In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import re
```

In [2]:
```
df=pd.read_csv('scaler_clustering.csv')
```

In [3]:
```
df.head()
```

Out[3]:

	Unnamed: 0	company_hash	email_hash	orgy
0	0	atrgxnnt xzaxv	6de0a4417d18ab14334c3f43397fc13b30c35149d70c05...	201
1	1	qtrxvzwt xzegwgbb rxbxnta	b0aaf1ac138b53cb6e039ba2c3d6604a250d02d5145c10...	201
2	2	ojzwnvwnxw vx	4860c670bcd48fb96c02a4b0ae3608ae6fdd98176112e9...	201
3	3	ngpgutaxv	effdede7a2e7c2af664c8a31d9346385016128d66bbc58...	201
4	4	qxen sqghu	6ff54e709262f55cb999a1c1db8436cb2055d8f79ab520...	201

In [4]:
```
df.shape
```

Out[4]:
```
(205843, 7)
```

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205843 entries, 0 to 205842
Data columns (total 7 columns):
 #   Column            Non-Null Count   Dtype  
---  ------            --------------   -----  
 0   Unnamed: 0        205843 non-null  int64  
 1   company_hash      205799 non-null  object 
 2   email_hash        205843 non-null  object 
 3   orgyear           205757 non-null  float64
 4   ctc               205843 non-null  int64  
 5   job_position      153281 non-null  object 
 6   ctc_updated_year  205843 non-null  float64
dtypes: float64(2), int64(2), object(3)
memory usage: 11.0+ MB
```

In [6]:
```
df.describe()
```

Out[6]:

	Unnamed: 0	orgyear	ctc	ctc_updated_year
count	205843.000000	205757.000000	2.058430e+05	205843.000000
mean	103273.941786	2014.882750	2.271685e+06	2019.628231
std	59741.306484	63.571115	1.180091e+07	1.325104
min	0.000000	0.000000	2.000000e+00	2015.000000
25%	51518.500000	2013.000000	5.300000e+05	2019.000000
50%	103151.000000	2016.000000	9.500000e+05	2020.000000
75%	154992.500000	2018.000000	1.700000e+06	2021.000000
max	206922.000000	20165.000000	1.000150e+09	2021.000000

In [7]:
```
df.describe(include=object)
```

Out[7]:

	company_hash	email_hash	job_positio
count	205799	205843	15328
unique	37299	153443	101
top	nvnv wgzohrnvzwj otqcxwto	bbace3cc586400bbc65765bc6a16b77d8913836cfc98b7...	Backen Enginee
freq	8337	10	4355

In [8]:
```
df=df[(df['orgyear']<2022) & (df['orgyear']>1900)]
```

Univariate Analysis

Non Visual Analysis

In [9]:
```
df['email_hash'].value_counts()
```

Out[9]:
```
bbace3cc586400bbc65765bc6a16b77d8913836cfc98b77c05488f02f5714a4b    10
3e5e49daa5527a6d5a33599b238bf9bf31e85b9efa9a94f1c88c5e15a6f31378     9
6842660273f70e9aa239026ba33bfe82275d6ab0d20124021b952b5bc3d07e6c     9
298528ce3160cc761e4dc37a07337ee2e0589df251d73645aae209b010210eee     9
d598d6f1fb21b45593c2afc1c2f76ae9f4cb7167156cdf93246d4192a89d8065     8
                                                                    ..
509c64729b4a325556a0e7e7af7fa12a1aa72405cf7367dd12019ddf88f036b8     1
fe49d1b2f7de7019e1773ae790669d9b5c1c7218a1849e748a87b19bc278e111     1
a1b8cb78f5fe704d7a4218f7e5438a12be71aadc68a70ecfa12a38923f762f70     1
bf41860f81866a7d6f70f5d77f6dd1d35747918fa8ddaf6a4e564f37cd1cb337     1
0bcfc1d05f2e8dc4147743a1313aa70a119b41b30d4a1f7e738a6a87d3712c31     1
Name: email_hash, Length: 152301, dtype: int64
```

In [10]:
```
df['company_hash'].value_counts()
```

Out[10]:
```
nvnv wgzohrnvzwj otqcxwto       8315
xzegojo                         5362
vbvkgz                          3461
zgn vuurxwvmrt vwwghzn          3275
wgszxkvzn                       3230
                                ...
cvrthqyq                           1
aqtvbch                            1
xnxcxnx ucn rna                    1
nyt hzxctqoxnj ge ihttzorvza       1
bvptbjnqxu td vbvkgz               1
Name: company_hash, Length: 37073, dtype: int64
```

In [11]:
```
df['job_position'].value_counts()
```

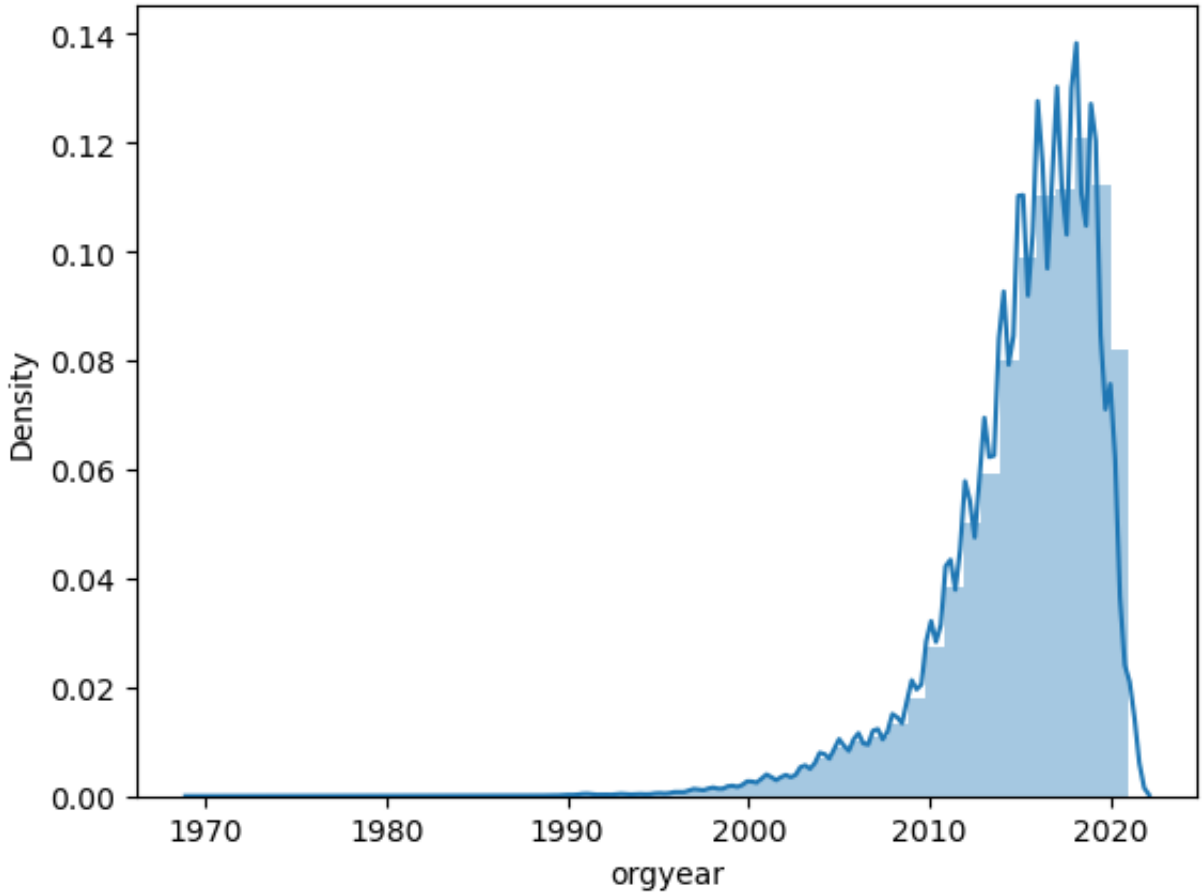
Out[11]:
```
Backend Engineer                                    43416
FullStack Engineer                                  24547
Other                                               17847
Frontend Engineer                                   10351
Engineering Leadership                               6823
                                                    ...
Compliance auditor                                     1
91                                                     1
Senior Software Development Engineer (Backend)         1
Messenger come driver                                  1
Android Application developer                          1
Name: job_position, Length: 1015, dtype: int64
```

Visual Analysis

In [12]:
```
sns.distplot(df['orgyear'])
```

```
/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619:
FutureWarning: `distplot` is a deprecated function and will be removed in
a future version. Please adapt your code to use either `displot` (a figur
e-level function with similar flexibility) or `histplot` (an axes-level f
unction for histograms).
  warnings.warn(msg, FutureWarning)
```

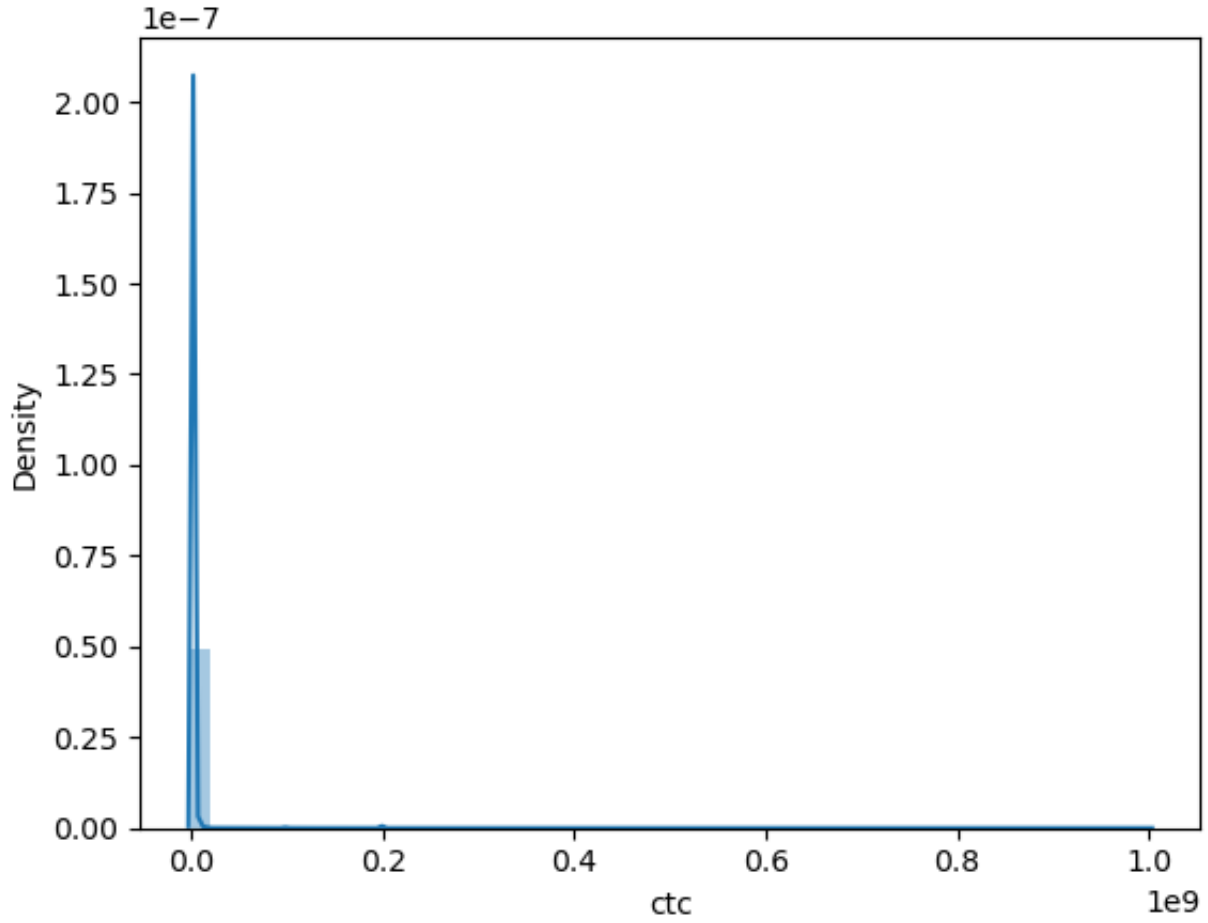
Out[12]:
```
<AxesSubplot:xlabel='orgyear', ylabel='Density'>
```



In [13]:
```
sns.distplot(df['ctc'])
```

```
/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

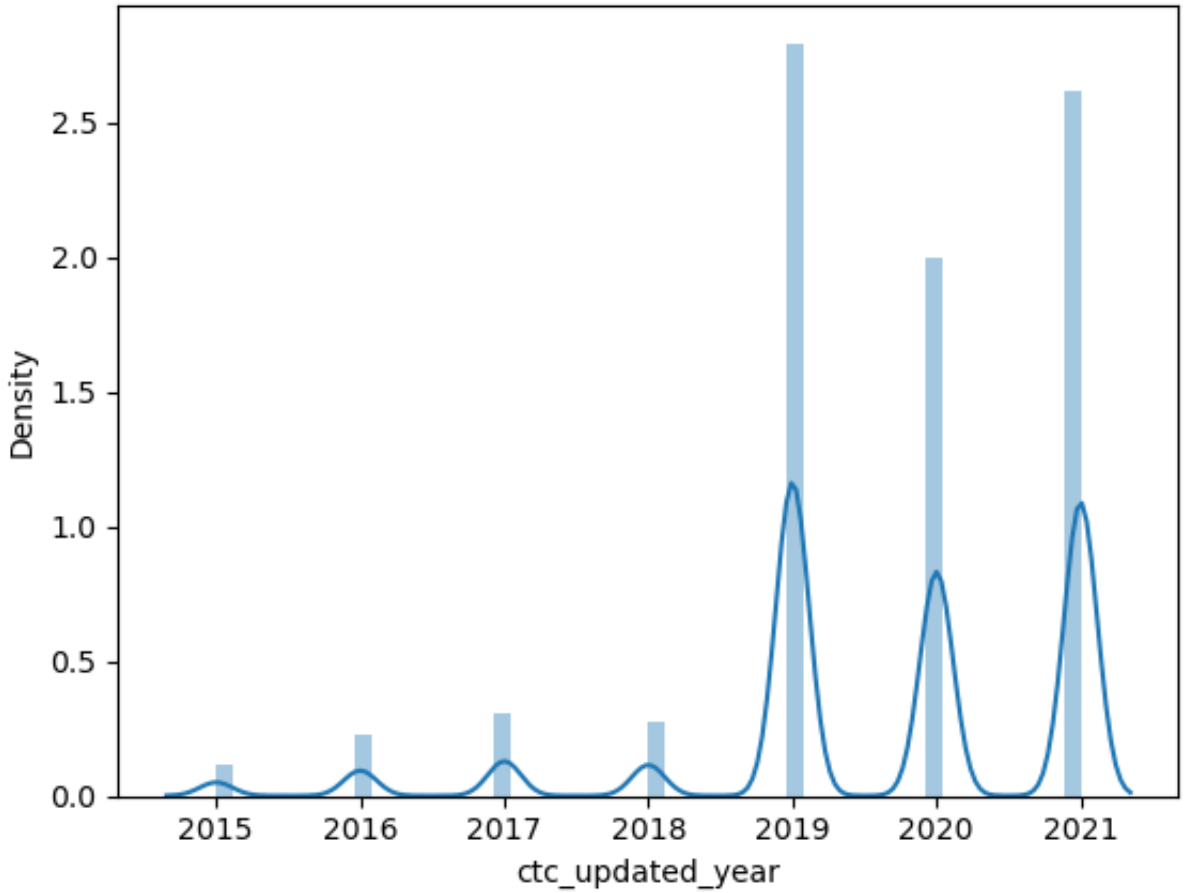
Out[13]:
```
<AxesSubplot:xlabel='ctc', ylabel='Density'>
```



In [14]:
```
sns.distplot(df['ctc_updated_year'])
```

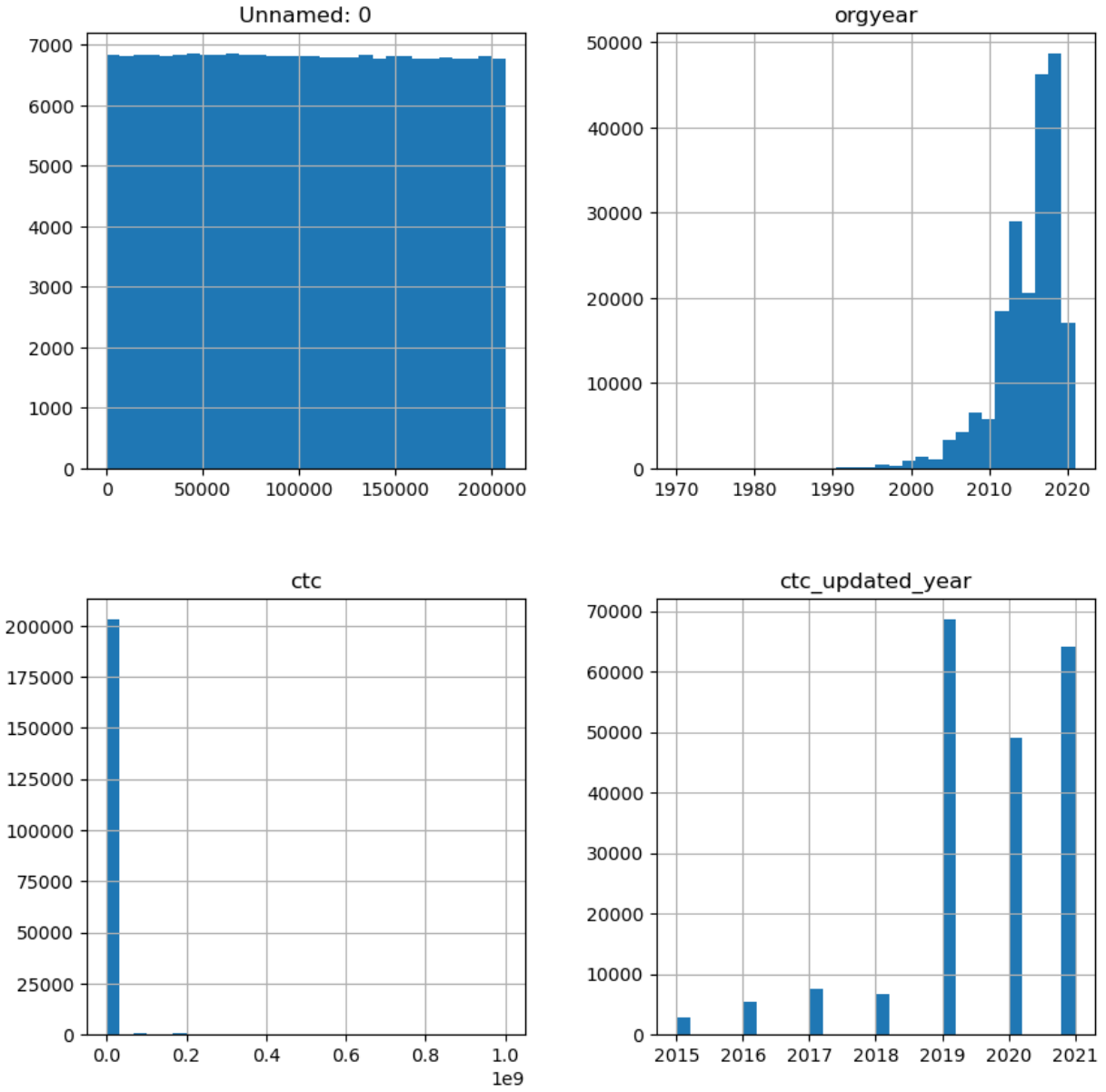
```
/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[14]:
```
<AxesSubplot:xlabel='ctc_updated_year', ylabel='Density'>
```



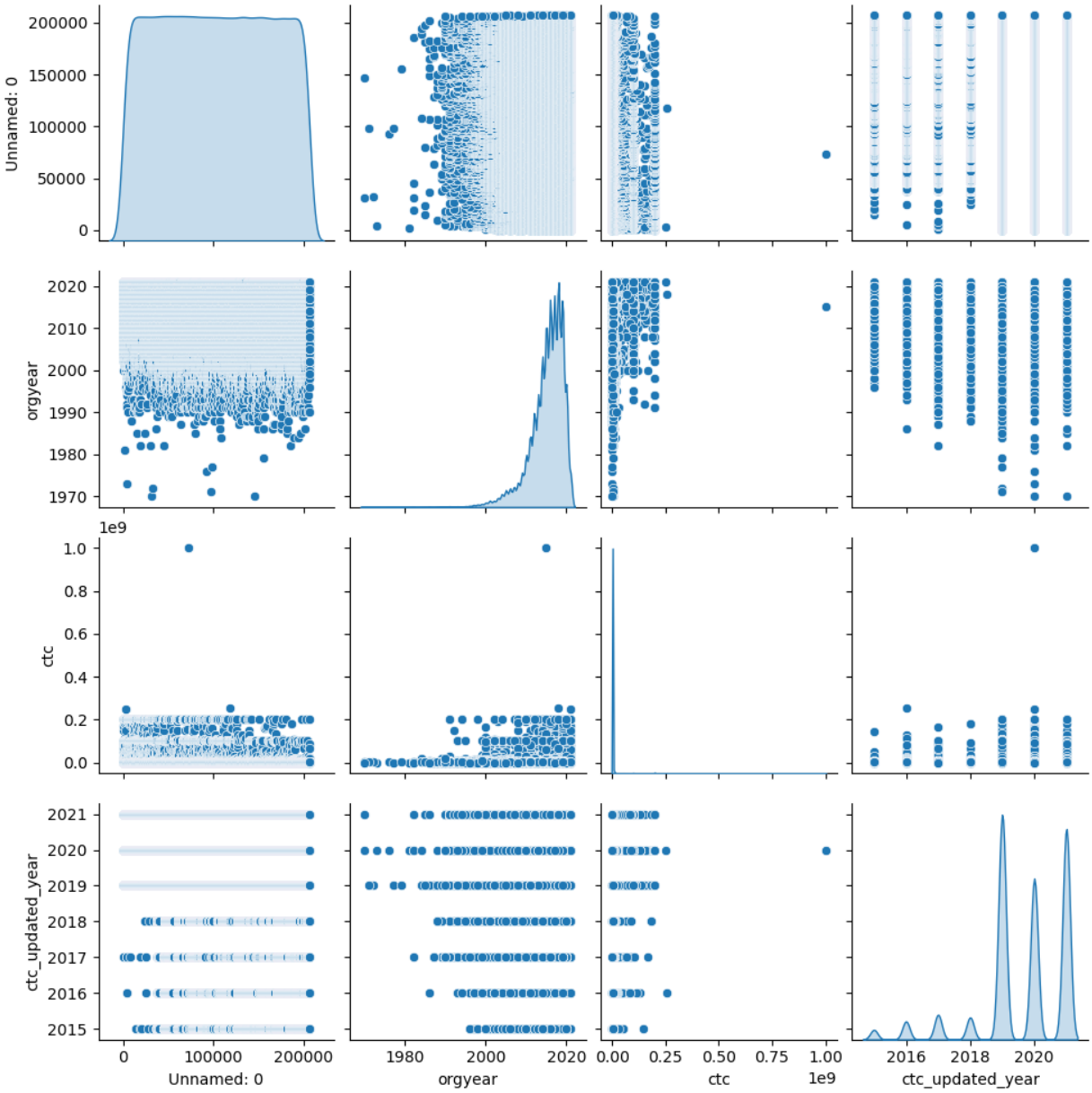
Bivariate Analysis

In [15]:
```
df.hist(figsize=(10,10),bins=30);
```



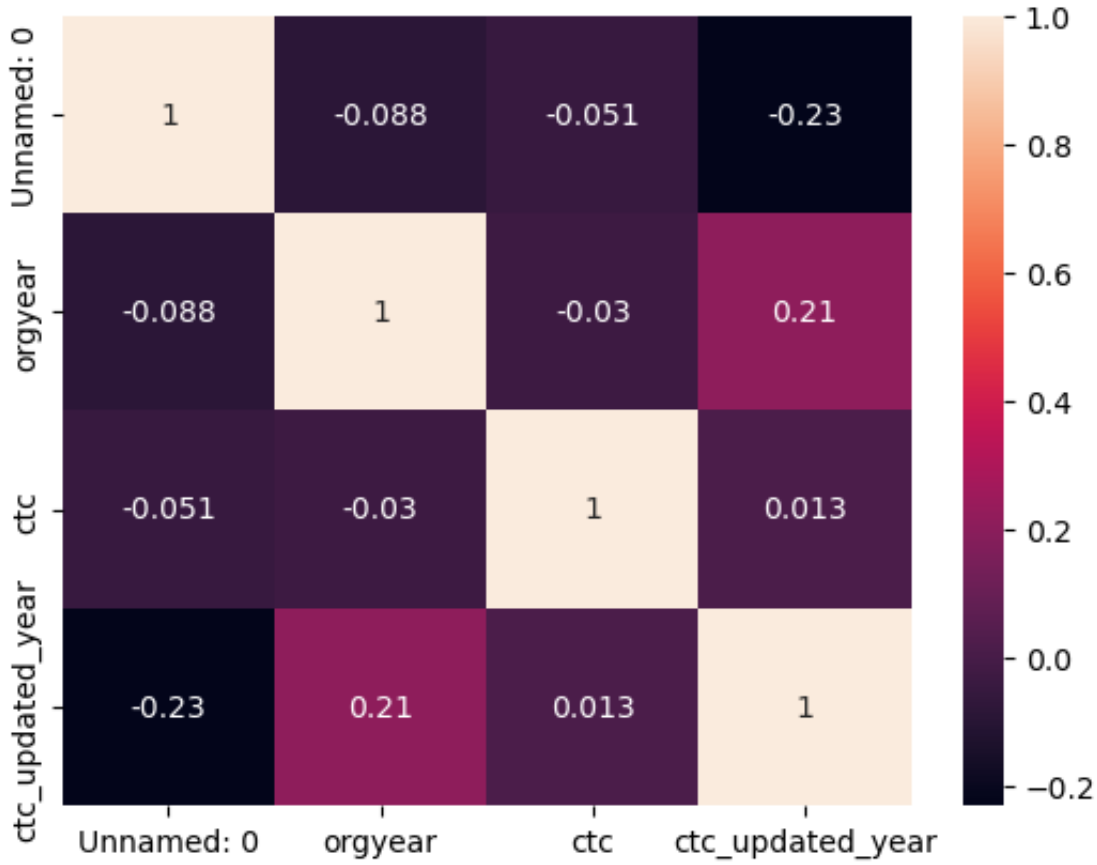
In [16]:
```
sns.pairplot(df, diag_kind='kde')
```

Out[16]:
```
<seaborn.axisgrid.PairGrid at 0x7fc924510460>
```



In [17]:
```
sns.heatmap(df.corr(),annot=True)
```

Out[17]:
```
<AxesSubplot:>
```

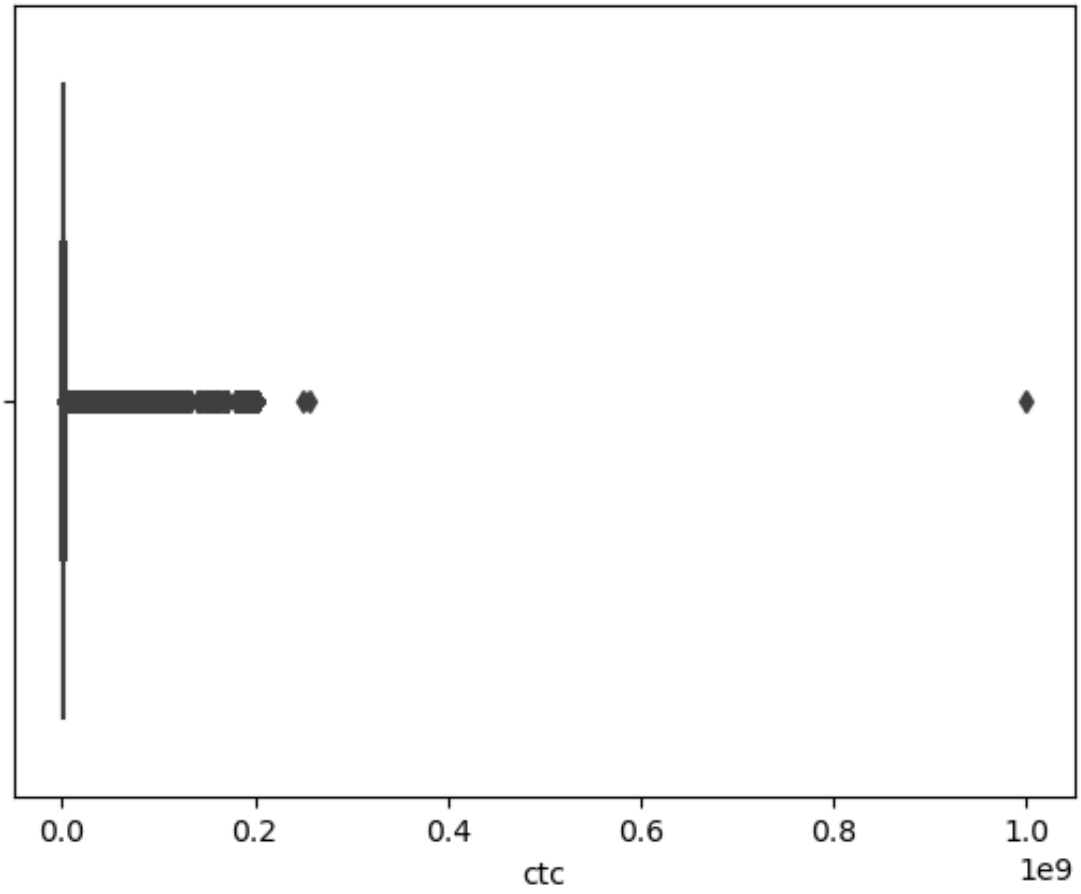


Outlier Detection

In [18]:
```
sns.boxplot(df['ctc'])
```

```
/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: Fut
ureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing othe
r arguments without an explicit keyword will result in an error or misint
erpretation.
  warnings.warn(
```

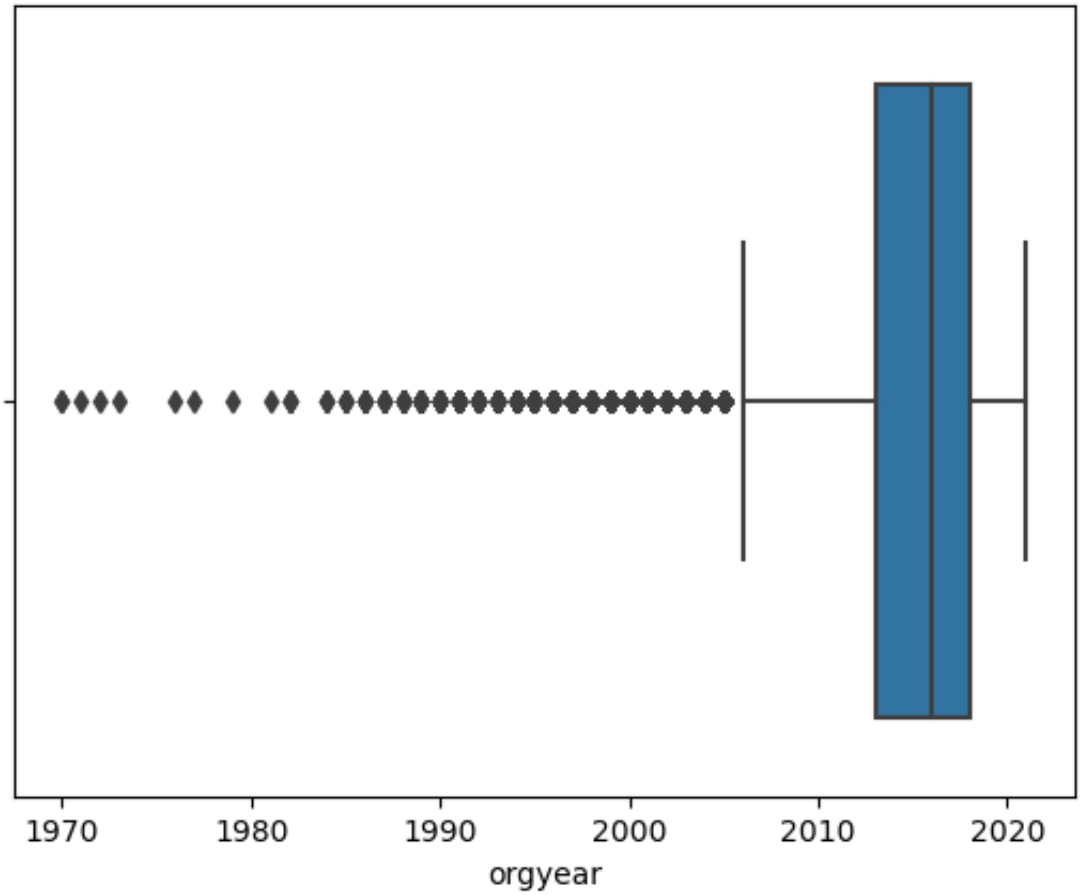
Out[18]:
```
<AxesSubplot:xlabel='ctc'>
```



In [19]:
```
sns.boxplot(df['orgyear'])
```

```
/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: Fut
ureWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing othe
r arguments without an explicit keyword will result in an error or misint
erpretation.
  warnings.warn(
```

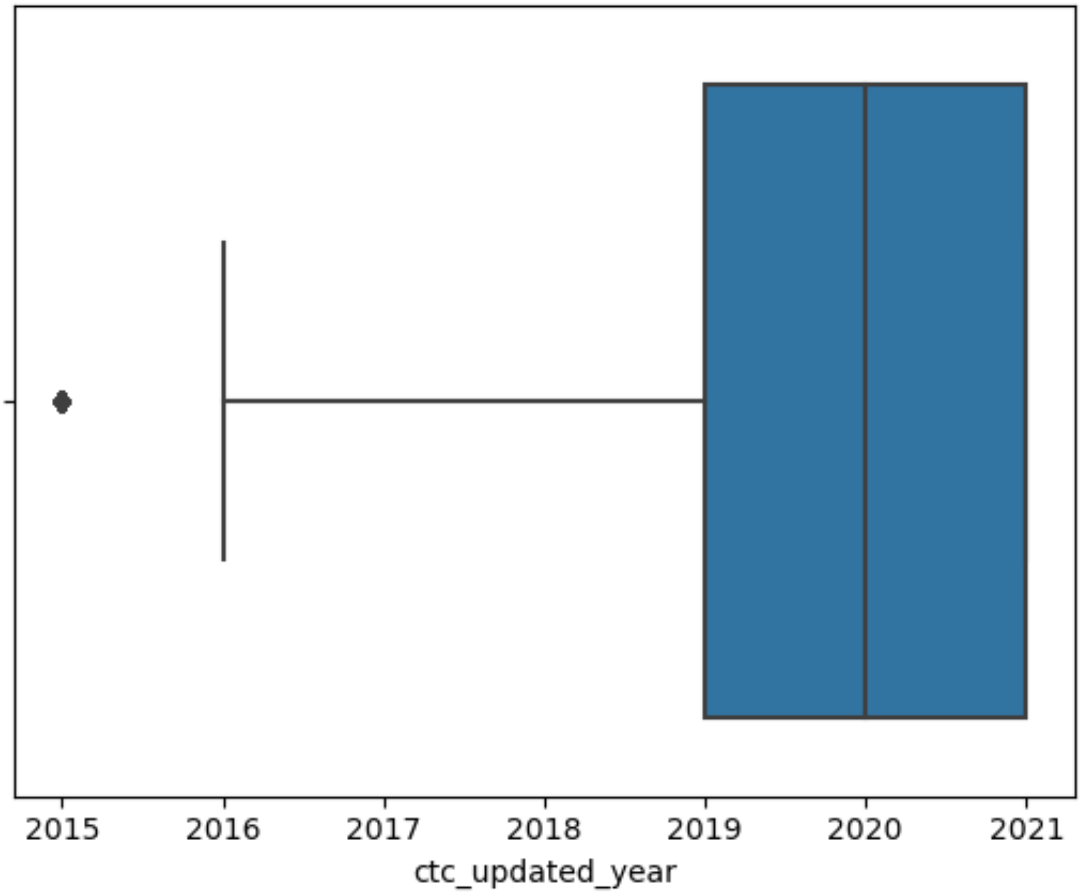
Out[19]:
```
<AxesSubplot:xlabel='orgyear'>
```



In [20]:
```
sns.boxplot(df['ctc_updated_year'])
```

```
/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[20]:
```
<AxesSubplot:xlabel='ctc_updated_year'>
```



Feature Engineering and Missing value Imputation

In [21]:
```
cat=df.describe(include=object).columns
```

In [22]:
```
from sklearn.impute import SimpleImputer
si=SimpleImputer(strategy='most_frequent')
for i in cat:
    v=df[i].values
    df[i]=si.fit_transform(v.reshape(-1,1))
```

In [23]:
```
from sklearn.impute import KNNImputer
ki=KNNImputer(n_neighbors=5)
c=df['orgyear'].values
df['orgyear']=ki.fit_transform(c.reshape(-1,1))
```

In [24]:
```
df.dropna(axis=0, inplace=True)
```

In [25]:
```
df.drop(['Unnamed: 0'], axis=1 , inplace=True)
```

In [26]:
```
df['job_position']=[re.sub('[^A-Za-z0-9 ]+', '', x) for x in df['job_posi
```

In [27]:
```
df['company_hash']=[re.sub('[^A-Za-z0-9 ]+', '', x) for x in df['company_
```

In [28]:
```
df['email_hash']=[re.sub('[^A-Za-z0-9 ]+', '', x) for x in df['email_hash
```

In [29]:
```
df.drop_duplicates(inplace=True)
```

In [30]:
```
df['years_of_experience']=2023-df['orgyear']
```

Manual Clustering

Clustering using Company

In [31]:
```
ch=df.groupby('company_hash')['ctc']
```

In [32]:
```
ch.mean()
```

Out[32]:
```
company_hash
0                                  100000.0
0000                               300000.0
01 ojztqsj                         550000.0
05mz exzytvrny uqxcvnt rxbxnta    1100000.0
1                                  100000.0
                                     ...
zyvzwt wgzohrnxzs tzsxzttqo        940000.0
zz                                 935000.0
zzb ztdnstz vacxogqj ucn rna       600000.0
zzgato                             130000.0
zzzbzb                             720000.0
Name: ctc, Length: 37073, dtype: float64
```

In [33]:
```
ch.median()
```

Out[33]:
```
company_hash
0                                  100000.0
0000                               300000.0
01 ojztqsj                         550000.0
05mz exzytvrny uqxcvnt rxbxnta    1100000.0
1                                  100000.0
                                     ...
zyvzwt wgzohrnxzs tzsxzttqo        940000.0
zz                                 935000.0
zzb ztdnstz vacxogqj ucn rna       600000.0
zzgato                             130000.0
zzzbzb                             720000.0
Name: ctc, Length: 37073, dtype: float64
```

In [34]:
```
ch.max()
```

Out[34]:
```
company_hash
0                                  100000
0000                               300000
01 ojztqsj                         830000
05mz exzytvrny uqxcvnt rxbxnta    1100000
1                                  100000
                                    ...
zyvzwt wgzohrnxzs tzsxzttqo        940000
zz                                1370000
zzb ztdnstz vacxogqj ucn rna       600000
zzgato                             130000
zzzbzb                             720000
Name: ctc, Length: 37073, dtype: int64
```

In [35]:
```
ch.min()
```

Out[35]:
```
company_hash
0                                  100000
0000                               300000
01 ojztqsj                         270000
05mz exzytvrny uqxcvnt rxbxnta    1100000
1                                  100000
                                    ...
zyvzwt wgzohrnxzs tzsxzttqo        940000
zz                                 500000
zzb ztdnstz vacxogqj ucn rna       600000
zzgato                             130000
zzzbzb                             720000
Name: ctc, Length: 37073, dtype: int64
```

In [36]:
```
ch.count()
```

Out[36]:
```
company_hash
0                                  2
0000                               1
01 ojztqsj                         2
05mz exzytvrny uqxcvnt rxbxnta     1
1                                  1
                                  ..
zyvzwt wgzohrnxzs tzsxzttqo        1
zz                                 2
zzb ztdnstz vacxogqj ucn rna       2
zzgato                             1
zzzbzb                             1
Name: ctc, Length: 37073, dtype: int64
```

Clustering using Job Position

In [37]:
```
jp=df.groupby('job_position')['ctc']
```

In [38]:
```
jp.mean()
```

Out[38]:
```
job_position
                                  650000.0
 SDE 2                           1200000.0
7                                 445000.0
7033771951                      100000000.0
737                               350000.0
                                     ...
student                          1715000.0
support escalation engineer      2000000.0
system engineer                   500000.0
system software engineer          610000.0
technology analyst                 82000.0
Name: ctc, Length: 1003, dtype: float64
```

In [39]:
```
jp.median()
```

Out[39]:
```
job_position
                                  650000.0
 SDE 2                           1200000.0
7                                 445000.0
7033771951                      100000000.0
737                               350000.0
                                     ...
student                          1715000.0
support escalation engineer      2000000.0
system engineer                   500000.0
system software engineer          610000.0
technology analyst                 82000.0
Name: ctc, Length: 1003, dtype: float64
```

In [40]:
```
jp.max()
```

Out[40]:
```
job_position
                                   700000
 SDE 2                            1200000
7                                  470000
7033771951                       100000000
737                                350000
                                      ...
student                           2400000
support escalation engineer       2000000
system engineer                    500000
system software engineer           610000
technology analyst                  82000
Name: ctc, Length: 1003, dtype: int64
```

In [41]:
```
jp.min()
```

Out[41]:
```
job_position
                                 600000
 SDE 2                          1200000
7                                420000
7033771951                     100000000
737                              350000
                                  ...
student                         1030000
support escalation engineer     2000000
system engineer                  500000
system software engineer         610000
technology analyst                82000
Name: ctc, Length: 1003, dtype: int64
```

In [42]:
```
jp.count()
```

Out[42]:
```
job_position
                                2
 SDE 2                          1
7                               2
7033771951                      1
737                             1
                               ..
student                         2
support escalation engineer     1
system engineer                 1
system software engineer        1
technology analyst              1
Name: ctc, Length: 1003, dtype: int64
```

Clustering using Years of Experience

In [43]:
```
yoe=df.groupby('years_of_experience')['ctc']
```

In [44]:
```
yoe.mean()
```

Out[44]:
```
years_of_experience
2.0     4.174413e+06
3.0     2.168744e+06
4.0     1.661865e+06
5.0     1.790543e+06
6.0     2.014689e+06
7.0     2.287597e+06
8.0     2.412350e+06
9.0     2.428622e+06
10.0    2.360870e+06
11.0    2.456325e+06
12.0    2.177922e+06
13.0    2.476629e+06
14.0    2.633404e+06
15.0    2.782173e+06
16.0    2.754114e+06
17.0    2.719412e+06
18.0    2.848160e+06
19.0    3.708461e+06
20.0    3.641774e+06
21.0    3.969004e+06
22.0    6.701403e+06
23.0    4.337689e+06
24.0    3.842169e+06
25.0    3.565805e+06
26.0    2.937248e+06
27.0    2.608812e+06
28.0    3.577553e+06
29.0    8.842375e+06
30.0    5.440609e+06
31.0    5.452064e+06
32.0    7.771547e+06
33.0    3.471514e+06
34.0    2.520909e+06
35.0    1.552200e+06
36.0    2.496067e+06
37.0    3.291250e+06
38.0    2.924000e+06
39.0    7.266667e+06
41.0    9.400000e+05
42.0    1.000000e+05
44.0    3.100000e+06
46.0    2.000000e+05
47.0    8.000000e+05
50.0    1.000000e+03
51.0    2.300000e+06
52.0    5.500000e+06
53.0    9.700000e+05
Name: ctc, dtype: float64
```

In [45]:
```
yoe.median()
```

Out[45]:
```
years_of_experience
2.0      700000.0
3.0      700000.0
4.0      680000.0
5.0      700000.0
6.0      750000.0
7.0      850000.0
8.0      949999.0
9.0     1000000.0
10.0    1200000.0
11.0    1360000.0
12.0    1500000.0
13.0    1600000.0
14.0    1680000.0
15.0    1750000.0
16.0    2000000.0
17.0    2100000.0
18.0    2450000.0
19.0    2500000.0
20.0    2500000.0
21.0    2700000.0
22.0    2600000.0
23.0    2750000.0
24.0    3000000.0
25.0    2900000.0
26.0    2800000.0
27.0    2600000.0
28.0    2200000.0
29.0    3000000.0
30.0    2500000.0
31.0    2300000.0
32.0    2000000.0
33.0    2700000.0
34.0    1500000.0
35.0    1300000.0
36.0    2650000.0
37.0    2750000.0
38.0    2620000.0
39.0    3000000.0
41.0     950000.0
42.0     100000.0
44.0    3100000.0
46.0     200000.0
47.0     800000.0
50.0       1000.0
51.0    2300000.0
52.0    5500000.0
53.0     970000.0
Name: ctc, dtype: float64
```

In [46]:
```
yoe.max()
```

Out[46]:
```
years_of_experience
2.0       250000000
3.0       200000000
4.0       200000000
5.0       255555555
6.0       200000000
7.0       200000000
8.0      1000150000
9.0       200000000
10.0      200000000
11.0      200000000
12.0      200000000
13.0      200000000
14.0      110000000
15.0      200000000
16.0      200000000
17.0      101500000
18.0      100000000
19.0      200000000
20.0      190000000
21.0      200000000
22.0      100000000
23.0      165000000
24.0      100000000
25.0      200000000
26.0       16000000
27.0       10500000
28.0      100000000
29.0      200000000
30.0      100000000
31.0      150000000
32.0      200000000
33.0       20000000
34.0       19800000
35.0        3600000
36.0        5501400
37.0        6500000
38.0        4000000
39.0       16300000
41.0        1800000
42.0         100000
44.0        3100000
46.0         200000
47.0         800000
50.0           1000
51.0        2300000
52.0        5500000
53.0        1800000
Name: ctc, dtype: int64
```

In [47]:
```
yoe.min()
```

Out[47]:
```
years_of_experience
2.0         1000
3.0           24
4.0           16
5.0          500
6.0         1000
7.0           15
8.0         1000
9.0            2
10.0           6
11.0         600
12.0        1000
13.0        1000
14.0        1000
15.0        2000
16.0        1000
17.0        1000
18.0        1000
19.0        1000
20.0        1500
21.0        1000
22.0        1000
23.0        2000
24.0       13000
25.0        1000
26.0       30000
27.0       21000
28.0       25000
29.0      135000
30.0       24000
31.0       62000
32.0       65000
33.0       90000
34.0      100000
35.0      180000
36.0       40000
37.0      500000
38.0     2500000
39.0     2500000
41.0       60000
42.0      100000
44.0     3100000
46.0      200000
47.0      800000
50.0        1000
51.0     2300000
52.0     5500000
53.0      140000
Name: ctc, dtype: int64
```

In [48]:
```
yoe.count()
```

Out[48]:
```
years_of_experience
2.0      3541
3.0     12781
4.0     22038
5.0     24068
6.0     22189
7.0     22061
8.0     19787
9.0     16115
10.0    11919
11.0    10131
12.0     7680
13.0     5550
14.0     3647
15.0     2653
16.0     2177
17.0     2015
18.0     1813
19.0     1423
20.0      996
21.0      668
22.0      701
23.0      491
24.0      338
25.0      279
26.0      232
27.0      133
28.0       94
29.0       64
30.0       69
31.0       47
32.0       75
33.0       37
34.0       22
35.0       10
36.0        6
37.0        8
38.0        5
39.0        3
41.0        4
42.0        1
44.0        1
46.0        1
47.0        1
50.0        1
51.0        1
52.0        1
53.0        2
Name: ctc, dtype: int64
```

In [49]:
```
df1=df.groupby(['company_hash','years_of_experience'])['ctc'].mean()
```

Merging the same with original dataset carefully and creating some flags showing learners with CTC greater than the Average of their Company's department having same Years of Experience - Call that flag designation with values [1,2,3]

In [50]:
```
df2=df.merge(df1,left_on=['company_hash','years_of_experience'],right_on=
```

In [51]:
```
df2['ctc_avg']=df2['ctc_avg'].round(2)
```

In [52]:
```
df2['designation']=np.where(df2['ctc']>df2['ctc_avg'],1,10)
df2['designation']=np.where(df2['ctc']<df2['ctc_avg'],3,df2['designation'
df2['designation']=np.where(df2['ctc']==df2['ctc_avg'],2,df2['designation
```

In [53]:
```
df2.drop('ctc_avg',axis=1, inplace=True)
```

Doing above analysis at Company & Job Position level. Name that flag Class with values [1,2,3]

In [54]:
```
df1=df.groupby(['company_hash','job_position'])['ctc'].mean()
```

In [55]:
```
df2=df2.merge(df1,left_on=['company_hash','job_position'],right_on=['comp
```

In [56]:
```
df2['ctc_avg']=df2['ctc_avg'].round(2)
```

In [57]:
```
df2['class']=np.where(df2['ctc']>df2['ctc_avg'],1,10)
df2['class']=np.where(df2['ctc']<df2['ctc_avg'],3,df2['class'])
df2['class']=np.where(df2['ctc']==df2['ctc_avg'],2,df2['class'])
```

In [58]:
```
df2.drop('ctc_avg',axis=1, inplace=True)
```

Repeating the same analysis at the Company level. Name that flag Tier with values [1,2,3]

In [59]:
```
df1=df.groupby(['company_hash'])['ctc'].mean()
```

In [60]:
```
df2=df2.merge(df1,left_on=['company_hash'],right_on=['company_hash'],suff
```

In [61]:
```
df2['ctc_avg']=df2['ctc_avg'].round(2)
```

In [62]:
```
df2['tier']=np.where(df2['ctc']>df2['ctc_avg'],1,10)
df2['tier']=np.where(df2['ctc']<df2['ctc_avg'],3,df2['tier'])
df2['tier']=np.where(df2['ctc']==df2['ctc_avg'],2,df2['tier'])
```

Based on the manual clustering done so far, answering few questions

Top 10 employees (earning more than most of the employees in the company) - Tier 1

In [63]:
```
df2[df2['tier']==1].sort_values(by=['ctc'],ascending=False).head(10)
```

Out[63]:

	company_hash	email_hash	orgyear
150927	obvqnuqxdwgb	5b4bed51797140db4ed52018a979db1e34cee49e27b488...	2018.0
7751	gnytqo	e7b2b159325f77e7abd5aa938371d7f9425530b36e703f...	2013.0
16317	ntwy bvyxzaqv	26946b7b12b7daa5b4f025d7c4c5c3ee0fbdb3bbe01356...	2016.0
80868	ovrtoegqwt	eb552f9d6f12d47656472a3f7c6a6625ebf3d699edb4b0...	2013.0
27284	vbvkgz	9e785d33821db67c01becc1c36f901d79d3142c1d13bd8...	2017.0
16581	ntwy bvyxzaqv	7a723f5b71698674b79bd2195c3bb58d3fcbf4ddb75a04...	2019.0
106973	zvz	0f7e7ebdae89364a5c20d32fed1886003a4375ed25531c...	2014.0
62012	ovu	a35a5abbe9fb056421bdd9aca4440acfb93e37c823564d...	2017.0
67124	ntrtutqegqbvzwt	979d02840c45c1d5790306130a0977aab05f2bd2679687...	2013.0
145249	vznxzg rvmo	634fd283565b8954513a6ad0e47cedb0fa8847923149fb...	2019.0

Top 10 employees of data science in Amazon / TCS etc earning more than their peers - Class 1

In [64]:
```
df2[(df2['class']==1)& (df2['job_position']=='Data Scientist')|(df2['job_
```

Out[64]:

	company_hash	email_hash	orgyear
148305	ihvaqvnxw xzoxsyno ucn rna	bd222ea783ee372da4e0ad60fdccec0b8f37999a032025...	2015.0
72609	mqxonrtwgzt v bvyxzaqv sqghu wgbuvzj	cda8d723438e81185d2ee8c348870a4612eea974cdb2db...	2017.0
16569	ntwy bvyxzaqv	6ad86d120e39db485331f9a0b2b1f15ce2a7bdaee778ab...	2021.0
144225	xzzgcv ogrhnxgzo	6b6dd66bae787dd4dd417e1777f8ea5a057257e9019995...	2016.0
126143	ptzgbt	4ddef8762b7585c6ee7b8c06834778f3aa00eb3be312b0...	2020.0
151738	wgqt wgbutnt	75f5b46d47310c3923e93329a62a1aa78d478803f0a685...	2016.0
14369	zgn vuurxwvmrt vwwghzn	544e75b477f8644eb71281133c62c19732547837e80e51...	2021.0
69624	zvsvqqg	15adaeb2eef9c0ee8a0f18e189bf426be390f5d1e911fd...	2021.0
2267	ztfstz ogenfvqt	3c64901d83458f3b7b8eed6fb529ee3a4c14d49339c398...	2017.0
52722	bgqsvz onvzrtj	2bede29959707d8c6f283d98319361c386baa6fa5c8028...	2021.0

Bottom 10 employees of data science in Amazon / TCS etc earning less than their peers - Class 3

In [65]:
```
df2[(df2['class']==3)& (df2['job_position']=='Data Scientist')|(df2['job_
```

Out[65]:

	company_hash	email_hash	orgyear
15788	ytfrtnn uvwpvqa tzntquqxot	8274b3188470cd1c4914e7face490111e27f239457e62d...	2018.0
104395	sggsrt	fb64af615420e06d46a1965f59068b34460fb3cbe70541...	2018.0
34709	uvjovet sqghu	3cc0c85d198d0e56a4cdefb6496333f59b97f87c293262...	2018.0
138449	vqxosrgmvr	3675f79c7e05de96ccf189c818b84b487cb1aa3f6b80e8...	2015.0
44901	nvnv wgzohrnvzwj otqcxwto	3175d03fd4618eb293d6f5a1d13d42a0c79f68e9acaaa3...	2020.0
145930	exznqhon ogrhnxgzo ucn rna	ab2dc9db23c3104f0b6b3dbd4cdd5bfb9e5829b8b7943d...	2017.0
143305	ovbohzs trtwnqg btwyvzxwo	e374eea75640881206a21894f69190138c2c0535277dc1...	2017.0
90145	onhatzn	bd9c04a574090e05b366a81cdb2f3f565d0c60fa8b1647...	2021.0
108835	bxyhu wgbbhzxwvnxgz	690f6fdab1ab7514a6a9325ebd6cfe910dbf12d46b6fde...	2018.0
35910	srgmvrtast xzntrrxstzwt ge nyxzso	8001bc017fbe95541d23f5780c3edb988b7d9b2225e39e...	2017.0

Bottom 10 employees (earning less than most of the employees in the company)- Tier 3

In [66]:
```
df2[df2['tier']==3].sort_values(by=['ctc'],ascending=False).tail(10)
```

Out[66]:

	company_hash	email_hash	orgyear
12503	zgn vuurxwvmrt vwwghzn	8d1e069a03fc437876b406b8c93bc7e07577f9836222bd...	2021.0
116155	mtznrtj ojontbo	7c8e0d8194db4deb41cbc9b3b6c428e0f9ab289436638e...	2016.0
45710	nvnv wgzohrnvzwj otqcxwto	8625d6d072e12dad0c5748ab010e1d0315736a359e2bb5...	2013.0
10048	xb v onhatzn	4eea97c023bd58395edce18538831df9a735180f88f79d...	2020.0
44446	nvnv wgzohrnvzwj otqcxwto	80ba0259f9f59034c4927cf3bd38dc9ce2eb60ff18135b...	2012.0
71341	gjg	b995d7a2ae5c6f8497762ce04dc5c04ad6ec734d70802a...	2018.0
192736	xm	b8a0bb340583936b5a7923947e9aec21add5ebc50cd60b...	2016.0
101949	xzntqcxtfmxn	23ad96d6b6f1ecf554a52f6e9b61677c7d73d8a409a143...	2013.0
101950	xzntqcxtfmxn	f2b58aeed3c074652de2cfd3c0717a5d21d6fbcf342a78...	2013.0
101946	xzntqcxtfmxn	3505b02549ebe2c95840ac6f0a35561a3b4cbe4b79cdb1...	2014.0

Top 10 employees in Amazon- X department - having 5/6/7 years of experience earning more than their peers - Tier X

In [67]:
```
df2[(df2['designation']==1)& (df2['years_of_experience']==5)|(df2['years_
```

Out[67]:

	company_hash	email_hash	orgyear
150927	obvqnuqxdwgb	5b4bed51797140db4ed52018a979db1e34cee49e27b488...	2018.0
7787	gnytqo	06c7cd6c1a50803bac81950ad450a0e78165cb7c09ddfe...	2018.0
40846	nvnv wgzohrnvzwj otqcxwto	10c67fa43291396c3f72c9ac34b99a6d9fb2c2007e6964...	2018.0
38310	nvnv wgzohrnvzwj otqcxwto	979235a69267e855c0361f670e5941138307caf43fa986...	2018.0
26901	vbvkgz	8dfe6251bd4ec533f02ddceb98b3dcebb9550ccd4ef2e6...	2018.0
38312	nvnv wgzohrnvzwj otqcxwto	97d25613e7bc3f47c87492d311f77232c105e4bc9ce642...	2018.0
17404	xzegojo	d418a571a2cf09cb8cbd9eceade19381777835f58d8f98...	2018.0
45892	nvnv wgzohrnvzwj otqcxwto	1b95e7ba0ee82100ca5a034239fa0203a1bec14280b82a...	2018.0
56196	xwxwx mvzp	2311bf023218afe93d650cac03abb7a40f7fa55c08d260...	2018.0
128470	nqvctrnqxvzsrt	3eb73d5c74bfd5f7a195d7f56e4922a46357c6ae0a8732...	2018.0

Top 10 companies (based on their CTC)

In [68]:
```
a=df2.groupby('company_hash')['ctc_avg'].mean()
```

In [69]:
```
a=a.sort_values(ascending=False).head(10)
```

In [70]:
```
a
```

Out[70]:
```
company_hash
whmxw rgsxwo uqxcvnt rxbxnta                          1.000150e+09
aveegaxr xzntqzvnxgzvr hzxctqoxnj                     2.500000e+08
uhxoovzwt xn vacxogqj vza exzvzwxvr otqcxwto rru      2.000000e+08
wvquvzntq                                             2.000000e+08
durgfxk ogrhnxgzo                                     2.000000e+08
qn                                                    2.000000e+08
ztbyvzo ogrhnxgzo ucn rna                             2.000000e+08
ihvrxnvo srgmvr rru                                   2.000000e+08
bvtonqg wxcxr vza xzntqxgqo                           2.000000e+08
egd z wrgha ntwyzgrgsxto                              2.000000e+08
Name: ctc_avg, dtype: float64
```

Top 2 positions in every company (based on their CTC)

In [71]:
```
df_grouped = df.groupby(['job_position']).apply(lambda x: x.sort_values('
```

In [72]:
```
df_grouped.groupby('company_hash')['ctc'].nlargest(2)
```

Out[72]:
```
company_hash                job_position
0                           Backend Engineer    2940      100000
                            Other               16824     100000
0000                        Other               197540    300000
01 ojztqsj                  Frontend Engineer   55241     830000
                            Android Engineer    74429     270000
                                                          ...
zz                          Backend Engineer    14670     500000
zzb ztdnstz vacxogqj ucn rna Backend Engineer   146629    600000
                            FullStack Engineer  72983     600000
zzgato                      Backend Engineer    117023    130000
zzzbzb                      Other               15838     720000
Name: ctc, Length: 50181, dtype: int64
```

Data processing for Unsupervised clustering - Label encoding/ One- hot encoding, Standardization of data

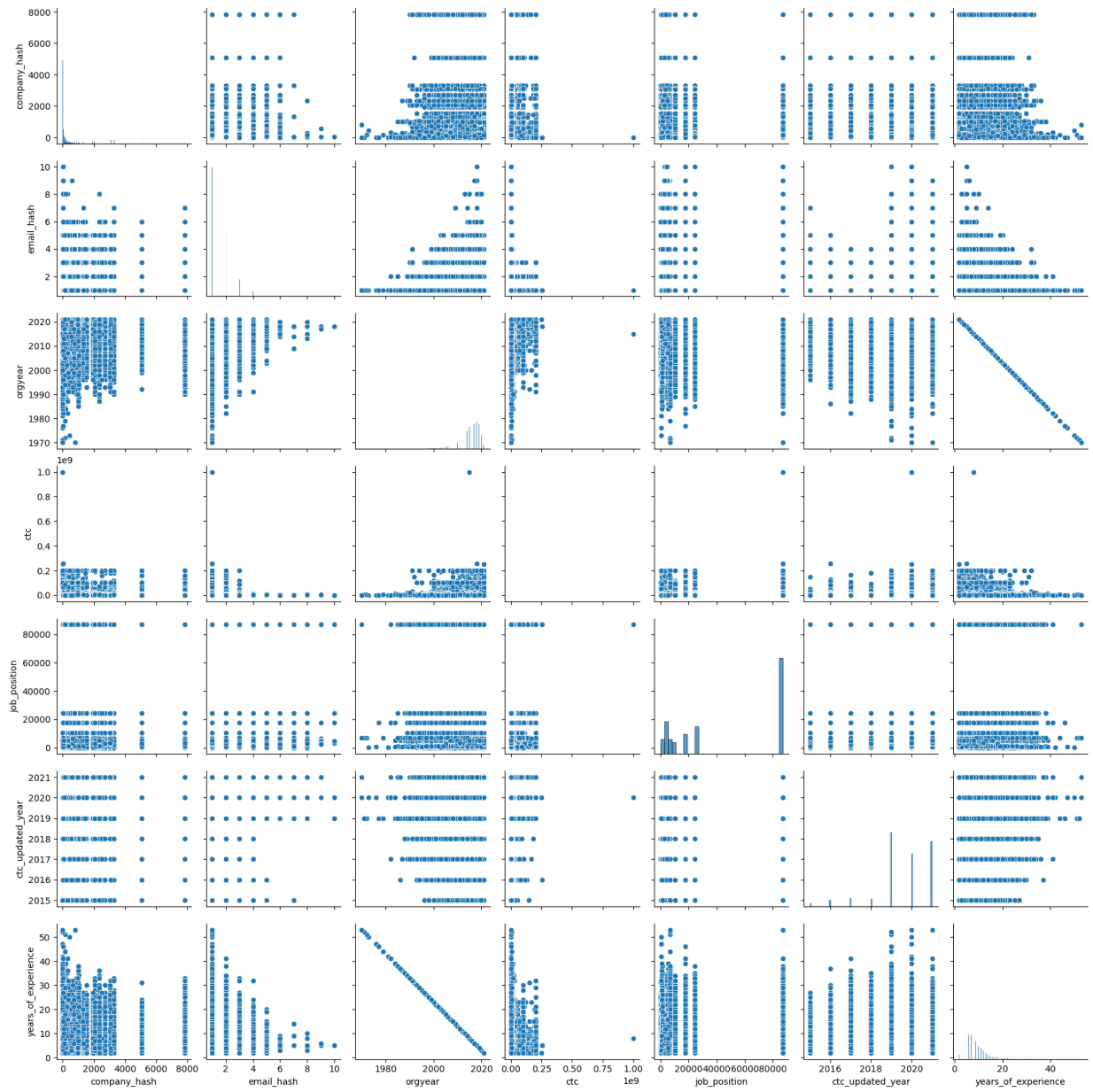
In [73]:
```
from category_encoders import CountEncoder
ce=CountEncoder()
df['email_hash']=ce.fit_transform(df['email_hash'])
```

In [74]:
```
from category_encoders import CountEncoder
ce=CountEncoder()
df['company_hash']=ce.fit_transform(df['company_hash'])
```

In [75]:
```
from category_encoders import CountEncoder
ce=CountEncoder()
df['job_position']=ce.fit_transform(df['job_position'])
```

In [76]:
```
sns.pairplot(df)
```

Out[76]:
```
<seaborn.axisgrid.PairGrid at 0x7fc8d0c3a9d0>
```



In [77]:
```
num_cols=df._get_numeric_data().columns
```

In [78]:
```
'''def outliers(data,feature):
  q1=data[feature].quantile(0.05)
  q3=data[feature].quantile(0.95)
  iqr=q3-q1
  ul=q3+1.5*iqr
  ll=q1-1.5*iqr
  return ul,ll'''
```

Out[78]:
```
'def outliers(data,feature):\n  q1=data[feature].quantile(0.05)\n  q3=dat
a[feature].quantile(0.95)\n  iqr=q3-q1\n  ul=q3+1.5*iqr\n  ll=q1-1.5*iqr\
n  return ul,ll'
```

In [79]:
```
'''for i in num_cols:
  ul,ll=outliers(df,i)
  df=df[(df[i]<ul) & (df[i]>ll)]'''
```

Out[79]:
```
'for i in num_cols:\n  ul,ll=outliers(df,i)\n  df=df[(df[i]<ul) & (df[i]>
ll)]'
```

Outlier Removal using Local Outlier Factor

In [80]:
```
from sklearn.neighbors import LocalOutlierFactor
clf = LocalOutlierFactor(n_neighbors=25, contamination=0.05)
is_o=clf.fit_predict(df)
```

In [81]:
```
df['lof']=is_o
df
```

Out[81]:

	company_hash	email_hash	orgyear	ctc	job_position	ctc_updated_year
0	9	2	2016.0	1100000	17847	2020.0
1	401	2	2018.0	449999	24541	2019.0
2	1	2	2015.0	2000000	86961	2020.0
3	68	1	2017.0	700000	86961	2019.0
4	6	2	2017.0	1400000	24541	2019.0
...	...	...	...	...	...	...
205838	17	2	2008.0	220000	86961	2019.0
205839	105	1	2017.0	500000	86961	2020.0
205840	159	1	2021.0	700000	86961	2021.0
205841	968	1	2019.0	5100000	86961	2019.0
205842	533	1	2014.0	1240000	86961	2016.0

195879 rows × 8 columns

In [82]:
```
df=df[df['lof']==1]
```

In [83]:
```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
X=sc.fit_transform(df)
```

Checking clustering tendency

In [84]:
```
from pyclustertend import hopkins
hopkins(X, X.shape[0])
```

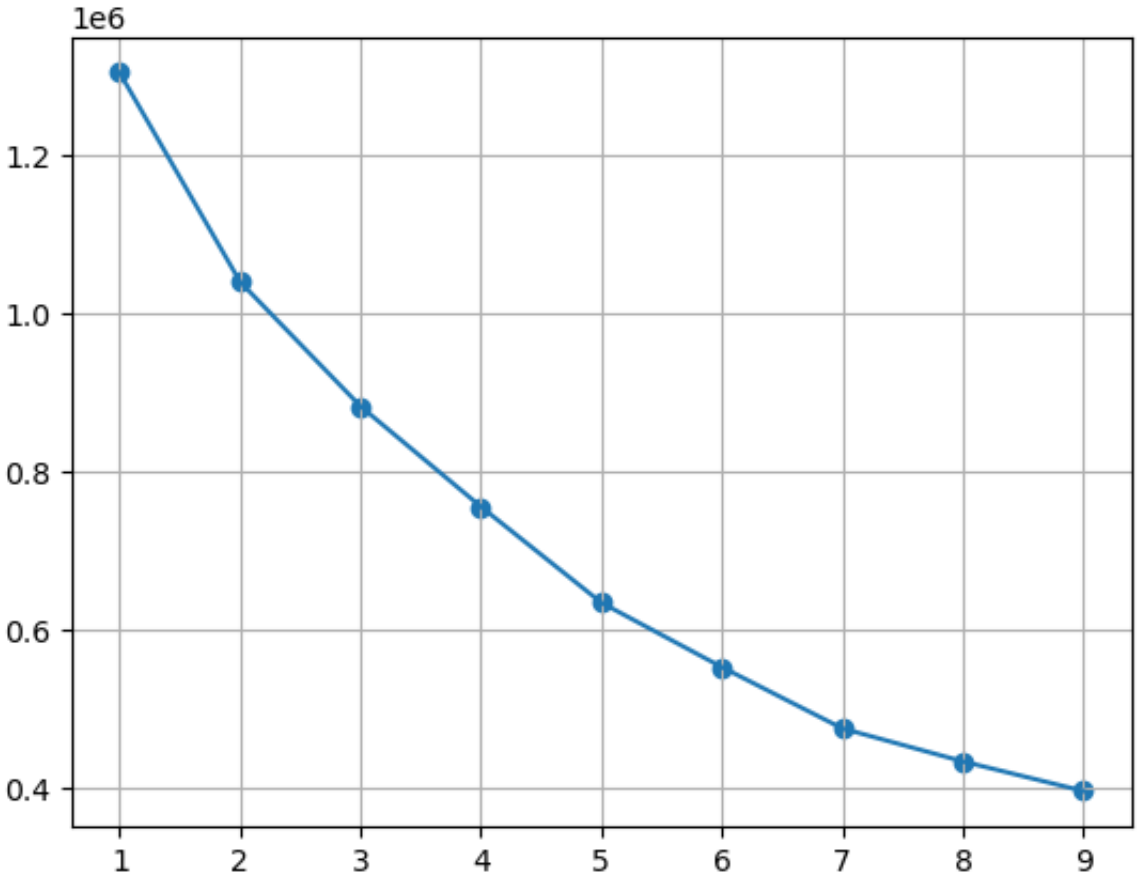
Out[84]:
```
0.003810942596027236
```

Elbow Method

In [85]:
```
num_clusters=np.arange(1,10)
```

In [86]:
```
from sklearn.cluster import KMeans
inertia=[]
for i in num_clusters:
    km=KMeans(n_clusters=i)
    km.fit(X)
    inertia.append(km.inertia_)
```

In [87]:
```
plt.plot(num_clusters,inertia)
plt.scatter(num_clusters,inertia)
plt.grid()
plt.show()
```



K-means clustering

In [88]:
```
km=KMeans(n_clusters=2)
km.fit(X)
```

Out[88]:
```
KMeans(n_clusters=2)
```

In [89]:
```
pred=km.fit_predict(X)
```

In [90]:
```
clusters = pd.DataFrame(X, columns=df.columns)
clusters['label'] = km.labels_
clusters.head(3)
```

Out[90]:

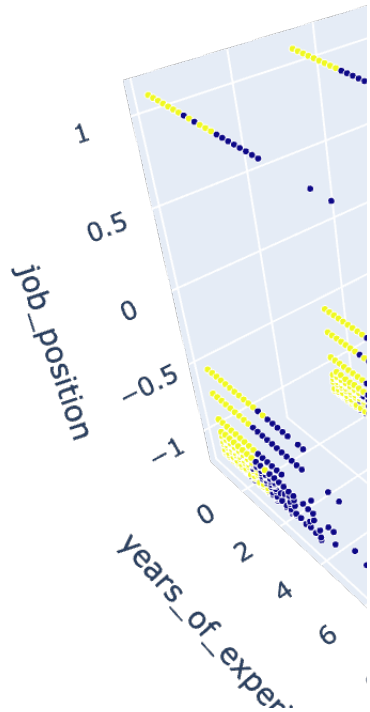
	company_hash	email_hash	orgyear	ctc	job_position	ctc_updated_year	ye
0	-0.484794	0.561983	0.224445	-0.099497	-0.710530	0.303744	
1	-0.264028	0.561983	0.698980	-0.157010	-0.534433	-0.451547	
2	-0.489299	0.561983	-0.012822	-0.019863	1.107632	0.303744	

In [91]:
```
np.unique(km.labels_)
```

Out[91]:
```
array([0, 1], dtype=int32)
```

In [92]:
```
import plotly.express as px

fig = px.scatter_3d(clusters, x='company_hash', y='years_of_experience',
fig.update_traces(marker=dict(size=2,
                              line=dict(width=2,
                                        color='DarkSlateGrey')),
                  selector=dict(mode='markers'))
fig.show()
```



Hierarchical clustering

In [93]:
```
q=pd.DataFrame(X)
sam=q.sample(20000).values
```

In [94]:
```
from sklearn.cluster import AgglomerativeClustering
hc = AgglomerativeClustering(n_clusters=3, affinity = 'euclidean', linkag
y_pred = hc.fit_predict(sam)
```

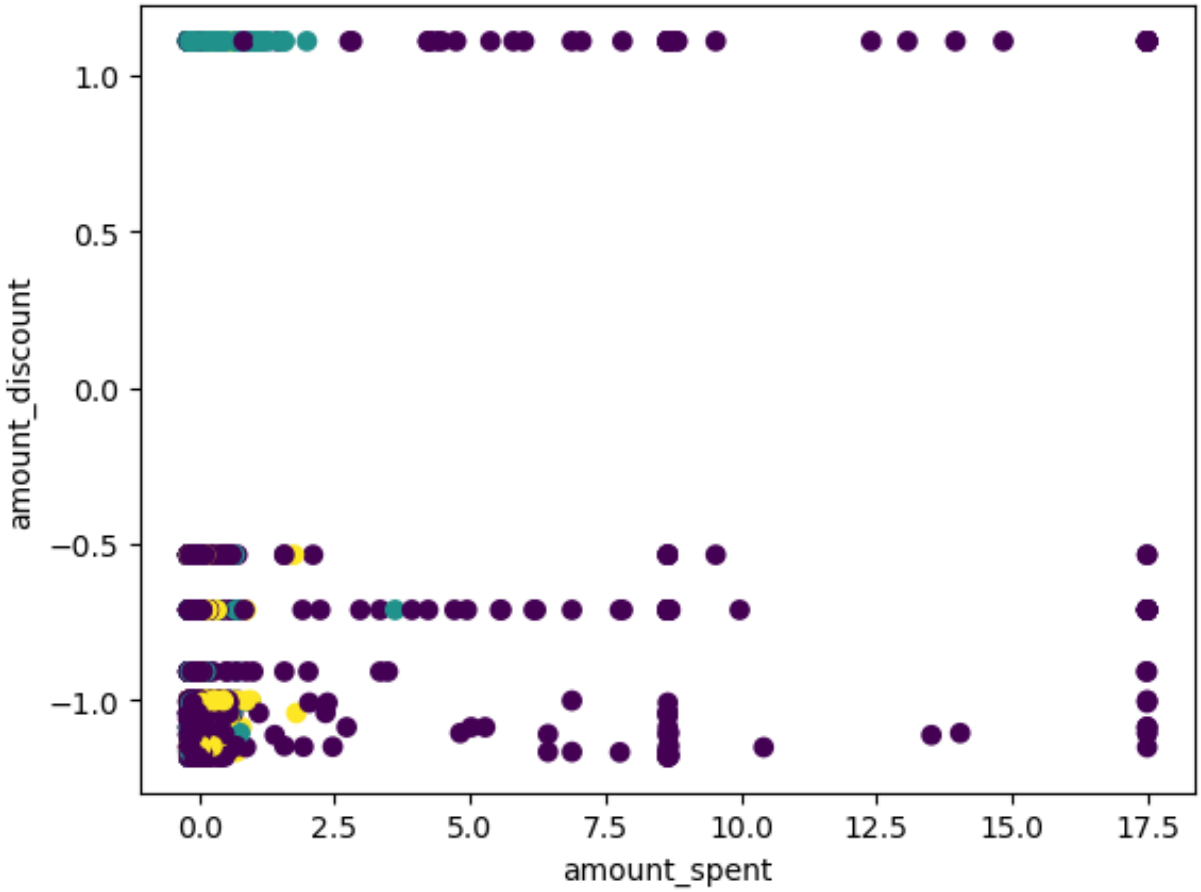
In [95]:
```
clusters = pd.DataFrame(sam, columns=df.columns)
clusters['label'] = hc.labels_
clusters.head(3)
```

Out[95]:

	company_hash	email_hash	orgyear	ctc	job_position	ctc_updated_year	ye
0	-0.380042	0.561983	-0.012822	-0.099497	1.107632	1.059034	
1	1.366938	-0.684951	0.936248	-0.072952	-0.534433	0.303744	
2	-0.484794	-0.684951	-0.487357	0.006682	1.107632	-3.472708	

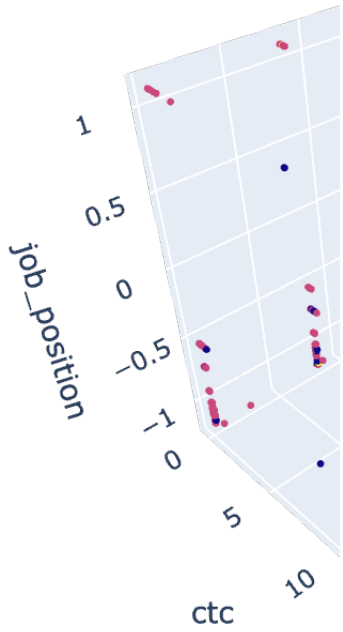
In [96]:
```
def viz_clusters(hc):
    plt.scatter(clusters['ctc'], clusters['job_position'], c=clusters['la
    plt.xlabel('amount_spent')
    plt.ylabel('amount_discount')
    plt.show()

viz_clusters(hc)
```



In [97]:
```
import plotly.express as px

fig = px.scatter_3d(clusters, x='company_hash', y='ctc', z='job_position'
fig.update_traces(marker=dict(size=2), selector=dict(mode='markers'))
fig.show()
```



In [ ]:

In [ ]: