Importando dataset

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from pandas.core.frame import DataFrame
```

```
%matplotlib inline
```

```
pf = pd.read_csv('drive/MyDrive/enron.csv', sep=',')
```

Tratamento de dados

```
pd.set_option("display.float_format", lambda x: "%.0f" % x)
```

.set_option personalizar alguns aspectos de seu comportamento, opções relacionadas à exibição sendo aquelas que o usuário tem mais chances de ajustar.

```
pf.columns
```

```
Index(['bonus', 'deferral_payments', 'deferred_income', 'director_fees',
       'email_address', 'exercised_stock_options', 'expenses', 'from_messages',
       'from_poi_to_this_person', 'from_this_person_to_poi', 'loan_advances',
       'long_term_incentive', 'other', 'poi', 'restricted_stock',
       'restricted_stock_deferred', 'salary', 'shared_receipt_with_poi',
       'to_messages', 'total_payments', 'total_stock_value', 'name'],
      dtype='object')
```

```
pf.rename(columns={'deferral_payments' : 'diferimento_pagamentos', 'deferred_income' : 're
```

	bonus	diferimento_pagamentos	renda_deferida	honorários do diretor	endereço_d
0	4175000	2869717	-3081055	NaN	phillip.allen@en
1	NaN	178980	NaN	NaN	
2	NaN	NaN	-5104	NaN	james.bannantine@en
3	1200000	1295738	-1386055	NaN	
4	400000	260455	-201641	NaN	frank.bay@en
...	...	...	...	...	
141	NaN	NaN	-25000	108579	
142	NaN	NaN	NaN	NaN	john.wodraska@en
143	NaN	NaN	NaN	NaN	
145	NaN	NaN	NaN	NaN	

.rename, renomear colunas em um DataFrame pandas:

Etapa 1

```
pf.shape
```

```
(146, 22)
```

.shape devolva uma tupla representando a dimensionalidade do DataFrame

```
pf.columns
pf.dtypes
```

```
bonus                      float64
deferral_payments          float64
deferred_income            float64
director_fees              float64
email_address               object
exercised_stock_options    float64
expenses                   float64
from_messages              float64
from_poi_to_this_person    float64
from_this_person_to_poi    float64
loan_advances              float64
long_term_incentive        float64
other                      float64
poi                           bool
```

```
restricted_stock             float64
restricted_stock_deferred    float64
salary                       float64
shared_receipt_with_poi      float64
to_messages                  float64
total_payments               float64
total_stock_value            float64
name                          object
dtype: object
```

Isso retorna uma série com o tipo de dados de cada coluna

```
pf.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 146 entries, 0 to 145
Data columns (total 22 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   bonus                      82 non-null     float64
 1   deferral_payments          39 non-null     float64
 2   deferred_income            49 non-null     float64
 3   director_fees              17 non-null     float64
 4   email_address              111 non-null    object
 5   exercised_stock_options    102 non-null    float64
 6   expenses                   95 non-null     float64
 7   from_messages              86 non-null     float64
 8   from_poi_to_this_person    86 non-null     float64
 9   from_this_person_to_poi    86 non-null     float64
 10  loan_advances              4 non-null      float64
 11  long_term_incentive        66 non-null     float64
 12  other                      93 non-null     float64
 13  poi                        146 non-null    bool
 14  restricted_stock           110 non-null    float64
 15  restricted_stock_deferred  18 non-null     float64
 16  salary                     95 non-null     float64
 17  shared_receipt_with_poi    86 non-null     float64
 18  to_messages                86 non-null     float64
 19  total_payments             125 non-null    float64
 20  total_stock_value          126 non-null    float64
 21  name                       146 non-null    object
dtypes: bool(1), float64(19), object(2)
memory usage: 24.2+ KB
```

Este método imprime informações sobre um DataFrame, incluindo o índice dtype e colunas, valores não nulos e uso de memória.

```
pf.poi.value_counts()
```

```
False    128
True      18
Name: poi, dtype: int64
```

.value_counts() retorna o valor da contagem para cada item exclusivo presente na coluna.

```
pf.mean()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: FutureWarning: Droppi
  """Entry point for launching an IPython kernel.
bonus                        2374235
deferral_payments            1642674
deferred_income             -1140475
director_fees                166805
exercised_stock_options      5987054
expenses                     108729
from_messages                   609
from_poi_to_this_person          65
from_this_person_to_poi          41
loan_advances              41962500
long_term_incentive          1470361
other                         919065
poi                                0
restricted_stock             2321741
restricted_stock_deferred     166411
salary                        562194
shared_receipt_with_poi         1176
to_messages                     2074
total_payments               5081526
total_stock_value            6773957
dtype: float64
```

.mean média dos valores

```
pf.median()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: FutureWarning: Droppi
  """Entry point for launching an IPython kernel.
bonus                         769375
deferral_payments             227449
deferred_income              -159792
director_fees                 108579
exercised_stock_options      1310814
expenses                       46950
from_messages                     41
from_poi_to_this_person           35
from_this_person_to_poi            8
loan_advances              41762500
long_term_incentive          442035
other                         52382
poi                                0
restricted_stock              451740
restricted_stock_deferred    -146975
salary                        259996
shared_receipt_with_poi          740
to_messages                     1211
total_payments               1101393
total_stock_value            1102872
dtype: float64
```

.median calcula a mediana dos elementos do objeto DataFrame

```
pf.var()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: FutureWarning: Droppi
  """Entry point for launching an IPython kernel.
bonus                        114775396172335
deferral_payments             26645521052091
deferred_income               16203896512113
director_fees                   102330514030
exercised_stock_options      964848252082402
expenses                        284659397866
from_messages                        3389406
from_poi_to_this_person                 7565
from_this_person_to_poi                10015
loan_advances               2216828541666667
long_term_incentive           35316388281937
other                         21061242250265
poi                                        0
restricted_stock             156707288526356
restricted_stock_deferred     17652554476485
salary                         7378661383806
shared_receipt_with_poi              1388432
to_messages                          6670344
total_payments               844583360181501
total_stock_value           1517708056259896
dtype: float64
```

.var() calcula a variância no Pandas através da função

```
pf.std()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: FutureWarning: Droppi
  """Entry point for launching an IPython kernel.
bonus                        10713328
deferral_payments             5161930
deferred_income               4025406
director_fees                  319891
exercised_stock_options      31062007
expenses                       533535
from_messages                    1841
from_poi_to_this_person            87
from_this_person_to_poi           100
loan_advances                47083209
long_term_incentive           5942759
other                         4589253
poi                                 0
restricted_stock             12518278
restricted_stock_deferred     4201494
salary                        2716369
shared_receipt_with_poi          1178
to_messages                      2583
total_payments               29061716
```

```
total_stock_value           38957773
dtype: float64
```

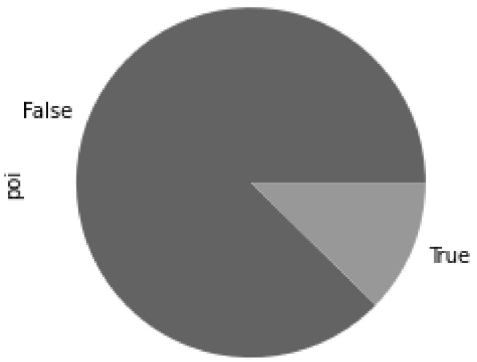
.std() calcula o desvio padrão das colunas ou linhas numéricas

```
pf.describe()
```

	bonus	deferral_payments	deferred_income	director_fees	exercised_stock_
count	82	39	49	17	
mean	2374235	1642674	-1140475	166805	
std	10713328	5161930	4025406	319891	3
min	70000	-102500	-27992891	3285	
25%	431250	81573	-694862	98784	
50%	769375	227449	-159792	108579	
75%	1200000	1002672	-38346	113784	
max	97343619	32083396	-833	1398517	31

Método para encontrar as estatísticas de uma moldura de dados

```
pf['poi'].value_counts(sort=False).plot.pie()
plt.show()
```



Etapa 2

```
sns.boxplot(data=pf[pf['bonus']<0.5*1e7], y='bonus', x='poi')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0e9bc98a90>
```

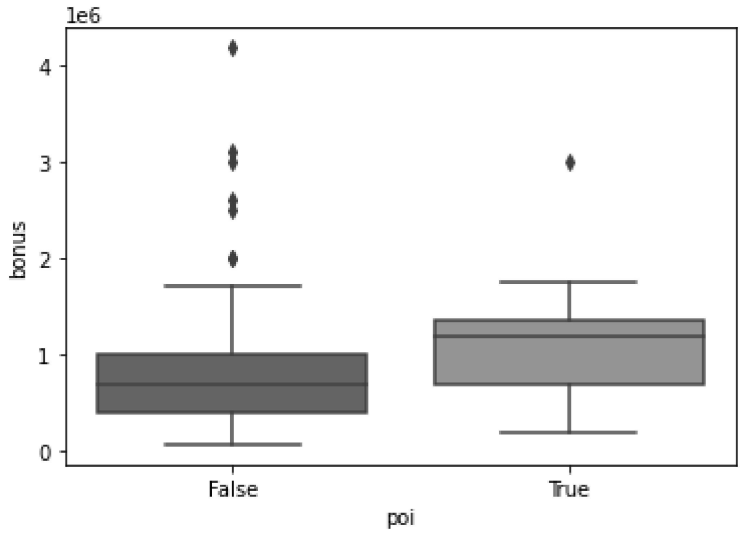


Gráfico boxplot aonde podemos ver duas variáveis, bônus e poi, também aqueles que receberam mais bônus estava envolvido, como consta a variedade true, observação a linha que está cortando o "meio" e a mediana da variável bônus

```
sns.lineplot(x="from_poi_to_this_person", y="from_this_person_to_poi",  hue="poi", data=pf
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0e9bc71910>
```

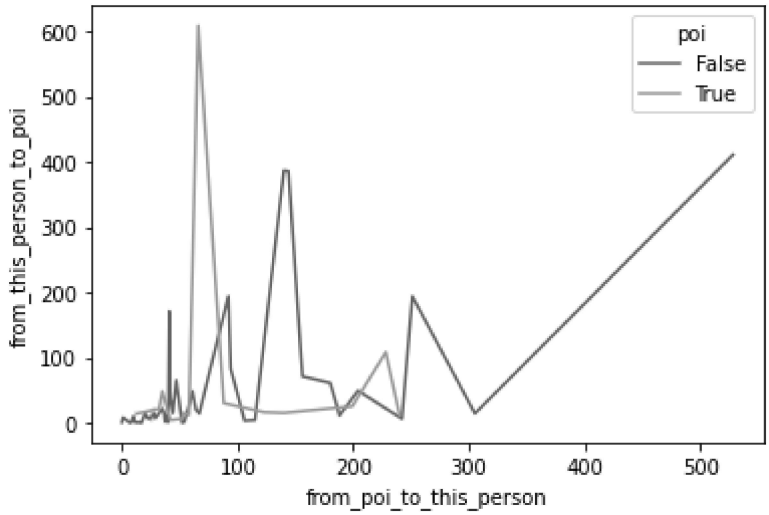


Grafico lineplot aonde pode utilizar quatro varieveis seria recomendavel utilizar duas varieval categoricas. teste feito com variavel que trocaram email com pessoas envolvidas no esquema.

```
sns.kdeplot(data=pf[pf['salary']<0.2*1e7], x="salary", y="bonus", hue="poi")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0e9bbabdd0>
```

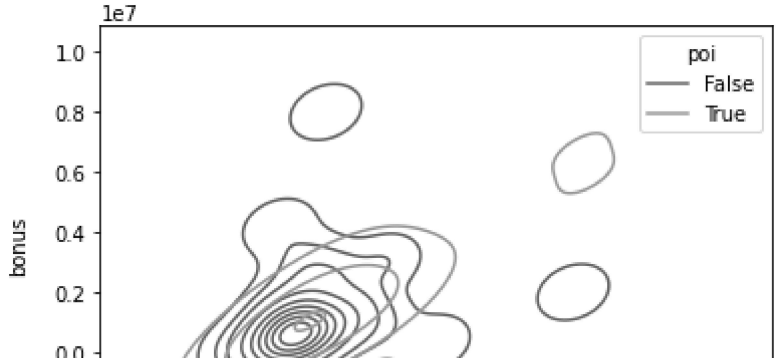


Grafico kdeplot com tres variaveis, mostra a relacão do bonus, salário e poi. mostrando que maior parte do pessoal que está envolvido ao escadalo possui salário, desde baixo a alto aumentando o bonus em si conforme o salário cresce.

Etapa 3

Dataset da empresa enron, ao qual se colapsou num esquema fraudulento, a partir desse dataset, com dados coletados sobre o esquema, tentar obter uma informção sobre o padrão de pessoas associada a corrupção.

```
pf["email_address"] = pf["email_address"].astype(bool)
```

.astype(bool) transforma a coluna email em boolean

```
pd.options.display.float_format = "{:,.2f}".format
```

formata os dado tipo float depois do , em dois casa

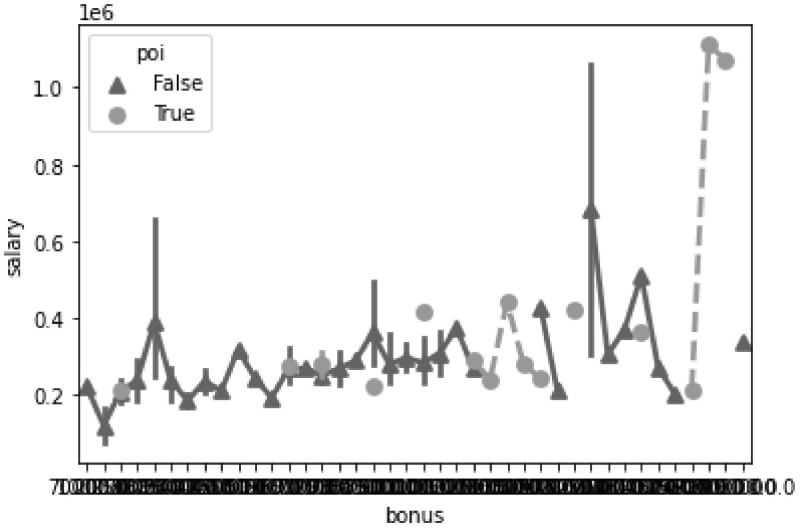
```
sns.boxenplot(data=pf[pf['salary']<0.5*1e7], x="email_address", y="salary", hue="poi")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0e9b8f6d90>
```

eixo x com email e y com salário e hue com poi, partir disso pode deduzir nessas visualizações que a maioria do pessoal com salário alto e tem email estao envolvidos ao esquema.

```
sns.pointplot(data=pf[pf['salary']<0.5*1e7], x="bonus", y="salary", hue="poi", markers=["^
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0e99023210>
```

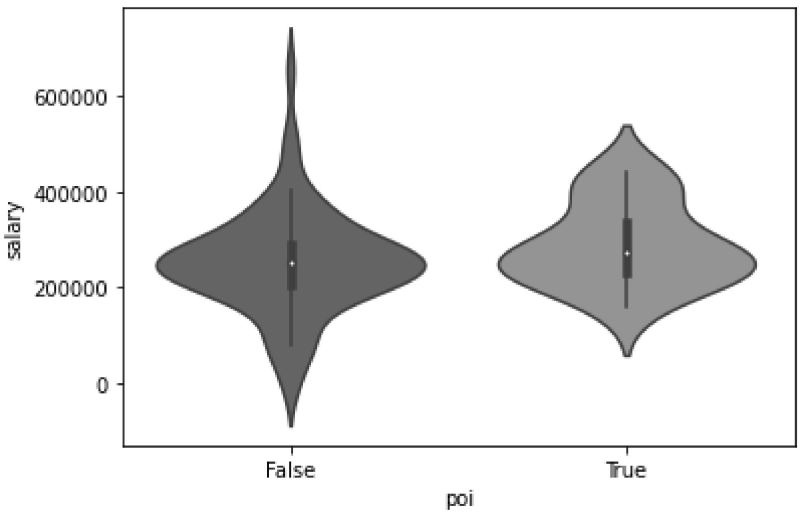


eixo x bonus, eixo y salário e poi como hue, podemos ver que pessoal que estava envolvido tinha o mesmo salário diferenciando mesmo o valor do bonus

Clique duas vezes (ou pressione "Enter") para editar

```
sns.violinplot(data=pf[pf['salary']<0.8*1e6], y='salary', x='poi')
```

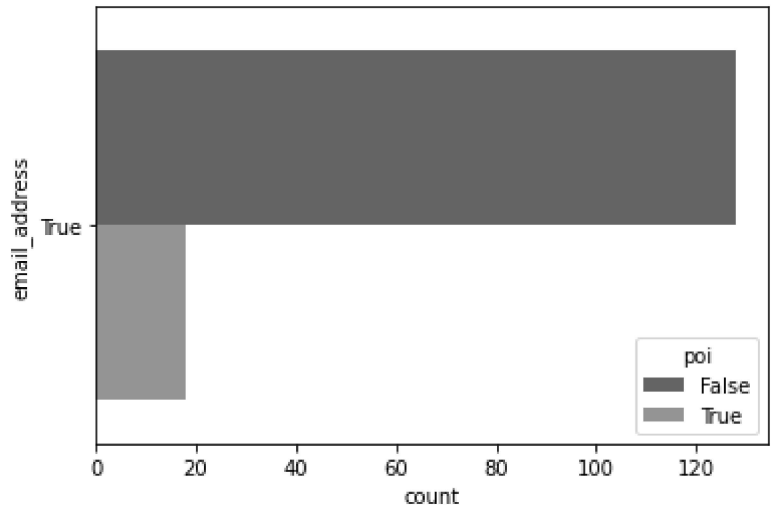
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0e98dfd3d0>
```



eixo x poi e eixo y salario, so comprovando o que foi descrito acima

```
sns.countplot(data=pf, y='email address', hue='poi')
```

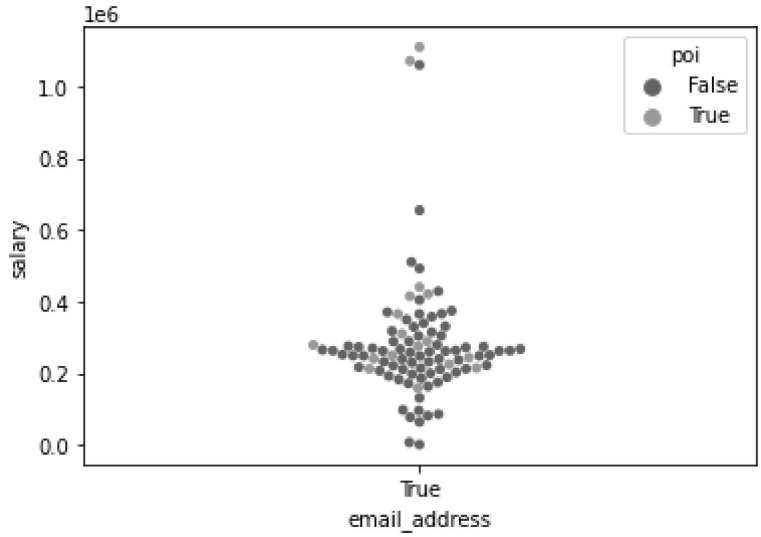
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0e98dddb10>
```



aqui a maioioria dos funcionario nao estava envolvido no escandalo com email

```
sns.swarmplot(data=pf[pf['salary']<0.2*1e8], x="email_address", y="salary", hue="poi")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0e98d56d10>
```

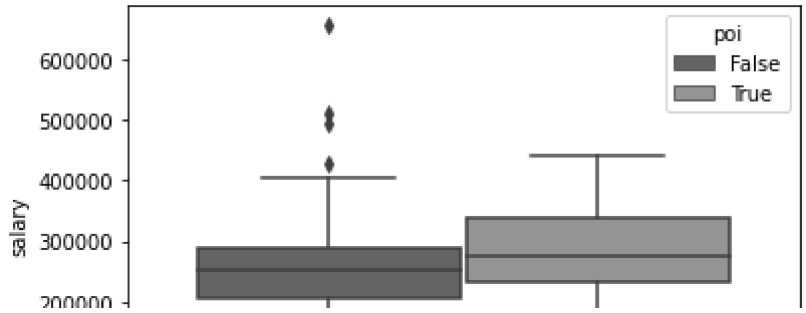


eixo x email eixo y salário e hue poi aonde vemos pessoal que utiliza email e esta envolvildo ao escandalo e mais no meio, sendo so dois superior

Clique duas vezes (ou pressione "Enter") para editar

```
sns.boxplot(data=pf[pf['salary']<0.8*1e6], x="email_address", y="salary", hue="poi")
```

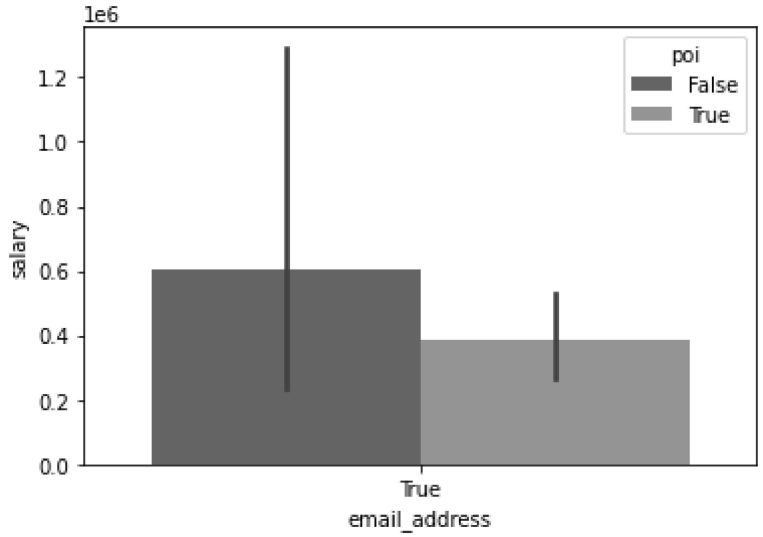
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0e98cbbd50>
```



aqui com boxplot fazendo mediana aonde a caixa do true está maior, mas nao necessariamente e o maior quantidade

```
sns.barplot(data=pf, x="email_address", y="salary", hue="poi")
```

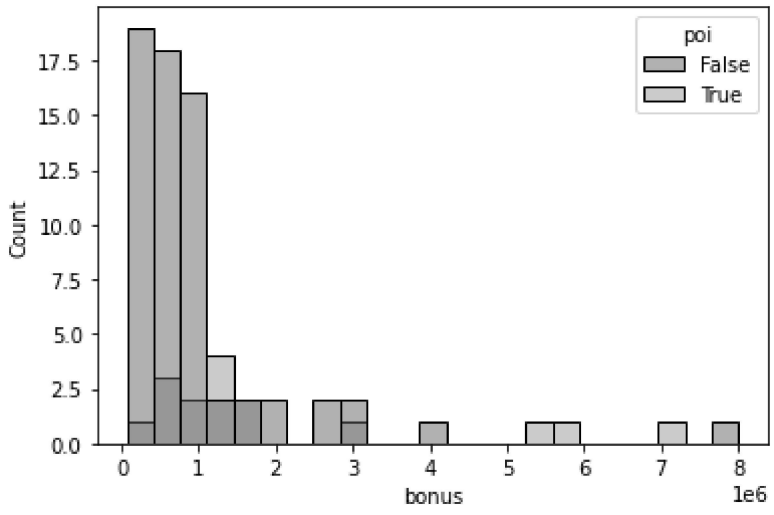
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0e98c52690>
```



ocorre ao contrario ja que o false e maior em quesito quantidade

```
sns.histplot(data=pf[pf['bonus']<0.2*1e8], x='bonus', hue='poi')
```

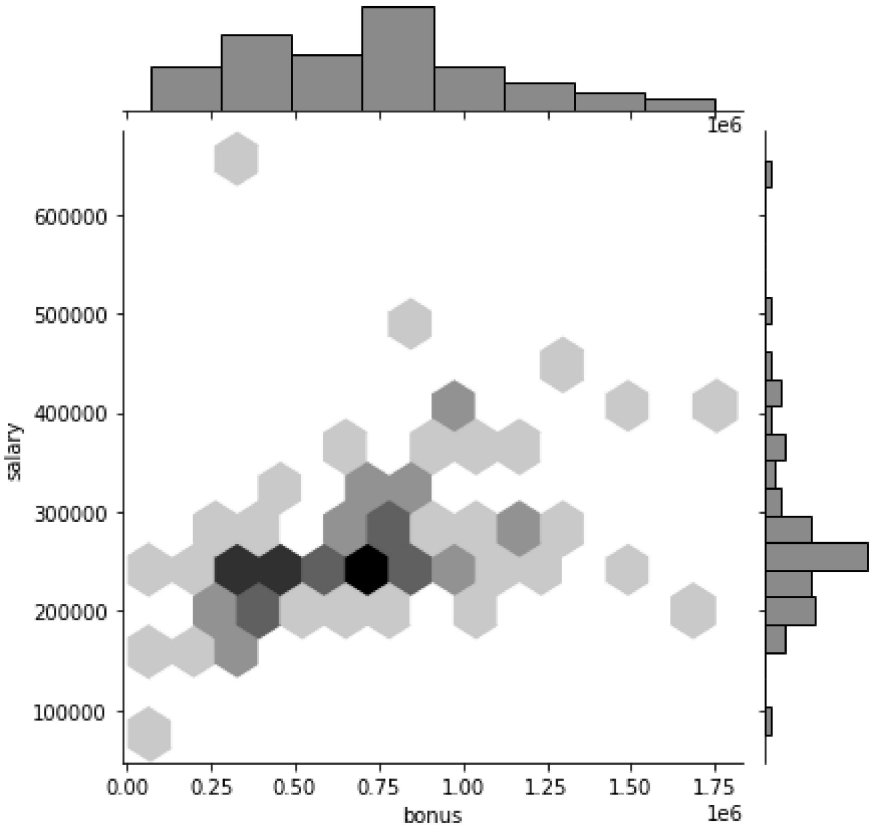
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0e98cc1890>
```



aqui e pra ver a contagem de bonus no eixo x com true e false

```
sns.jointplot(data=pf[pf['bonus']<2.0*1e6], x='bonus', y='salary',  kind="hex")
```

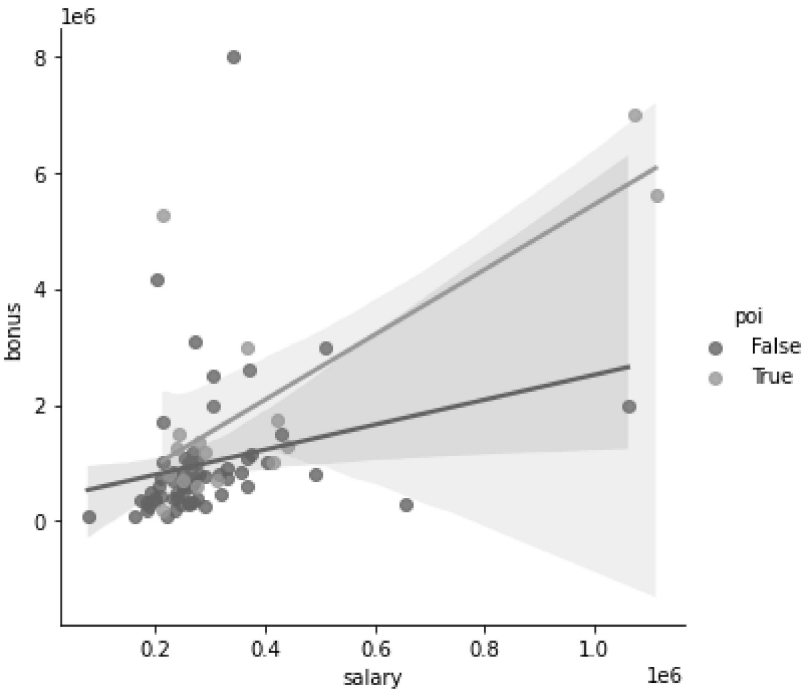
```
<seaborn.axisgrid.JointGrid at 0x7f0e98ea21d0>
```



aqui esta pra ver aonde e mais ocorrente entre salario e bonus

```
sns.lmplot(data=pf[pf['bonus']<0.2*1e8], x="salary", y="bonus", hue="poi")
```

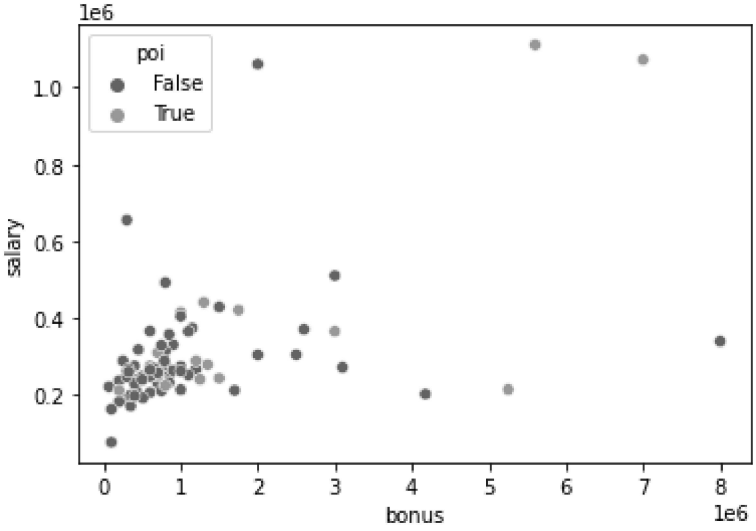
```
<seaborn.axisgrid.FacetGrid at 0x7f0e9899bf10>
```



Traçando uma linear entre false e true, conforme bonus vai indo, linear do false tente a aumentar, porem a do True aumenta mais conforme salário for maior

```
sns.scatterplot(data=pf[pf['salary']<0.5*1e7], x="bonus", y="salary", hue="poi")
```

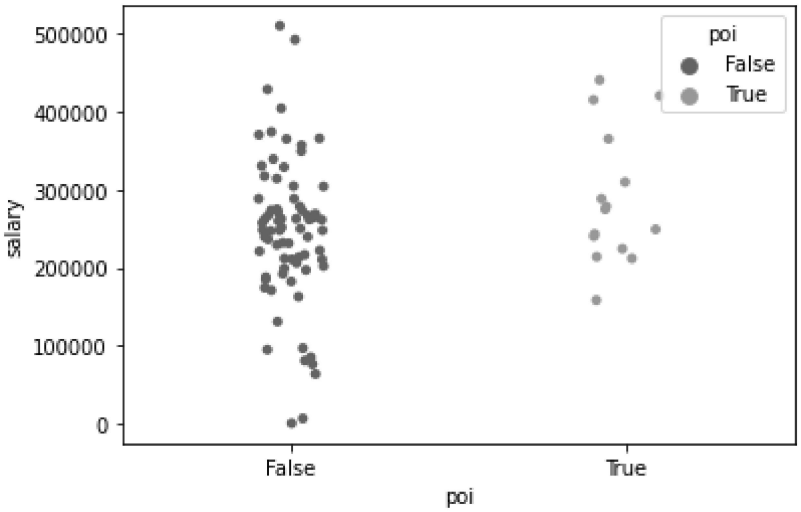
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0e98925090>
```



Aqui pra ver mais detalhado o bonus eixo x e y salário, como pode ver há 3 com bônus altos, sendo 2 com salário já altos

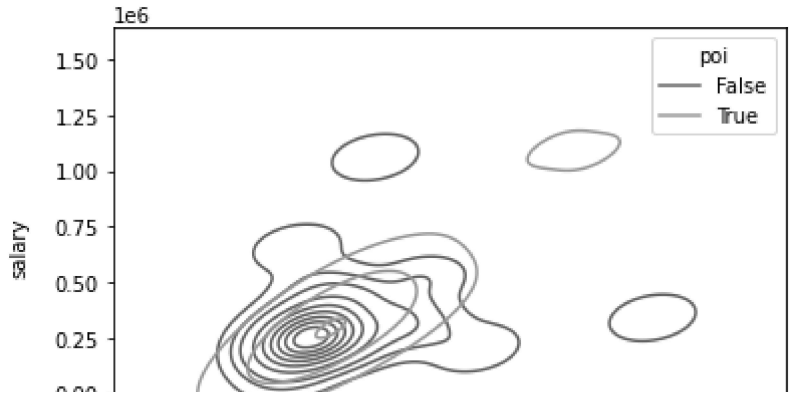
```
sns.stripplot(data=pf[pf['salary']<0.6*1e6], x="poi", y="salary", hue="poi")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0e9c3c90d0>
```



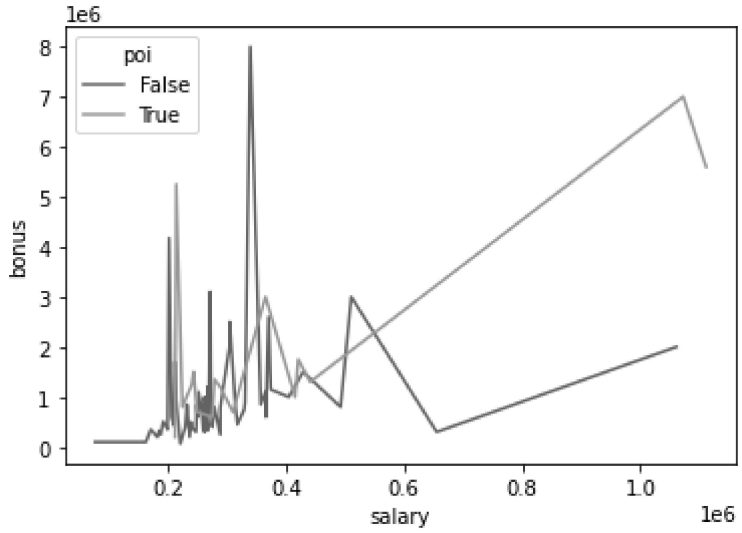

```
sns.kdeplot(data=pf[pf['salary']<1.0*1e7], x="bonus", y="salary", hue="poi")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0e98802490>
```



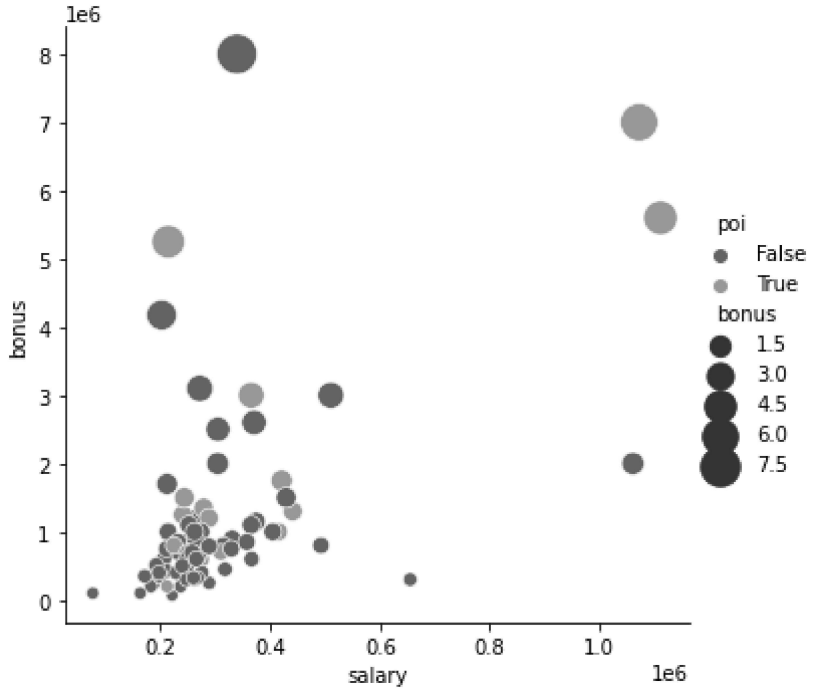

```
sns.lineplot(data=pf[pf['salary']<0.2*1e8], x="salary", y="bonus",  hue="poi")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0e987504d0>
```




```
sns.relplot(data=pf[pf['bonus']<0.2*1e8], x="salary", y="bonus",  hue="poi", size="bonus",
```

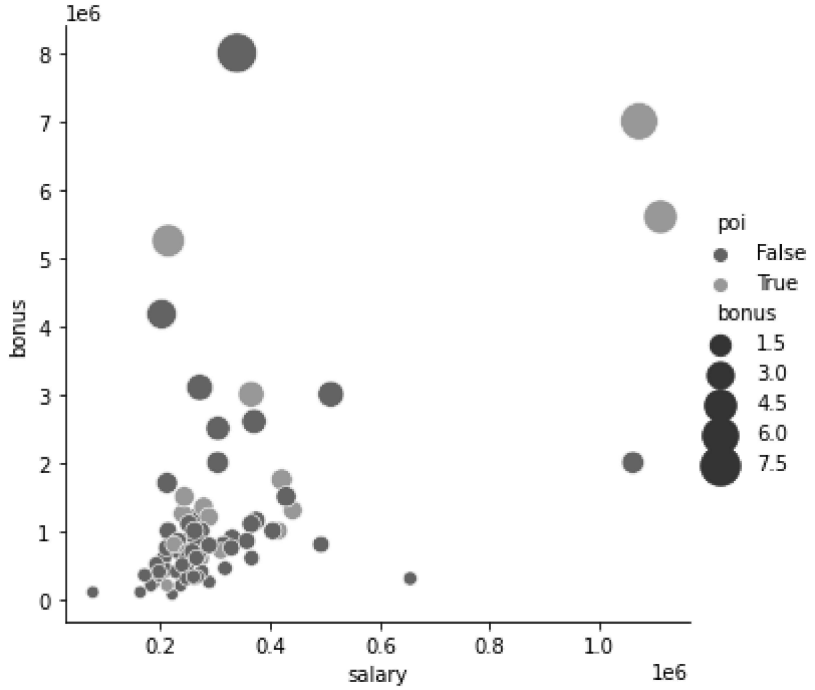
```
<seaborn.axisgrid.FacetGrid at 0x7f0e9875ac50>
```



Etapa 4

```
sns.relplot(data=pf[pf['bonus']<0.2*1e8], x="salary", y="bonus",  hue="poi", size="bonus",
```

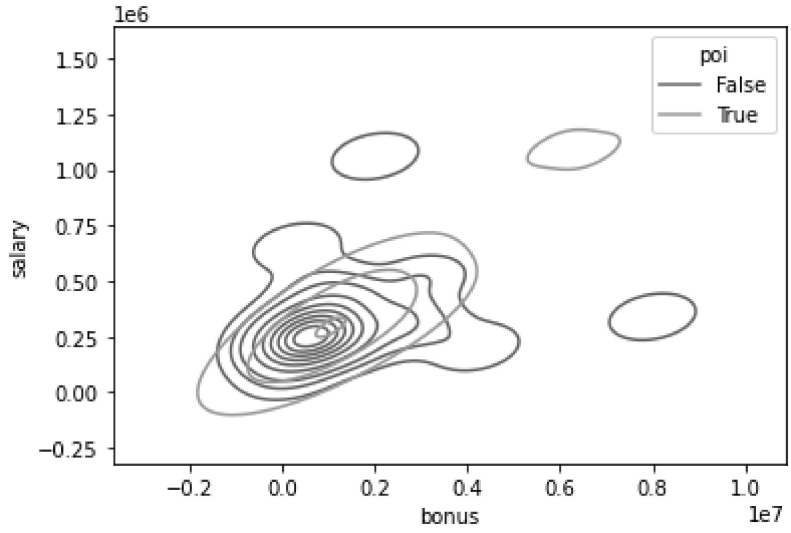
```
<seaborn.axisgrid.FacetGrid at 0x7f0e9861f310>
```



Eixo x salário e y bonus, com hue o poi, partir disso podemos ver que a maioria do pessoal que participou do esquema recebia relativamente o mesmo salário, diferenciando mesmo o bonus anual da empresa

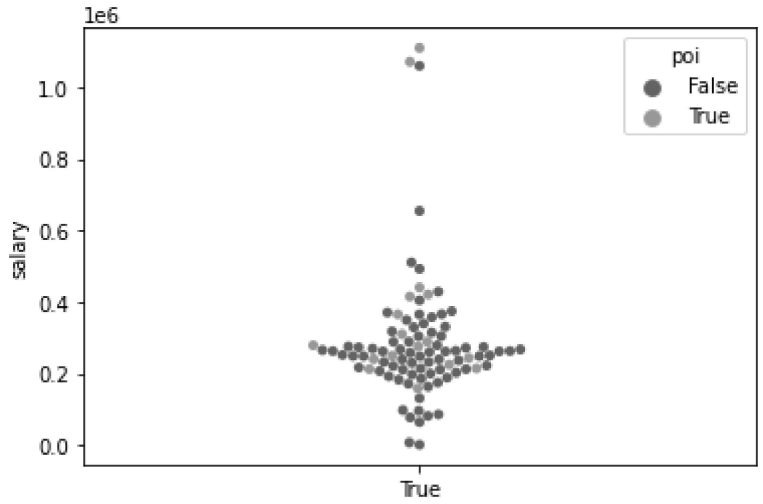
```
sns.kdeplot(data=pf[pf['salary']<1.0*1e7], x="bonus", y="salary", hue="poi")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0e98559710>
```



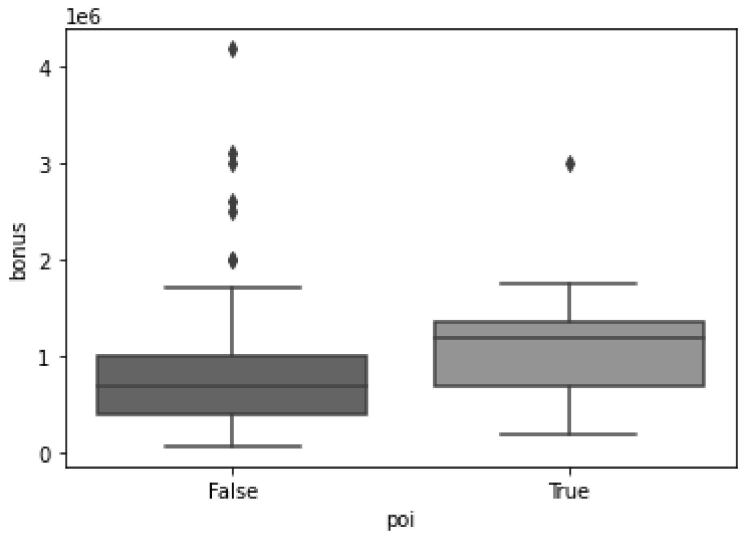

```
sns.swarmplot(data=pf[pf['salary']<0.2*1e8], x="email_address", y="salary", hue="poi")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0e984e9990>
```




```
sns.boxplot(data=pf[pf['bonus']<0.5*1e7], y='bonus', x='poi')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0e9843f5d0>
```




```
sns.lmplot(data=pf[pf['bonus']<0.2*1e8], x="salary", y="bonus", hue="poi")
```

```
<seaborn.axisgrid.FacetGrid at 0x7f0e9b887dd0>
```

Conclusão

Pessoal envolvido no exquema tiveram um bonus no geral bem mais, mesmo sendo minoria como tendo cargos relativamente baixo partindo do salário como pretexto, se benificiaram bastante com bonus anual da empresa como mostra a visualizações do grafico