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

In [2]:
```
pd.__version__
```

Out[2]:
```
'2.2.2'
```

In [3]:
```
emp=pd.read_excel(r"C:\Users\Jan Saida\Downloads\Rawdata.xlsx")
```

In [4]:
```
emp
```

Out[4]:

	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 [5]:
```
id(emp)
```

Out[5]:
```
2512612543040
```

In [6]:
```
emp.columns
```

Out[6]:
```
Index(['Name', 'Domain', 'Age', 'Location', 'Salary', 'Exp'], dtype='object')
```

In [7]:
```
emp.shape
```

Out[7]:
```
(6, 6)
```

In [8]:
```
emp.head()
```

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

In [9]:
```
emp.tail()
```

Out[9]:

	Name	Domain	Age	Location	Salary	Exp
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]:
```
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 [11]:
```
emp.isnull()
```

Out[11]:

	Name	Domain	Age	Location	Salary	Exp
0	False	False	False	False	False	False
1	False	False	False	False	False	False
2	False	False	True	True	False	False
3	False	False	True	False	False	True
4	False	False	False	True	False	False
5	False	False	False	False	False	False

In [12]:
```
emp.isna()
```

Out[12]:

	Name	Domain	Age	Location	Salary	Exp
0	False	False	False	False	False	False
1	False	False	False	False	False	False
2	False	False	True	True	False	False
3	False	False	True	False	False	True
4	False	False	False	True	False	False
5	False	False	False	False	False	False

In [13]:
```
emp.isnull().sum()
```

Out[13]:
```
Name        0
Domain      0
Age         2
Location    2
Salary      0
Exp         1
dtype: int64
```

DATA CLEANING OR DATA CLEANSING

In [15]:
```
emp['Name']
```

Out[15]:
```
0      Mike
1    Teddy^
2     Uma#r
3      Jane
4    Uttam*
5       Kim
Name: Name, dtype: object
```

In [16]:
```
emp['Name']=emp['Name'].str.replace(r'\W','',regex=True)
```

In [17]:
```
emp['Name']
```

Out[17]:
```
0     Mike
1    Teddy
2     Umar
3     Jane
4    Uttam
5      Kim
Name: Name, dtype: object
```

In [18]:
```
emp['Domain']=emp['Domain'].str.replace(r'\W','',regex=True)
```

In [19]:
```
emp['Domain']
```

Out[19]:
```
0    Datascience
1        Testing
2    Dataanalyst
3      Analytics
4     Statistics
5            NLP
Name: Domain, dtype: object
```

In [20]:
```
emp['Age']=emp['Age'].str.replace(r'\W','',regex=True)
```

In [21]:
```
emp['Age']
```

Out[21]:
```
0    34years
1       45yr
2        NaN
3        NaN
4       67yr
5       55yr
Name: Age, dtype: object
```

In [22]:
```
emp['Age']=emp['Age'].str.extract(r'(\d+)')
```

In [23]:
```
emp['Age']
```

Out[23]:
```
0     34
1     45
2    NaN
3    NaN
4     67
5     55
Name: Age, dtype: object
```

In [24]:
```
emp
```

Out[24]:

	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 [25]:
```
emp['Location']=emp['Location'].str.replace(r'\W','',regex=True)
```

In [26]:
```
emp['Location']
```

Out[26]:
```
0       Mumbai
1    Bangalore
2          NaN
3     Hyderbad
4          NaN
5        Delhi
Name: Location, dtype: object
```

In [27]:
```
emp['Salary']=emp['Salary'].str.replace(r'\W','',regex=True)
```

In [28]:
```
emp['Salary']
```

Out[28]:
```
0     5000
1    10000
2    15000
3    20000
4    30000
5    60000
Name: Salary, dtype: object
```

In [29]:
```
emp['Exp']=emp['Exp'].str.extract(r'(\d+)')
```

In [30]:
```
emp['Exp']
```

Out[30]:
```
0      2
1      3
2      4
3    NaN
4      5
5     10
Name: Exp, dtype: object
```

In [31]:
```
emp
```

Out[31]:

	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 [32]:
```
clean_data=emp.copy()
```

In [33]:
```
clean_data
```

Out[33]:

	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

EDA TECHNIQUES

1- MISSING VALUE TREATMENT

In [36]:
```
clean_data
```

Out[36]:

	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 [37]:
```
clean_data.isnull().sum()
```

Out[37]:
```
Name        0
Domain      0
Age         2
Location    2
Salary      0
Exp         1
dtype: int64
```

In [38]:
```
clean_data['Age']
```

Out[38]:
```
0     34
1     45
2    NaN
3    NaN
4     67
5     55
Name: Age, dtype: object
```

In [39]:
```
import numpy as np
```

In [40]:
```
clean_data['Age']=clean_data['Age'].fillna(np.mean(pd.to_numeric(clean_data['Age'])))
```

In [41]:
```
clean_data['Age']
```

Out[41]:
```
0       34
1       45
2    50.25
3    50.25
4       67
5       55
Name: Age, dtype: object
```

In [42]:
```
clean_data['Exp']
```

Out[42]:
```
0      2
1      3
2      4
3    NaN
4      5
5     10
Name: Exp, dtype: object
```

In [43]:
```
clean_data['Exp']=clean_data['Exp'].fillna(np.mean(pd.to_numeric(clean_data['Exp'])))
```

In [44]:
```
clean_data['Exp']
```

Out[44]:
```
0      2
1      3
2      4
3    4.8
4      5
5     10
Name: Exp, dtype: object
```

In [45]:
```
clean_data['Location']=clean_data['Location'].fillna(clean_data['Location'].mode()[0])
```

In [46]:
```
clean_data['Location']
```

Out[46]:
```
0       Mumbai
1    Bangalore
2    Bangalore
3     Hyderbad
4    Bangalore
5        Delhi
Name: Location, dtype: object
```

In [47]:
```
clean_data
```

Out[47]:

	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 [48]:
```
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 [49]:
```
clean_data['Age']=clean_data['Age'].astype(int)
```

In [50]:
```
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      object
 5   Exp       6 non-null      object
dtypes: int32(1), object(5)
memory usage: 396.0+ bytes
```

In [51]:
```
clean_data['Salary']=clean_data['Salary'].astype(int)
```

In [52]:
```
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      object
dtypes: int32(2), object(4)
memory usage: 372.0+ bytes
```

In [53]:
```
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 [54]:
```
clean_data['Exp']=clean_data['Exp'].astype(int)
```

In [55]:
```
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 [56]:
```
clean_data
```

Out[56]:

	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 [57]:
```
clean_data.to_csv('clean_data.csv')
```

In [58]:
```
import os
os.getcwd()
```

Out[58]:
```
'C:\\Users\\Jan Saida'
```

In [59]:
```
clean_data
```

Out[59]:

	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 [60]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

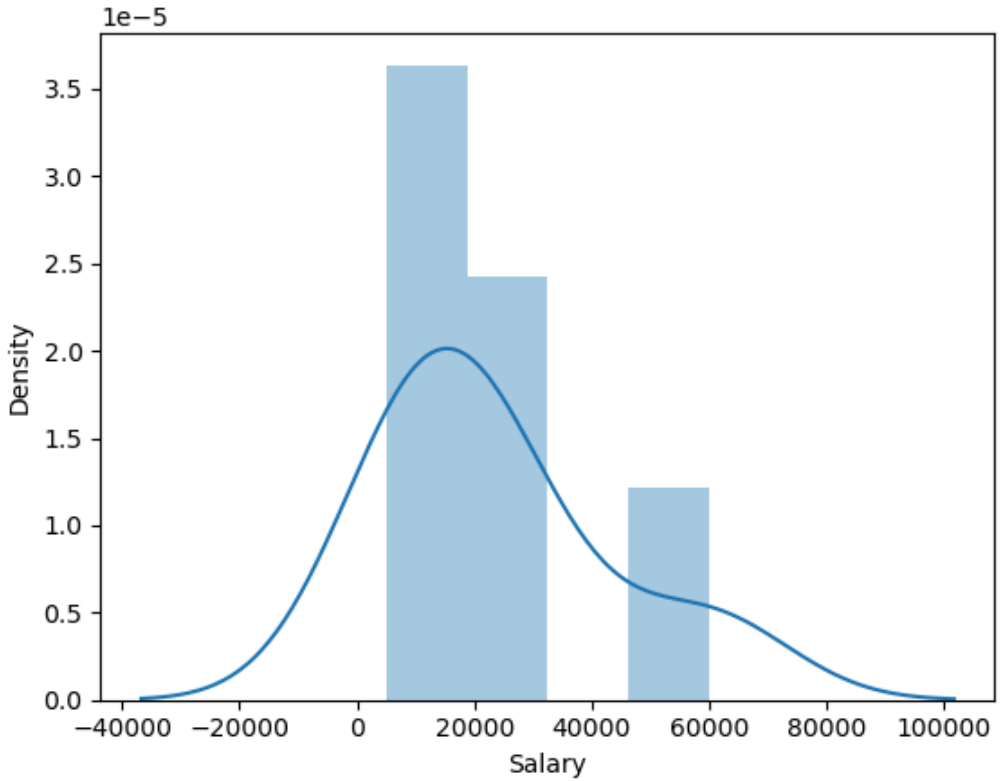
In [61]:
```
import warnings
warnings.filterwarnings('ignore')
```

UNIVARIATE ANALYSIS

In [63]:
```
clean_data['Salary']
```

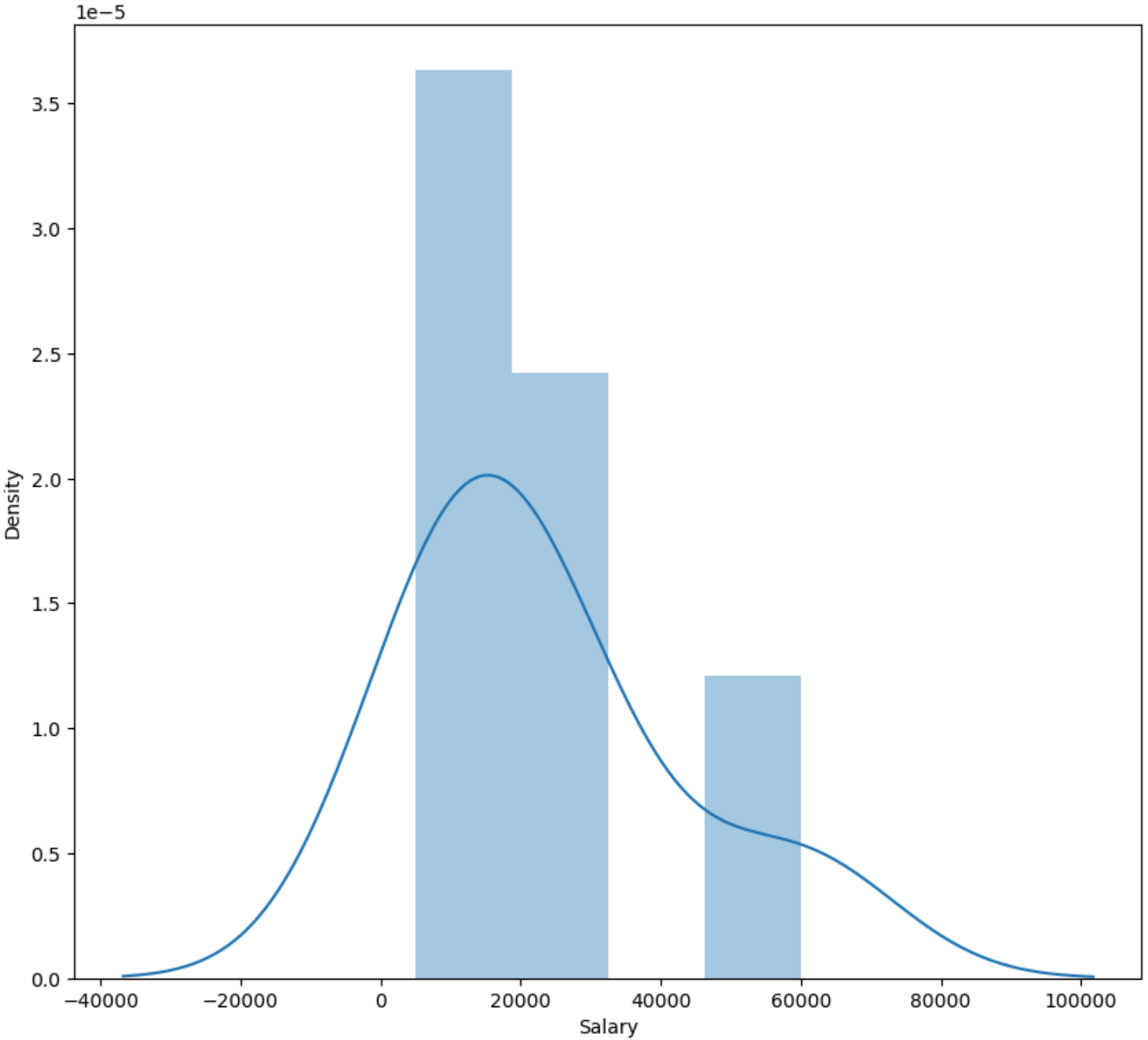
Out[63]:
```
0     5000
1    10000
2    15000
3    20000
4    30000
5    60000
Name: Salary, dtype: int32
```

In [64]:
```
vis1=sns.distplot(clean_data['Salary'])
```

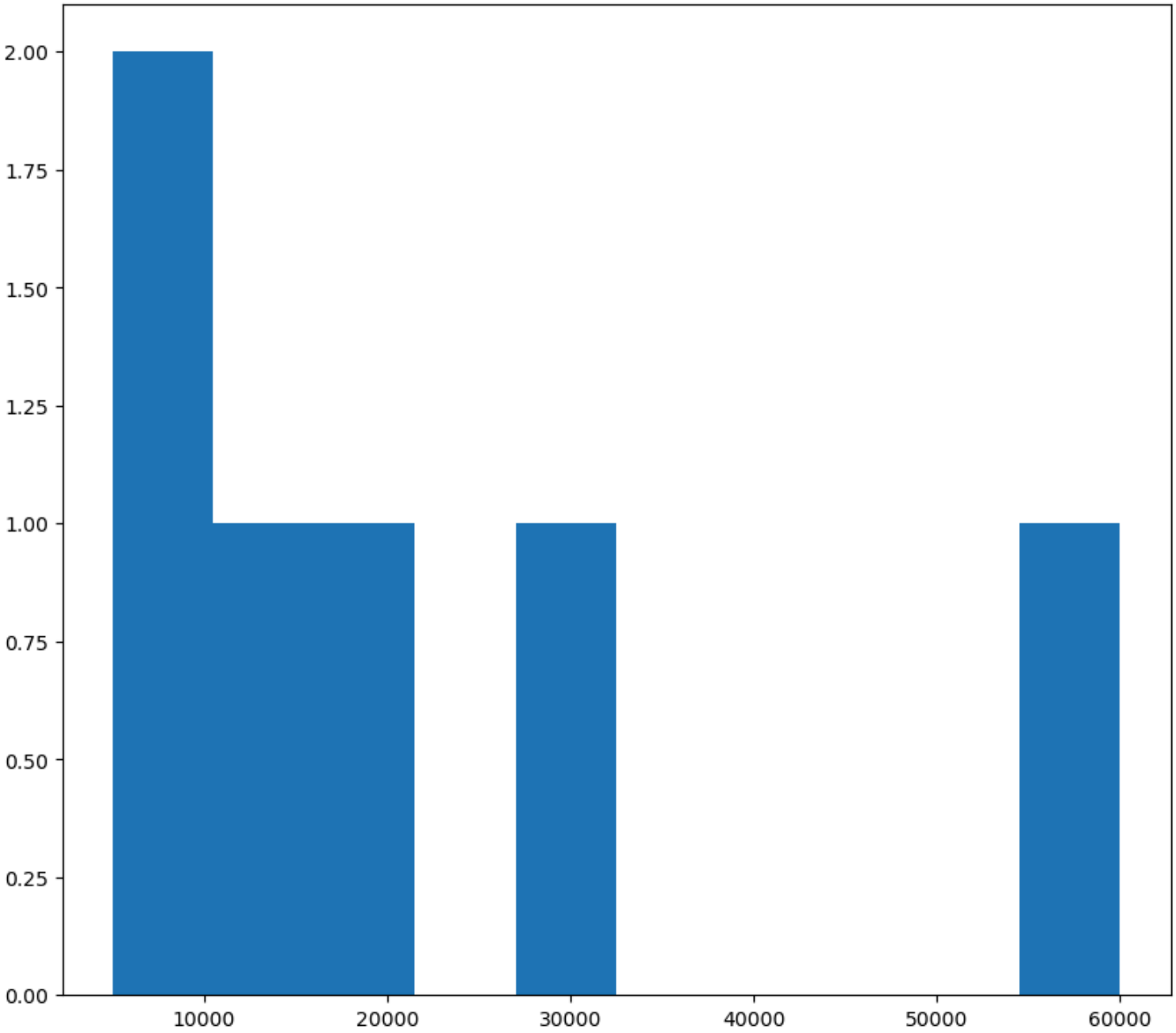


In [65]:
```
plt.rcParams['figure.figsize']=10,9
```

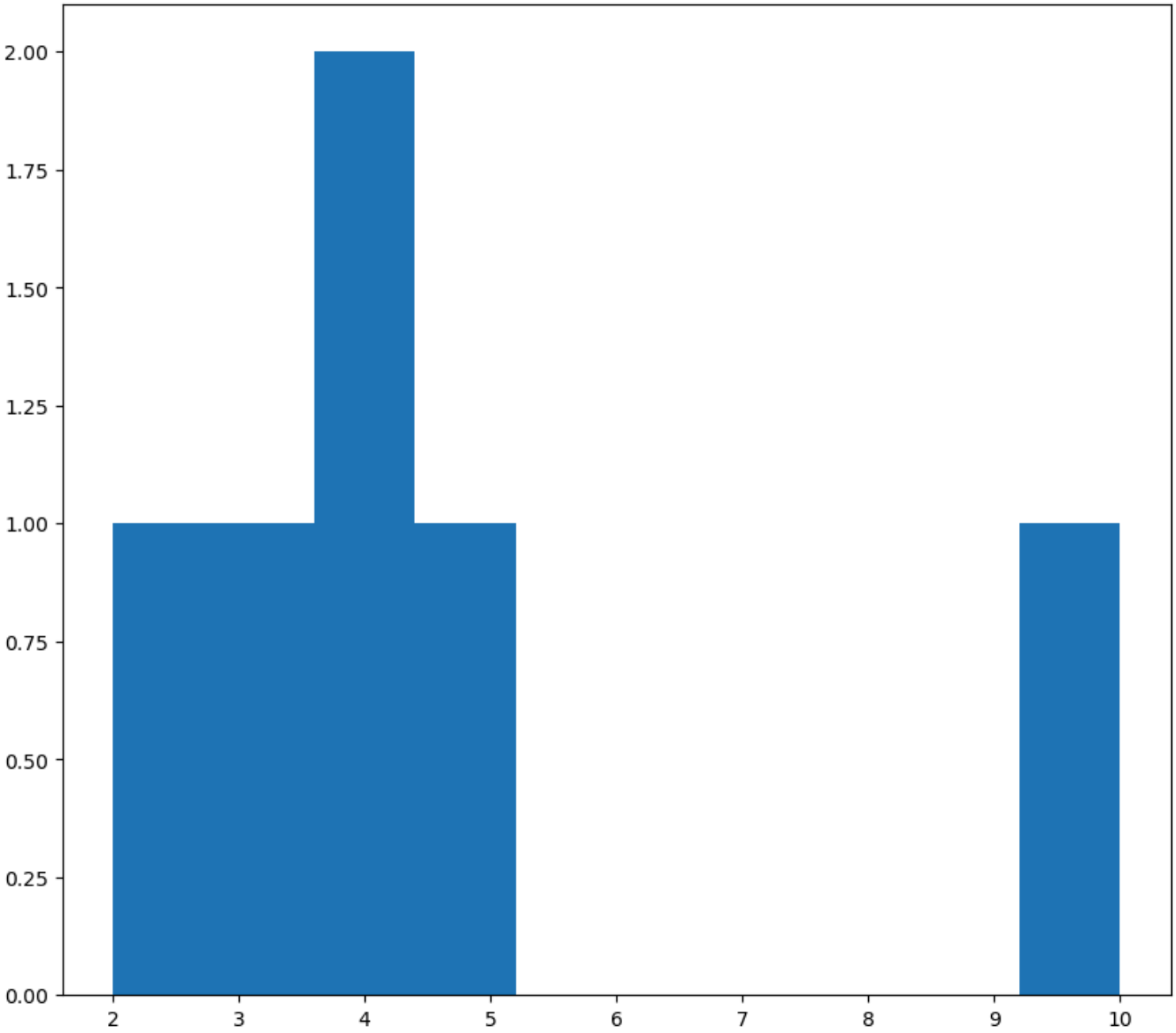
In [66]:
```
vis1=sns.distplot(clean_data['Salary'])
```


1e−5
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
Density
−40000
−20000
0
20000
40000
60000
80000
100000
Salary

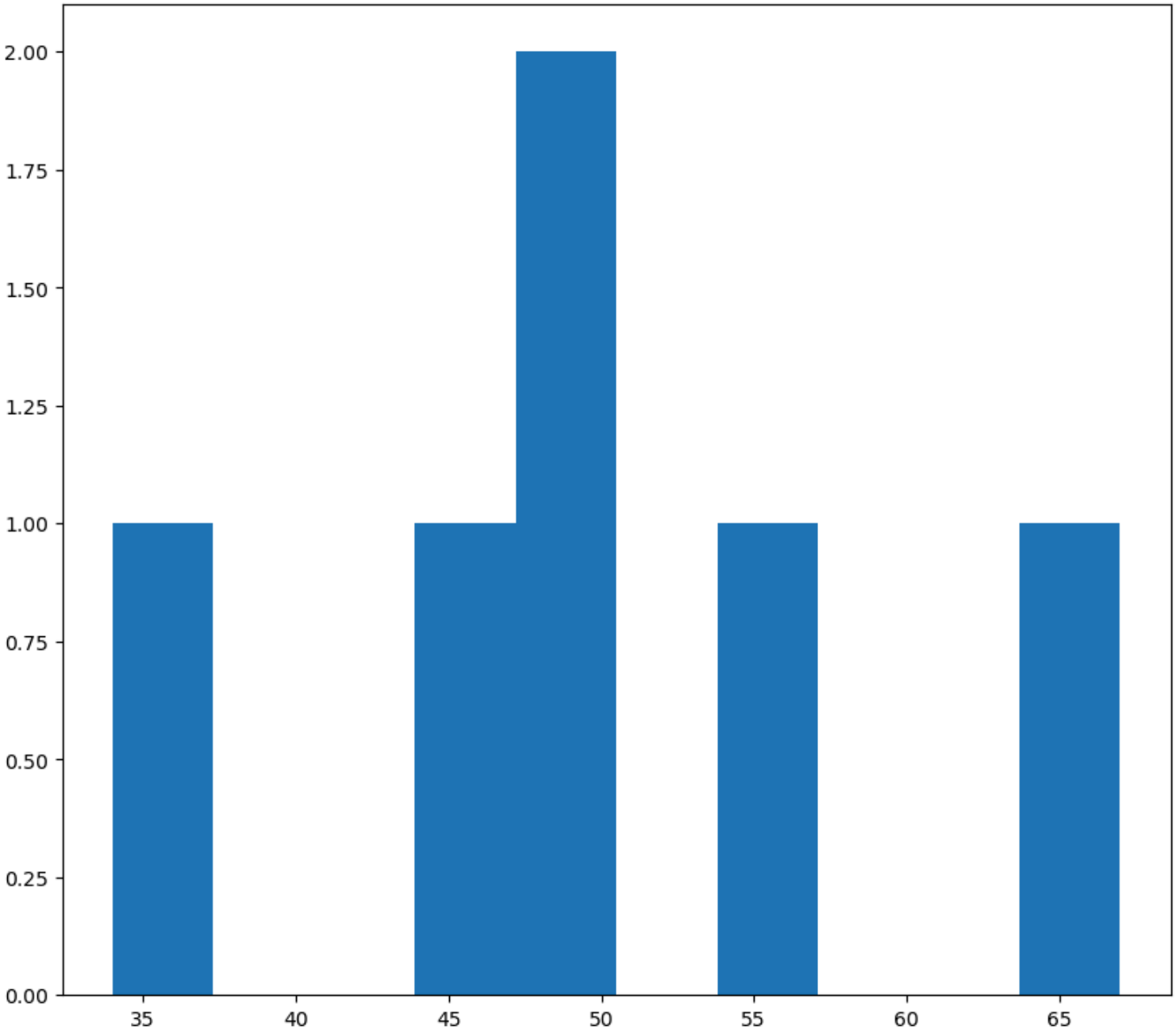
In [67]:
```
vis2=plt.hist(clean_data['Salary'])
```


2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
10000
20000
30000
40000
50000
60000

In [68]:
```
vis3=plt.hist(clean_data['Exp'])
```

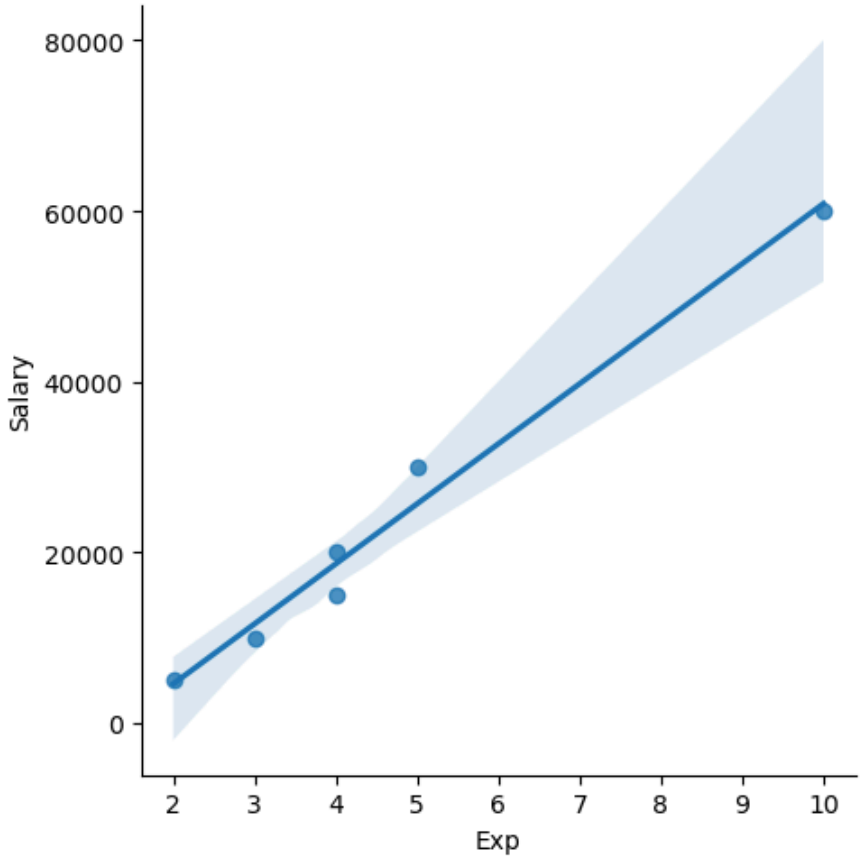

2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
2
3
4
5
6
7
8
9
10

In [69]:
```
vis4=plt.hist(clean_data['Age'])
```

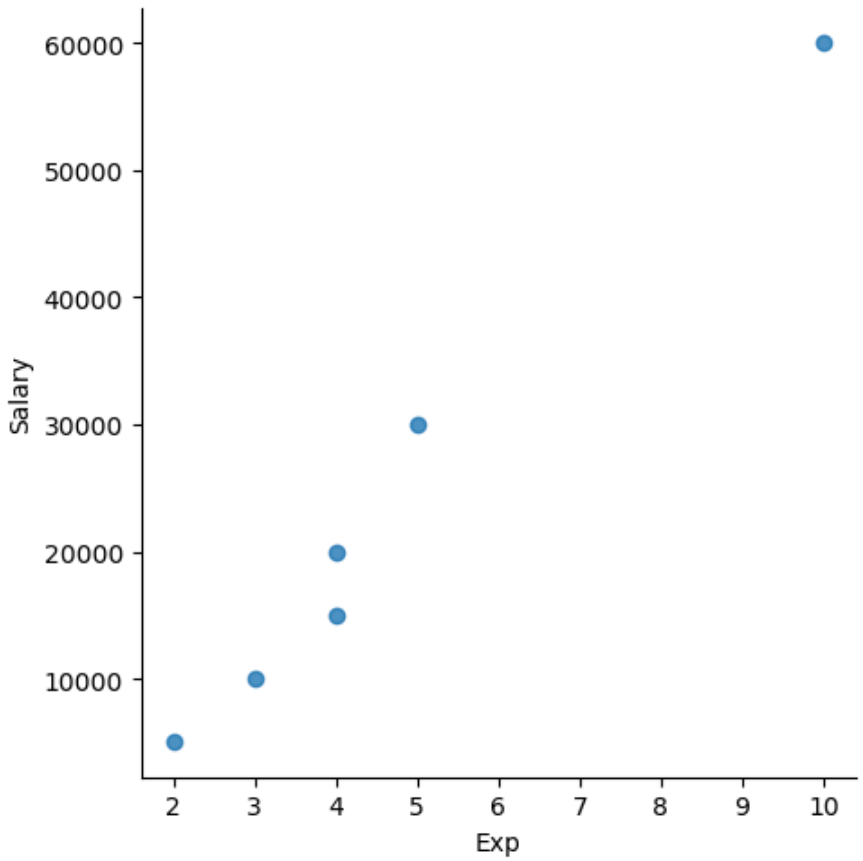

2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
35
40
45
50
55
60
65

BIVARIATE ANALYSIS

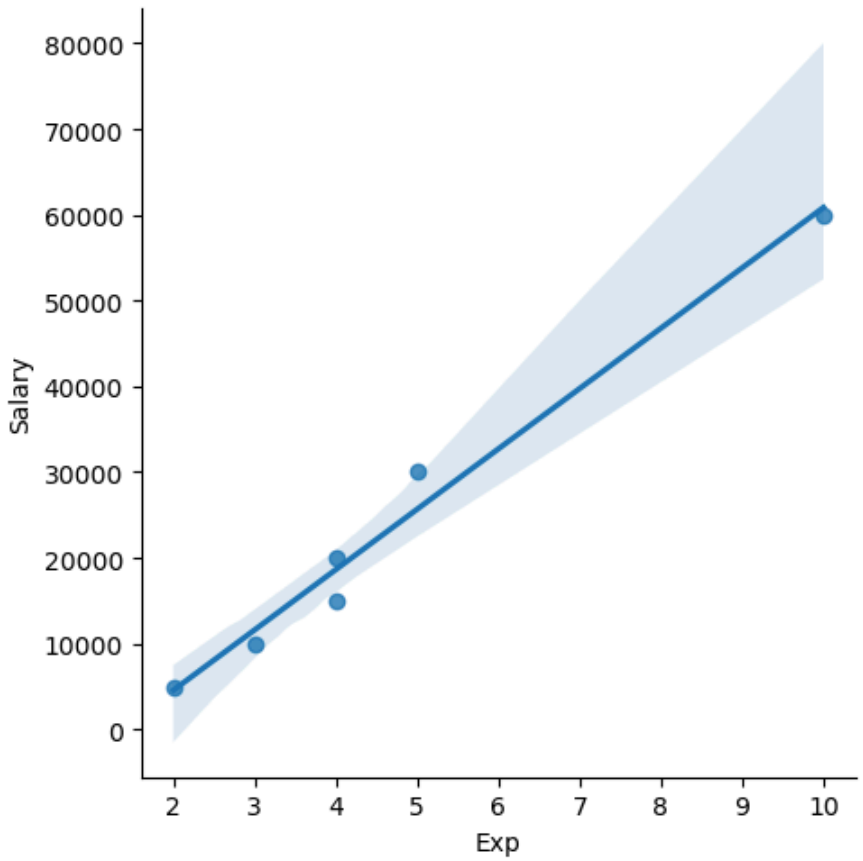
In [71]:
```
vis5=sns.lmplot(data=clean_data, x='Exp',y='Salary')
```



In [72]:
```
vis6=sns.lmplot(data=clean_data,x='Exp',y='Salary',fit_reg=False)
```



In [73]:
```
vis6=sns.lmplot(data=clean_data,x='Exp',y='Salary',fit_reg=True)
```



In [74]:
```
clean_data
```

Out[74]:

	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 [75]:
```
clean_data[:]
```

Out[75]:

	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 [76]:
```
clean_data[4:]
```

Out[76]:

	Name	Domain	Age	Location	Salary	Exp
4	Uttam	Statistics	67	Bangalore	30000	5
5	Kim	NLP	55	Delhi	60000	10

In [77]:
```
clean_data[:3]
```

Out[77]:

	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

In [78]:
```
clean_data[::2]
```

Out[78]:

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Datascience	34	Mumbai	5000	2
2	Umar	Dataanalyst	50	Bangalore	15000	4
4	Uttam	Statistics	67	Bangalore	30000	5

In [79]:
```
clean_data[2::]
```

Out[79]:

	Name	Domain	Age	Location	Salary	Exp
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 [80]:
```
clean_data[2,3]
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
File ~\anaconda3\Lib\site-packages\pandas\core\indexes\base.py:3805, in Index.get_loc(self, key)
   3804 try:
-> 3805     return self._engine.get_loc(casted_key)
   3806 except KeyError as err:

File index.pyx:167, in pandas._libs.index.IndexEngine.get_loc()

File index.pyx:196, in pandas._libs.index.IndexEngine.get_loc()

File pandas\\_libs\\hashtable_class_helper.pxi:7081, in pandas._libs.hashtable.PyObjectHashTable.get_item()

File pandas\\_libs\\hashtable_class_helper.pxi:7089, in pandas._libs.hashtable.PyObjectHashTable.get_item()

KeyError: (2, 3)

The above exception was the direct cause of the following exception:

KeyError                                  Traceback (most recent call last)
Cell In[80], line 1
----> 1 clean_data[2,3]

File ~\anaconda3\Lib\site-packages\pandas\core\frame.py:4102, in DataFrame.__getitem__(self, key)
   4100 if self.columns.nlevels > 1:
   4101     return self._getitem_multilevel(key)
-> 4102 indexer = self.columns.get_loc(key)
   4103 if is_integer(indexer):
   4104     indexer = [indexer]

File ~\anaconda3\Lib\site-packages\pandas\core\indexes\base.py:3812, in Index.get_loc(self, key)
   3807     if isinstance(casted_key, slice) or (
   3808         isinstance(casted_key, abc.Iterable)
   3809         and any(isinstance(x, slice) for x in casted_key)
   3810     ):
   3811         raise InvalidIndexError(key)
-> 3812     raise KeyError(key) from err
   3813 except TypeError:
   3814     # If we have a listlike key, _check_indexing_error will raise
   3815     #  InvalidIndexError. Otherwise we fall through and re-raise
   3816     #  the TypeError.
   3817     self._check_indexing_error(key)

KeyError: (2, 3)
```

In [90]:
```
clean_data
```

Out[90]:

	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 [92]:
```
y_iv=clean_data.drop(['Name','Domain','Age','Location','Salary'],axis=1)
```

In [94]:
```
y_iv
```

Out[94]:

	Exp
0	2
1	3
2	4
3	4
4	5
5	10

In [96]:
```
y_iv.columns
```

Out[96]:
```
Index(['Exp'], dtype='object')
```

In [98]:
```
x_dv=clean_data.drop(['Exp'],axis=1)
```

In [100…
```
x_dv
```

Out[100...

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

In [102...
```
x_dv.columns
```

Out[102...
```
Index(['Name', 'Domain', 'Age', 'Location', 'Salary'], dtype='object')
```

In [104...
```
clean_data.columns
```

Out[104...
```
Index(['Name', 'Domain', 'Age', 'Location', 'Salary', 'Exp'], dtype='object')
```

In [106...
```
imputation=pd.get_dummies(clean_data)
```

In [108...
```
imputation
```

Out[108...

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

In [110...
```
imputation.columns
```

Out[110…

```
Index(['Age', 'Salary', 'Exp', 'Name_Jane', 'Name_Kim', 'Name_Mike',
       'Name_Teddy', 'Name_Umar', 'Name_Uttam', 'Domain_Analytics',
       'Domain_Dataanalyst', 'Domain_Datascience', 'Domain_NLP',
       'Domain_Statistics', 'Domain_Testing', 'Location_Bangalore',
       'Location_Delhi', 'Location_Hyderbad', 'Location_Mumbai'],
      dtype='object')
```

In [114…

```
imputation.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6 entries, 0 to 5
Data columns (total 19 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Age                 6 non-null      int32
 1   Salary              6 non-null      int32
 2   Exp                 6 non-null      int32
 3   Name_Jane           6 non-null      bool
 4   Name_Kim            6 non-null      bool
 5   Name_Mike           6 non-null      bool
 6   Name_Teddy          6 non-null      bool
 7   Name_Umar           6 non-null      bool
 8   Name_Uttam          6 non-null      bool
 9   Domain_Analytics    6 non-null      bool
 10  Domain_Dataanalyst  6 non-null      bool
 11  Domain_Datascience  6 non-null      bool
 12  Domain_NLP          6 non-null      bool
 13  Domain_Statistics   6 non-null      bool
 14  Domain_Testing      6 non-null      bool
 15  Location_Bangalore  6 non-null      bool
 16  Location_Delhi      6 non-null      bool
 17  Location_Hyderbad   6 non-null      bool
 18  Location_Mumbai     6 non-null      bool
dtypes: bool(16), int32(3)
memory usage: 300.0 bytes
```

In [ ]: