Scaler Case Study

Problem Statement

Scaler is an online tech-versity offering intensive computer science & Data Science courses through live classes delivered by tech leaders and subject matter experts. The meticulously structured program enhances the skills of software professionals by offering a modern curriculum with exposure to the latest technologies. It is a product by InterviewBit.

You are working as a data scientist with the analytics vertical of Scaler, focused on profiling the best companies and job positions to work for from the Scaler database. You are provided with the information for a segment of learners and tasked to cluster them on the basis of their job profile, company, and other features. Ideally, these clusters should have similar characteristics.

Data Dictionary:

- ‘Unnamed 0’- Index of the dataset
- Email_hash- Anonymised Personal Identifiable Information (PII)
- Company_hash- Current employer of the learner
- orgyear- Employment start date
- CTC- Current CTC
- Job_position- Job profile in the company
- CTC_updated_year: Year in which CTC got updated (Yearly increments, Promotions)

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import re
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.mixture import GaussianMixture
```

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

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

Out[3]:

	Unnamed: 0	company_hash	email_hash	orgyear
0	0	atrgxnnt xzaxv	6de0a4417d18ab14334c3f43397fc13b30c35149d70c05...	2016.0
1	1	qtrxvzwt xzegwgbb rxbxnta	b0aaf1ac138b53cb6e039ba2c3d6604a250d02d5145c10...	2018.0
2	2	ojzwnvwnxw vx	4860c670bcd48fb96c02a4b0ae3608ae6fdd98176112e9...	2015.0
3	3	ngpgutaxv	effdede7a2e7c2af664c8a31d9346385016128d66bbc58...	2017.0
4	4	qxen sqghu	6ff54e709262f55cb999a1c1db8436cb2055d8f79ab520...	2017.0

Statistical Summary

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

Out[4]: (205843, 7)

In [5]:
```
df.drop(columns="Unnamed: 0", inplace = True)
```

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

Out[6]:

	company_hash	email_hash	orgyear	ctc	j
0	atrgxnnt xzaxv	6de0a4417d18ab14334c3f43397fc13b30c35149d70c05...	2016.0	1100000	
1	qtrxvzwt xzegwgbb rxbxnta	b0aaf1ac138b53cb6e039ba2c3d6604a250d02d5145c10...	2018.0	449999	
2	ojzwnvwnxw vx	4860c670bcd48fb96c02a4b0ae3608ae6fdd98176112e9...	2015.0	2000000	
3	ngpgutaxv	effdede7a2e7c2af664c8a31d9346385016128d66bbc58...	2017.0	700000	
4	qxen sqghu	6ff54e709262f55cb999a1c1db8436cb2055d8f79ab520...	2017.0	1400000	

In [7]:
```
df["company_hash"].isnull().sum()
```

Out[7]: 44

In [8]:
```
df.shape
```

Out[8]: (205843, 6)

In [9]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205843 entries, 0 to 205842
Data columns (total 6 columns):
 #   Column            Non-Null Count   Dtype  
---  ------            --------------   -----  
 0   company_hash      205799 non-null  object 
 1   email_hash        205843 non-null  object 
 2   orgyear           205757 non-null  float64
 3   ctc               205843 non-null  int64  
 4   job_position      153281 non-null  object 
 5   ctc_updated_year  205843 non-null  float64
dtypes: float64(2), int64(1), object(3)
memory usage: 9.4+ MB
```

In [10]:
```
df.describe()
```

Out[10]:

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

Exploratory Data Analysis

In [11]:
```
df.nunique()
```

Out[11]:
```
company_hash         37299
email_hash          153443
orgyear                 77
ctc                   3360
job_position          1017
ctc_updated_year         7
dtype: int64
```

In [12]:

```
for i in df.columns:
    print(df[i].value_counts(normalize=True)*100)
    print("\n\n")
```

```
nvnv wgzohrnvzwj otqcxwto        4.051040
xzegojo                          2.614687
vbvkgz                           1.691456
zgn vuurxwvmrt vwwghzn           1.657442
wgszxkvzn                        1.574352
                                   ...
onvqmhwpo                        0.000486
bvsxw ogenfvqt uqxcvnt rxbxnta   0.000486
agsbv ojontbo                    0.000486
vnnhzt xzegwgb                   0.000486
bvptbjnqxu td vbvkgz             0.000486
Name: company_hash, Length: 37299, dtype: float64



bbace3cc586400bbc65765bc6a16b77d8913836cfc98b77c05488f02f5714a4b
0.004858
6842660273f70e9aa239026ba33bfe82275d6ab0d20124021b952b5bc3d07e6c
0.004372
298528ce3160cc761e4dc37a07337ee2e0589df251d73645aae209b010210eee
0.004372
3e5e49daa5527a6d5a33599b238bf9bf31e85b9efa9a94f1c88c5e15a6f31378
0.004372
b4d5afa09bec8689017d8b29701b80d664ca37b83cb883376b2e95191320da66
0.003886

...
bb2fe5e655ada7f7b7ac4a614db0b9c560e796bdfcaa4e5367e69eedfea93876
0.000486
d6cdef97e759dbf1b7522babccbbbd5f164a75d1b4139e02c945958720f1ed79
0.000486
700d1190c17aaa3f2dd9070e47a4c042ecd9205333545dbfaee0f85644d00306
0.000486
c2a1c9e4b9f4e1ed7d889ee4560102c1e2235b2c1a0e59cea95a6fe55c658407
0.000486
0bcfc1d05f2e8dc4147743a1313aa70a119b41b30d4a1f7e738a6a87d3712c31
0.000486
Name: email_hash, Length: 153443, dtype: float64



2018.0    12.274674
2019.0    11.385761
2017.0    11.294391
2016.0    11.199133
2015.0    10.016670
            ...
2107.0     0.000486
1972.0     0.000486
```

```
2101.0     0.000486
208.0      0.000486
200.0      0.000486
Name: orgyear, Length: 77, dtype: float64
```

```
600000     3.804842
400000     3.691163
1000000    3.682904
500000     3.518215
800000     3.280170
             ...
1916000    0.000486
5340000    0.000486
2305000    0.000486
4225000    0.000486
3327000    0.000486
Name: ctc, Length: 3360, dtype: float64
```

```
Backend Engineer                 28.414481
FullStack Engineer               16.125286
Other                            11.789459
Frontend Engineer                 6.796015
Engineering Leadership            4.481964
                                    ...
ayS                               0.000652
Principal Product Engineer        0.000652
Senior Director of Engineering    0.000652
Seller Support Associate          0.000652
Android Application developer     0.000652
Name: job_position, Length: 1017, dtype: float64
```

```
2019.0    33.369121
2021.0    31.565805
2020.0    24.020248
2017.0     3.673188
2018.0     3.277255
2016.0     2.672425
2015.0     1.421958
Name: ctc_updated_year, dtype: float64
```

In [13]:
```
df.isnull().sum()/len(df)*100
```

Out[13]:
```
company_hash         0.021376
email_hash           0.000000
orgyear              0.041779
ctc                  0.000000
job_position        25.534995
ctc_updated_year     0.000000
dtype: float64
```

Graphical and Non-Graphical Analysis

In [14]:
```
object_list = []
numerical_list = []

for i in df.columns:
    if df[i].dtype == "O":
        object_list.append(i)
    else:
        numerical_list.append(i)
```

In [15]:
```
object_list
```

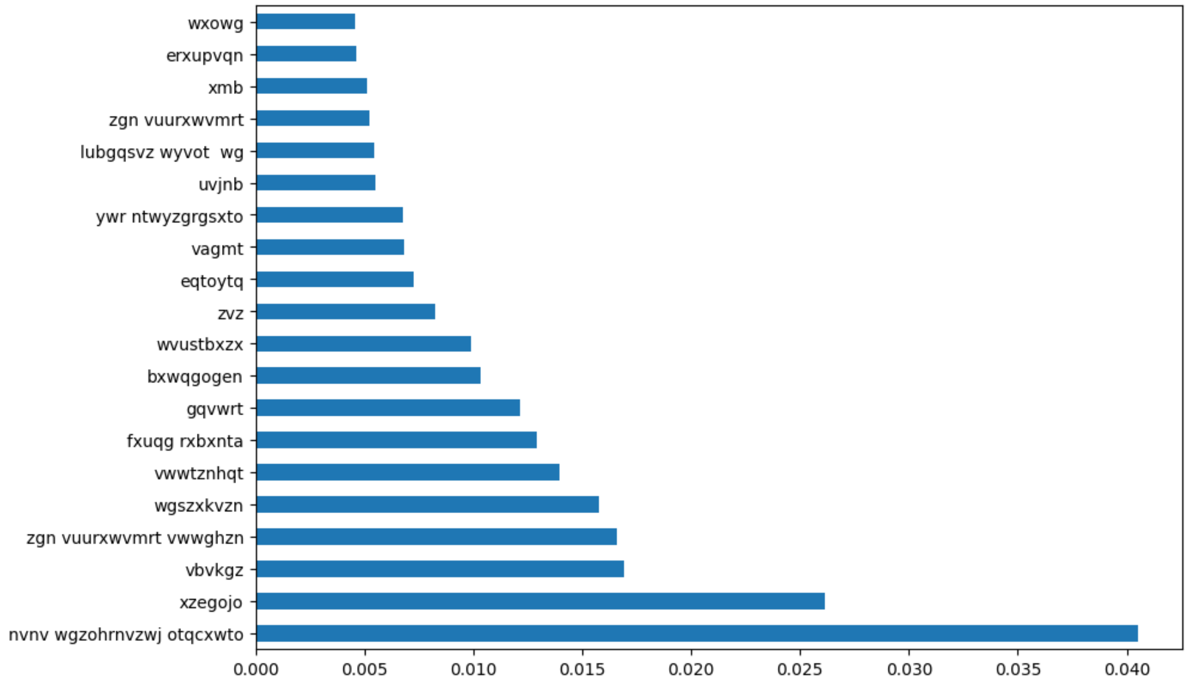
Out[15]:
```
['company_hash', 'email_hash', 'job_position']
```

In [16]:
```
numerical_list
```

Out[16]:
```
['orgyear', 'ctc', 'ctc_updated_year']
```

In [17]:
```
plt.figure(figsize=(10,7))
df["company_hash"].value_counts(normalize= True).sort_values(ascend
```

Out[17]: <Axes: >



In [ ]:
```
# Above are the top occuring Companies in this which consitutes of
```

In [18]:
```
df["company_hash"].value_counts(normalize= True).sort_values(ascend
```

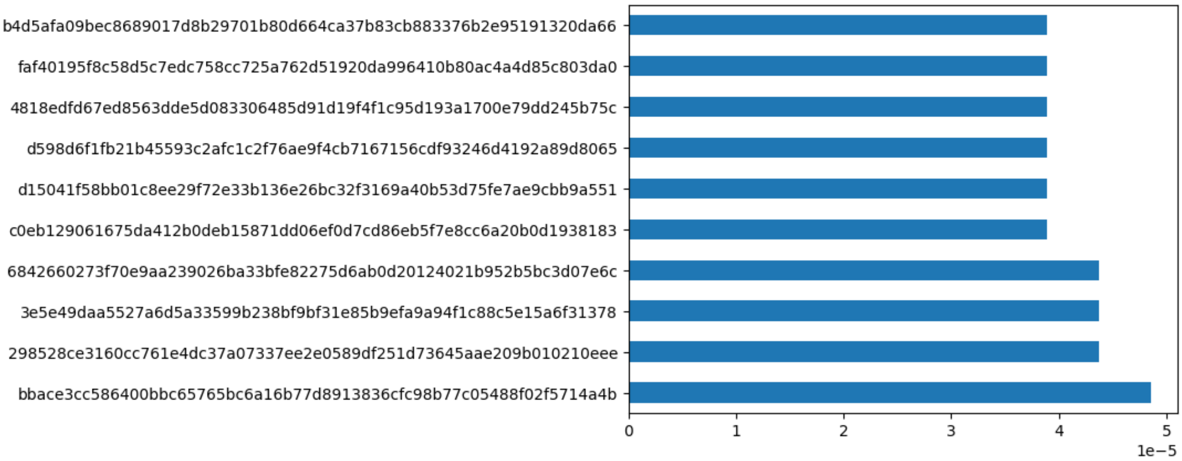
Out[18]: 23.474846816554017

In [19]:
```
df["company_hash"].value_counts(normalize= True).sort_values(ascend
```

Out[19]:
```
nvnv wgzohrnvzwj otqcxwto    0.040510
xzegojo                      0.026147
vbvkgz                       0.016915
zgn vuurxwvmrt vwwghzn       0.016574
wgszxkvzn                    0.015744
Name: company_hash, dtype: float64
```

Type *Markdown* and LaTeX: α^2

In [20]:
```
df["email_hash"].value_counts(normalize= True).sort_values(ascendin
plt.show()
```



In [21]:
```
df["email_hash"].value_counts(normalize= True).sort_values(ascendin
```

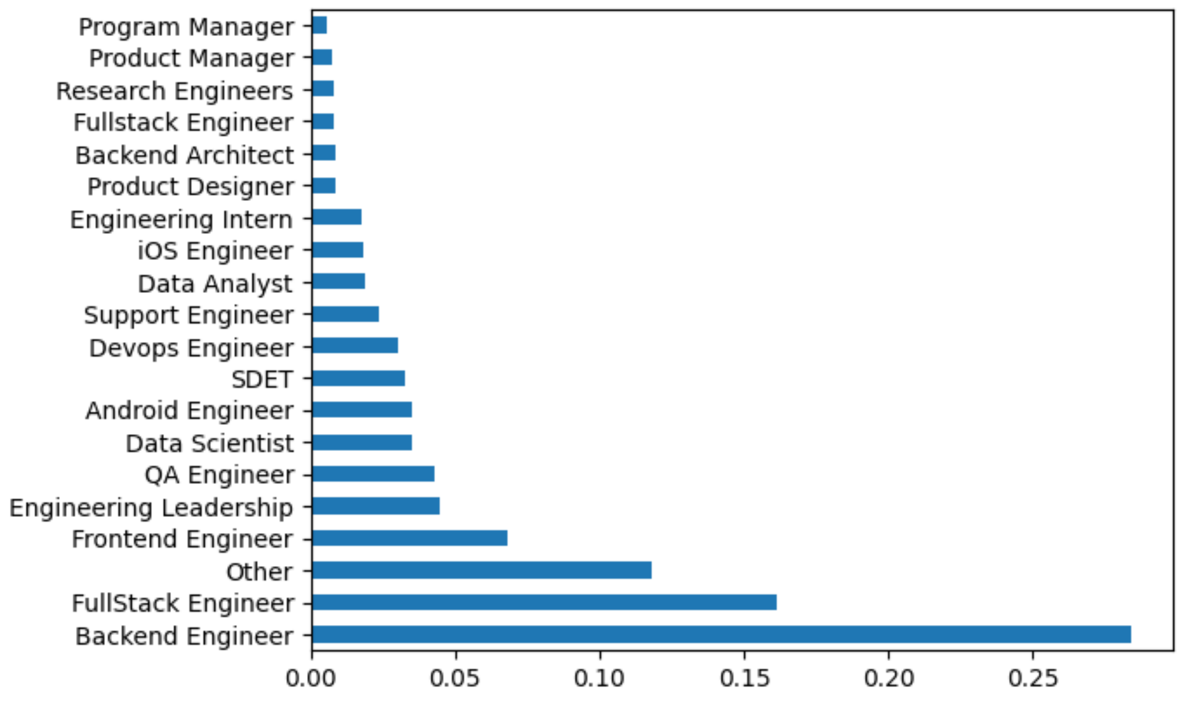
Out[21]:
```
0.0007530010736337888
```

In [22]:
```
df["email_hash"].value_counts(normalize= True).head(5)
```

Out[22]:
```
bbace3cc586400bbc65765bc6a16b77d8913836cfc98b77c05488f02f5714a4b
0.000049
6842660273f70e9aa239026ba33bfe82275d6ab0d20124021b952b5bc3d07e6c
0.000044
298528ce3160cc761e4dc37a07337ee2e0589df251d73645aae209b010210eee
0.000044
3e5e49daa5527a6d5a33599b238bf9bf31e85b9efa9a94f1c88c5e15a6f31378
0.000044
b4d5afa09bec8689017d8b29701b80d664ca37b83cb883376b2e95191320da66
0.000039
Name: email_hash, dtype: float64
```

Type *Markdown* and LaTeX: α^2

In [23]:
```
df["job_position"].value_counts(normalize= True).sort_values(ascend
plt.show()
```



In [24]:
```
df["job_position"].value_counts(normalize= True).sort_values(ascend
```

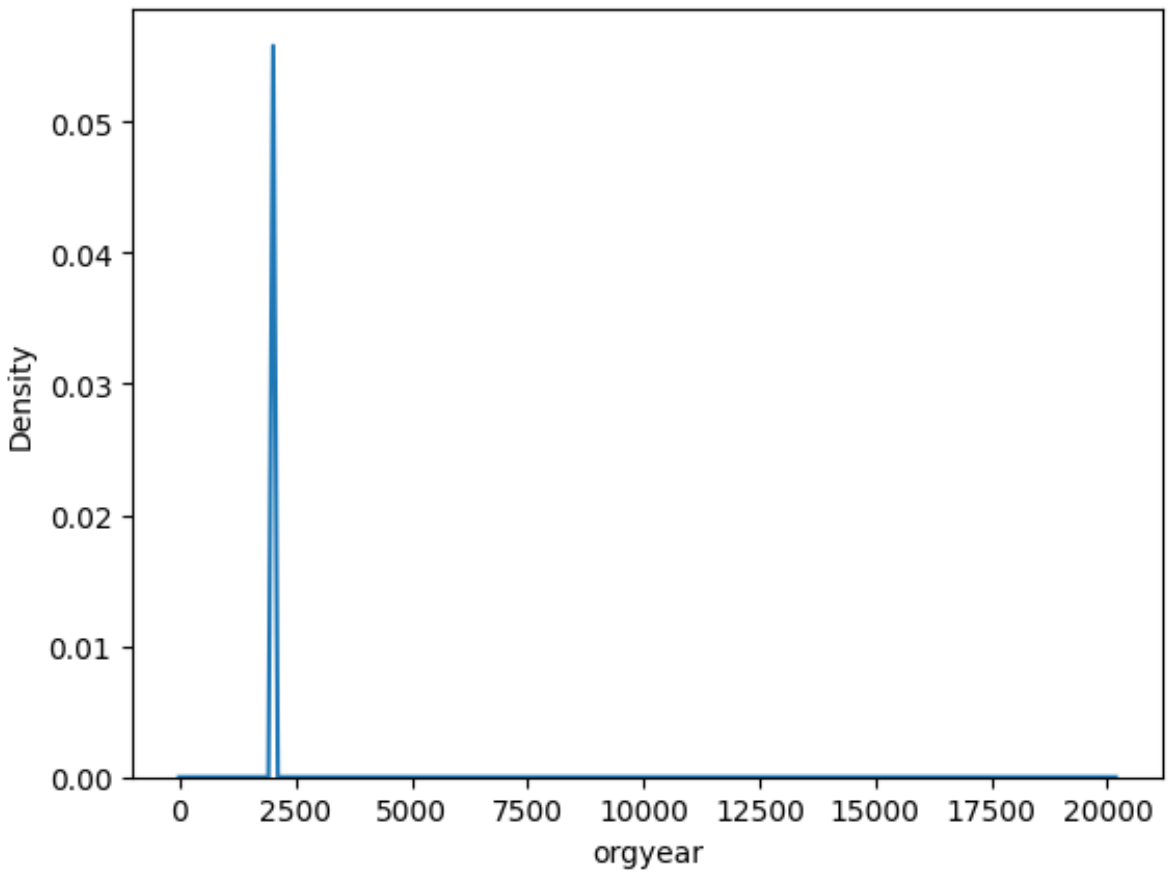
Out[24]: 97.55677481227288

In [25]:
```
df["job_position"].value_counts(normalize= True).sort_values(ascend
```

Out[25]:
```
Backend Engineer          28.414481
FullStack Engineer        16.125286
Other                     11.789459
Frontend Engineer          6.796015
Engineering Leadership     4.481964
QA Engineer                4.297336
Data Scientist             3.502065
Android Engineer           3.494888
SDET                       3.240454
Devops Engineer            3.008853
Support Engineer           2.350585
Data Analyst               1.895864
iOS Engineer               1.791481
Engineering Intern         1.756252
Product Designer           0.857249
Backend Architect          0.839634
Fullstack Engineer         0.825282
Research Engineers         0.801143
Product Manager            0.757432
Program Manager            0.531051
Name: job_position, dtype: float64
```

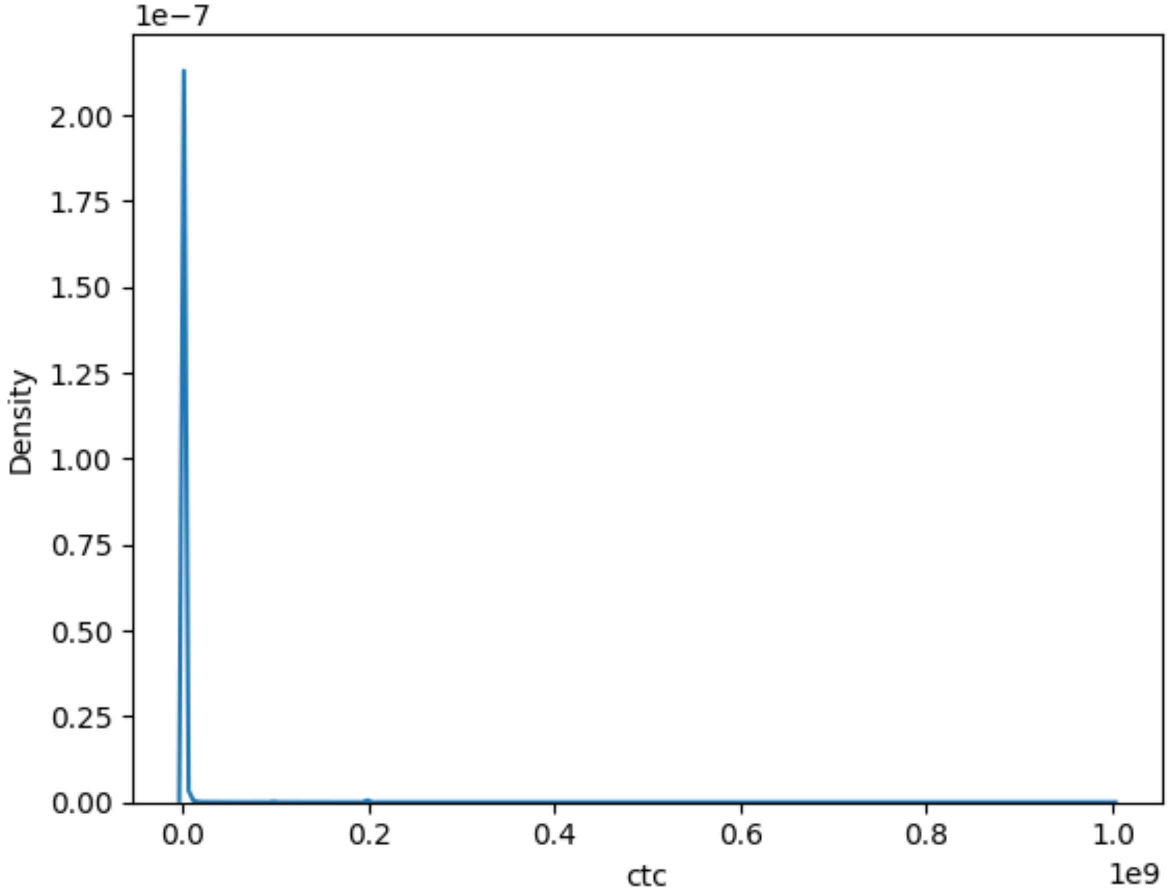
In [27]:
```
sns.kdeplot(df["orgyear"])
```

Out[27]: <Axes: xlabel='orgyear', ylabel='Density'>



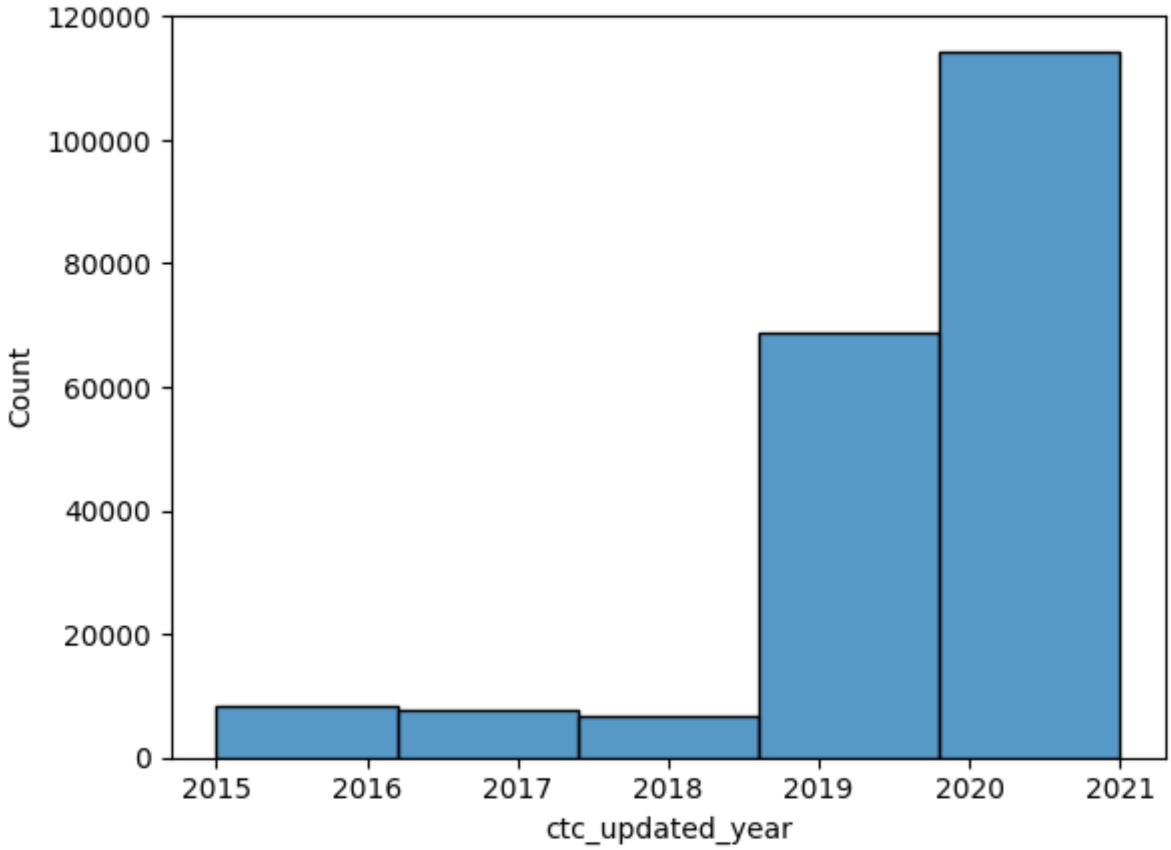
In [28]:
```
sns.kdeplot(df["ctc"])
```

Out[28]: <Axes: xlabel='ctc', ylabel='Density'>



In [30]:
```
sns.histplot(df["ctc_updated_year"], bins = 5)
```

Out[30]:
```
<Axes: xlabel='ctc_updated_year', ylabel='Count'>
```



In [31]:
```
# Due to outliers/high income of some professionals, distribution v
# Due to incorrect inputs from the users , distribution is skwewed.

# Most professionals have got ctc updated in the span of 3 years(20
```

Preprocessing Data

Null Values Imputations and Duplicated Values

In [32]:
```
df.nunique()/len(df)*100
```

Out[32]:
```
company_hash        18.120121
email_hash          74.543706
orgyear              0.037407
ctc                  1.632312
job_position         0.494066
ctc_updated_year     0.003401
dtype: float64
```

In [33]:
```
df.isnull().sum()/len(df)*100
```
Out[33]:
```
company_hash         0.021376
email_hash           0.000000
orgyear              0.041779
ctc                  0.000000
job_position        25.534995
ctc_updated_year     0.000000
dtype: float64
```

In [34]:
```
df.isnull().sum()
```
Out[34]:
```
company_hash           44
email_hash              0
orgyear                86
ctc                     0
job_position        52562
ctc_updated_year        0
dtype: int64
```

In [35]:
```
df.shape
```
Out[35]:
```
(205843, 6)
```

In [36]:
```
#As the percentage of null values in the below columns are very low
```

In [37]:
```
df.dropna(subset=["company_hash" , "orgyear"] , inplace = True)
```

In [38]:
```
df.isnull().sum()/len(df)*100
```
Out[38]:
```
company_hash         0.000000
email_hash           0.000000
orgyear              0.000000
ctc                  0.000000
job_position        25.524396
ctc_updated_year     0.000000
dtype: float64
```

In [39]:
```
# It has been observed that there are multiple duplicate values
# With respect to the rows , there are duplicate rows with original
# Hence , we are dropping those rows
```

In [40]:
```
temp = df[df.duplicated(subset=['company_hash', 'email_hash', 'orgy
       'ctc_updated_year'], keep = False)].sort_values(["email_hash
```

In [41]:
```
temp.head(40)
```
Out[41]:

	company_hash	email_hash	orgyear	
51568	gunhb	0000d58fbc18012bf6fa2605a7b0357d126ee69bc41032...	2021.0	130(

122325	gunhb	0000d58fbc18012bf6fa2605a7b0357d126ee69bc41032...	2021.0	130(
30512	ocu xnivz gbvz	00036c2c5212d88d07acdc5bda7eef5653f8b09bbe30b7...	2011.0	230(
35942	ocu xnivz gbvz	00036c2c5212d88d07acdc5bda7eef5653f8b09bbe30b7...	2011.0	230(
33768	ko	00037a2e4fcfe2830d91270102aaaf105a324a3ce17075...	2012.0	180(
34435	ko	00037a2e4fcfe2830d91270102aaaf105a324a3ce17075...	2012.0	180(
77648	sgrabvz ovwyo	00083d053a4ebf8e8eb99c08c63e0183a70caa0ce348a5...	2014.0	300(
139004	sgrabvz ovwyo	00083d053a4ebf8e8eb99c08c63e0183a70caa0ce348a5...	2014.0	300(
36797	bxzanxwprt	000be203953f54199c95d736e86a75096d0019592cc27c...	2013.0	20(
40946	bxzanxwprt	000be203953f54199c95d736e86a75096d0019592cc27c...	2013.0	20(
57197	xzegojo	000e23cee1f1c00d338672c6dbff0ea7a560916ccac258...	2010.0	100(
66811	xzegojo	000e23cee1f1c00d338672c6dbff0ea7a560916ccac258...	2010.0	100(
64538	nvnv wgzohrnvzwj otqcxwto	001059a637996b0b09d5fcbcd8b40d8e1f6cfa62b18b10...	2019.0	50(
105936	nvnv wgzohrnvzwj otqcxwto	001059a637996b0b09d5fcbcd8b40d8e1f6cfa62b18b10...	2019.0	50(
108551	nvnv wgzohrnvzwj otqcxwto	001059a637996b0b09d5fcbcd8b40d8e1f6cfa62b18b10...	2019.0	50(
3400	ctqntdux	001439ba74b1c44ff593eae85574ba7bc94d86eb399f02...	2018.0	30(
23462	ctqntdux	001439ba74b1c44ff593eae85574ba7bc94d86eb399f02...	2018.0	30(
59729	rvqotz nghmqg	00152a894efe1da15c1164467c09012a2e9fae65b907e1...	2012.0	90(
61015	rvqotz nghmqg	00152a894efe1da15c1164467c09012a2e9fae65b907e1...	2012.0	90(
188728	ltvcxg xzaxv ucn rna	0018d91337b46826a70a961962abbd7a8a8e8036e678bc...	2019.0	55(
192803	ltvcxg xzaxv ucn rna	0018d91337b46826a70a961962abbd7a8a8e8036e678bc...	2019.0	55(
3125	wgzwtznqxd	001944b076fabdf04328f934c7a93fd69de81114836c3d...	2018.0	36(
5043	wgzwtznqxd	001944b076fabdf04328f934c7a93fd69de81114836c3d...	2018.0	36(
90113	ytfrtnn uvwpvqa tzntquqxot	001b08c2b2993420c397fe98bf5c73ca17eca761f190ae...	2012.0	154(
144946	ytfrtnn uvwpvqa tzntquqxot	001b08c2b2993420c397fe98bf5c73ca17eca761f190ae...	2012.0	154(
4435	nvnv wgzohrnvzwj otqcxwto	001b3125da5372767bc5c560066e7e53525f2aece726e6...	2017.0	150(

27460	nvnv wgzohrnvzwj otqcxwto	001b3125da5372767bc5c560066e7e53525f2aece726e6...	2017.0	150(
136720	nvnv wgzohrnvzwj otqcxwto	001b3125da5372767bc5c560066e7e53525f2aece726e6...	2017.0	36(
143866	nvnv wgzohrnvzwj otqcxwto	001b3125da5372767bc5c560066e7e53525f2aece726e6...	2017.0	36(
202754	nvnv wgzohrnvzwj otqcxwto	001b3125da5372767bc5c560066e7e53525f2aece726e6...	2017.0	36(
42452	nvnv wgzohrnvzwj otqcxwto	001bfdb02614b9fc3a288a67944236bb8f3526a146bb1e...	2015.0	90(
57692	nvnv wgzohrnvzwj otqcxwto	001bfdb02614b9fc3a288a67944236bb8f3526a146bb1e...	2015.0	90(
20114	ntwy bvyxzaqv	001da11b06165648239bf15bdbeafd2db64a9fc9b3523c...	2017.0	20(
23577	ntwy bvyxzaqv	001da11b06165648239bf15bdbeafd2db64a9fc9b3523c...	2017.0	20(
27189	btaxwxnj	001fae6fc1e79d270f4a480bfb5a4f5c540ae0c024e794...	2010.0	200(
27206	btaxwxnj	001fae6fc1e79d270f4a480bfb5a4f5c540ae0c024e794...	2010.0	200(
121735	xatvmgd wqtvnxgzo rru	00206e51a50d3080eb3782321ba75c780fc8602b87078d...	2017.0	30(
121999	xatvmgd wqtvnxgzo rru	00206e51a50d3080eb3782321ba75c780fc8602b87078d...	2017.0	30(
38686	gqvwrt	0022e8883afda9ec717eceda94ea8aab89cfbf5ec3b359...	2015.0	83(
62738	gqvwrt	0022e8883afda9ec717eceda94ea8aab89cfbf5ec3b359...	2015.0	83(

In [42]:
```
df[df.duplicated(subset=['company_hash', 'email_hash', 'orgyear', '
       'ctc_updated_year'], keep = False)].sort_values(["email_hash
```

Out[42]:
```
company_hash            0
email_hash              0
orgyear                 0
ctc                     0
job_position        27231
ctc_updated_year        0
dtype: int64
```

In [43]:
```
index_to_drop = temp[temp["job_position"].isnull()].index
df2  = df.drop(index_to_drop)
```

In [44]:
```
df2.shape
```

Out[44]:
```
(178482, 6)
```

In [45]:
```
df2.isnull().sum()/len(df2)
```

Out[45]:
```
company_hash        0.000000
email_hash          0.000000
orgyear             0.000000
ctc                 0.000000
job_position        0.141617
ctc_updated_year    0.000000
dtype: float64
```

In [47]:
```
df2["orgyear"].value_counts().tail(50)
```

Out[47]:
```
1996.0     125
1995.0      84
1993.0      64
1991.0      64
1994.0      59
1992.0      42
2024.0      41
1990.0      34
1989.0      21
0.0         17
2025.0      11
1988.0      10
2026.0       8
1986.0       8
1987.0       6
3.0          6
2031.0       5
2029.0       5
1985.0       4
1984.0       3
1982.0       3
2.0          3
2028.0       3
20165.0      2
1970.0       2
6.0          2
5.0          2
1.0          2
91.0         2
1979.0       1
83.0         1
209.0        1
2204.0       1
1977.0       1
1900.0       1
201.0        1
```

```
38.0          1
1976.0        1
1971.0        1
4.0           1
206.0         1
2027.0        1
1973.0        1
1981.0        1
2106.0        1
2107.0        1
1972.0        1
2101.0        1
208.0         1
200.0         1
Name: orgyear, dtype: int64
```

In [48]:

```
# There were multiple incorrect values in orgyear , imputing correc


# if values are less than 10, it is possible , user was mentioning

# if values are greater than 2021 , imputed to 2021

#Rest were hardcoded
```

In [49]:

```
def Orgyear_fixing(x):

    if x["orgyear"]<=10:
        k = x["ctc_updated_year"]-x["orgyear"]
        return k
    if x["orgyear"] > 2021:
        return 2021
    if x["orgyear"]>=200 and x["orgyear"]<=202:
        return x["orgyear"]*10
    if x["orgyear"]==206.0:
        return 2006
    if x["orgyear"]==209.0:
        return 2009
    if x["orgyear"]==208.0:
        return 2008
    if x["orgyear"] == 91.0:
        return 1991
    if x["orgyear"] == 83.0:
        return 1983
    if x["orgyear"] == 38.0:
        return 2021
    if x["orgyear"] == 1900.0:
        return x["ctc_updated_year"]
    else:
        return x["orgyear"]
```

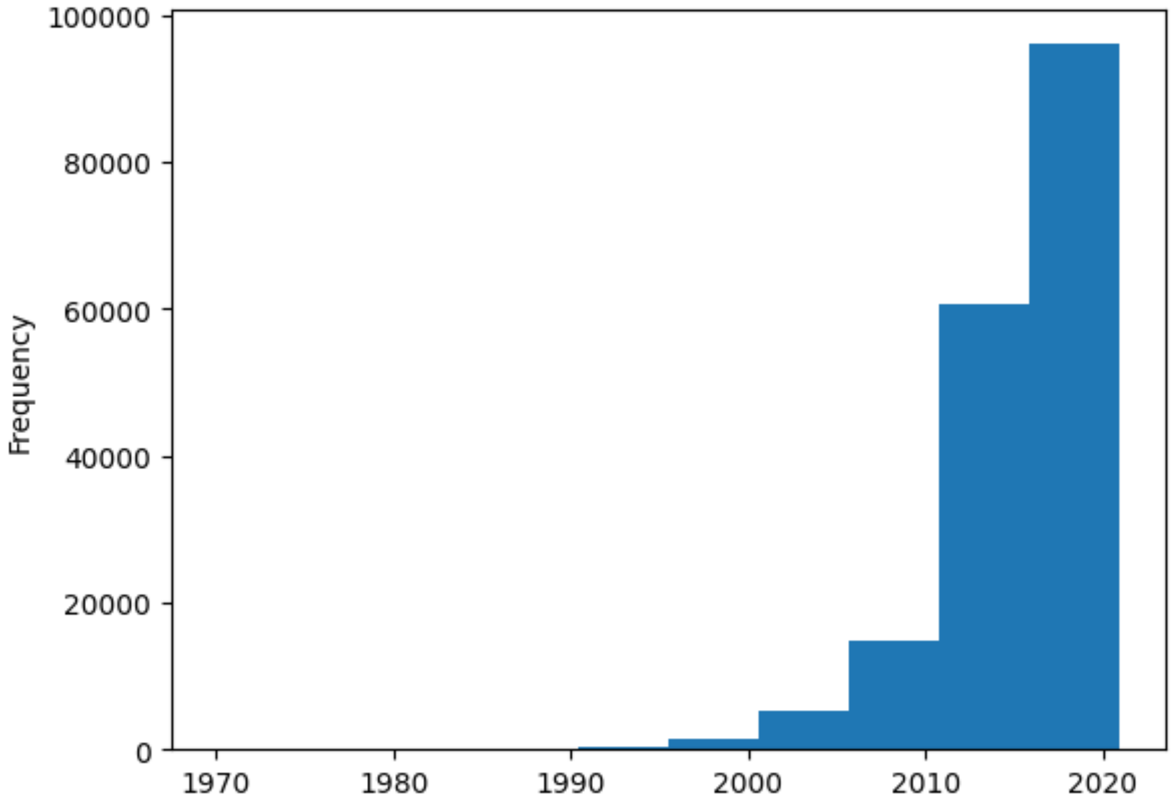
In [50]:
```
df2["orgyear"] = df2.apply(Orgyear_fixing , axis = 1)
```

In [51]:
```
df2["orgyear"].value_counts().tail(50)
```

Out[51]:
```
2018.0    21515
2016.0    20186
2017.0    20009
2019.0    19052
2015.0    18200
2014.0    14911
2013.0    11099
2020.0    11007
2012.0     9408
2011.0     7079
2010.0     5147
2021.0     4255
2009.0     3377
2008.0     2425
2007.0     2004
2006.0     1870
2005.0     1666
2004.0     1314
2003.0      909
2001.0      642
2002.0      618
2000.0      456
1999.0      310
1998.0      265
1997.0      219
1996.0      125
1995.0       84
1991.0       66
1993.0       64
1994.0       59
1992.0       42
1990.0       34
1989.0       21
1988.0       10
1986.0        8
1987.0        6
1985.0        4
1982.0        3
1984.0        3
1970.0        2
1972.0        1
1981.0        1
1973.0        1
1976.0        1
1971.0        1
1977.0        1
1983.0        1
1979.0        1
Name: orgyear, dtype: int64
```

In [52]:
```
df2["orgyear"].plot(kind = "hist")
```

Out[52]: <Axes: ylabel='Frequency'>



In [53]:
```
df2["orgyear"].describe()
```

Out[53]:
```
count    178482.000000
mean       2015.004034
std           4.250953
min        1970.000000
25%        2013.000000
50%        2016.000000
75%        2018.000000
max        2021.000000
Name: orgyear, dtype: float64
```

In [60]:
```
# distribution of the orgyear looks cleaner and better after imputi


# Highest Number of professionals have joined within the range of 2
```

In [55]:
```
# For Job positions, according to business logic, Imputing Others i
```

In [56]:
```
df2["job_position"].fillna("Others" , inplace=True)
```

In [57]:
```
df2.isnull().sum()
```

Out[57]:
```
company_hash        0
email_hash          0
orgyear             0
ctc                 0
job_position        0
ctc_updated_year    0
dtype: int64
```

In [58]:
```
df3 = df2.copy()
```

Feature Engineering

Adding Two features

- TYOE : Total Years of Experience
- Exp After ctc update

In [59]:
```
df3["TYOE"] = 2023 - df3["orgyear"]
df3["Exp After ctc update"] = 2023 - df3["ctc_updated_year"]
```

Analysis on Company , Job Position , TYOE

In [64]:
```
ctc_summary = df3.groupby(by = ["company_hash" , "job_position" , "
ctc_summary.reset_index(inplace = True )
```

In [65]:
```
merged_data_frame = df3.merge(ctc_summary , how = "inner" , on = ["
```

In [66]:
```
merged_data_frame.head(20)
```

Out[66]:

	company_hash	email_hash	orgyear	ctc
0	atrgxnnt xzaxv	6de0a4417d18ab14334c3f43397fc13b30c35149d70c05...	2016.0	1100000
1	qtrxvzwt xzegwgbb rxbxnta	b0aaf1ac138b53cb6e039ba2c3d6604a250d02d5145c10...	2018.0	449999
2	qtrxvzwt xzegwgbb rxbxnta	f4fa64972185ac2b73e99c0cc10d1bf50d6dbfbc9a2cba...	2018.0	620000
3	qtrxvzwt xzegwgbb rxbxnta	be3bcde831f8816f2bad9781f1282f09908f803c2fafb3...	2018.0	950000

4	qtrxvzwt xzegwgbb rxbxnta	ddf45c7b7bd4c461890121c416b2fdff9ba34fbaea2ad4...	2018.0	750000
5	qtrxvzwt xzegwgbb rxbxnta	4a1fcd83b7e904c089f71c77897d4f67728a919776f176...	2018.0	850000
6	qtrxvzwt xzegwgbb rxbxnta	a14e42082606250faf5a138be230ca0d1504377799daf8...	2018.0	600000
7	qtrxvzwt xzegwgbb rxbxnta	fb16948f8da112d0742984f7604d6c4b47c950172f441d...	2018.0	1200000
8	ojzwnvwnxw vx	4860c670bcd48fb96c02a4b0ae3608ae6fdd98176112e9...	2015.0	2000000
9	ngpgutaxv	effdede7a2e7c2af664c8a31d9346385016128d66bbc58...	2017.0	700000
10	ngpgutaxv	30a88256b5586ba59b25e6fe78fada76950fd65ca9f250...	2017.0	1200000
11	ngpgutaxv	2fab5e919a339803876fb532a618ab93c7b83c49746dd7...	2017.0	1750000
12	ngpgutaxv	803bccee8b046cc228a77cc32e5f22704dab529b336ff7...	2017.0	800000
13	ngpgutaxv	c9e14b4d46b1a76974a2e06bc546886cff85bd441f21b8...	2017.0	1600000
14	ngpgutaxv	400aea75dc1316022b8c4436c60a0646fbea2962e26a5a...	2017.0	1210000
15	ngpgutaxv	65ffc5106a7a9e3efb6c94fe3535b9d7d84a0b6b5347aa...	2017.0	850000
16	qxen sqghu	6ff54e709262f55cb999a1c1db8436cb2055d8f79ab520...	2017.0	1400000
17	yvuuxrj hzbvqqxta bvqptnxzs ucn rna	18f2c4aa2ac9dd3ae8ff74f32d30413f5165565b90d8f2...	2018.0	700000
18	lubgqsvz wyvot wg	9bf128ae3f4ea26c7a38b9cdc58cf2acbb8592100c4128...	2018.0	1500000
19	lubgqsvz wyvot wg	76e48bc7e0f9c6a8e4147ad476cdad4c7c9ffa6c621081...	2018.0	1500000

In [67]:
```
# Label function is implemented to create labels for the users, on
# Company , TYOE.

# If the max salary is equal to the ctc , then 1 category is chosen
# users are already high paid,
# in the current organisation.
# If ctc exceeds the 75 percentile , it is labeled as 1 , as the us
# if salary is between the IQR range,it is labeled as 2 , as there
# in the current organisation,
# if ctc is below 25 percentile ,it is labeled as 2, stating that t

def labelling(x):
    if x["max"] ==x["ctc"]:
        return 1
    else:
        if x["ctc"]>x["75%"]:
            return 1
        elif x["ctc"]>=x["25%"] and x["ctc"]<=x["75%"] :
            return 2
        else:
            return 3
```

In [68]:
```
merged_data_frame.head()
```

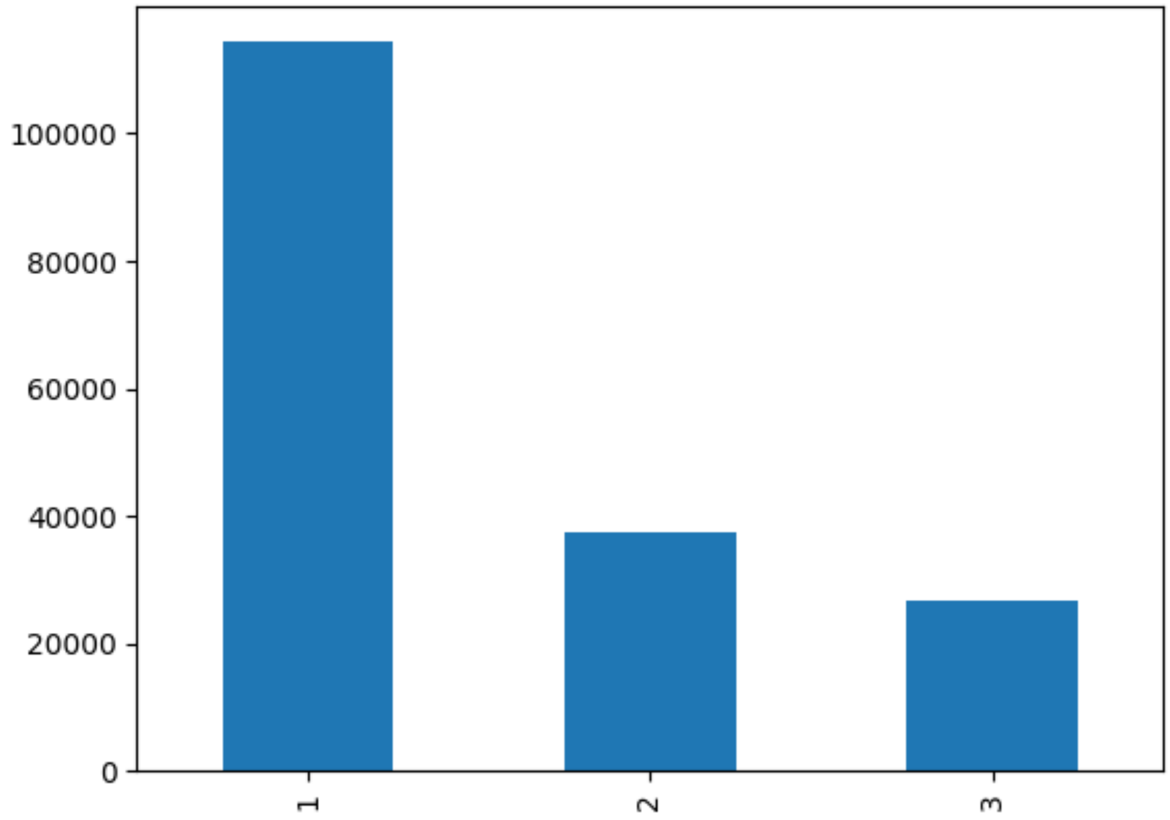
Out[68]:

	company_hash	email_hash	orgyear	ctc	j
0	atrgxnnt xzaxv	6de0a4417d18ab14334c3f43397fc13b30c35149d70c05...	2016.0	1100000	
1	qtrxvzwt xzegwgbb rxbxnta	b0aaf1ac138b53cb6e039ba2c3d6604a250d02d5145c10...	2018.0	449999	
2	qtrxvzwt xzegwgbb rxbxnta	f4fa64972185ac2b73e99c0cc10d1bf50d6dbfbc9a2cba...	2018.0	620000	
3	qtrxvzwt xzegwgbb rxbxnta	be3bcde831f8816f2bad9781f1282f09908f803c2fafb3...	2018.0	950000	
4	qtrxvzwt xzegwgbb rxbxnta	ddf45c7b7bd4c461890121c416b2fdff9ba34fbaea2ad4...	2018.0	750000	

In [69]:
```
merged_data_frame["Designation"] = merged_data_frame.apply(labellin
```

In [70]:
```
merged_data_frame["Designation"].value_counts().plot(kind = "bar")
```

Out[70]: <Axes: >



In [71]:
```
merged_data_frame["Designation"].value_counts()
```

Out[71]:
```
1    114250
2     37422
3     26810
Name: Designation, dtype: int64
```

Analysis on Company Level

In [72]:
```
ctc_summary_company = df3.groupby(by = "company_hash")["ctc"].descr
ctc_summary_company.reset_index(inplace = True)
```

In [73]:
```
merged_data_frame2 = merged_data_frame.copy()
```

In [74]:
```
merged_data_frame2.head()
```

Out[74]:

	company_hash	email_hash	orgyear	ctc	j
0	atrgxnnt xzaxv	6de0a4417d18ab14334c3f43397fc13b30c35149d70c05...	2016.0	1100000	
1	qtrxvzwt xzegwgbb rxbxnta	b0aaf1ac138b53cb6e039ba2c3d6604a250d02d5145c10...	2018.0	449999	
2	qtrxvzwt xzegwgbb rxbxnta	f4fa64972185ac2b73e99c0cc10d1bf50d6dbfbc9a2cba...	2018.0	620000	
3	qtrxvzwt xzegwgbb rxbxnta	be3bcde831f8816f2bad9781f1282f09908f803c2fafb3...	2018.0	950000	
4	qtrxvzwt xzegwgbb rxbxnta	ddf45c7b7bd4c461890121c416b2fdff9ba34fbaea2ad4...	2018.0	750000	

In [75]:
```
merged_data_frame2.drop(["count","mean","std","min","25%","50%","75
merged_data_frame2.head()
```

Out[75]:

	company_hash	email_hash	orgyear	ctc	j
0	atrgxnnt xzaxv	6de0a4417d18ab14334c3f43397fc13b30c35149d70c05...	2016.0	1100000	
1	qtrxvzwt xzegwgbb rxbxnta	b0aaf1ac138b53cb6e039ba2c3d6604a250d02d5145c10...	2018.0	449999	
2	qtrxvzwt xzegwgbb rxbxnta	f4fa64972185ac2b73e99c0cc10d1bf50d6dbfbc9a2cba...	2018.0	620000	
3	qtrxvzwt xzegwgbb rxbxnta	be3bcde831f8816f2bad9781f1282f09908f803c2fafb3...	2018.0	950000	
4	qtrxvzwt xzegwgbb rxbxnta	ddf45c7b7bd4c461890121c416b2fdff9ba34fbaea2ad4...	2018.0	750000	

In [76]:
```
merged_data_frame2 = merged_data_frame2.merge(ctc_summary_company ,
```

In [77]:
```
merged_data_frame2.head()
```

Out[77]:

	company_hash	email_hash	orgyear	ctc	j
0	atrgxnnt xzaxv	6de0a4417d18ab14334c3f43397fc13b30c35149d70c05...	2016.0	1100000	
1	atrgxnnt xzaxv	a309a8c6610af7e9f0a88cfb67f9a0095b0dde63475475...	2019.0	500000	
2	atrgxnnt xzaxv	ffc974693a2bfd0326c707d8460d6783861a9497e538e2...	2017.0	1700000	
3	atrgxnnt xzaxv	b4dcd1e7ac426014a32ae303e4b527325d482e4d2c4bef...	2014.0	1000000	
4	atrgxnnt xzaxv	0d2f25432591093f5907a8681d600f869bbe7c2ae39cd7...	2017.0	600000	

In [78]:
```
merged_data_frame2["Tier"] = merged_data_frame2.apply(labelling , a
```

In [79]:
```
merged_data_frame2.head()
```

Out[79]:

	company_hash	email_hash	orgyear	ctc	j
0	atrgxnnt xzaxv	6de0a4417d18ab14334c3f43397fc13b30c35149d70c05...	2016.0	1100000	
1	atrgxnnt xzaxv	a309a8c6610af7e9f0a88cfb67f9a0095b0dde63475475...	2019.0	500000	
2	atrgxnnt xzaxv	ffc974693a2bfd0326c707d8460d6783861a9497e538e2...	2017.0	1700000	
3	atrgxnnt xzaxv	b4dcd1e7ac426014a32ae303e4b527325d482e4d2c4bef...	2014.0	1000000	
4	atrgxnnt xzaxv	0d2f25432591093f5907a8681d600f869bbe7c2ae39cd7...	2017.0	600000	

Analysis On Company and Job Position Level

In [80]:
```
ctc_summary_company_job = df3.groupby(by = ["company_hash" , "job_p
ctc_summary_company_job.reset_index(inplace = True)
ctc_summary_company_job.head()
```

Out[80]:

	company_hash	job_position	count	mean	std	min	25%	50%
0	0	Other	1.0	100000.0	NaN	100000.0	100000.0	100000.0
1	0000	Other	1.0	300000.0	NaN	300000.0	300000.0	300000.0
2	01 ojztqsj	Android Engineer	1.0	270000.0	NaN	270000.0	270000.0	270000.0
3	01 ojztqsj	Frontend Engineer	1.0	830000.0	NaN	830000.0	830000.0	830000.0
4	05mz exzytvrny uqxcvnt rxbxnta	Backend Engineer	1.0	1100000.0	NaN	1100000.0	1100000.0	1100000.0

In [81]:
```
merged_data_frame3 = merged_data_frame2.copy()
merged_data_frame3.drop(["count","mean","std","min","25%","50%","75
merged_data_frame3 = merged_data_frame3.merge(ctc_summary_company_j
merged_data_frame3["Class"] = merged_data_frame2.apply(labelling ,
merged_data_frame3.head()
```

Out[81]:

	company_hash	email_hash	orgyear	ctc
0	atrgxnnt xzaxv	6de0a4417d18ab14334c3f43397fc13b30c35149d70c05...	2016.0	1100000
1	atrgxnnt xzaxv	696f674fbc0d337b20152f91c43082bafaa243da70932c...	2014.0	1070000
2	atrgxnnt xzaxv	a309a8c6610af7e9f0a88cfb67f9a0095b0dde63475475...	2019.0	500000
3	atrgxnnt xzaxv	b4dcd1e7ac426014a32ae303e4b527325d482e4d2c4bef...	2014.0	1000000
4	atrgxnnt xzaxv	ffc974693a2bfd0326c707d8460d6783861a9497e538e2...	2017.0	1700000

In [82]:
```
merged_data_frame3["Class"].value_counts()/len(merged_data_frame3)
```

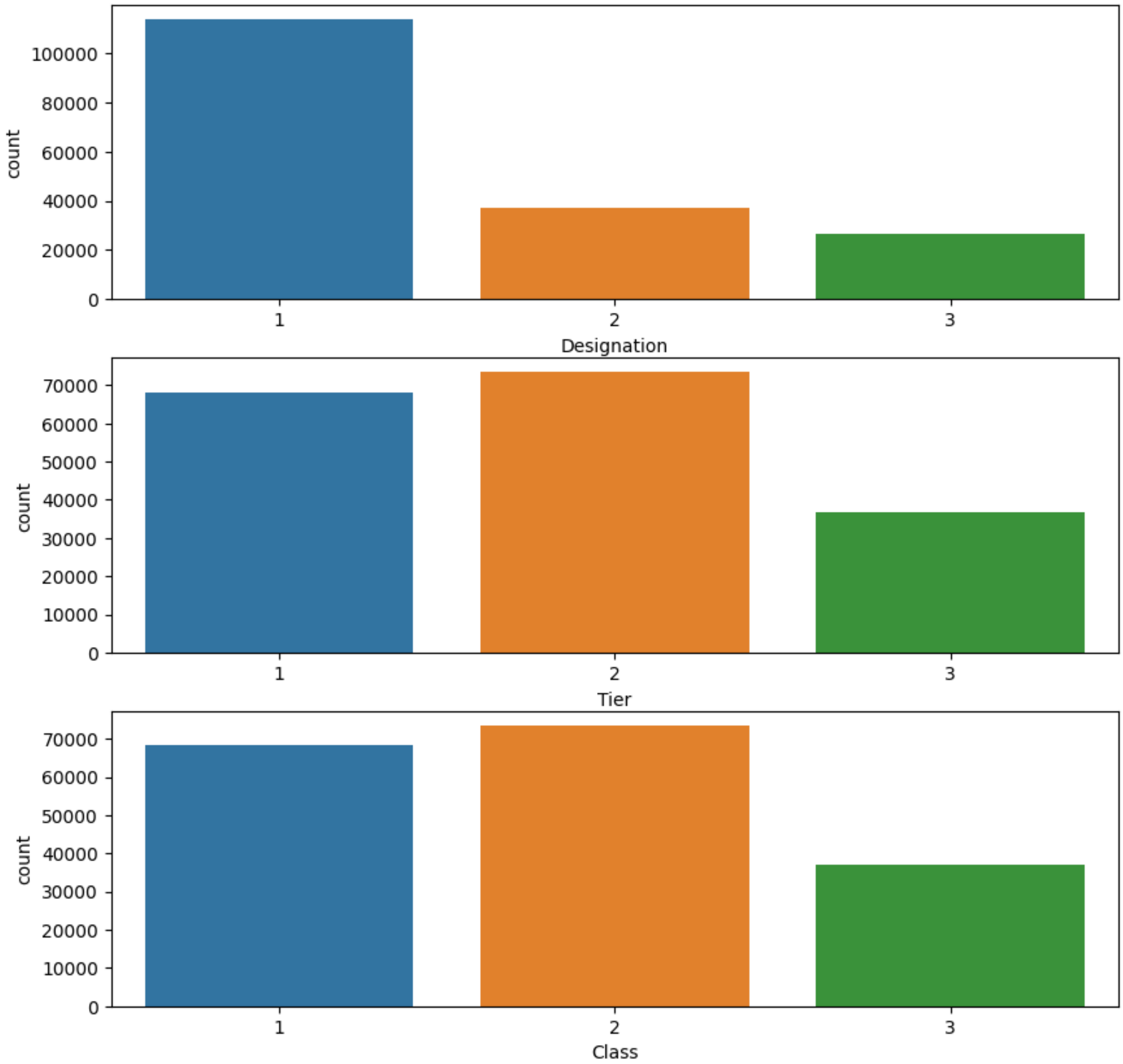
Out[82]:
```
2    0.411459
1    0.381977
3    0.206564
Name: Class, dtype: float64
```

In [83]:
```
merged_data_frame3.drop(["count","mean","std","min","25%","50%","75
merged_data_frame3.head()
```

Out[83]:

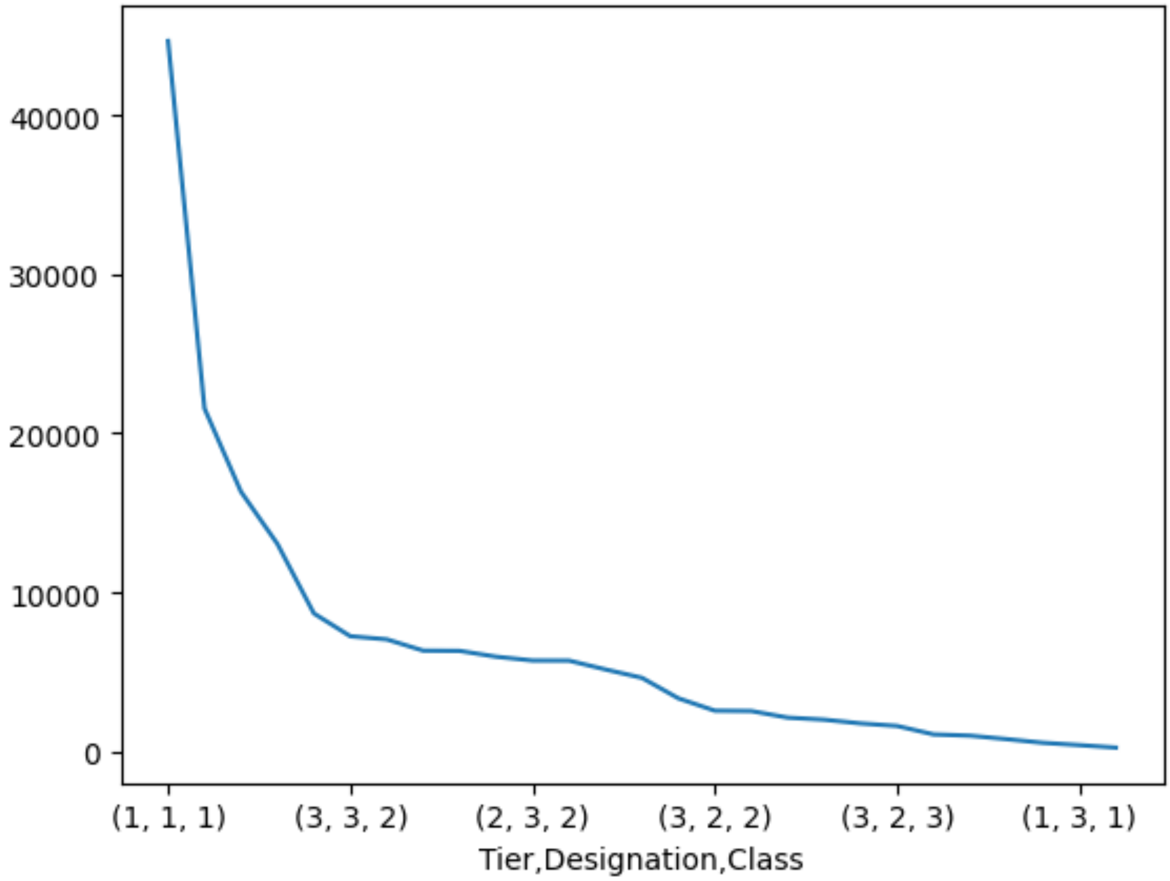
	company_hash	email_hash	orgyear	ctc	j
0	atrgxnnt xzaxv	6de0a4417d18ab14334c3f43397fc13b30c35149d70c05...	2016.0	1100000	
1	atrgxnnt xzaxv	696f674fbc0d337b20152f91c43082bafaa243da70932c...	2014.0	1070000	
2	atrgxnnt xzaxv	a309a8c6610af7e9f0a88cfb67f9a0095b0dde63475475...	2019.0	500000	
3	atrgxnnt xzaxv	b4dcd1e7ac426014a32ae303e4b527325d482e4d2c4bef...	2014.0	1000000	
4	atrgxnnt xzaxv	ffc974693a2bfd0326c707d8460d6783861a9497e538e2...	2017.0	1700000	

In [107]:
```
fig, axes = plt.subplots(3 , figsize = [10,10])
sns.countplot(x = merged_data_frame3["Designation"] , ax = axes[0])
sns.countplot(x = merged_data_frame3["Tier"] , ax = axes[1])
sns.countplot(x = merged_data_frame3["Class"] , ax = axes[2])
plt.show()
```

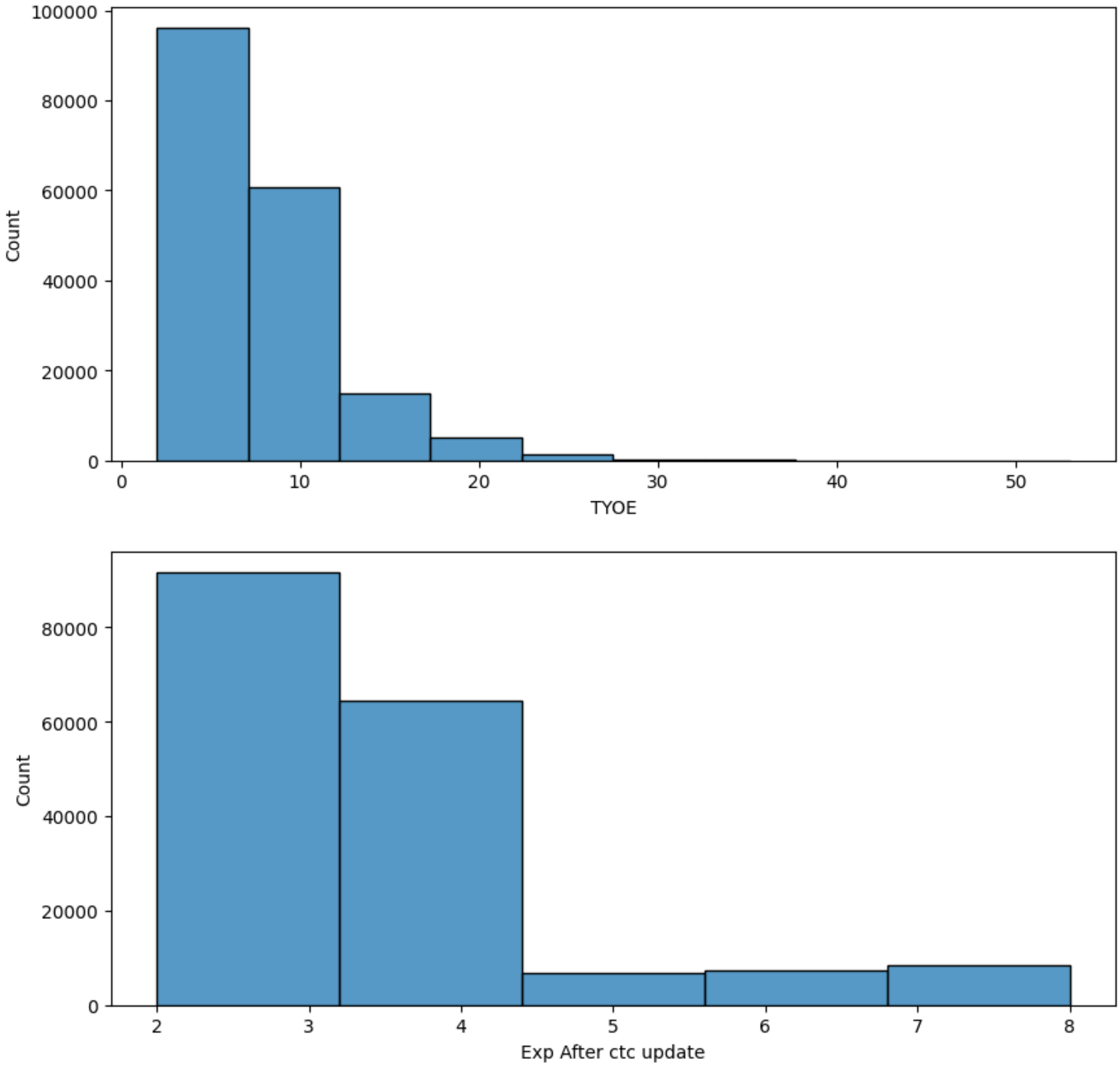


In [117]:
```
merged_data_frame3[["Tier" , "Designation" , "Class"]].value_counts
```

Out[117]: <Axes: xlabel='Tier,Designation,Class'>



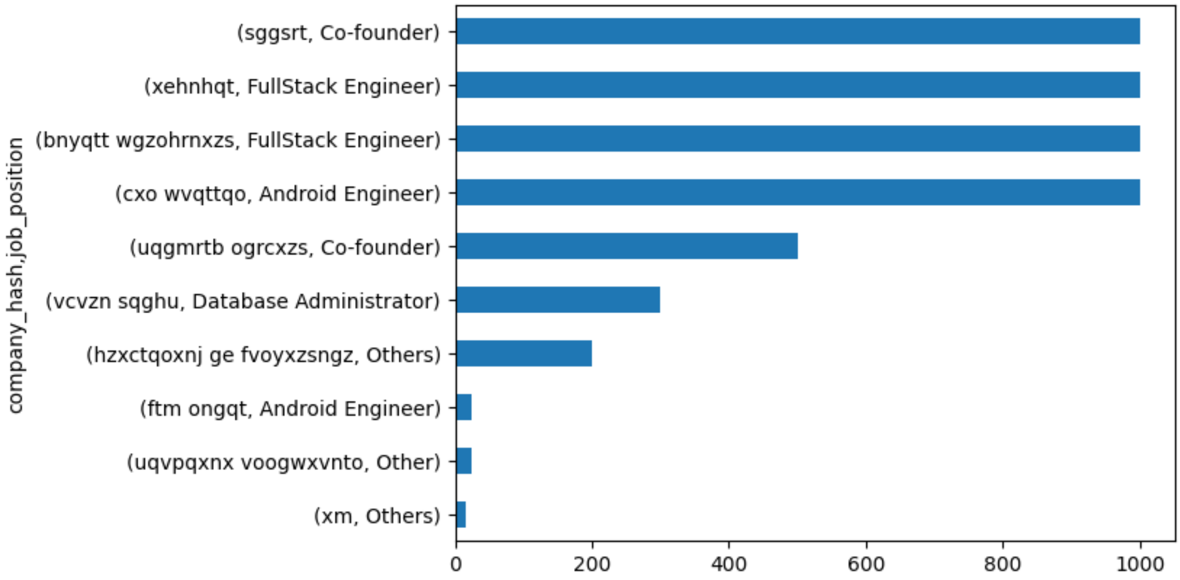
In [113]:
```
fig, axes = plt.subplots(2 , figsize = [10,10])
sns.histplot(x = merged_data_frame3["TYOE"] , ax = axes[0] ,bins=10
sns.histplot(x = merged_data_frame3["Exp After ctc update"] , ax =
plt.show()
```



In [235]:
```
# It has been observed that most people have got their appraisals b
```

In [126]:
```
merged_data_frame3.groupby(["company_hash" , "job_position"])["ctc"
```

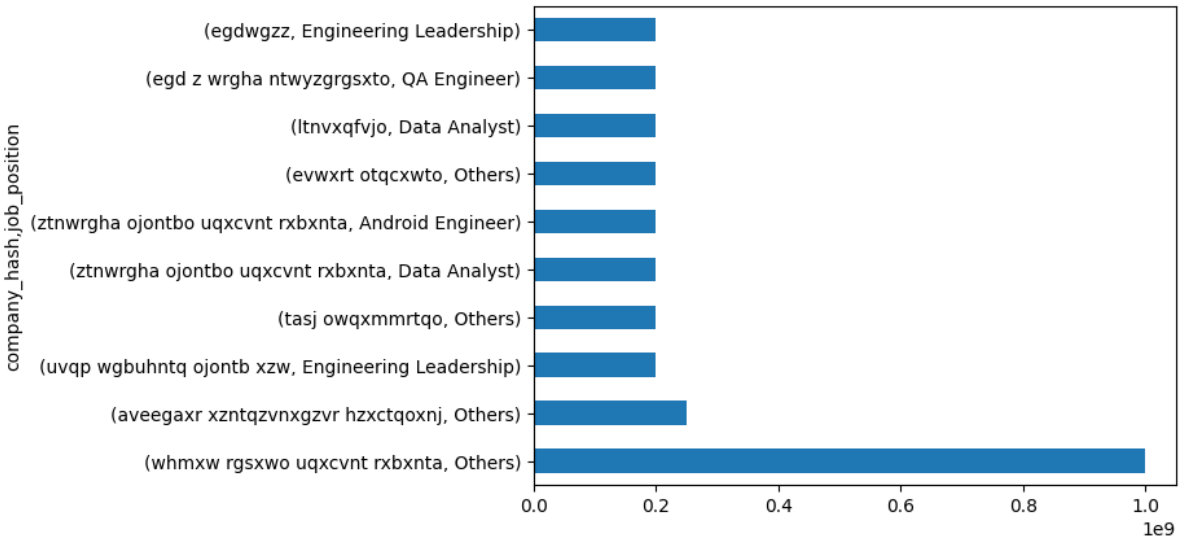
Out[126]: <Axes: ylabel='company_hash,job_position'>



In [236]:
```
# Lowest Combination of Company and Position with respect to ctc.

# These People can be targeted.
```

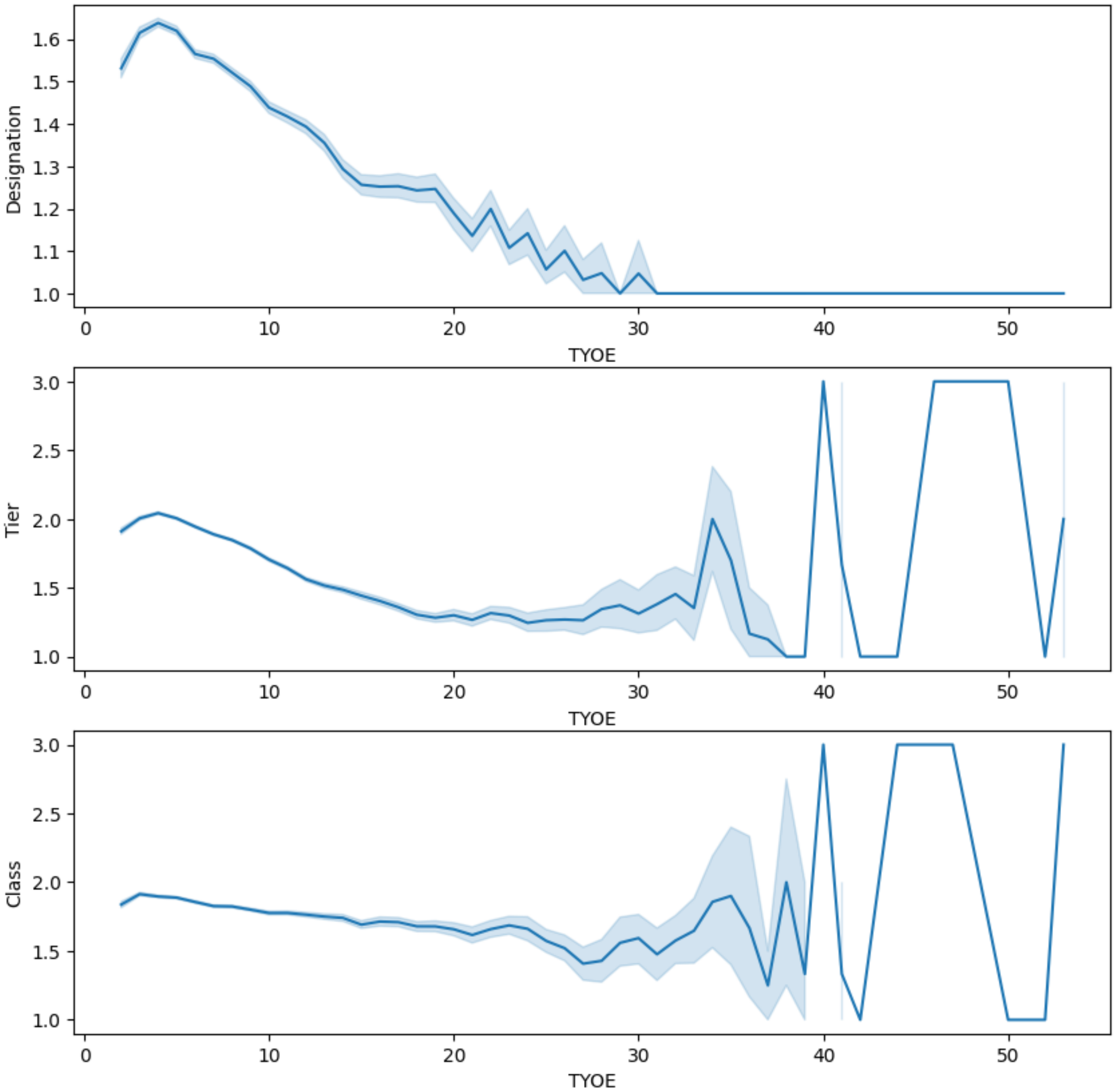
In [127]:
```
merged_data_frame3.groupby(["company_hash" , "job_position"])["ctc"
```

Out[127]: <Axes: ylabel='company_hash,job_position'>



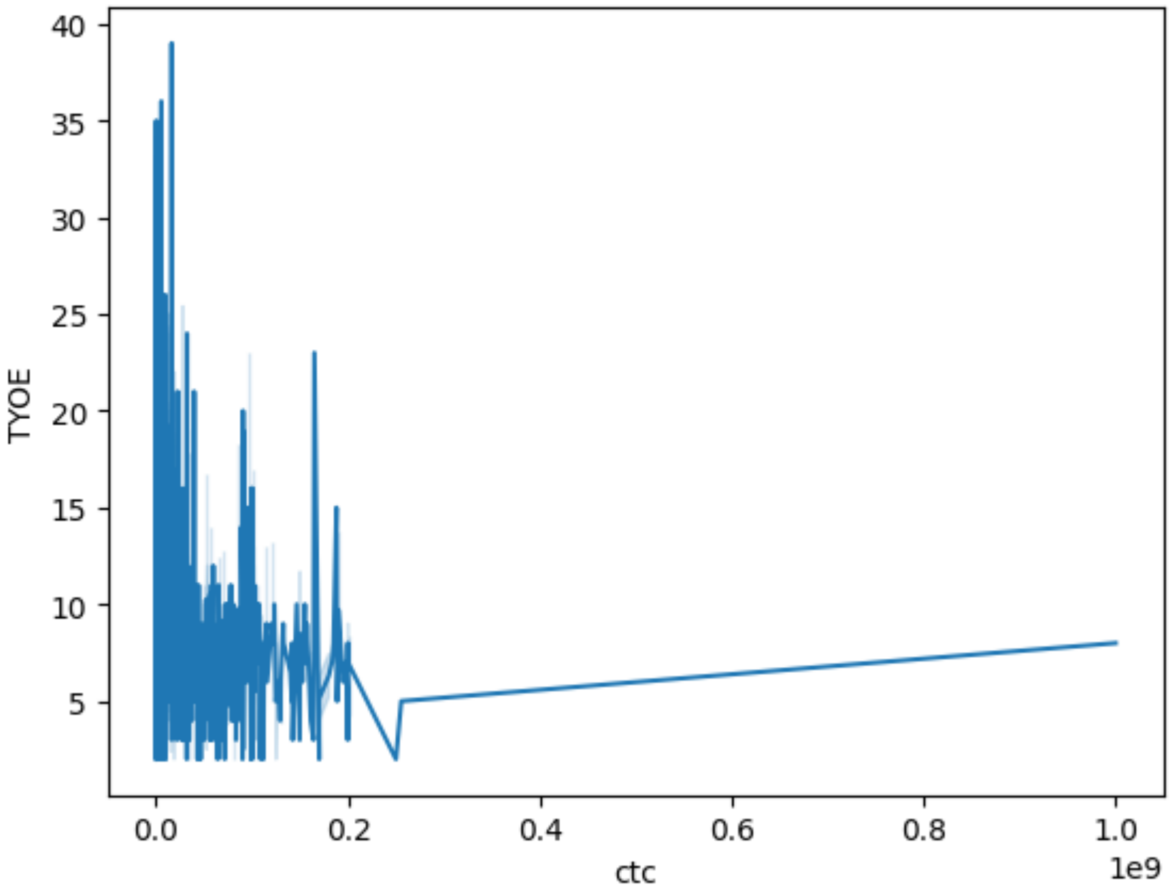
In [237]:
```
# Highest Combination of Company and Position with respect to ctc.
```

In [130]:
```
fig, axes = plt.subplots(3 , figsize = [10,10])
sns.lineplot(x =merged_data_frame3["TYOE"] ,y = merged_data_frame3[
sns.lineplot(x =merged_data_frame3["TYOE"] ,y = merged_data_frame3[
sns.lineplot(x =merged_data_frame3["TYOE"] ,y = merged_data_frame3[
plt.show()
```



In [135]:
```
sns.lineplot(y =merged_data_frame3["TYOE"] ,x = merged_data_frame3[
```

Out[135]: <Axes: xlabel='ctc', ylabel='TYOE'>



In [239]:
```
# Designation flag is highest with 0-10 TYOE.
# Tier flag is highest with 40-50 TYOE.
# Class is similar to TIER.
```

Answering Questions :

In [ ]:

In [ ]:
```
# Top 10 employees (earning more than most of the employees in the
```

In [138]:
```
merged_data_frame3[(merged_data_frame3["Tier"]==1 )& (merged_data_f
```

Out[138]:

	company_hash	email_hash	orgyear	
152570	whmxw rgsxwo uqxcvnt rxbxnta	29a71dd13adf6d2d497571a565bb3096cf66cb46cd1ece...	2015.0	100(
135810	obvqnuqxdwgb	5b4bed51797140db4ed52018a979db1e34cee49e27b488...	2018.0	25!
97637	aveegaxr xzntqzvnxgzvr hzxctqoxnj	06d231f167701592a69cdd7d5c825a0f5b30f0347a4078...	2021.0	25(
96544	oygud 10x wgbbtqwt otqcxwto ucn rna	c84272422e4917b67dcadfc8c2e6dffbe4d018b9235ba6...	2013.0	20(
130287	vznxzg rvmo	634fd283565b8954513a6ad0e47cedb0fa8847923149fb...	2019.0	20(
159884	wrxwpgzwvqt qtnvxr ucn rna	3ad257a31e5448532319f105d5dd2097b5457001aab61e...	2017.0	20(
31945	fxuqg rxbxnta	89f343bf01094accb8b0b2c799499daf6bf881321db2e4...	2017.0	20(
108664	xzaxvmhrro	189dfe129dde29338bdbff63ced8c02dc3c2135fe6decc...	2017.0	20(
119751	axctqoxexta tztqsj ogrhnxgzo ucn rna	83f825e4d64d19bd374ea9ea4d5a16a0a22c08eb92e7ca...	2018.0	20(
130128	20152019	a947ac358ba9670159e9b8350ed4e64ee3fad1715521aa...	2019.0	20(

In [ ]:
```
#Top 10 employees of data science in Amazon / TCS etc earning more
```

In [140]:
```
merged_data_frame3[(merged_data_frame3["Class"]==1) & (merged_data_
```

Out[140]:

	company_hash	email_hash	orgyear	
62644	mqxonrtwgzt v bvyxzaqv sqghu wgbuvzj	cda8d723438e81185d2ee8c348870a4612eea974cdb2db...	2017.0	200(
70605	zgzt	268a5aa92f0b6d0c675fc9cc1e300eb0c5930a3a139a23...	2021.0	200(
133258	ihvaqvnxw xzoxsyno ucn rna	bd222ea783ee372da4e0ad60fdccec0b8f37999a032025...	2015.0	200(
91373	ptnovvr qtnvxr rru	72ed7ced98573f71c8f95bc8b75aac4f0677e8872c6bec...	2019.0	199ε
82473	myvoyjvb owyggr	ee8dd42d6ea8365909147d861c7978d19f727a8075ba96...	2020.0	102ξ
114252	eqvhzygetq hov	2e1d492bc09bfe0d4cc9757a9c63a296c1527af1c8ecc8...	2021.0	100(
167864	bvzyvnnvz wgrrtst	0a358600d0689dbe6c1bae2e27aeca2f248591361b6e65...	2021.0	100(
110867	ptzgbt	4ddef8762b7585c6ee7b8c06834778f3aa00eb3be312b0...	2020.0	100(
129169	xzzgcv ogrhnxgzo	6b6dd66bae787dd4dd417e1777f8ea5a057257e9019995...	2016.0	100(
121330	utqtzsg	e7722fb701c61e5cad82c39ee8bf3debe160d429b72c64...	2015.0	100(

In [ ]:
```
#Bottom 10 employees of data science in Amazon / TCS etc earning le
```

In [142]:
```
merged_data_frame3[(merged_data_frame3["Class"]==3) & (merged_data_
```

Out[142]:

	company_hash	email_hash	orgyear	c
96391	bxyhu wgbbhzxwvnxgz	690f6fdab1ab7514a6a9325ebd6cfe910dbf12d46b6fde...	2018.0	40(
130968	exznqhon ogrhnxgzo ucn rna	ab2dc9db23c3104f0b6b3dbd4cdd5bfb9e5829b8b7943d...	2017.0	72(
114165	tkap	4ed3d04bca6467a839f7a4f878bc15737c3c4afa9cb3a5...	2012.0	80(
107096	nyt mgongz wgzohrnxzs sqghu mws	cf663c71fc96db1ea5658342e2d73050b40ca479d324de...	2016.0	80(
138819	ovuxtznzxnqg	d920a8aa9b63eb317a34bc6cfc4010ec1bb1146f149cb3...	2016.0	90(
13336	ytfrtnn uvwpvqa tzntquqxot	8274b3188470cd1c4914e7face490111e27f239457e62d...	2018.0	100(
13200	hmtq	f091e63c9cc72c1159ad686e32a0a813a617976e44843e...	2017.0	108(
144370	nqtzavp	c5731552cb81ade004c50badb162f8d1ca616743c11343...	2013.0	140(
71267	eqttrvzwtq	10ea984d5c781f1faabc8867f4f4103a1fbf2ec76587bd...	2012.0	240(
149720	ojqvwhot hzxctqoxnj	84737b1d7c2ff2008c2c976f1b28d336d1caaa23159ca2...	2018.0	250(

In [ ]:
```
#Bottom 10 employees (earning less than most of the employees in th
```

In [143]:
```
merged_data_frame3[merged_data_frame3["Class"]==3].sort_values(by =
```

Out[143]:

	company_hash	email_hash	orgyear	ctc
175325	xm	b8a0bb340583936b5a7923947e9aec21add5ebc50cd60b...	2016.0	15
144477	hzxctqoxnj ge fvoyxzsngz	f7e5e788676100d7c4146740ada9e2f8974defc01f571d...	2021.0	200
61587	gjg	b995d7a2ae5c6f8497762ce04dc5c04ad6ec734d70802a...	2018.0	600
8668	xb v onhatzn	4eea97c023bd58395edce18538831df9a735180f88f79d...	2020.0	1000
156986	wgd vhngbgnxct xzw	4d18008fc2cb66e4b90f3798ccbbc4792dfd4bad5a7a87...	2016.0	1000
167944	kvrgqv sqghu	ae625c7063c1f8194deadfb28905d5dcc6f9077274a083...	2017.0	1000
51945	sttpoegqsttpo	1694233be08738b7b50bdb7649b792f0ab8a514c01bec9...	2016.0	1000
149182	uvsotshqg hgr	fc6c6989648ca9a8e78932e583b3f4e6f75a43e0e6c84a...	2015.0	1000
87662	onvqnhu	d9476096e4e5d6f0b0f6079b0543145f62b43c82478bbc...	2018.0	1000
115915	cxo wvqttqo	daa966561c4087398b3c3b13855ce17adcf5e08dda803f...	2012.0	1000

In [ ]:
```
#Top 10 companies (based on their CTC)
```

In [148]:
```
merged_data_frame3[merged_data_frame3["Tier"]==1].sort_values("ctc"
```

Out[148]:

	company_hash	ctc
152570	whmxw rgsxwo uqxcvnt rxbxnta	1000150000
135810	obvqnuqxdwgb	255555555
97637	aveegaxr xzntqzvnxgzvr hzxctqoxnj	250000000
71487	qmo	200000000
87626	onvqnhu	200000000
75662	mvlvl vhng rna	200000000
49532	otre tburgjta	200000000
131945	evwxrxg	200000000
131955	ogzj	200000000
49545	otre tburgjta	200000000

In [ ]:
```
#Top 10 employees in Amazon- X department - having 5/6/7 years of e
```

In [149]:
```
merged_data_frame3[(merged_data_frame3["Class"]==1) & (merged_data_
```

Out[149]:

	company_hash	email_hash	orgyear	
135810	obvqnuqxdwgb	5b4bed51797140db4ed52018a979db1e34cee49e27b488...	2018.0	255
115634	ztnowqxmto	23f778fcb9c8c1cfc177fa5a1c892feca9e24e069e57f5...	2018.0	200
129381	wyvrrtzst xzonxnhnxgz	7e447c2a4390a212cb825a72991d04251b2d943a1daf8d...	2016.0	200
131945	evwxrxg	70a9894df841c880c220dbfd764e664b9e920be7f1a6b5...	2016.0	200
131619	i wgzztin mhoxztoo ogrhnxgzo ucn rna	0a5eaf16728b44b9b5c8ac562df307860433f2fc7ab003...	2017.0	200
130185	guug bgmxrto	9f36d2d7710f7c61aa1a31b86f6bf2d5b5664d71e011f2...	2017.0	200
130094	vour	bc78793b18787e45a5f9509e2acbc4c03095f466b81707...	2018.0	200
129897	dgq avnv tdwyvzst	df86594f0ff614dc9426cab6c87c2dd4a36caad56eeba4...	2016.0	200
129896	bvqctr xzegwgbb ucn rna	a9ac257b8552a8ae1e607f7481d9ce5887fc1aa5970c5d...	2018.0	200
129885	ngfvqao xzaxv	d5feb863469a246ff703b80fec5a9eaedee1e86ed64f61...	2016.0	200

In [151]:
```
merged_data_frame4 = merged_data_frame3.copy()
```

In [152]:
```
from sklearn.preprocessing import LabelEncoder
label = LabelEncoder()
merged_data_frame4["company_hash"] = label.fit_transform(merged_dat
merged_data_frame4["job_position"] = label.fit_transform(merged_dat
merged_data_frame4["email_hash"] = label.fit_transform(merged_data_
```

In [154]:
```
merged_data_encoded = merged_data_frame4.copy()
```

In [155]:
```
merged_data_encoded.head()
```

Out[155]:

	company_hash	email_hash	orgyear	ctc	job_position	ctc_updated_year	TYOE	up
0	968	65738	2016.0	1100000	458	2020.0	7.0	
1	968	63094	2014.0	1070000	458	2018.0	9.0	
2	968	97587	2019.0	500000	140	2020.0	4.0	
3	968	108337	2014.0	1000000	140	2018.0	9.0	
4	968	153209	2017.0	1700000	208	2020.0	6.0	

In [156]:
```
scaler = StandardScaler()
scaled = scaler.fit_transform(merged_data_encoded)
scaled
```

Out[156]:
```
array([[-1.65380593e+00, -2.46594672e-01,  2.34293053e-01, ...,
        -6.87605701e-01,  2.34872799e-01,  2.34872799e-01],
       [-1.65380593e+00, -3.06330884e-01, -2.36190995e-01, ...,
        -6.87605701e-01,  2.34872799e-01,  1.57384581e+00],
       [-1.65380593e+00,  4.72973628e-01,  9.40019125e-01, ...,
        -6.87605701e-01,  1.57384581e+00, -1.10410021e+00],
       ...,
       [ 1.50179732e+00, -1.16143532e+00, -2.36190995e-01, ...,
        -6.87605701e-01, -1.10410021e+00, -1.10410021e+00],
       [-7.55502670e-01, -4.33349408e-01, -4.71433019e-01, ...,
        -6.87605701e-01, -1.10410021e+00, -1.10410021e+00],
       [-1.45781735e+00, -2.83737763e-01, -9.48971086e-04, ...,
        -6.87605701e-01, -1.10410021e+00, -1.10410021e+00]])
```

In [159]:
```
final_database = pd.DataFrame(data = scaled , columns=merged_data_e
```

In [160]:
```
final_database
```

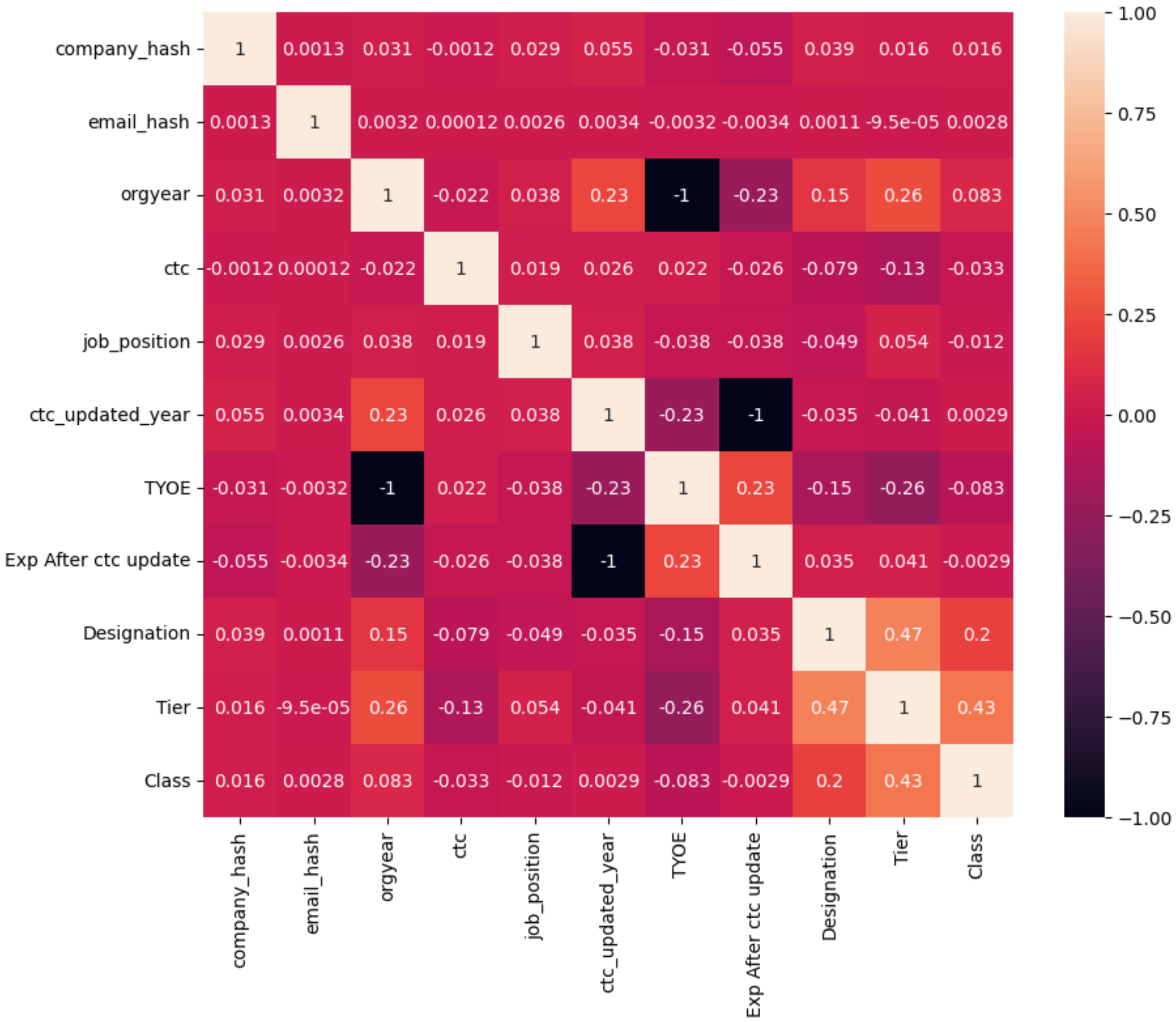
Out[160]:

	company_hash	email_hash	orgyear	ctc	job_position	ctc_updated_year	
0	-1.653806	-0.246595	0.234293	-0.104926	0.700790	0.374759	-
1	-1.653806	-0.306331	-0.236191	-0.107351	0.700790	-1.117971	
2	-1.653806	0.472974	0.940019	-0.153427	-0.989553	0.374759	-
3	-1.653806	0.715850	-0.236191	-0.113010	-0.989553	-1.117971	
4	-1.653806	1.729648	0.469535	-0.056425	-0.628096	0.374759	-
...	...	...	...	...	...	...	
178477	1.386022	-1.054728	-0.706675	-0.024090	0.711421	-1.117971	
178478	1.361067	0.037062	-0.471433	-0.125943	0.711421	-0.371606	
178479	1.501797	-1.161435	-0.236191	0.231836	0.711421	-3.357065	
178480	-0.755503	-0.433349	-0.471433	-0.186974	0.711421	-2.610700	
178481	-1.457817	-0.283738	-0.000949	0.000160	0.711421	-0.371606	

178482 rows × 11 columns

In [167]:
```
plt.figure(figsize=(10,8))
sns.heatmap(final_database.corr(), annot=True)
```

Out[167]: <Axes: >



Modelling

In [168]:

```
wcss = []
for k in range(1, 10):
    model = KMeans(n_clusters = k)
    model.fit(scaled)
    wcss.append(model.inertia_)
```

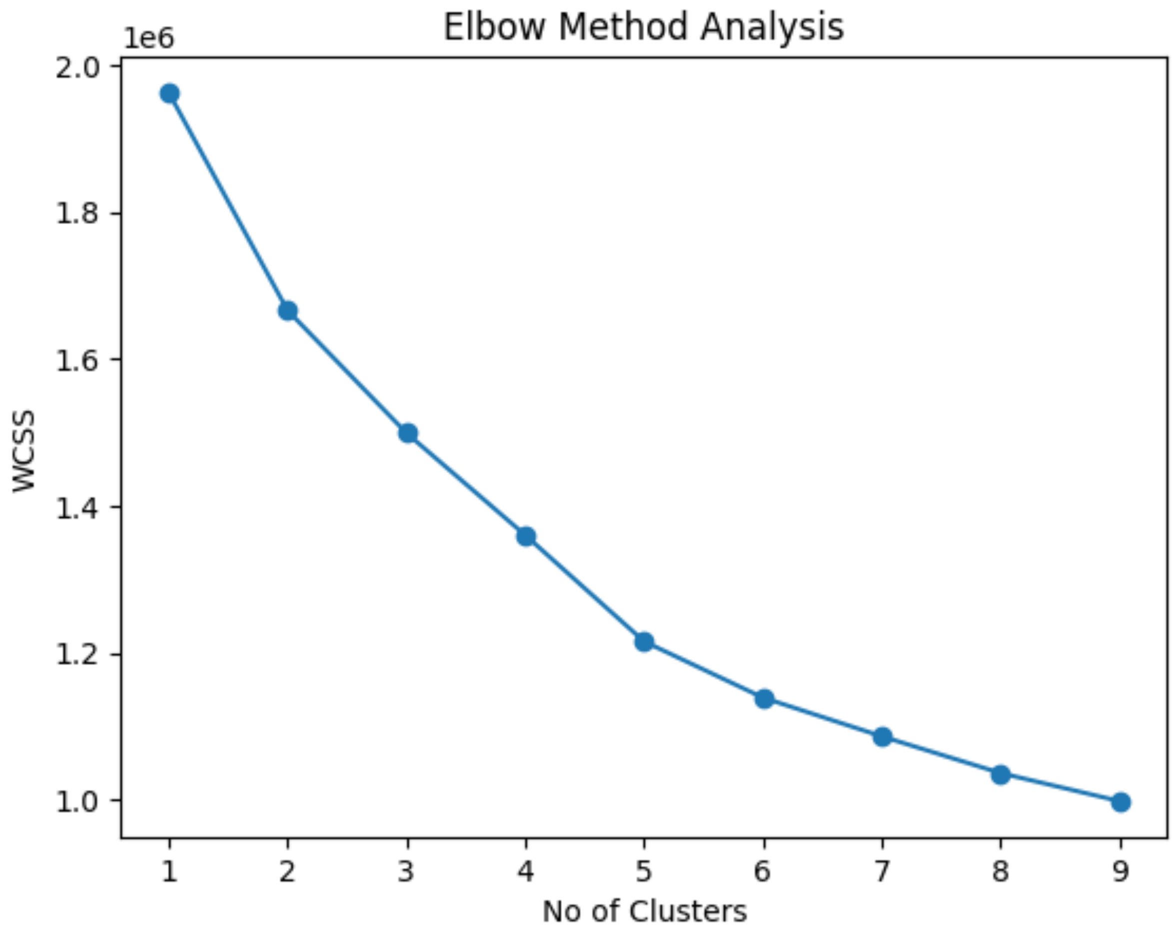
```
/Users/arjunarora/Library/Python/3.9/lib/python/site-packages/skle
arn/cluster/_kmeans.py:870: FutureWarning: The default value of `n
_init` will change from 10 to 'auto' in 1.4. Set the value of `n_i
nit` explicitly to suppress the warning
  warnings.warn(
/Users/arjunarora/Library/Python/3.9/lib/python/site-packages/skle
arn/cluster/_kmeans.py:870: FutureWarning: The default value of `n
_init` will change from 10 to 'auto' in 1.4. Set the value of `n_i
nit` explicitly to suppress the warning
  warnings.warn(
/Users/arjunarora/Library/Python/3.9/lib/python/site-packages/skle
arn/cluster/_kmeans.py:870: FutureWarning: The default value of `n
_init` will change from 10 to 'auto' in 1.4. Set the value of `n_i
nit` explicitly to suppress the warning
  warnings.warn(
/Users/arjunarora/Library/Python/3.9/lib/python/site-packages/skle
arn/cluster/_kmeans.py:870: FutureWarning: The default value of `n
_init` will change from 10 to 'auto' in 1.4. Set the value of `n_i
nit` explicitly to suppress the warning
  warnings.warn(
/Users/arjunarora/Library/Python/3.9/lib/python/site-packages/skle
arn/cluster/_kmeans.py:870: FutureWarning: The default value of `n
_init` will change from 10 to 'auto' in 1.4. Set the value of `n_i
nit` explicitly to suppress the warning
  warnings.warn(
/Users/arjunarora/Library/Python/3.9/lib/python/site-packages/skle
arn/cluster/_kmeans.py:870: FutureWarning: The default value of `n
_init` will change from 10 to 'auto' in 1.4. Set the value of `n_i
nit` explicitly to suppress the warning
  warnings.warn(
/Users/arjunarora/Library/Python/3.9/lib/python/site-packages/skle
arn/cluster/_kmeans.py:870: FutureWarning: The default value of `n
_init` will change from 10 to 'auto' in 1.4. Set the value of `n_i
nit` explicitly to suppress the warning
  warnings.warn(
/Users/arjunarora/Library/Python/3.9/lib/python/site-packages/skle
arn/cluster/_kmeans.py:870: FutureWarning: The default value of `n
_init` will change from 10 to 'auto' in 1.4. Set the value of `n_i
nit` explicitly to suppress the warning
  warnings.warn(
/Users/arjunarora/Library/Python/3.9/lib/python/site-packages/skle
arn/cluster/_kmeans.py:870: FutureWarning: The default value of `n
_init` will change from 10 to 'auto' in 1.4. Set the value of `n_i
nit` explicitly to suppress the warning
  warnings.warn(
```

In [170]:
```
wcss
```

Out[170]:
```
[1963302.0000000014,
 1666297.4840638482,
 1499237.3064775488,
 1360032.679713246,
 1215155.7995885897,
 1138622.9874097463,
 1085842.49840083,
 1035769.5332563792,
 997644.3358722621]
```

In [177]:
```
plt.plot(range(1, 10), wcss, '-o')
plt.title("Elbow Method Analysis")
plt.xlabel("No of Clusters")
plt.ylabel("WCSS")
```

Out[177]:
```
Text(0, 0.5, 'WCSS')
```



In [240]:
```
# From the elbow method , there should be 5 Clusters
```

In [214]:
```
kmean = KMeans(n_clusters = 5)
kmean.fit(scaled)
kmean.labels_ , kmean.inertia_
```

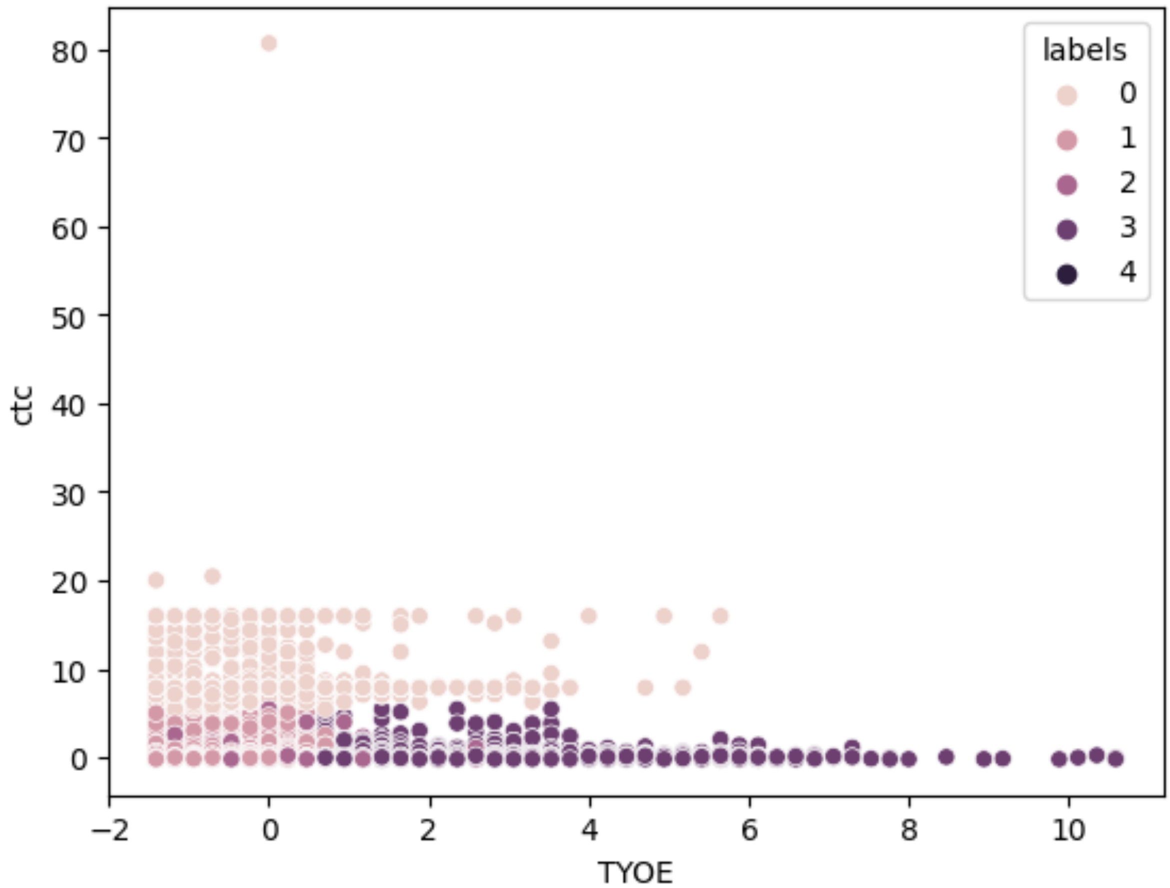
```
/Users/arjunarora/Library/Python/3.9/lib/python/site-packages/skle
arn/cluster/_kmeans.py:870: FutureWarning:

The default value of `n_init` will change from 10 to 'auto' in 1.4
. Set the value of `n_init` explicitly to suppress the warning
```

Out[214]:
```
(array([1, 2, 1, ..., 2, 2, 1], dtype=int32), 1215155.2024409636)
```

In [215]:
```
final_database["labels"] = kmean.labels_
```

In [216]:
```
sns.scatterplot(x = final_database["TYOE"] , y = final_database["ct
```

Out[216]:
```
<Axes: xlabel='TYOE', ylabel='ctc'>
```



In [217]:
```
import plotly.express as px

fig = px.scatter_3d(final_datase, x='Class', y='Tier', z='ctc', col
fig.update_traces(marker=dict(size=2), selector=dict(mode='markers'
fig.show()
```

In [218]:
```
import plotly.express as px

fig = px.scatter_3d(final_datase, x='job_position', y='TYOE', z='ct
fig.update_traces(marker=dict(size=2), selector=dict(mode='markers'
fig.show()
```

Hierarchical Clustering

In [203]:
```
hc_d=final_database.sample(5000)
X = hc_d.copy()
```

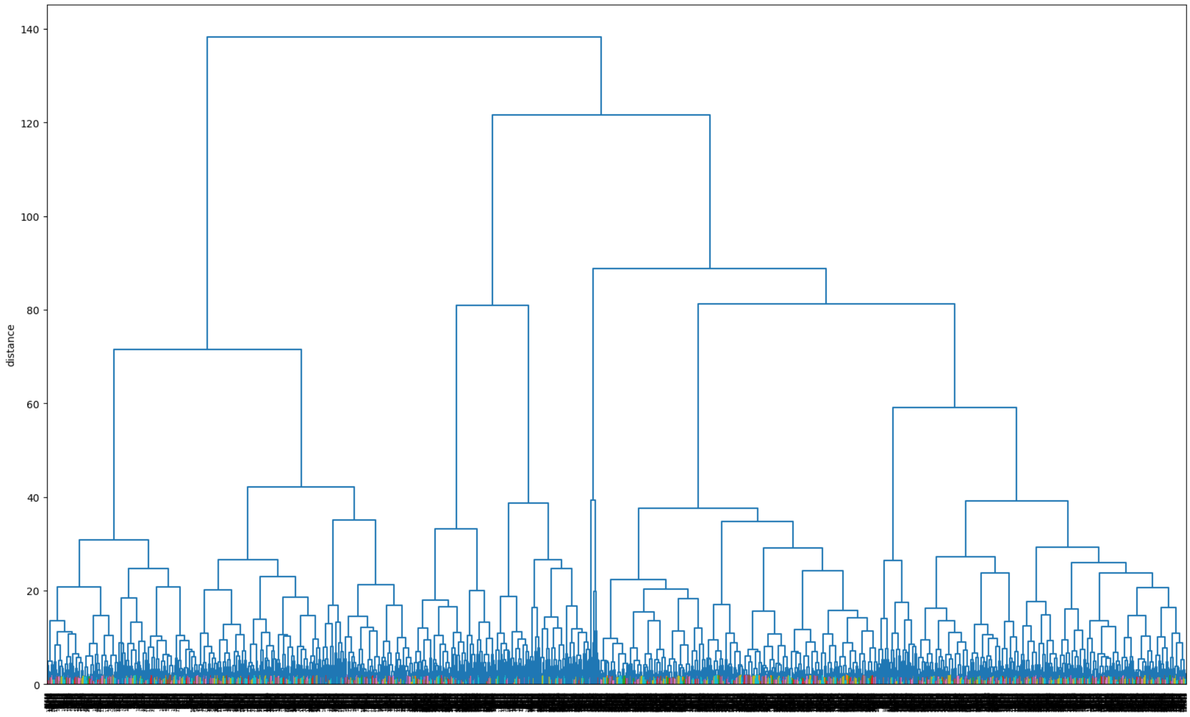
In [204]:
```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
scaler.fit(X)
X = scaler.transform(X)
```

In [205]:
```
scaled_df = pd.DataFrame(X, columns=hc_d.columns, index=hc_d.index)
```

In [206]:
```
import scipy.cluster.hierarchy as sch

Z = sch.linkage(scaled_df, method='ward', metric='euclidean')
```

In [207]:
```
fig, ax = plt.subplots(figsize=(20, 12))
sch.dendrogram(Z, labels=scaled_df.index, ax=ax, color_threshold=2)
plt.xticks(rotation=90)
ax.set_ylabel('distance')
```

Out[207]: Text(0, 0.5, 'distance')



In [ ]:
```
# Numbers of clusters are inconclusive with the help of Dendogram.
```

In [220]:
```
from sklearn.cluster import AgglomerativeClustering

hc = AgglomerativeClustering(n_clusters = 5, affinity='euclidean',
hc.fit(X)
```

```
/Users/arjunarora/Library/Python/3.9/lib/python/site-packages/skle
arn/cluster/_agglomerative.py:983: FutureWarning:

Attribute `affinity` was deprecated in version 1.2 and will be rem
oved in 1.4. Use `metric` instead
```

Out[220]:
```
AgglomerativeClustering(affinity='euclidean', n_clusters=5)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [227]:
```
clusters = pd.DataFrame(X, columns=final_database.columns)
clusters['HC_labels'] = hc.labels_
clusters.head(10)
```

Out[227]:

	company_hash	email_hash	orgyear	ctc	job_position	ctc_updated_year	TY
0	-1.496995	-0.075133	-3.727946	0.503442	-0.351254	-2.730877	3.727
1	-0.221412	-1.123954	1.168806	-0.151704	-0.971477	1.107643	-1.168
2	0.720863	1.382563	0.679131	-0.099858	0.743576	-1.195469	-0.679
3	-0.238557	0.715556	0.923968	-0.128137	1.406945	0.339939	-0.923
4	-0.088870	-0.177754	0.923968	18.630710	-0.604736	0.339939	-0.923
5	1.458639	-0.739708	0.679131	-0.137564	-0.604736	-0.427765	-0.679
6	-0.920455	0.147582	1.413643	-0.163016	-0.367433	1.107643	-1.413
7	0.852605	-0.143562	0.923968	-0.161130	2.976381	1.107643	-0.923
8	-0.131866	1.256018	0.923968	-0.062152	0.754362	-0.427765	-0.923
9	1.080556	-1.586445	1.168806	-0.109284	0.754362	1.107643	-1.168

In [234]:
```
import plotly.express as px

fig = px.scatter_3d(final_datase, x='job_position', y='Class', z='c
fig.update_traces(marker=dict(size=2), selector=dict(mode='markers'
fig.show()
```

In [230]:
```

gmm = GaussianMixture(n_components=3).fit(scaled)
```

In [231]:
```
final_database["GMM Labels"]=gmm.predict(scaled)
```

In [233]:

```
fig = px.scatter_3d(final_database, x='Tier', y='Class', z='ctc', c
fig.update_traces(marker=dict(size=2), selector=dict(mode='markers'
fig.show()
```

Actionable Insights

1. Organisations with the highest Employees in the data set

- nvnv wgzohrnvzwj otqcxwto
- xzegojo
- vbvkgz
- zgn vuurxwvmrt vwwghzn

2. Highest Current Jobs with percentages.

- Backend Engineer 28.414481
- FullStack Engineer 16.125286
- Frontend Engineer 6.796015
- Engineering Leadership 4.481964
- QA Engineer 4.297336

3. Due to outliers/high income of some professionals, ctc distribution seems normal but skewed to the left.
4. Most professionals have got ctc updated in the span of 3 years(2019-2021)
5. It has been observed that there are multiple duplicate values, where an entry is repeated but with Null Job Position in the data set.
6. There are mutplitple incorrect values in the orgyear , which do not make sense. Hence, orgyear distribution is skwewed. Values have been imputed with Business Logic.
7. Highest Number of professionals have joined within the range of 2010-2020.
8. Highesr number of Professionals are Class 2 and tier 2.
9. Lowest number of Professionals are Class 3 and tier 3.
10. Tier - Designation - Class Vs CTC. Below combinations should be focused as they have low ctc.

- 131
- 323
- 322
- 232

11. Designation flag is highest with 0-10 TYOE.
12. Tier flag is highest with 40-50 TYOE.
13. Class is similar to TIER.
14. Positive co relation between CTC and TYOE
15. It has been observed that most people have got their appraisals between 2-4 years.
16. Elbow method suggests 5 Clusters

In [ ]:

Recommendations

1. There should be 9 clusters according to business sense. Tier X Class. But as the elbow method suggests we are working on 5-6 Clusters
2. Data does not provide clear clusters.
3. Company should focus on lower Tier - Designation - Class combinations mentioned.
4. Company should target professionals with more than 2-4 years of Experience after CTC Update.
5. Company should target Data Professionals as current data suggests very few of them in the industry , considering the Future AI Transition.

In [ ]:

```

```