Raw data to clean data conversion using python EDA

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

In [2]:
```
emp = pd.read_excel(r'C:\Users\Rachana Jena\Downloads\Rawdata.xlsx')
```

In [3]:
```
emp
```

Out[3]:

	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 [4]:
```
emp.columns
```

Out[4]:
```
Index(['Name', 'Domain', 'Age', 'Location', 'Salary', 'Exp'], dtype='object')
```

In [5]:
```
emp.shape
```

Out[5]:
```
(6, 6)
```

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

Out[6]:

	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 [7]:
```
emp.tail()
```

Out[7]:

	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 [8]:
```
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 [9]:
```
emp
```

Out[9]:

	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]:
```
emp['Domain']
```

Out[10]:
```
0     Datascience#$
1           Testing
2    Dataanalyst^^#
3       Ana^^lytics
4        Statistics
5               NLP
Name: Domain, dtype: object
```

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.isnull().sum()
```

Out[12]:
```
Name        0
Domain      0
Age         2
Location    2
Salary      0
Exp         1
dtype: int64
```

In [13]:
```
emp['Name']
```

Out[13]:
```
0      Mike
1    Teddy^
2     Uma#r
3      Jane
4    Uttam*
5       Kim
Name: Name, dtype: object
```

In [14]:
```
emp['Name'] = emp['Name'].str.replace(r'\W','',regex=True)
```

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

Out[15]:
```
0     Datascience#$
1           Testing
2    Dataanalyst^^#
3       Ana^^lytics
4        Statistics
5               NLP
Name: Domain, dtype: object
```

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

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

Out[17]:
```
0    Datascience
1        Testing
2    Dataanalyst
3      Analytics
4     Statistics
5            NLP
Name: Domain, dtype: object
```

In [18]:
```
emp
```

Out[18]:

	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	Umar	Dataanalyst	NaN	NaN	1$5%000	4> yrs
3	Jane	Analytics	NaN	Hyderbad	2000^0	NaN
4	Uttam	Statistics	67-yr	NaN	30000-	5+ year
5	Kim	NLP	55yr	Delhi	6000^$0	10+

In [19]:
```
emp['Age'] = emp['Age'].str.replace(r'\W','',regex=True)
```

In [20]:
```
emp['Age']
```

Out[20]:
```
0    34years
1       45yr
2        NaN
3        NaN
4       67yr
5       55yr
Name: Age, dtype: object
```

In [21]:
```
emp
```

Out[21]:

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Datascience	34years	Mumbai	5^00#0	2+
1	Teddy	Testing	45yr	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	67yr	NaN	30000-	5+ year
5	Kim	NLP	55yr	Delhi	6000^$0	10+

In [22]:
```
emp['Location'] = emp['Location'].str.replace(r'\W','',regex=True)
```

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

Out[23]:
```
0       Mumbai
1    Bangalore
2          NaN
3     Hyderbad
4          NaN
5        Delhi
Name: Location, dtype: object
```

In [24]:
```
emp['Salary']
```

Out[24]:
```
0     5^00#0
1    10%%000
2    1$5%000
3     2000^0
4     30000-
5    6000^$0
Name: Salary, dtype: object
```

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

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

Out[26]:
```
0     5000
1    10000
2    15000
3    20000
4    30000
5    60000
Name: Salary, dtype: object
```

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

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

Out[28]:
```
0    34years
1       45yr
2        NaN
3        NaN
4       67yr
5       55yr
Name: Age, dtype: object
```

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

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

Out[30]:
```
0     34
1     45
2    NaN
3    NaN
4     67
5     55
Name: Age, 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> yrs
3	Jane	Analytics	NaN	Hyderbad	20000	NaN
4	Uttam	Statistics	67	NaN	30000	5+ year
5	Kim	NLP	55	Delhi	60000	10+

In [32]:
```
emp['Exp']
```

Out[32]:
```
0         2+
1         <3
2     4> yrs
3        NaN
4    5+ year
5        10+
Name: Exp, dtype: object
```

In [33]:
```
emp['Exp'] = emp['Exp'].str.extract(r'(\d+)')
```

In [34]:
```
emp['Exp']
```

Out[34]:
```
0      2
1      3
2      4
3    NaN
4      5
5     10
Name: Exp, dtype: object
```

In [35]:
```
emp
```

Out[35]:

	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 [36]:
```
clean_data = emp.copy()
```

Till now we have raw data we use regex to clean the data and removed all noise characted from the dataset

You can also work in same in things in sql query as well

- missing values treatment for numerical data

In [37]:
```
clean_data
```

Out[37]:

	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 [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
```

Out[45]:

	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 [46]:
```
clean_data['Location'].isnull().sum()
```

Out[46]:
```
2
```

In [47]:
```
clean_data['Location']
```

Out[47]:
```
0       Mumbai
1    Bangalore
2          NaN
3     Hyderbad
4          NaN
5        Delhi
Name: Location, dtype: object
```

In [48]:
```
clean_data
```

Out[48]:

	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 [49]:
```
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  4 non-null      object
 4   Salary    6 non-null      object
 5   Exp       6 non-null      object
dtypes: object(6)
memory usage: 420.0+ bytes
```

In [50]:
```
clean_data['Age'] = clean_data['Age'].astype(int)
```

In [51]:
```
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  4 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 [52]:
```
clean_data['Salary'] = clean_data['Salary'].astype(int)
clean_data['Exp'] = clean_data['Exp'].astype(int)
```

In [53]:
```
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  4 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 [54]:
```
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 [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  4 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	NaN	15000	4
3	Jane	Analytics	50	Hyderbad	20000	4
4	Uttam	Statistics	67	NaN	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\\Rachana Jena'
```

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	NaN	15000	4
3	Jane	Analytics	50	Hyderbad	20000	4
4	Uttam	Statistics	67	NaN	30000	5
5	Kim	NLP	55	Delhi	60000	10

EDA technique lets apply

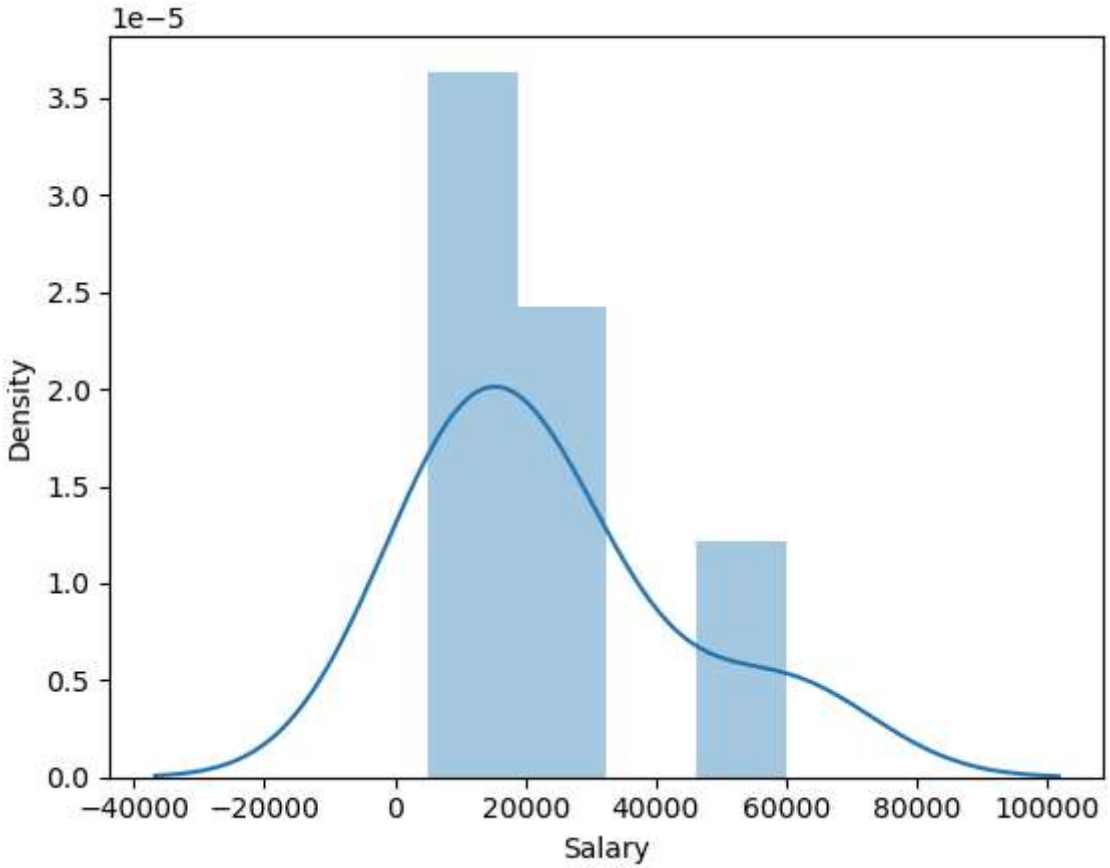
In [61]:
```
import matplotlib.pyplot as plt # visualization
import seaborn as sns
```

In [62]:
```
import warnings
warnings.filterwarnings('ignore')
```

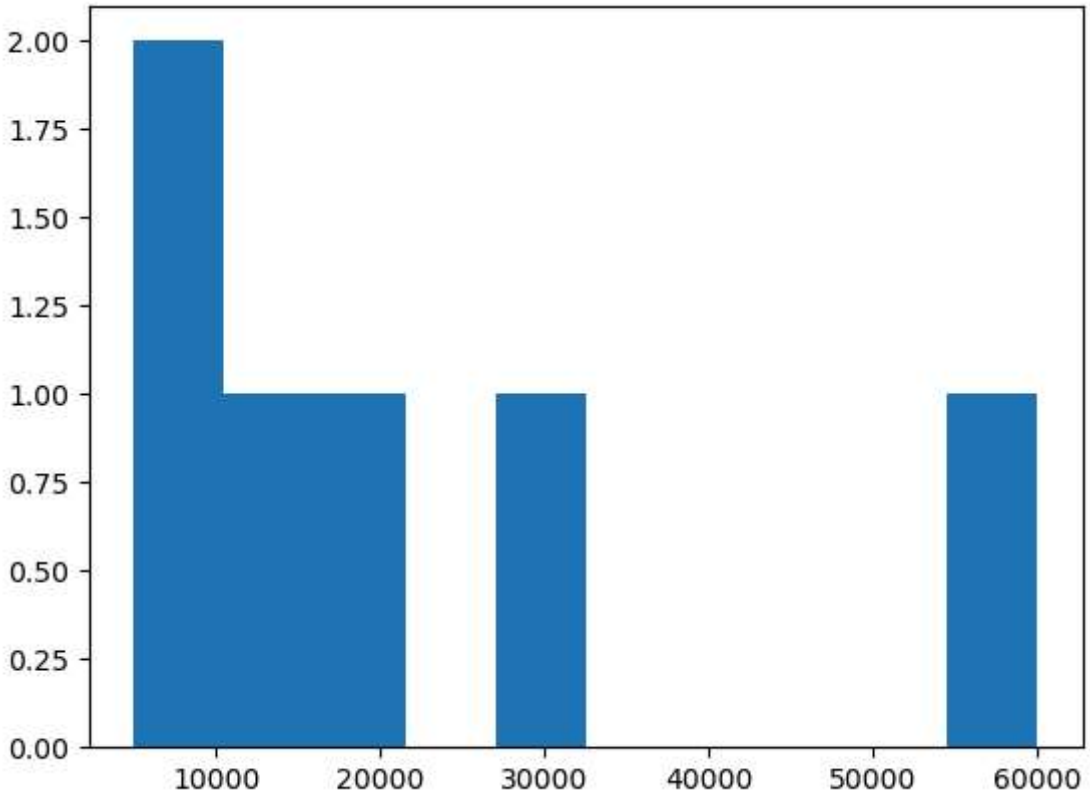
In [63]:
```
clean_data['Salary']
```

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

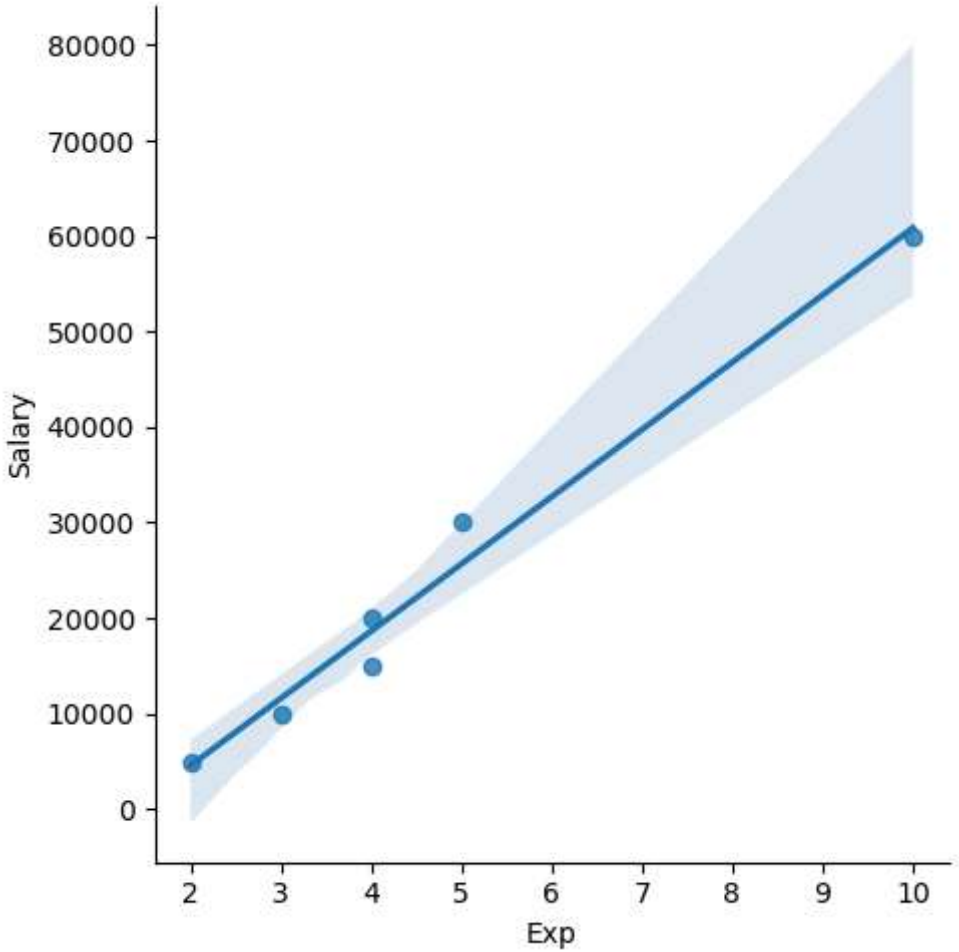
In [64]:
```
vis1 = sns.distplot(clean_data['Salary'])
```



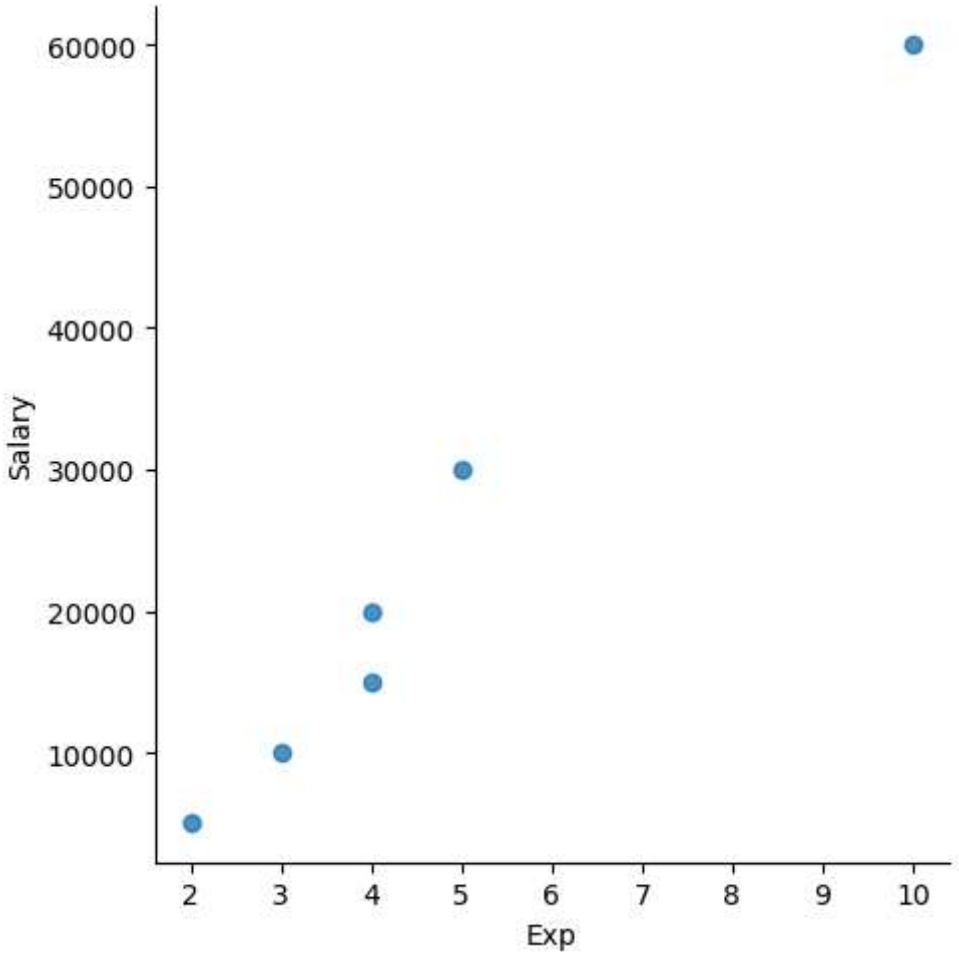
In [65]:
```
vis2 = plt.hist(clean_data['Salary'])
```



In [66]:
```
vis4 = sns.lmplot(data=clean_data,x = 'Exp', y='Salary')
```



In [67]:
```
vis5 = sns.lmplot(data=clean_data,x = 'Exp', y='Salary', fit_reg = False)
```



In [68]:
```
clean_data[:]
```

Out[68]:

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

In [69]:
```
clean_data[::-1]
```

Out[69]:

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

In [70]:
```
clean_data.columns
```

Out[70]:
```
Index(['Name', 'Domain', 'Age', 'Location', 'Salary', 'Exp'], dtype='object')
```

In [71]:
```
X_iv = clean_data[['Name', 'Domain', 'Age', 'Location', 'Exp']]
```

In [72]:
```
X_iv
```

Out[72]:

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

In [73]:
```
y_dv = clean_data[['Salary']]
```

In [74]:
```
y_dv
```

Out[74]:

	Salary
0	5000
1	10000
2	15000
3	20000
4	30000
5	60000

In [75]:
```
emp
```

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	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 [76]:
```
clean_data
```

Out[76]:

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

In [77]:
```
X_iv
```

Out[77]:

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

In [78]:
```
y_dv
```

Out[78]:

	Salary
0	5000
1	10000
2	15000
3	20000
4	30000
5	60000

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

Out[79]:

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

In [80]:
```
imputation = pd.get_dummies(clean_data)
```

In [81]:
```
imputation
```

Out[81]:

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

In [82]:
```
clean_data
```

Out[82]:

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

In [83]:
```
imputation
```

Out[83]:

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

raw data with lot of regex, missing, uncleandata regex, clean fill missing numerical & cateigroica clean_dataset (data cleaning) 3 month - 5mont outlier treatement, univati, bivariate, corelation split the data into x_i.v & y_dv impute cateogrica data to numerical eda part complete

Next step

- we splitn x_iv -- x_train, x_test
- we split y_dv -- y_train, y_test

- build the ml model with x_train & y_train

In [ ]: