting datetime in ISO format

[29]:
```
df = pd.read_csv("./data/dataset.csv")
df.summary_date = pd.to_datetime(df.summary_date, format="%Y.%m.%d")
df['summary_date'] = df['summary_date'].apply(lambda x: x.isoformat())
df.set_index('summary_date')
```

[29]:

summary_date	app_id	app_type	app_name	campaign_id	campaign_name	ad_id	ad_name
2022-06-14T00:00:00	com.super.app	android	Super App 2000	campaign_16	Super campaign 16	ad_16L8hGR	Super AD 16L8hGR
2022-07-10T00:00:00	com.super.app	android	Super App 2000	campaign_16	Super campaign 16	ad_16z49oF	Super AD 16z49oF
2022-07-05T00:00:00	com.super.app	android	Super App 2000	campaign_16	Super campaign 16	ad_16DpJ5e	Super AD 16DpJ5e
2022-06-01T00:00:00	com.super.app	android	Super App 2000	campaign_16	Super campaign 16	ad_16sutOl	Super AD 16sutOl
2022-07-25T00:00:00	com.super.app	android	Super App 2000	campaign_16	Super campaign 16	ad_16ZEk4H	Super AD 16ZEk4H
…	…	…	…	…	…	…	…
2022-06-25T00:00:00	com.super.app	android	Super App 2000	campaign_16	Super campaign 16	ad_16tke86	Super AD 16tke86
2022-06-11T00:00:00	com.super.app	android	Super App 2000	campaign_16	Super campaign 16	ad_16L8hGR	Super AD 16L8hGR
2022-07-09T00:00:00	com.super.app	android	Super App 2000	campaign_16	Super campaign 16	ad_16z49oF	Super AD 16z49oF
2022-07-08T00:00:00	com.super.app	android	Super App 2000	campaign_16	Super campaign 16	ad_16DpJ5e	Super AD 16DpJ5e
2022-07-02T00:00:00	com.super.app	android	Super App 2000	campaign_16	Super campaign 16	ad_16sutOl	Super AD 16sutOl

```
2022-06-11T00:00:00  Super campaign 16  ad_16Qnvp6  Super AD 16Qnvp6
2022-07-09T00:00:00  Super campaign 16  ad_1658MjI  Super AD 1658MjI
2022-07-08T00:00:00  Super campaign 16  ad_16VAIpg  Super AD 16VAIpg
2022-07-02T00:00:00  Super campaign 16  ad_16L8hGR  Super AD 16L8hGR

                     impressions  clicks  installs     spend   event_name  \
summary_date
2022-06-14T00:00:00         82.0    46.0       NaN   0.00455  super_event
2022-07-10T00:00:00       1942.0  1226.0       2.0   0.88920  super_event
2022-07-05T00:00:00        116.0     2.0       NaN   0.00715  super_event
2022-06-01T00:00:00      15550.0    18.0       4.0   0.52455  super_event
2022-07-25T00:00:00          0.0     0.0       2.0   0.00000  super_event
…                              …       …         …         …            …
2022-06-25T00:00:00      62954.0   162.0       NaN  11.77605  super_event
2022-06-11T00:00:00         28.0     6.0       NaN   0.00455  super_event
2022-07-09T00:00:00        868.0   470.0       NaN   0.41340  super_event
2022-07-08T00:00:00       7376.0  4366.0       4.0   3.01665  super_event
2022-07-02T00:00:00          0.0     0.0       2.0   0.00000  super_event

                     events_d0  events_d7  unique_events_d0  unique_events_d7
summary_date
2022-06-14T00:00:00        NaN        NaN               NaN               NaN
2022-07-10T00:00:00        0.0        0.0               0.0               0.0
2022-07-05T00:00:00        NaN        NaN               NaN               NaN
2022-06-01T00:00:00        0.0        0.0               0.0               0.0
2022-07-25T00:00:00        0.0        0.0               0.0               0.0
…                            …          …                 …                 …
2022-06-25T00:00:00        NaN        NaN               NaN               NaN
2022-06-11T00:00:00        NaN        NaN               NaN               NaN
2022-07-09T00:00:00        NaN        NaN               NaN               NaN
2022-07-08T00:00:00        0.0        0.0               0.0               0.0
2022-07-02T00:00:00        0.0        0.0               0.0               0.0

[1570 rows x 16 columns]
```

1.2 Exploratory analysis

1.3 Initial overview

[30]:
```
df.head()
```

[30]:
```
          summary_date         app_id app_type        app_name  campaign_id  \
0  2022-06-14T00:00:00  com.super.app  android  Super App 2000  campaign_16
1  2022-07-10T00:00:00  com.super.app  android  Super App 2000  campaign_16
2  2022-07-05T00:00:00  com.super.app  android  Super App 2000  campaign_16
3  2022-06-01T00:00:00  com.super.app  android  Super App 2000  campaign_16
4  2022-07-25T00:00:00  com.super.app  android  Super App 2000  campaign_16
```

```
       campaign_name       ad_id           ad_name  impressions  clicks  \
0  Super campaign 16  ad_16L8hGR  Super AD 16L8hGR         82.0    46.0
1  Super campaign 16  ad_16z49oF  Super AD 16z49oF       1942.0  1226.0
2  Super campaign 16  ad_16DpJ5e  Super AD 16DpJ5e        116.0     2.0
3  Super campaign 16  ad_16sutOl  Super AD 16sutOl      15550.0    18.0
4  Super campaign 16  ad_16ZEk4H  Super AD 16ZEk4H          0.0     0.0

   installs    spend   event_name  events_d0  events_d7  unique_events_d0  \
0       NaN  0.00455  super_event        NaN        NaN               NaN
1       2.0  0.88920  super_event        0.0        0.0               0.0
2       NaN  0.00715  super_event        NaN        NaN               NaN
3       4.0  0.52455  super_event        0.0        0.0               0.0
4       2.0  0.00000  super_event        0.0        0.0               0.0

   unique_events_d7
0               NaN
1               0.0
2               NaN
3               0.0
4               0.0
```

[31]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1570 entries, 0 to 1569
Data columns (total 17 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   summary_date      1570 non-null   object
 1   app_id            1570 non-null   object
 2   app_type          1570 non-null   object
 3   app_name          1570 non-null   object
 4   campaign_id       1570 non-null   object
 5   campaign_name     1570 non-null   object
 6   ad_id             1453 non-null   object
 7   ad_name           1453 non-null   object
 8   impressions       1489 non-null   float64
 9   clicks            1489 non-null   float64
 10  installs          826 non-null    float64
 11  spend             1489 non-null   float64
 12  event_name        1570 non-null   object
 13  events_d0         826 non-null    float64
 14  events_d7         826 non-null    float64
 15  unique_events_d0  826 non-null    float64
 16  unique_events_d7  826 non-null    float64
dtypes: float64(8), object(9)
memory usage: 208.6+ KB
```

[32]:
```
df.describe()
```

[32]:

	impressions	clicks	installs	spend	events_d0	\
count	1.489000e+03	1.489000e+03	826.000000	1489.000000	826.000000	
mean	2.289176e+05	5.648466e+03	954.670702	51.666529	4.322034	
std	6.223744e+05	4.006779e+04	5317.702577	294.343442	10.845378	
min	0.000000e+00	0.000000e+00	2.000000	0.000000	0.000000	
25%	4.800000e+01	6.000000e+00	4.000000	0.011700	0.000000	
50%	4.984000e+03	9.000000e+01	12.000000	0.689650	0.000000	
75%	7.424400e+04	1.288000e+03	42.000000	9.286550	2.000000	
max	5.825656e+06	1.215918e+06	83240.000000	4803.045000	82.000000	

	events_d7	unique_events_d0	unique_events_d7
count	826.000000	826.000000	826.000000
mean	5.491525	3.573850	4.210654
std	14.054200	8.846297	10.448334
min	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000
50%	0.000000	0.000000	0.000000
75%	4.000000	2.000000	2.000000
max	116.000000	58.000000	68.000000

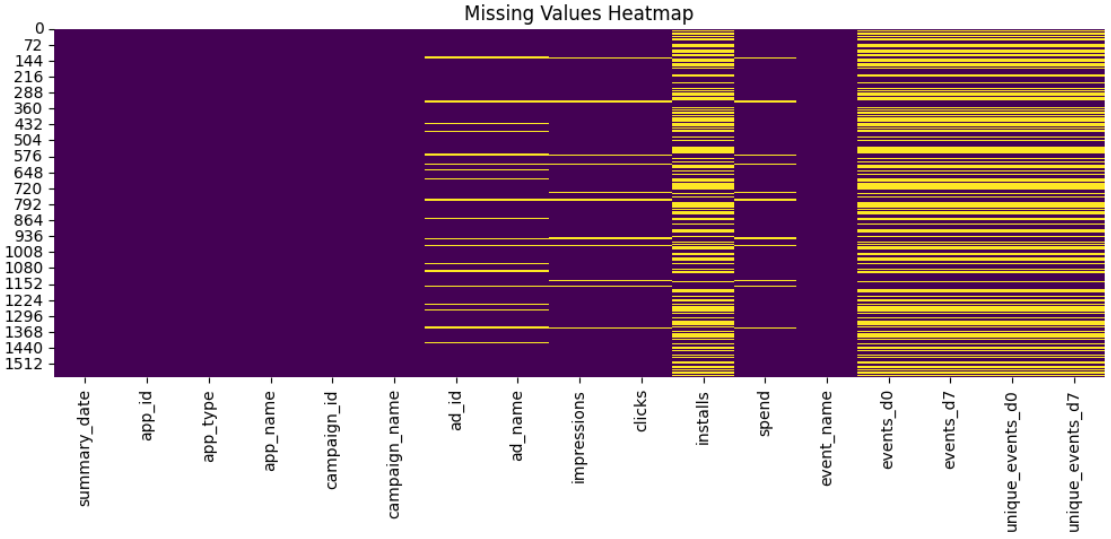
1.4 Missing values

[33]:
```
df.isnull().sum()
```

[33]:
```
summary_date          0
app_id                0
app_type              0
app_name              0
campaign_id           0
campaign_name         0
ad_id               117
ad_name             117
impressions          81
clicks               81
installs            744
spend                81
event_name            0
events_d0           744
events_d7           744
unique_events_d0    744
unique_events_d7    744
dtype: int64
```

[34]:
```
plt.figure(figsize=(12, 4))
sns.heatmap(df.isnull(), cbar=False, cmap='viridis')
```

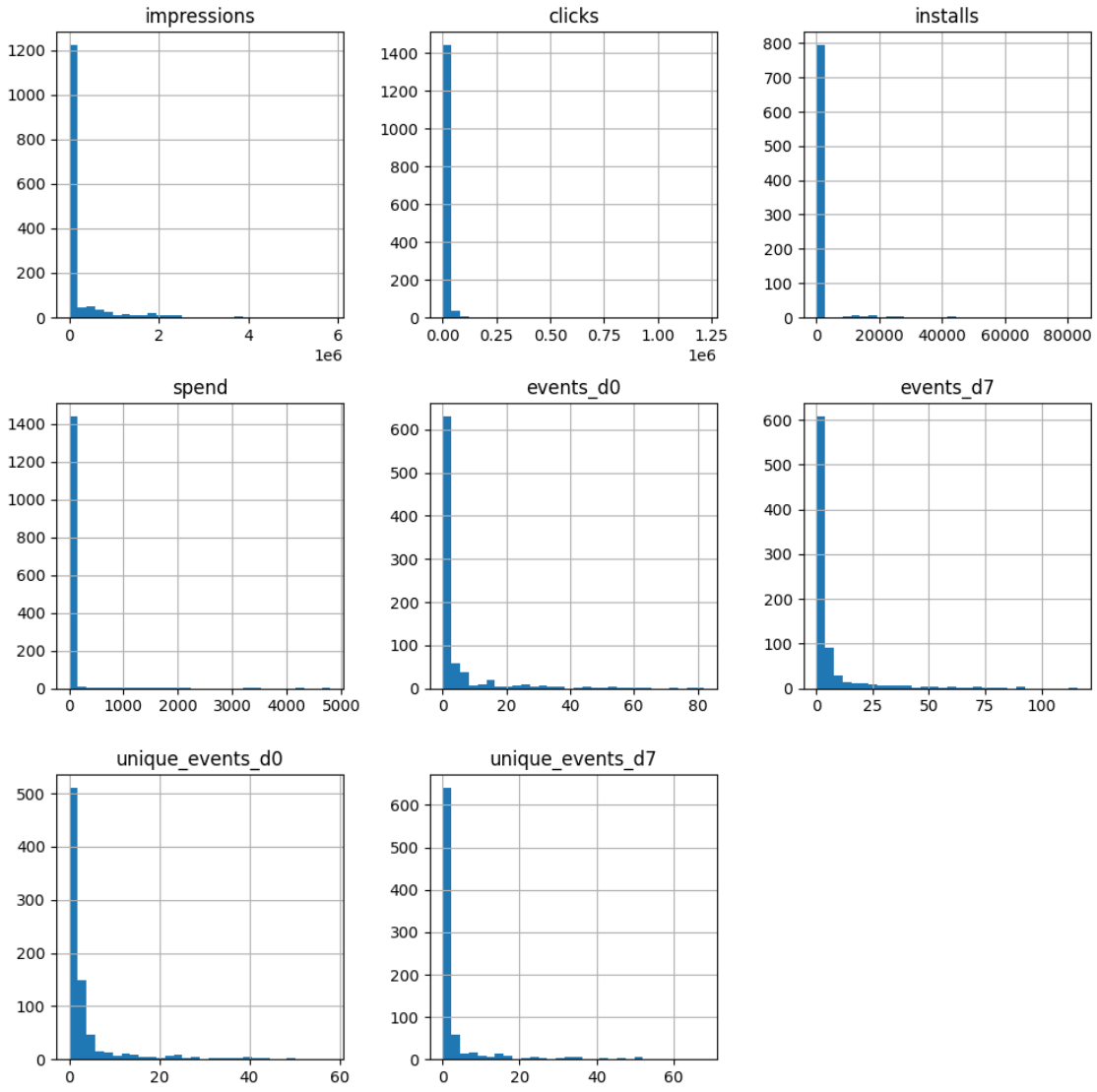
```
plt.title('Missing Values Heatmap')
plt.show()
```



Here we can see the yellow lines as missing values, the events and install variables have the majority of NA's in our dataset

1.5 Distribution exploration

[35]:
```
df.hist(figsize=(12, 12), bins=30)
plt.tick_params(axis='x', labelrotation=90)
plt.show()
```



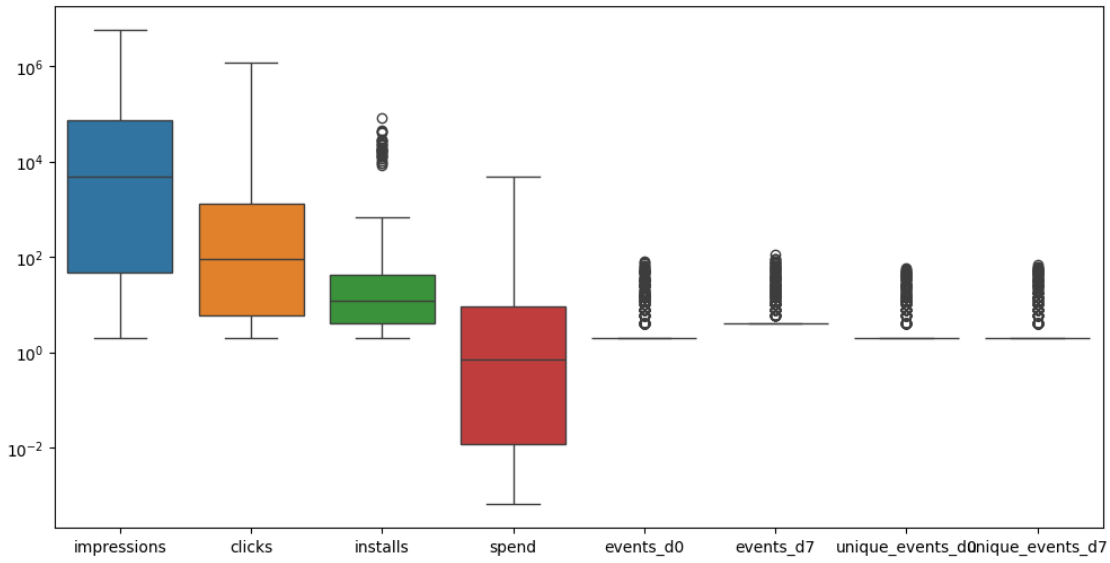
For the quantitative variables we can grasp here that the variabes values are very concentrated in the lower values, while having many outliers existing with values sensibly higher than the median.

1.6 Outliers exploration

[36]:
```
plt.figure(figsize=(12, 6))
sns.boxplot(data=df.select_dtypes(include=['float64']), log_scale=True)
plt.show()
```

```
/home/luko/winclap/WinClap-dataEng-technicalTest/.venv/lib/python3.11/site-
packages/numpy/lib/_function_base_impl.py:4779: RuntimeWarning: invalid value
encountered in subtract
  diff_b_a = subtract(b, a)
/home/luko/winclap/WinClap-dataEng-technicalTest/.venv/lib/python3.11/site-
```

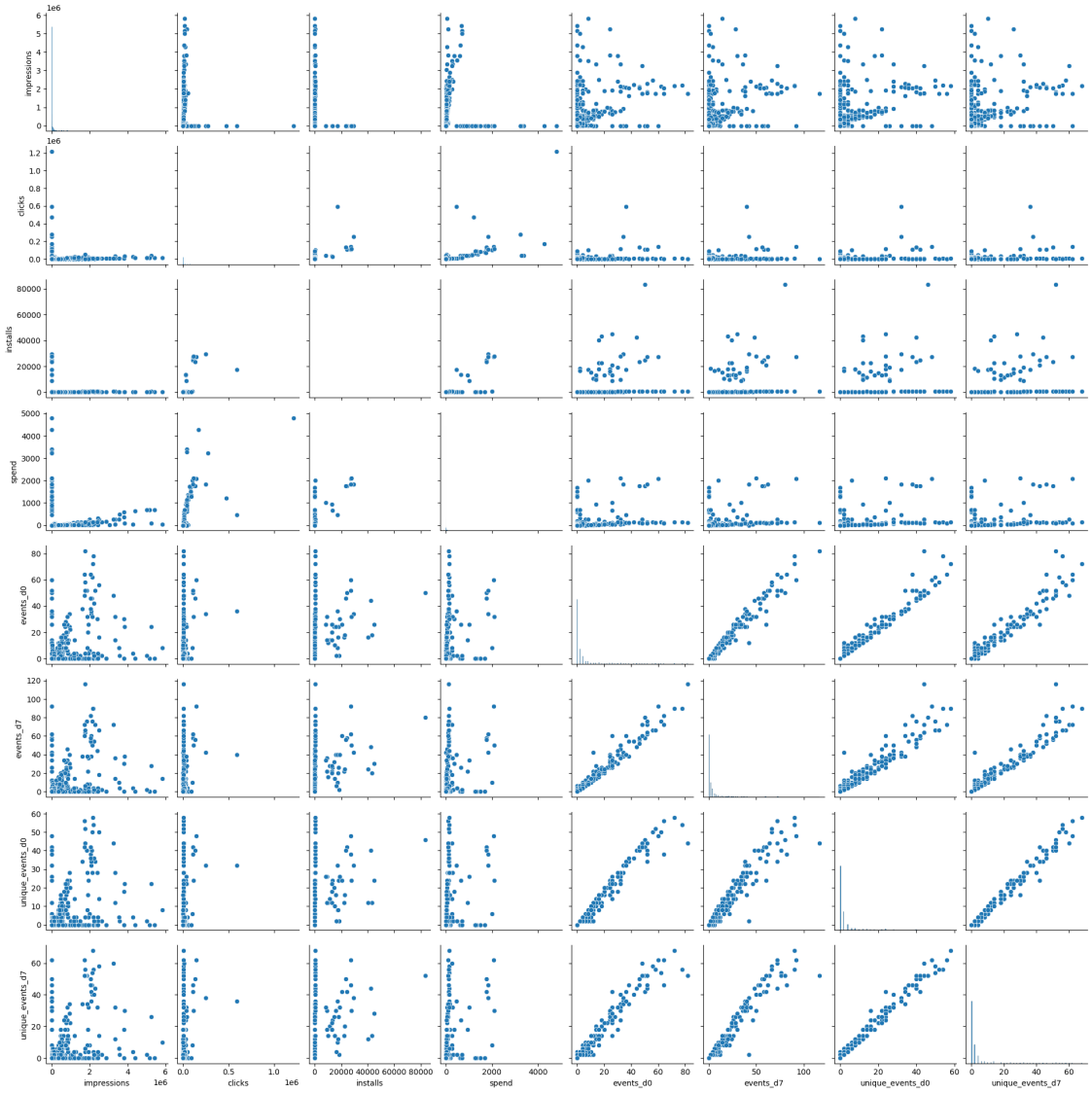
```
packages/numpy/lib/_function_base_impl.py:4779: RuntimeWarning: invalid value
encountered in subtract
  diff_b_a = subtract(b, a)
/home/luko/winclap/WinClap-dataEng-technicalTest/.venv/lib/python3.11/site-
packages/numpy/lib/_function_base_impl.py:4779: RuntimeWarning: invalid value
encountered in subtract
  diff_b_a = subtract(b, a)
/home/luko/winclap/WinClap-dataEng-technicalTest/.venv/lib/python3.11/site-
packages/numpy/lib/_function_base_impl.py:4779: RuntimeWarning: invalid value
encountered in subtract
  diff_b_a = subtract(b, a)
```



Here we can confirm the precence of the ditribution outliers with the log scale. But we must take in consideration that the majority of outliers appear in the variables with more missing values. So it needs further analysis focusing in those variables.

1.7 Relationships exploration

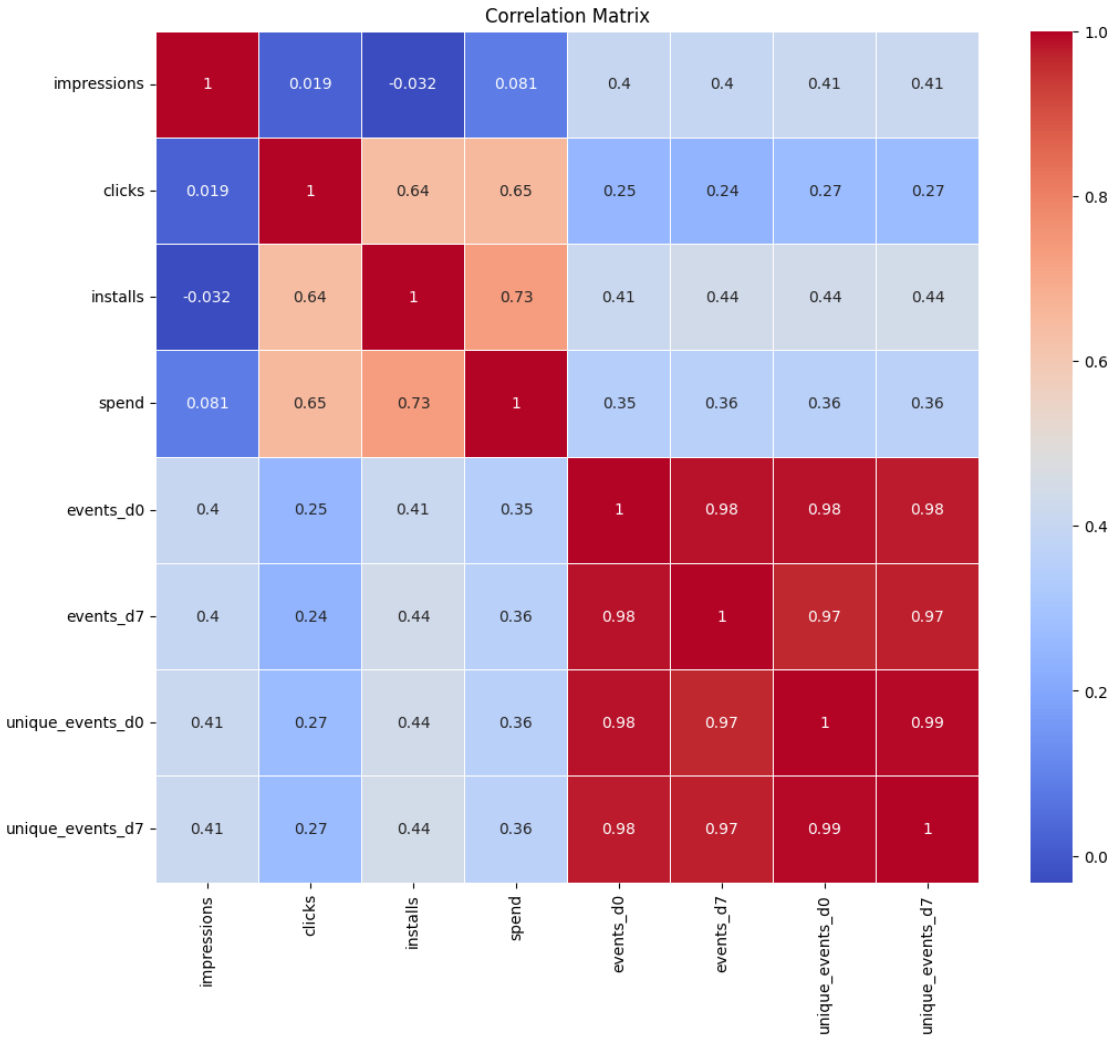
[37]:
```
sns.pairplot(df.select_dtypes(include=['float64']))
plt.show()
```



We can see there are strong correlation for some relationships, while we can argue some others have a deffinite behaviour in the variables interaction that needs futher analysis.

1.8 Correlation exploration

[38]:
```
corr_matrix = df.select_dtypes(include=['int','float']).corr()
plt.figure(figsize=(12, 10))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Correlation Matrix')
plt.show()
```

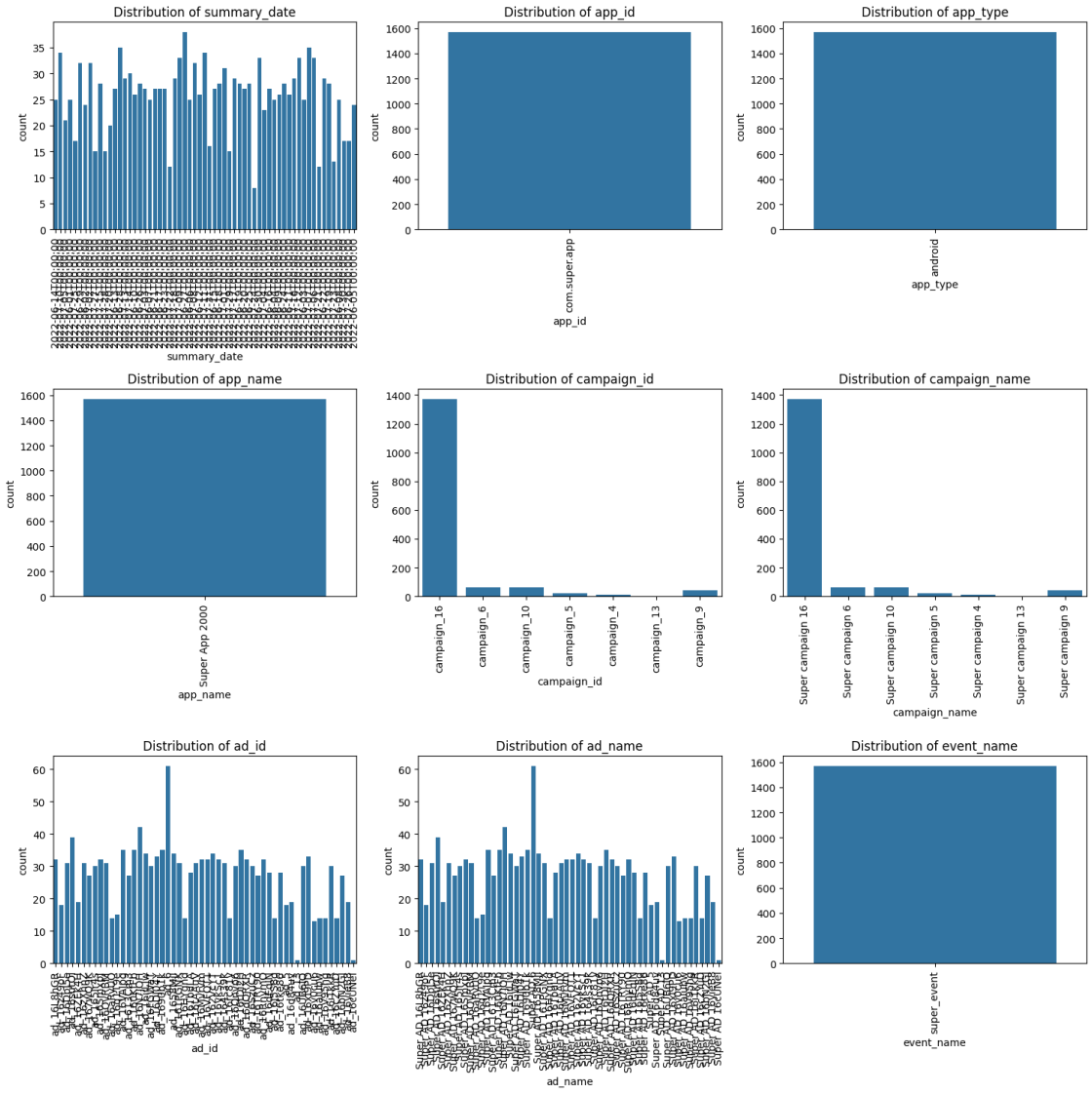


Theres a strong corralation between the events, but also between the spend and the installs

1.9 Categorical variables

[39]:
```
categorical_columns = df.select_dtypes(include=['object']).columns


num_cols = len(categorical_columns)
ncols = 3
nrows = (num_cols + ncols - 1) // ncols

fig, axes = plt.subplots(nrows=nrows, ncols=ncols, figsize=(15, nrows * 5))

axes = axes.flatten()
```

```
for i, col in enumerate(categorical_columns):
    sns.countplot(data=df, x=col, ax=axes[i])
    axes[i].set_title(f'Distribution of {col}')
    axes[i].tick_params(axis='x', rotation=90)

for j in range(i + 1, len(axes)):
    fig.delaxes(axes[j])

plt.tight_layout()
plt.show()
```



For some variables theres only one value, campaign_id and campaign_name (which its distribution is logicaly the same) distribution is very concentrated in `campaign_16`. The adds variable is more

equally distributed.

2 Report

2.1 Grouping by campaign and adding CTR and CPI columns

[40]:
```
def insert_ctr_cpi(df: pd.DataFrame)->pd.DataFrame:
    df["CTR"] = df.clicks/df.impressions
    df["CPI"] = df.spend/df.installs
    return df

df = pd.DataFrame(
    df.groupby(['campaign_id', 'campaign_name'])
    .sum(numeric_only=True)
    [['impressions','clicks','installs','spend']]
    )
df = insert_ctr_cpi(df)
df
```

[40]:

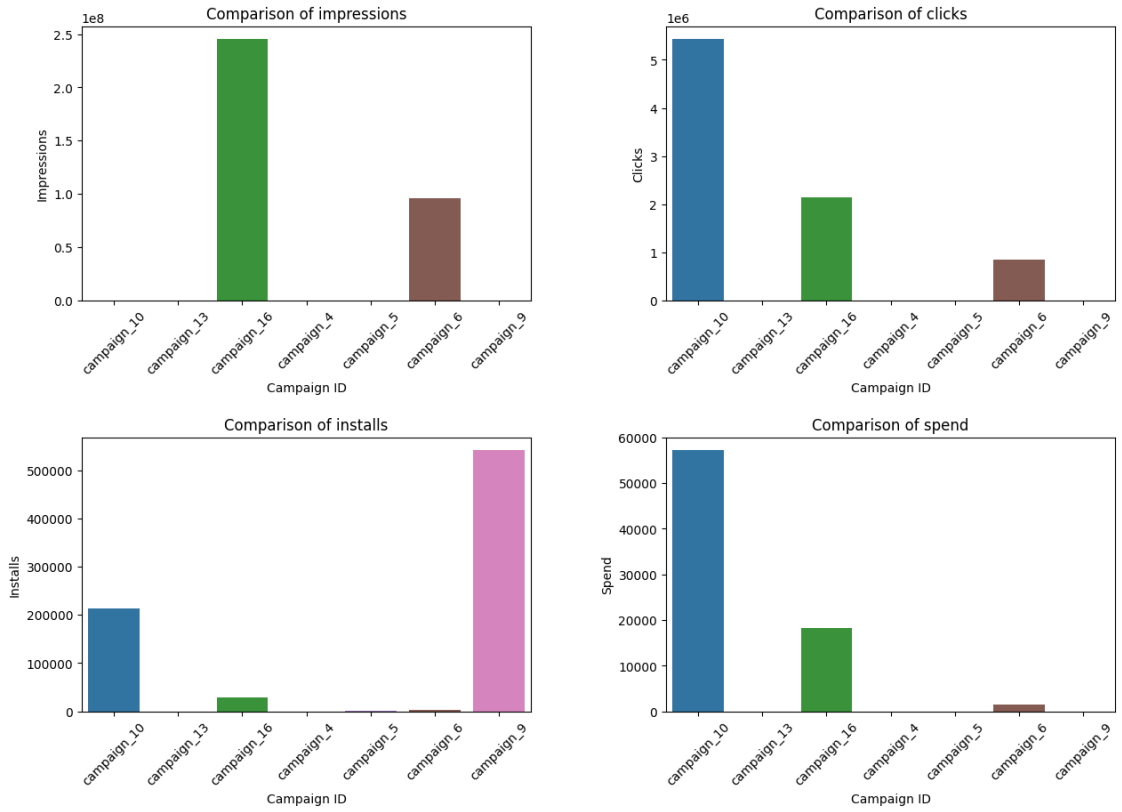
campaign_id	campaign_name	impressions	clicks	installs	spend
campaign_10	Super campaign 10	0.0	5429764.0	213682.0	57194.6700
campaign_13	Super campaign 13	0.0	0.0	2.0	0.0000
campaign_16	Super campaign 16	245177026.0	2132984.0	28776.0	18314.5937
campaign_4	Super campaign 4	0.0	0.0	26.0	0.0000
campaign_5	Super campaign 5	0.0	0.0	1072.0	0.0000
campaign_6	Super campaign 6	95681244.0	847818.0	3776.0	1422.1974
campaign_9	Super campaign 9	0.0	0.0	541224.0	0.0000

campaign_id	campaign_name	CTR	CPI
campaign_10	Super campaign 10	inf	0.267663
campaign_13	Super campaign 13	NaN	0.000000
campaign_16	Super campaign 16	0.008700	0.636454
campaign_4	Super campaign 4	NaN	0.000000
campaign_5	Super campaign 5	NaN	0.000000
campaign_6	Super campaign 6	0.008861	0.376641
campaign_9	Super campaign 9	NaN	0.000000

2.2 Impressions, clicks, install and spend by campaign

[41]:
```
campaign_grouping = df.groupby('campaign_id').sum()

metrics = ['impressions', 'clicks', 'installs', 'spend']


n_metrics = len(metrics)
```

```
ncols = 2
nrows = (n_metrics + ncols - 1) // ncols

fig, axes = plt.subplots(nrows=nrows, ncols=ncols, figsize=(15, nrows * 5))

axes = axes.flatten() # type: ignore

for i, metric in enumerate(metrics):
    sns.barplot(x=campaign_grouping.index, y=campaign_grouping[metric],␣
 ↪ax=axes[i], hue=campaign_grouping.index, legend=False)
    axes[i].set_title(f'Comparison of {metric}')
    axes[i].set_ylabel(metric.capitalize())
    axes[i].set_xlabel('Campaign ID')
    axes[i].tick_params(axis='x', rotation=45)

for j in range(i + 1, len(axes)):
    fig.delaxes(axes[j])

plt.subplots_adjust(hspace=.5, wspace=0.3)
plt.figure(figsize=(12, 4))
plt.tight_layout()
plt.show()
```



```
<Figure size 1200x400 with 0 Axes>
```

We can see the spend was singificantly higher in campaign 10 and 16, but it seems that the spendidure was transfered to clicks in campaign 10 but without impressions info and for campaign 16 highlights for its impression values. For campaign 9 we have no data for the spend variable but has the higher istalls number. We can compare campaign 6 with 16 by its CTR as we have data for clicks and impressions. As 16, 10 and 6 are the campaigns with spend data we can compare them by its CPI.

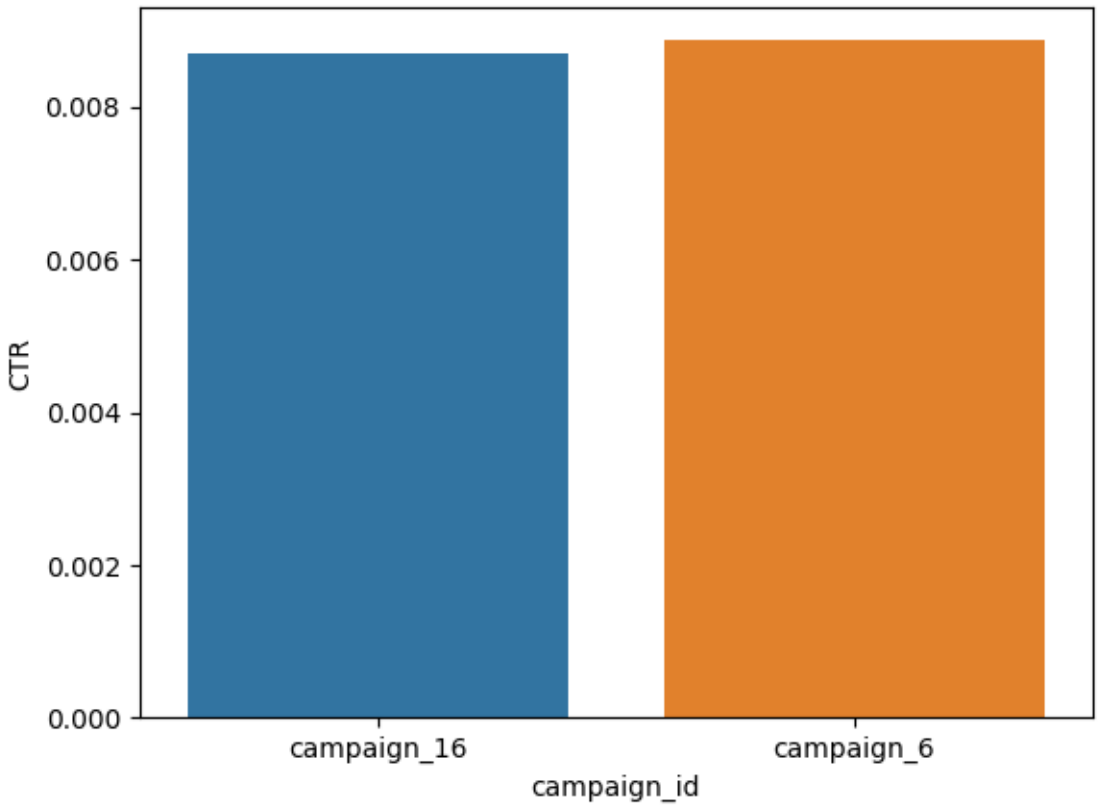
[42]:
```
df.loc["campaign_10"].CTR.values[0]
```

[42]:
```
np.float64(inf)
```

2.3 CTR comparison

[43]:
```
ctr_df = df[df.CTR.notna() & (df.CTR != 0) & (df.CTR!=np.inf)]
ctr_df=ctr_df.reset_index()
sns.barplot(x=ctr_df.campaign_id, y=ctr_df["CTR"], hue=ctr_df.index,␣
↪legend=False)
```

[43]:
```
<Axes: xlabel='campaign_id', ylabel='CTR'>
```

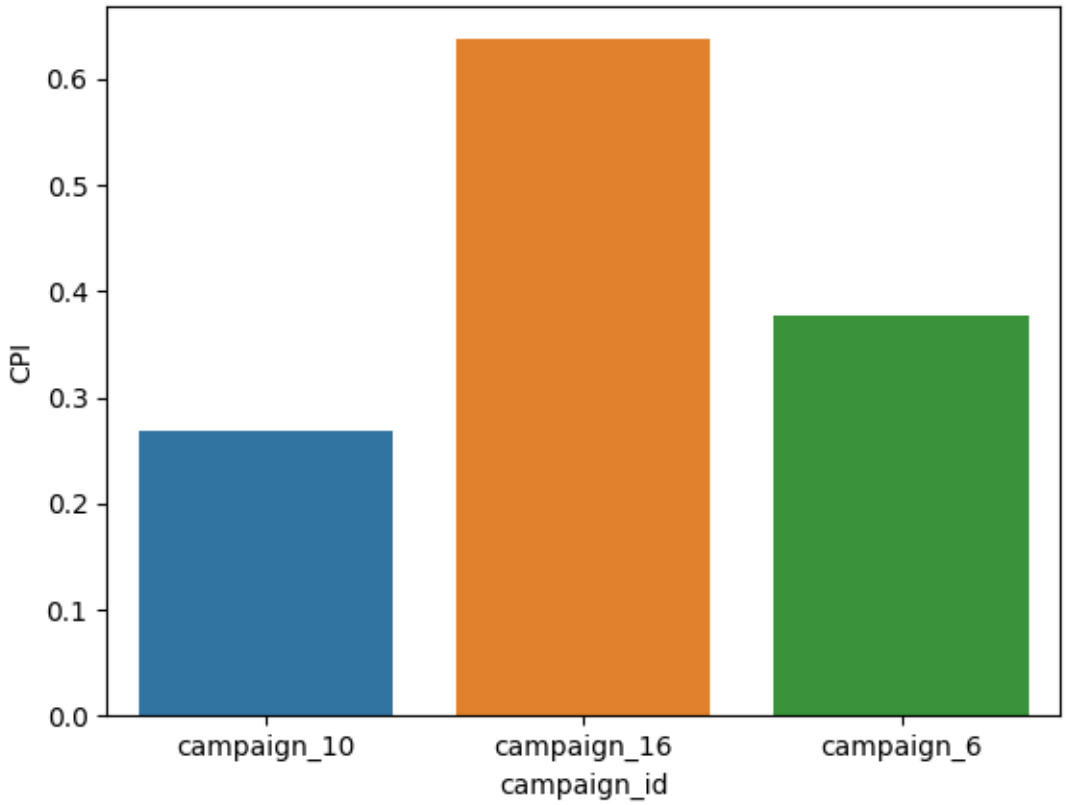


We can observe, while the volume of clicks and impressions of campaign 16 (see previous chart) was higher, the CTR metric is very similar in both.

2.4 CPI comparison

[63]:
```
CPI_df = df[df.CPI.notna() & (df.CPI != 0) & (df.CPI!=np.inf)]
CPI_df=CPI_df.reset_index()
# CPI_df
sns.barplot(x=CPI_df.campaign_id, y=CPI_df["CPI"], hue=CPI_df.campaign_id.
 ↪to_list(), legend=False)
```

[63]:
```
<Axes: xlabel='campaign_id', ylabel='CPI'>
```



The campaing 16 was the most succesfull in terms of CPI, overpassing the campaign 10 which is the campaing with higher spendidure numbers, almost three times campaign 16 numbers. Campaign 6 falls in the middle being smaller in terms of overall numbers.