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

In [6]:
```
emp = pd.read_excel(r'C:\sample\datafiles\Rawdata.xlsx')
```

emp

In [8]:
```
emp
```

Out[8]:

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Datascience#$	34 years	Mumbai	5^00#0	2+
1	Teddy^	Testing	45' yr	Bangalore	10%%000	<3
2	Uma#r	Dataanalyst^^#	NaN	NaN	1$5%000	4> yrs
3	Jane	Ana^^lytics	NaN	Hyderbad	2000^0	NaN
4	Uttam*	Statistics	67-yr	NaN	30000-	5+ year
5	Kim	NLP	55yr	Delhi	6000^$0	10+

In [10]:
```
id(emp)
```

Out[10]:
```
3128203585360
```

In [16]:
```
emp.columns
```

Out[16]:
```
Index(['Name', 'Domain', 'Age', 'Location', 'Salary', 'Exp'], dtype='object')
```

In [23]:
```
emp.shape
```

Out[23]:
```
(6, 6)
```

In [30]:
```
emp.head()
```

Out[30]:

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Datascience#$	34 years	Mumbai	5^00#0	2+
1	Teddy^	Testing	45' yr	Bangalore	10%%000	<3
2	Uma#r	Dataanalyst^^#	NaN	NaN	1$5%000	4> yrs
3	Jane	Ana^^lytics	NaN	Hyderbad	2000^0	NaN
4	Uttam*	Statistics	67-yr	NaN	30000-	5+ year

In [36]:
```
emp.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6 entries, 0 to 5
Data columns (total 6 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   Name      6 non-null      object
 1   Domain    6 non-null      object
 2   Age       4 non-null      object
 3   Location  4 non-null      object
 4   Salary    6 non-null      object
 5   Exp       5 non-null      object
dtypes: object(6)
memory usage: 420.0+ bytes
```

In [46]:
```
emp.isnull().sum()
```

Out[46]:
```
Name        0
Domain      0
Age         2
Location    2
Salary      0
Exp         1
dtype: int64
```

In [54]:
```
emp['Name']
```

Out[54]:
```
0      Mike
1    Teddy^
2     Uma#r
3      Jane
4    Uttam*
5       Kim
Name: Name, dtype: object
```

In [75]:
```
emp['Name'] = emp['Name'].str.replace(r'\W','',regex=True) # non word char
```

In [77]:
```
emp['Name']
```

Out[77]:
```
0     Mike
1    Teddy
2     Umar
3     Jane
4    Uttam
5      Kim
Name: Name, dtype: object
```

In [81]:
```
emp['Domain'] = emp['Domain'].str.replace(r'\W','',regex=True) # non word char
```

In [83]:
```
emp['Domain']
```

Out[83]:
```
0    Datascience
1        Testing
2    Dataanalyst
3      Analytics
4     Statistics
5            NLP
Name: Domain, dtype: object
```

In [94]:
```
emp['Age'] = emp['Age'].str.replace(r'\W','',regex=True) # non word char
```

In [98]:
```
emp['Age'] = emp['Age'].str.extract('(\\d+)')
```

In [102…
```
emp['Age']
```

Out[102…
```
0     34
1     45
2    NaN
3    NaN
4     67
5     55
Name: Age, dtype: object
```

In [106…
```
emp
```

Out[106…

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Datascience	34	Mumbai	5^00#0	2+
1	Teddy	Testing	45	Bangalore	10%%000	<3
2	Umar	Dataanalyst	NaN	NaN	1$5%000	4> yrs
3	Jane	Analytics	NaN	Hyderbad	2000^0	NaN
4	Uttam	Statistics	67	NaN	30000-	5+ year
5	Kim	NLP	55	Delhi	6000^$0	10+

In [109…
```
emp['Location'] = emp['Location'].str.replace(r'\W','',regex=True) # non word char
```

In [116…
```
emp
```

Out[116…

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Datascience	34	Mumbai	5^00#0	2+
1	Teddy	Testing	45	Bangalore	10%%000	<3
2	Umar	Dataanalyst	NaN	NaN	1$5%000	4> yrs
3	Jane	Analytics	NaN	Hyderbad	2000^0	NaN
4	Uttam	Statistics	67	NaN	30000-	5+ year
5	Kim	NLP	55	Delhi	6000^$0	10+

In [120…
```
emp['Salary'] = emp['Salary'].str.replace(r'\W','',regex=True) # non word char
```

In [122…
```
emp['Salary']
```

Out[122…
```
0     5000
1    10000
2    15000
3    20000
4    30000
5    60000
Name: Salary, dtype: object
```

In [126…
```
emp
```

Out[126…

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Datascience	34	Mumbai	5000	2+
1	Teddy	Testing	45	Bangalore	10000	<3
2	Umar	Dataanalyst	NaN	NaN	15000	4> yrs
3	Jane	Analytics	NaN	Hyderbad	20000	NaN
4	Uttam	Statistics	67	NaN	30000	5+ year
5	Kim	NLP	55	Delhi	60000	10+

In [131…
```
emp['Exp'] = emp['Exp'].str.replace(r'\W','',regex=True) # non word char
```

In [133…
```
emp['Exp'] = emp['Exp'].str.extract('(\\d+)')
```

In [135…
```
emp
```

Out[135…

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Datascience	34	Mumbai	5000	2
1	Teddy	Testing	45	Bangalore	10000	3
2	Umar	Dataanalyst	NaN	NaN	15000	4
3	Jane	Analytics	NaN	Hyderbad	20000	NaN
4	Uttam	Statistics	67	NaN	30000	5
5	Kim	NLP	55	Delhi	60000	10

In [139…
```
clean_data = emp.copy()
```

In [143…
```
clean_data
```

Out[143…

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Datascience	34	Mumbai	5000	2
1	Teddy	Testing	45	Bangalore	10000	3
2	Umar	Dataanalyst	NaN	NaN	15000	4
3	Jane	Analytics	NaN	Hyderbad	20000	NaN
4	Uttam	Statistics	67	NaN	30000	5
5	Kim	NLP	55	Delhi	60000	10

In [149…
```
clean_data
```

Out[149…

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Datascience	34	Mumbai	5000	2
1	Teddy	Testing	45	Bangalore	10000	3
2	Umar	Dataanalyst	NaN	NaN	15000	4
3	Jane	Analytics	NaN	Hyderbad	20000	NaN
4	Uttam	Statistics	67	NaN	30000	5
5	Kim	NLP	55	Delhi	60000	10

In [154…
```
clean_data.isnull().sum()
```

Out[154…
```
Name        0
Domain      0
Age         2
Location    2
Salary      0
Exp         1
dtype: int64
```

In [159…
```
clean_data['Age']
```

Out[159…
```
0     34
1     45
2    NaN
3    NaN
4     67
5     55
Name: Age, dtype: object
```

In [165…
```
import numpy as np
```

In [167…
```
clean_data['Age'] = clean_data['Age'].fillna(np.mean(pd.to_numeric(clean_data['Age'])))
```

In [169…
```
clean_data['Age']
```

Out[169…
```
0       34
1       45
2    50.25
3    50.25
4       67
5       55
Name: Age, dtype: object
```

In [171…
```
clean_data['Exp'] = clean_data['Exp'].fillna(np.mean(pd.to_numeric(clean_data['Exp'])))
```

In [173…
```
clean_data
```

Out[173…

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Datascience	34	Mumbai	5000	2
1	Teddy	Testing	45	Bangalore	10000	3
2	Umar	Dataanalyst	50.25	NaN	15000	4
3	Jane	Analytics	50.25	Hyderbad	20000	4.8
4	Uttam	Statistics	67	NaN	30000	5
5	Kim	NLP	55	Delhi	60000	10

In [178…
```
clean_data['Location'] = clean_data['Location'].fillna(clean_data['Location'].mode()[0])
```

In [182…
```
clean_data
```

Out[182…

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Datascience	34	Mumbai	5000	2
1	Teddy	Testing	45	Bangalore	10000	3
2	Umar	Dataanalyst	50.25	Bangalore	15000	4
3	Jane	Analytics	50.25	Hyderbad	20000	4.8
4	Uttam	Statistics	67	Bangalore	30000	5
5	Kim	NLP	55	Delhi	60000	10

In [188…
```
clean_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6 entries, 0 to 5
Data columns (total 6 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   Name      6 non-null      object
 1   Domain    6 non-null      object
 2   Age       6 non-null      object
 3   Location  6 non-null      object
 4   Salary    6 non-null      object
 5   Exp       6 non-null      object
dtypes: object(6)
memory usage: 420.0+ bytes
```

In [192…
```
emp.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6 entries, 0 to 5
Data columns (total 6 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   Name      6 non-null      object
 1   Domain    6 non-null      object
 2   Age       4 non-null      object
 3   Location  4 non-null      object
 4   Salary    6 non-null      object
 5   Exp       5 non-null      object
dtypes: object(6)
memory usage: 420.0+ bytes
```

In [195…
```
clean_data['Age'] = clean_data['Age'].astype(int)
```

In [212…
```
clean_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6 entries, 0 to 5
Data columns (total 6 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   Name      6 non-null      object
 1   Domain    6 non-null      object
 2   Age       6 non-null      int32 
 3   Location  6 non-null      object
 4   Salary    6 non-null      int32 
 5   Exp       6 non-null      int32 
dtypes: int32(3), object(3)
memory usage: 348.0+ bytes
```

In [213…
```
clean_data['Salary'] = clean_data['Salary'].astype(int)
```

In [216…
```
clean_data['Exp'] = clean_data['Exp'].astype(int)
```

In [222…
```
clean_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6 entries, 0 to 5
Data columns (total 6 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   Name      6 non-null      object
 1   Domain    6 non-null      object
 2   Age       6 non-null      int32 
 3   Location  6 non-null      object
 4   Salary    6 non-null      int32 
 5   Exp       6 non-null      int32 
dtypes: int32(3), object(3)
memory usage: 348.0+ bytes
```

In [224…
```
clean_data['Name'] = clean_data['Name'].astype('category')
clean_data['Domain'] = clean_data['Domain'].astype('category')
clean_data['Location'] = clean_data['Location'].astype('category')
```

In [230…
```
clean_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6 entries, 0 to 5
Data columns (total 6 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Name      6 non-null      category
 1   Domain    6 non-null      category
 2   Age       6 non-null      int32
 3   Location  6 non-null      category
 4   Salary    6 non-null      int32
 5   Exp       6 non-null      int32
dtypes: category(3), int32(3)
memory usage: 866.0 bytes
```

In [240…
```
clean_data.to_csv('C:\\Users\\Administrator\\clean_data.csv')
```

In [238…
```
import os
os.getcwd()
```

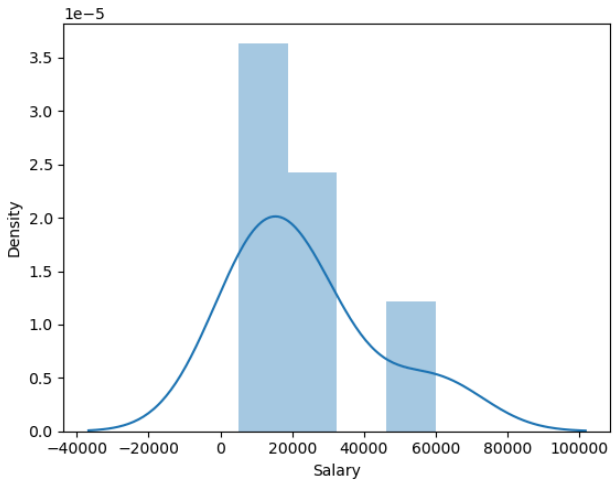
Out[238… 'C:\\Users\\Administrator'

In [244…
```
import matplotlib.pyplot as plt
```

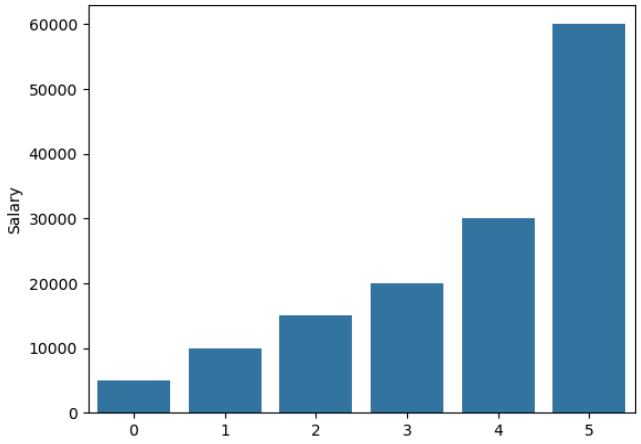
In [246…
```
import seaborn as sns
```

In [248…
```
import warnings
warnings.filterwarnings('ignore')
```

In [263…
```
vs1 = sns.distplot(clean_data['Salary']) # displot plot
plt.show(vs1)
```

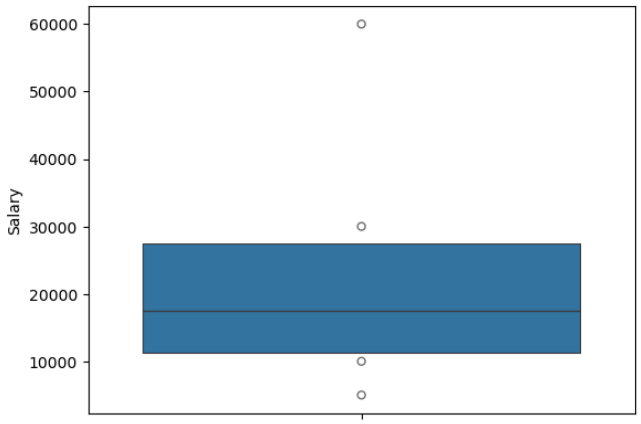


In [277…
```
vs2 = sns.barplot(clean_data['Salary']) # bar plot
plt.show(vs2)
```



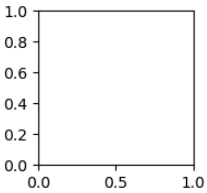
In [279…
```
vs3 = sns.dogplot(clean_data['Salary'])
```

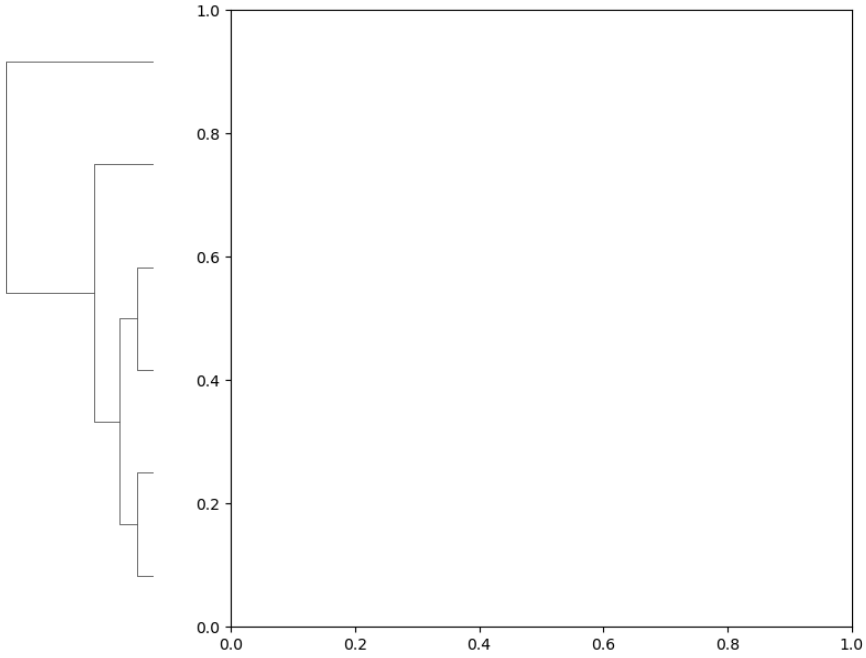
In [284…
```
vs4 = sns.boxenplot(clean_data['Salary'])
```



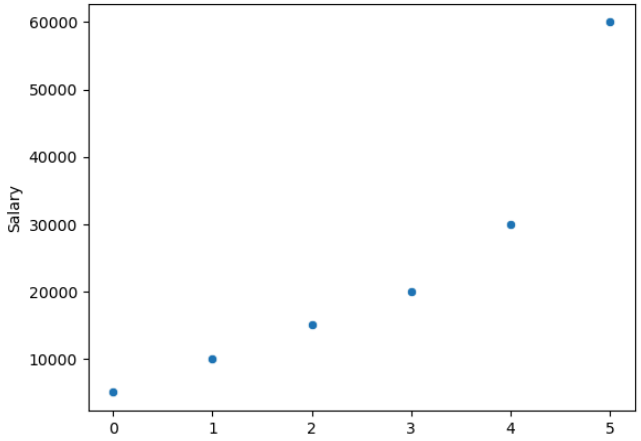
In [294…
```
vs4 = sns.clustermap(clean_data['Salary'])
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[294], line 1
----> 1 vs4 = sns.clustermap(clean_data['Salary'])

File C:\ProgramData\anaconda3\Lib\site-packages\seaborn\matrix.py:1258, in clustermap(data, pivot_kws, method, metric, z_score, standard_scale, figsize, cbar_kws, row_cluster, col_cluster, row_linkage, col_linkage, row_colors, col_colors, mask, dendrogram_ratio, colors_ratio, cbar_pos, tree_kws, **kwargs)
   1250     raise RuntimeError("clustermap requires scipy to be available")
   1252 plotter = ClusterGrid(data, pivot_kws=pivot_kws, figsize=figsize,
   1253                       row_colors=row_colors, col_colors=col_colors,
   1254                       z_score=z_score, standard_scale=standard_scale,
   1255                       mask=mask, dendrogram_ratio=dendrogram_ratio,
   1256                       colors_ratio=colors_ratio, cbar_pos=cbar_pos)
-> 1258 return plotter.plot(metric=metric, method=method,
   1259                     colorbar_kws=cbar_kws,
   1260                     row_cluster=row_cluster, col_cluster=col_cluster,
   1261                     row_linkage=row_linkage, col_linkage=col_linkage,
   1262                     tree_kws=tree_kws, **kwargs)

File C:\ProgramData\anaconda3\Lib\site-packages\seaborn\matrix.py:1129, in ClusterGrid.plot(self, metric, method, colorbar_kws, row_cluster, col_cluster, row_linkage, col_linkage, tree_kws, **kws)
   1125     kws.pop("square")
   1127 colorbar_kws = {} if colorbar_kws is None else colorbar_kws
-> 1129 self.plot_dendrograms(row_cluster, col_cluster, metric, method,
   1130                       row_linkage=row_linkage, col_linkage=col_linkage,
   1131                       tree_kws=tree_kws)
   1132 try:
   1133     xind = self.dendrogram_col.reordered_ind

File C:\ProgramData\anaconda3\Lib\site-packages\seaborn\matrix.py:984, in ClusterGrid.plot_dendrograms(self, row_cluster, col_cluster, metric, method, row_linkage, col_linkage, tree_kws)
    982 # PLot the column dendrogram
    983 if col_cluster:
--> 984     self.dendrogram_col = dendrogram(
    985         self.data2d, metric=metric, method=method, label=False,
    986         axis=1, ax=self.ax_col_dendrogram, linkage=col_linkage,
    987         tree_kws=tree_kws
    988     )
    989 else:
    990     self.ax_col_dendrogram.set_xticks([])

File C:\ProgramData\anaconda3\Lib\site-packages\seaborn\matrix.py:687, in dendrogram(data, linkage, axis, label, metric, method, rotate, tree_kws, ax)
    684 if _no_scipy:
    685     raise RuntimeError("dendrogram requires scipy to be installed")
--> 687 plotter = _DendrogramPlotter(data, linkage=linkage, axis=axis,
    688                               metric=metric, method=method,
    689                               label=label, rotate=rotate)
    690 if ax is None:
    691     ax = plt.gca()

File C:\ProgramData\anaconda3\Lib\site-packages\seaborn\matrix.py:495, in _DendrogramPlotter.__init__(self, data, linkage, metric, method, axis, label, rotate)
    492 self.rotate = rotate
    494 if linkage is None:
--> 495     self.linkage = self.calculated_linkage
    496 else:
    497     self.linkage = linkage

File C:\ProgramData\anaconda3\Lib\site-packages\seaborn\matrix.py:562, in _DendrogramPlotter.calculated_linkage(self)
    558         msg = ("Clustering large matrix with scipy. Installing "
    559                "`fastcluster` may give better performance.")
    560         warnings.warn(msg)
--> 562 return self._calculate_linkage_scipy()

File C:\ProgramData\anaconda3\Lib\site-packages\seaborn\matrix.py:530, in _DendrogramPlotter._calculate_linkage_scipy(self)
    529 def _calculate_linkage_scipy(self):
--> 530     linkage = hierarchy.linkage(self.array, method=self.method,
    531                                 metric=self.metric)
    532     return linkage

File C:\ProgramData\anaconda3\Lib\site-packages\scipy\cluster\hierarchy.py:1033, in linkage(y, method, metric, optimal_ordering)
   1029 if not xp.all(xp.isfinite(y)):
   1030     raise ValueError("The condensed distance matrix must contain only "
   1031                      "finite values.")
-> 1033 n = int(distance.num_obs_y(y))
   1034 method_code = _LINKAGE_METHODS[method]
   1036 y = np.asarray(y)

File C:\ProgramData\anaconda3\Lib\site-packages\scipy\spatial\distance.py:2605, in num_obs_y(Y)
   2603 k = Y.shape[0]
   2604 if k == 0:
-> 2605     raise ValueError("The number of observations cannot be determined on "
   2606                      "an empty distance matrix.")
   2607 d = int(np.ceil(np.sqrt(k * 2)))
   2608 if (d * (d - 1) / 2) != k:

ValueError: The number of observations cannot be determined on an empty distance matrix.
```

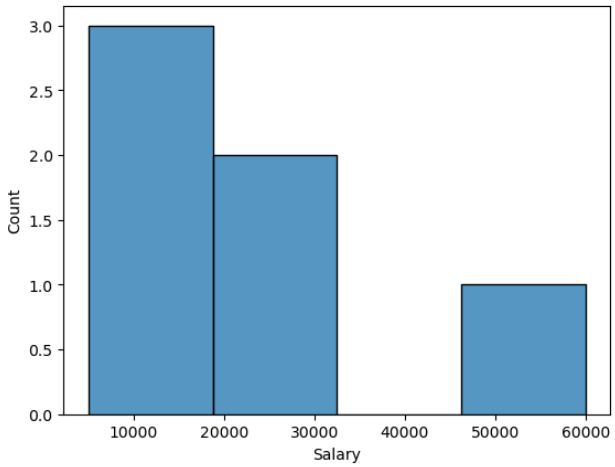




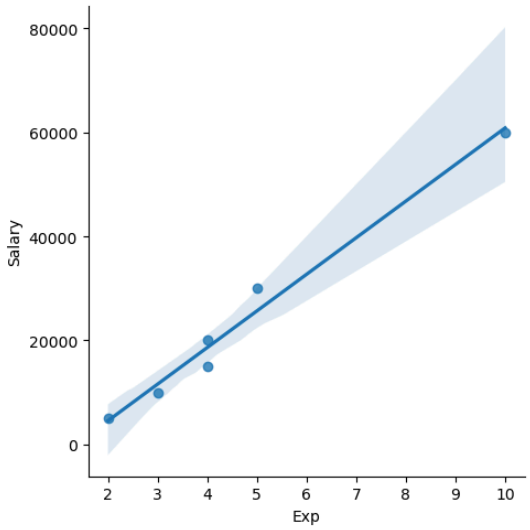
In [303…
```
vs4 = sns.scatterplot(clean_data['Salary'])
```



In [308…
```
vs5 = sns.histplot(clean_data['Salary'])
```



In [314…
```
vs6 = sns.lmplot(data=clean_data, x= 'Exp',y= 'Salary', fit_reg=True)
```



In [320…
```
clean_data
```

Out[320…

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Datascience	34	Mumbai	5000	2
1	Teddy	Testing	45	Bangalore	10000	3
2	Umar	Dataanalyst	50	Bangalore	15000	4
3	Jane	Analytics	50	Hyderbad	20000	4
4	Uttam	Statistics	67	Bangalore	30000	5
5	Kim	NLP	55	Delhi	60000	10

In [326…
```
clean_data.columns
```

Out[326…
```
Index(['Name', 'Domain', 'Age', 'Location', 'Salary', 'Exp'], dtype='object')
```

In [334…
```
imputation = pd.get_dummies(clean_data, dtype=int)
```

In [336…
```
imputation
```

Out[336…

	Age	Salary	Exp	Name_Jane	Name_Kim	Name_Mike	Name_Teddy	Name_Umar	Name_Uttam	Domain_Analytics	Domain_Dataanalyst	Domain_Datascience	Domain_NLP	Dom
0	34	5000	2	0	0	1	0	0	0	0	0	1	0	
1	45	10000	3	0	0	0	1	0	0	0	0	0	0	
2	50	15000	4	0	0	0	0	1	0	0	1	0	0	
3	50	20000	4	1	0	0	0	0	0	1	0	0	0	
4	67	30000	5	0	0	0	0	0	1	0	0	0	0	
5	55	60000	10	0	1	0	0	0	0	0	0	0	1	

In [ ]:

In [ ]: