Raw Data to Clean Data conversion using python EDA

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

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

Out[2]:
```
'2.2.3'
```

In [3]:
```
# pip install --upgrade openpyxl
```

In [4]:
```
emp=pd.read_excel(r"C:\Users\shaik\Downloads\Rawdata.xlsx")
```

In [5]:
```
emp
```

Out[5]:

	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 [6]:
```
id(emp)
```

Out[6]:
```
2168099765760
```

In [7]:
```
emp.head()
```

Out[7]:

	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 [8]:
```
emp.tail()
```

Out[8]:

	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 [9]:

```
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 [10]:

```
emp
```

Out[10]:

	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 [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 cleansing

In [14]:
```
emp
```

Out[14]:

	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 [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) # non word character.
```

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
```

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

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

Out[20]:
```
0    Datascience
1        Testing
2    Dataanalyst
3      Analytics
4     Statistics
5            NLP
Name: Domain, dtype: object
```

In [21]:
```
emp['Age'] = emp['Age'].str.replace(r'\W','',regex=True)
```

In [22]:
```
emp['Age']
```

Out[22]:
```
0    34years
1       45yr
2        NaN
3        NaN
4       67yr
5       55yr
Name: Age, dtype: object
```

In [23]:
```
emp['Age'] = emp['Age'].str.extract('(\\d+)')  # r(r'(\\d+)')
```

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

Out[24]:
```
0     34
1     45
2    NaN
3    NaN
4     67
5     55
Name: Age, dtype: object
```

In [25]:
```
emp
```

Out[25]:

	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 [26]:
```
emp['Location'] = emp['Location'].str.replace(r'\W','',regex=True)
```

In [27]:
```
emp['Location']
```

Out[27]:
```
0       Mumbai
1    Bangalore
2          NaN
3     Hyderbad
4          NaN
5        Delhi
Name: Location, dtype: object
```

In [28]:
```
emp
```

Out[28]:

	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 [29]:
```
emp['Salary'] = emp['Salary'].str.replace(r'\W','',regex=True)
```

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

Out[30]:
```
0     5000
1    10000
2    15000
3    20000
4    30000
5    60000
Name: Salary, dtype: object
```

In [31]:
```
emp['Exp'] = emp['Exp'].str.extract('(\\d+)')
```

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

Out[32]:

```
0      2
1      3
2      4
3    NaN
4      5
5     10
Name: Exp, dtype: object
```

In [33]:

```
emp
```

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

In [34]:

```
clean_data = emp.copy()
```

In [35]:

```
clean_data
```

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

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

you can also work in same thing in sql query as well

EDA Technique Lets Apply

Missing Value Treatment For Numerical Data

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['A
```

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['Location'] = clean_data['Location'].fillna(clean_data['Location'].mc
```

In [43]:
```
clean_data['Location']
```

```
Out[43]: 0       Mumbai
         1    Bangalore
         2    Bangalore
         3     Hyderbad
         4    Bangalore
         5        Delhi
         Name: Location, dtype: object
```

```
In [44]: clean_data['Exp'] = clean_data['Exp'].fillna(np.mean(pd.to_numeric(clean_data['E
```

```
In [45]: clean_data['Exp']
```

```
Out[45]: 0      2
         1      3
         2      4
         3    4.8
         4      5
         5     10
         Name: Exp, dtype: object
```

```
In [46]: clean_data
```

Out[46]:

	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 [47]: 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 [48]: clean_data['Age'] = clean_data['Age'].astype(int)
```

```
In [49]: clean_data['Age']
```

```
Out[49]: 0    34
         1    45
         2    50
         3    50
         4    67
         5    55
         Name: Age, dtype: int64
```

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

```
In [51]: clean_data
```

Out[51]:

	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	4.8
4	Uttam	Statistics	67	Bangalore	30000	5
5	Kim	NLP	55	Delhi	60000	10

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

```
In [53]: clean_data['Exp']
```

```
Out[53]: 0     2
         1     3
         2     4
         3     4
         4     5
         5    10
         Name: Exp, dtype: int64
```

```
In [54]: 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      int64 
 3   Location  6 non-null      object
 4   Salary    6 non-null      int64 
 5   Exp       6 non-null      int64 
dtypes: int64(3), object(3)
memory usage: 420.0+ bytes
```

```
In [55]: 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 [56]: 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      int64   
 3   Location  6 non-null      category
 4   Salary    6 non-null      int64   
 5   Exp       6 non-null      int64   
dtypes: category(3), int64(3)
memory usage: 938.0 bytes
```

In [57]:

```
clean_data
```

Out[57]:

	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 [58]:

```
clean_data.to_csv('clean_data.csv')
```

In [59]:

```
import os
os.getcwd() # from os give the saved current working directly.
```

Out[59]:

```
'C:\\Users\\shaik'
```

In [60]:

```
clean_data
```

Out[60]:

	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

EDA TECHNIQUE LETS APPLY

In [71]:

```
import matplotlib.pyplot as plt # visualization
import seaborn as sus
```

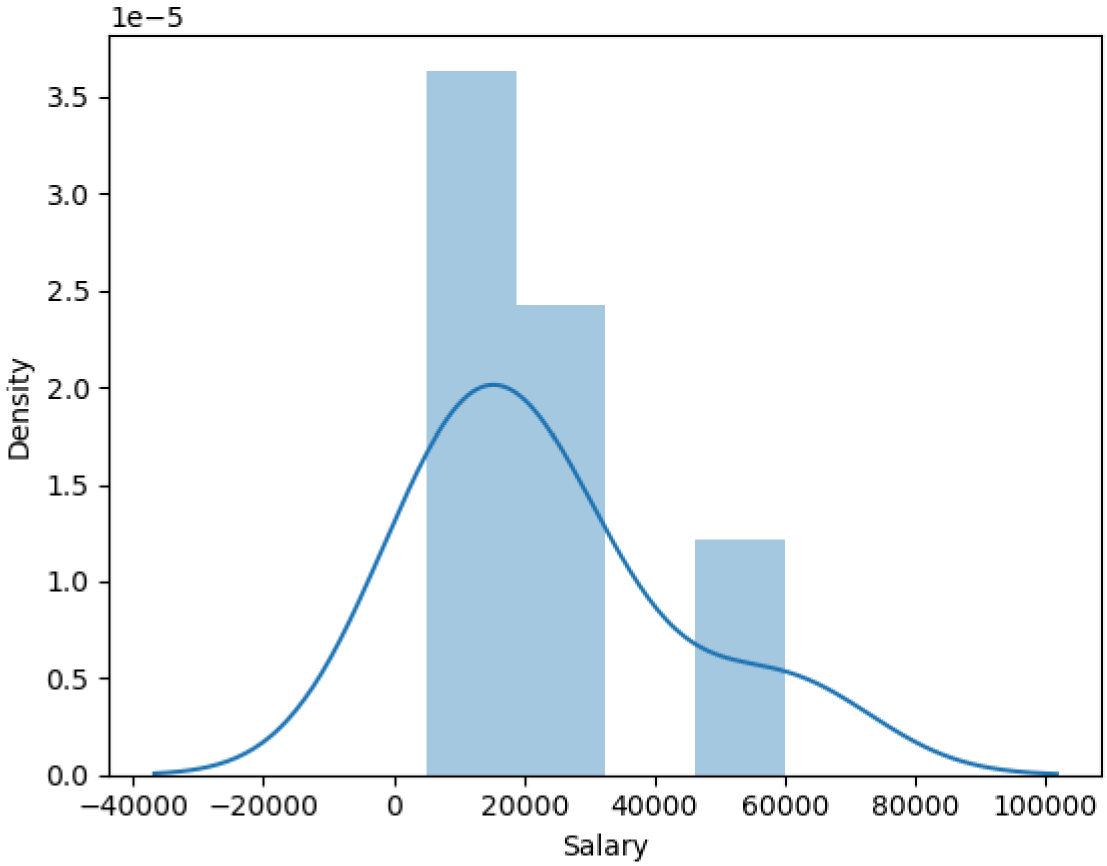
```
In [72]: import warnings
         warnings.filterwarnings('ignore')
```

```
In [75]: clean_data['Salary']
```

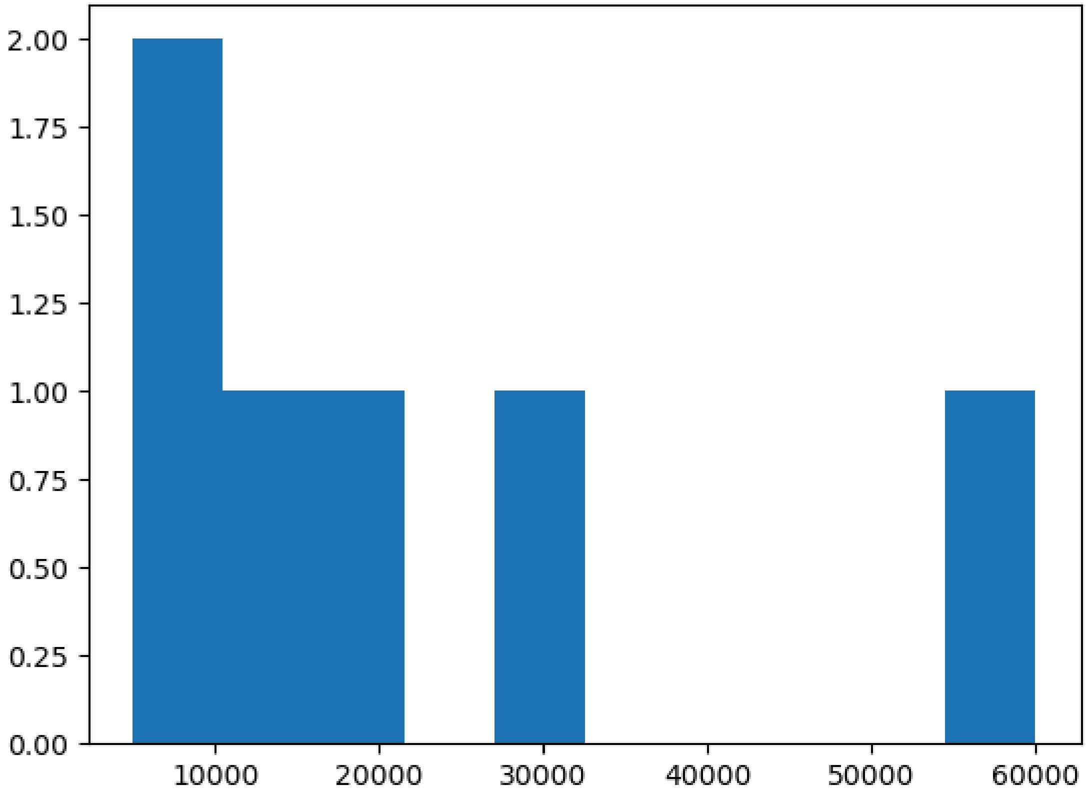
```
Out[75]: 0     5000
         1    10000
         2    15000
         3    20000
         4    30000
         5    60000
         Name: Salary, dtype: int64
```

```
In [84]: import seaborn as sns
         import matplotlib.pyplot as plt
```

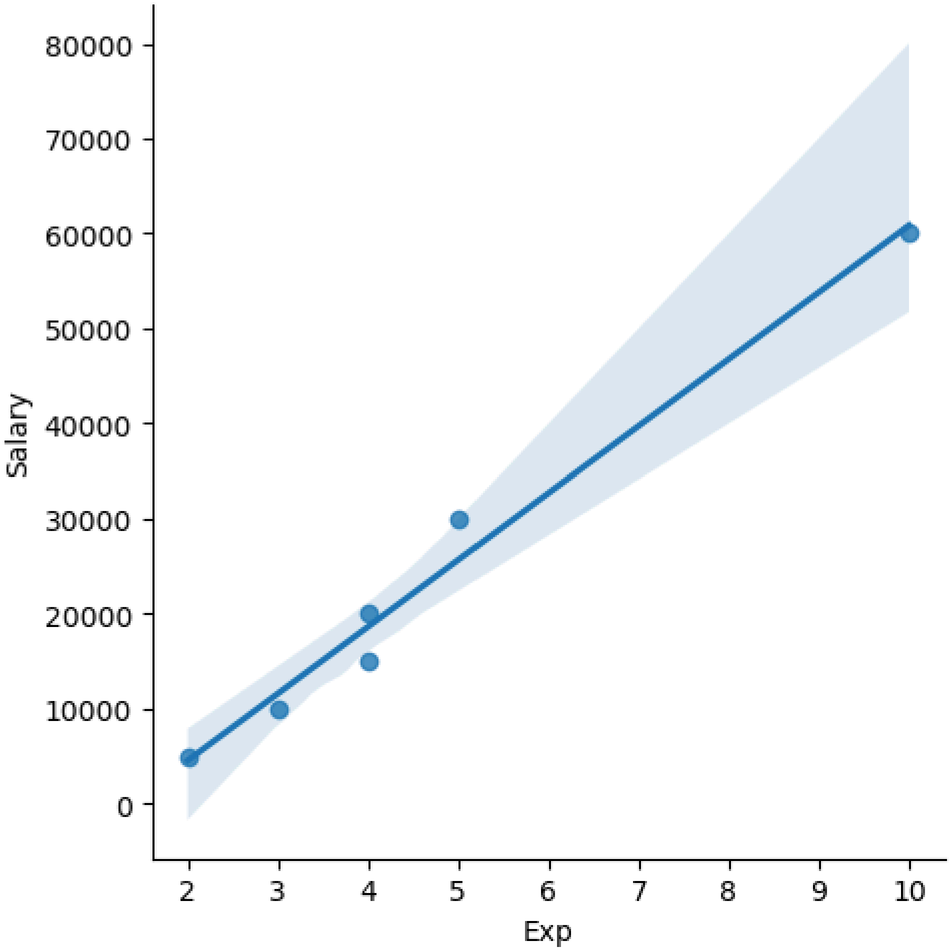
```
In [90]: vis1 = sns.distplot(clean_data['Salary'])
```



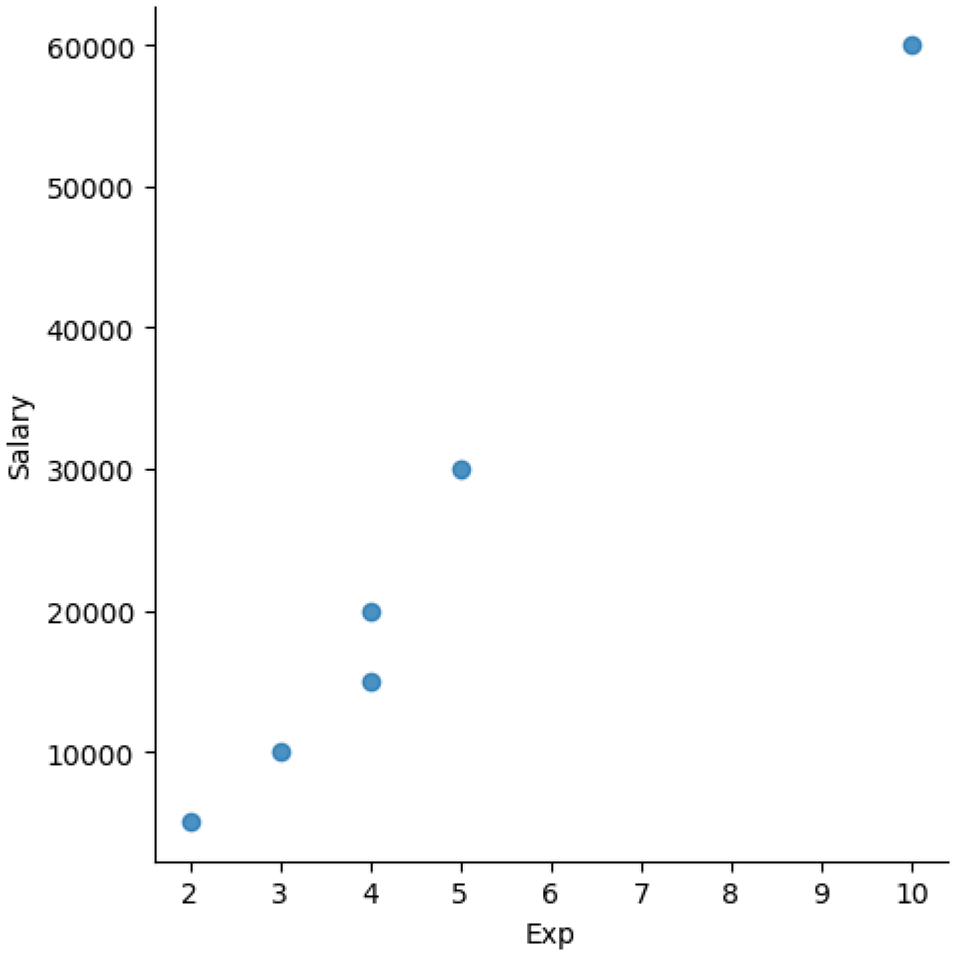
```
In [88]: vis2 = plt.hist(clean_data['Salary'])
```



```
In [91]: vis4 = sns.lmplot(data=clean_data,x = 'Exp', y='Salary')
```



```
In [92]: vis5 = sns.lmplot(data=clean_data,x = 'Exp', y='Salary', fit_reg = False)
```



In [94]:
```
clean_data[:]
```

Out[94]:

	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 [96]:
```
clean_data[0:6:2]
```

Out[96]:

	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 [97]:
```
clean_data[::-1]
```

Out[97]:

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

In [98]:

```
clean_data.columns
```

Out[98]:

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

In [100…

```
x_iv = clean_data[['Name','Domain','Age','Location','Exp']]
```

In [101…

```
x_iv
```

Out[101…

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

In [102…

```
y_dv = clean_data[['Salary']]
```

In [103…

```
y_dv
```

Out[103…

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

In [105…

```
emp
```

Out[105…

	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 [106…

```
clean_data
```

Out[106…

	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 [108…

```
x_iv
```

Out[108…

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

In [109…

```
y_dv
```

Out[109…

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

In [110…

```
clean_data
```

Out[110…

	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 [111…

```
impurtation = pd.get_dummies(clean_data)
```

In [112…

```
impurtation
```

Out[112…

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

In [113…

```
clean_data
```

Out[113…

	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 [114…

```
impurtation
```

Out[114…

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

In [ ]: