raw-data-to-clean-data

November 21, 2024

1 Raw data to clene data Conversion using python EDA

[1]:
```
import numpy as np
import pandas as pd # import the library
```

[2]:
```
pd.__version__ #check the version
```

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

[3]:
```
emp = pd.read_excel('Rawdata.xlsx')
```

[4]:
```
emp
```

[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+

[5]:
```
id(emp) # chech the id
```

[5]:
```
140189539870432
```

[6]:
```
# Check columns
emp.columns
```

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

[7]:
```
emp.shape # gives rows and columns
```

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

[8]:
```
emp.head() # see top 5 rows
```

[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

[9]:
```
emp.tail() # see buttom 5 rows
```

[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+

[10]:
```
emp.info() # see the information
```

```
<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: 416.0+ bytes
```

[11]:
```
emp
```

[11]:

	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+

[12]:
```
# Checking missing values
emp.isnull().sum()
```

```
Name        0
Domain      0
```

```
Age         2
Location    2
Salary      0
Exp         1
dtype: int64
```

2 DATA CLEANING OR DATA CLEANSING

[13]:
```
emp['Name'] # to read the name column
```

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

[14]:
```
# Remove unwanted symbols - Special symbols
emp['Name'] = emp['Name'].str.replace(r'\W','',regex=True) #non Word Character
```

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

[15]:
```
0     Mike
1    Teddy
2     Umar
3     Jane
4    Uttam
5      Kim
Name: Name, dtype: object
```

[16]:
```
# To remove unwanted characters
emp['Domain'] = emp['Domain'].str.replace(r'\W','',regex=True)
```

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

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

[18]:
```
emp # See the data
```

[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+

[19]:
```
# Check whether there is any unwanted symbols
emp['Age'] = emp['Age'].str.replace(r'\W','',regex=True)
```

[20]:
```
emp['Age'] #Prints tha Age columns
```

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

[21]:
```
emp['Age'] = emp['Age'].str.extract(r'(\d+)') # for extract only numbers from␣
 ↪the age column
```

[22]:
```
emp['Age'] # Prints the asg columns
```

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

[23]:
```
emp
```

[23]:
	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+

[24]:
```
emp['Location']= emp['Location'].str.replace(r'\W','',regex=True) #Here we␣
 ↪replace the raw string
```

[25]:
```
emp['Location'] #disply the location columns
```

[25]:
```
0       Mumbai
1    Bangalore
2          NaN
3     Hyderbad
4          NaN
5        Delhi
Name: Location, dtype: object
```

[26]:
```
emp
```

[26]:

	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+

[27]:
```
emp['Salary']=emp['Salary'].str.replace(r'\W','',regex=True) # For replace the␣
↪unwanted symbols
```

[28]:
```
emp['Salary'] #prints the salary column
```

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

[29]:
```
emp
```

[29]:

	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+

[30]:
```
emp['Exp']=emp['Exp'].str.extract(r'(\d+)') #For extract the integers
```

[31]:
```
emp['Exp'] # prints Exp
```

[31]:
```
0      2
1      3
2      4
3    NaN
4      5
5     10
Name: Exp, dtype: object
```

[32]:
```
emp
```

[32]:

	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

[33]:
```
clean_data = emp.copy() # Copy the dataset
```

[34]:
```
clean_data
```

[34]:

	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

3 Till now we have raw data we use regex to clean the data and removed all unwanted symbols, nums, char from the dataset

4 we can also work with sql

5 Using EDA technique

6 Missing value Treatment for numerical data

[35]:
```
clean_data.isnull().sum()
```

[35]:
```
Name        0
Domain      0
Age         2
Location    2
```

```
Salary      0
Exp         1
dtype: int64
```

[36]:
```
clean_data['Age']
```

[36]:
```
0     34
1     45
2    NaN
3    NaN
4     67
5     55
Name: Age, dtype: object
```

[37]:
```
import numpy as np # import numpy
```

[38]:
```
# For numerical value  we use mean, median and mode strategy
clean_data['Age']= clean_data['Age'].fillna(np.mean(pd.
 ↪to_numeric(clean_data['Age']))) #using mean strategy
```

[39]:
```
clean_data['Age'] #Displayes the cleane Age column
```

[39]:
```
0       34
1       45
2    50.25
3    50.25
4       67
5       55
Name: Age, dtype: object
```

[40]:
```
clean_data['Exp']=clean_data['Exp'].fillna(np.mean(pd.
 ↪to_numeric(clean_data['Exp']))) #here also goes the same
```

[41]:
```
clean_data['Exp'] #prints it
```

[41]:
```
0      2
1      3
2      4
3    4.8
4      5
5     10
Name: Exp, dtype: object
```

[42]:
```
clean_data['Location'].isnull().sum() #here we check rather there is any␣
 ↪missing value or not
```

[42]:
```
2
```

[43]:
```
clean_data['Location'] #displayes the location column
```

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

[44]:
```
# For categorical value we use mode strategy or KNN strategy
clean_data['Location'] = clean_data['Location'].fillna(clean_data['Location'].
 ↪mode()[0]) #using mode strategy
```

[45]:
```
clean_data['Location'] # prints the location column
```

[45]:
```
0       Mumbai
1    Bangalore
2    Bangalore
3     Hyderbad
4    Bangalore
5        Delhi
Name: Location, dtype: object
```

[46]:
```
clean_data # displayes the clean_data
```

[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

[47]:
```
clean_data.info() # see the informations
```

```
<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: 416.0+ bytes
```

[48]:
```
#For change the type -- numerical
clean_data['Age'] = clean_data['Age'].astype(int) #Chenge flote or string type␣
 ↪to integer type
```

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

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

[50]:
```
clean_data['Salary']=clean_data['Salary'].astype(int) #change the type
```

[51]:
```
clean_data['Salary']
```

[51]:
```
0     5000
1    10000
2    15000
3    20000
4    30000
5    60000
Name: Salary, dtype: int64
```

[52]:
```
clean_data['Exp']=clean_data['Exp'].astype(int) #change the type
```

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

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

[54]:
```
# change the type -- categorical
clean_data['Name'] = clean_data['Name'].astype('category')
clean_data['Domain'] = clean_data['Name'].astype('category')
clean_data['Location'] = clean_data['Name'].astype('category')
```

[55]:
```
clean_data.info() #see the information
```

```
<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: 950.0 bytes
```

7 Data set is cleaned

[56]:
```
# SAVE THE CLEAN FILE
# We have to convert this file to csv to extract the dataframe
clean_data.to_csv('clean_data.csv') #change the file to csv
```

[57]:
```
#get the location
import os
os.getcwd()
```

[57]: '/content'

[58]:
```
clean_data
```

[58]:

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Mike	34	Mike	5000	2
1	Teddy	Teddy	45	Teddy	10000	3
2	Umar	Umar	50	Umar	15000	4
3	Jane	Jane	50	Jane	20000	4
4	Uttam	Uttam	67	Uttam	30000	5
5	Kim	Kim	55	Kim	60000	10

8 Data Visualization

9 EDA technique lets Apply

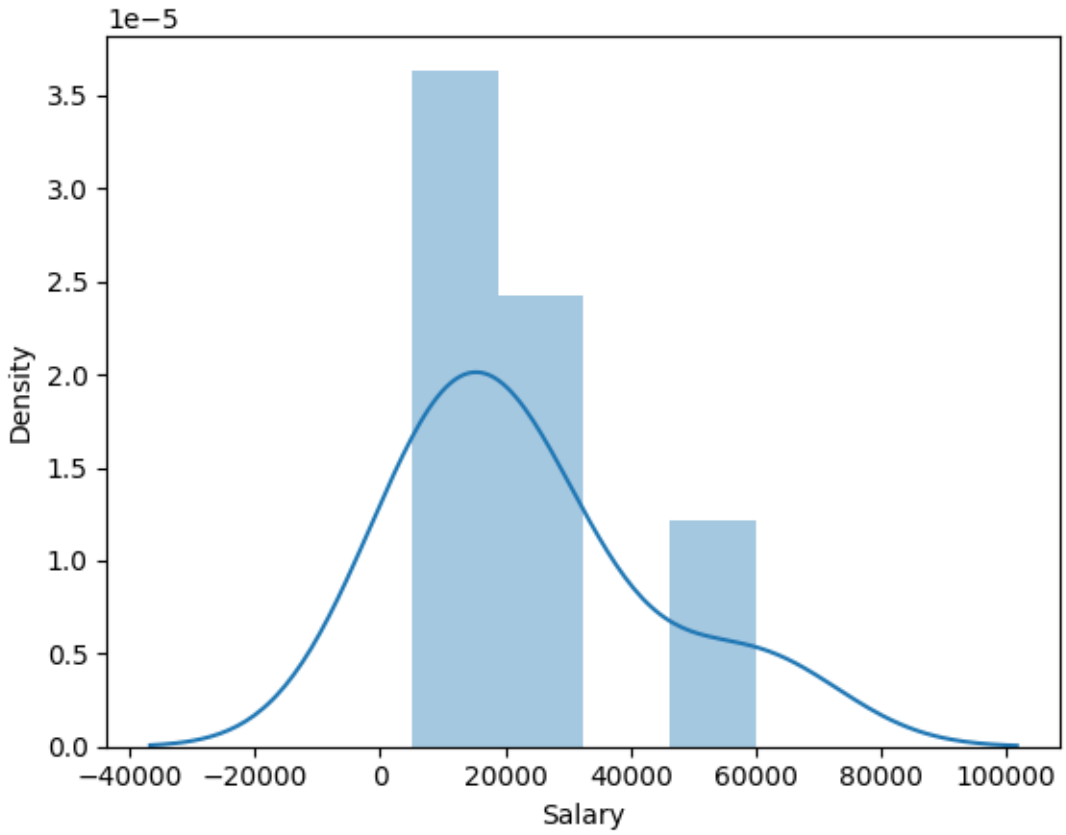
[59]:
```
import matplotlib.pyplot as plt # For visualization
import seaborn as sns           # For advance visualization
```

[60]:
```
import warnings
warnings.filterwarnings('ignore') # Ignore all warnings
```

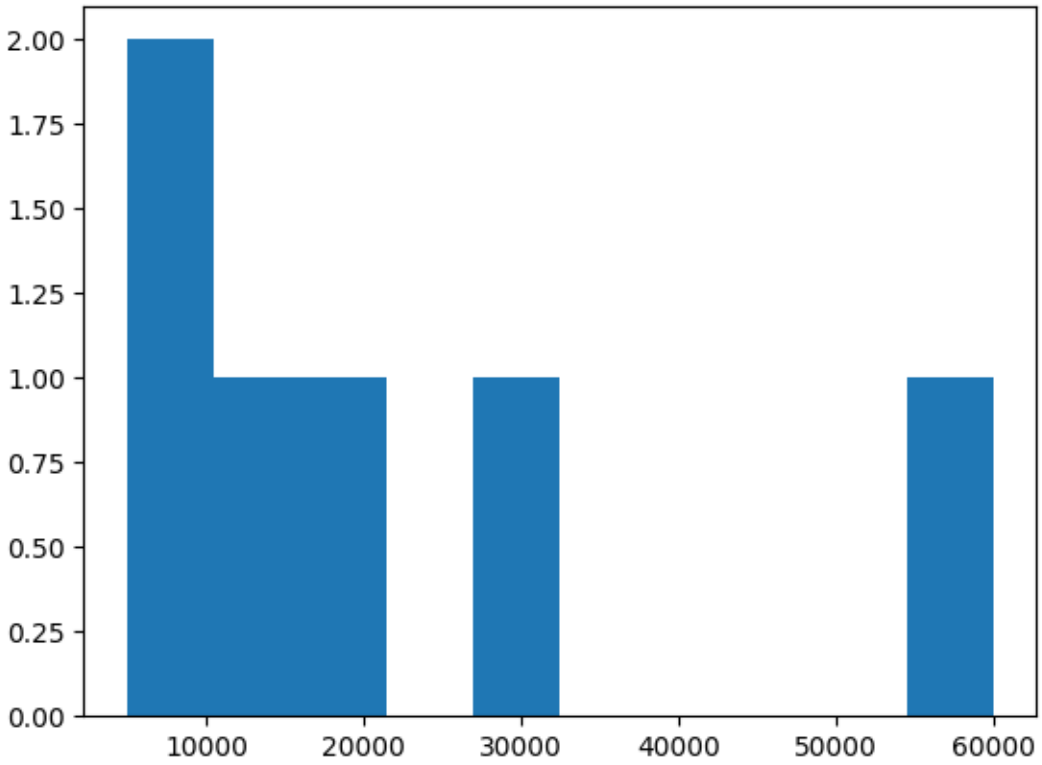
[61]:
```
clean_data['Salary'] # display the salary column
```

[61]:
```
0     5000
1    10000
2    15000
3    20000
4    30000
5    60000
Name: Salary, dtype: int64
```

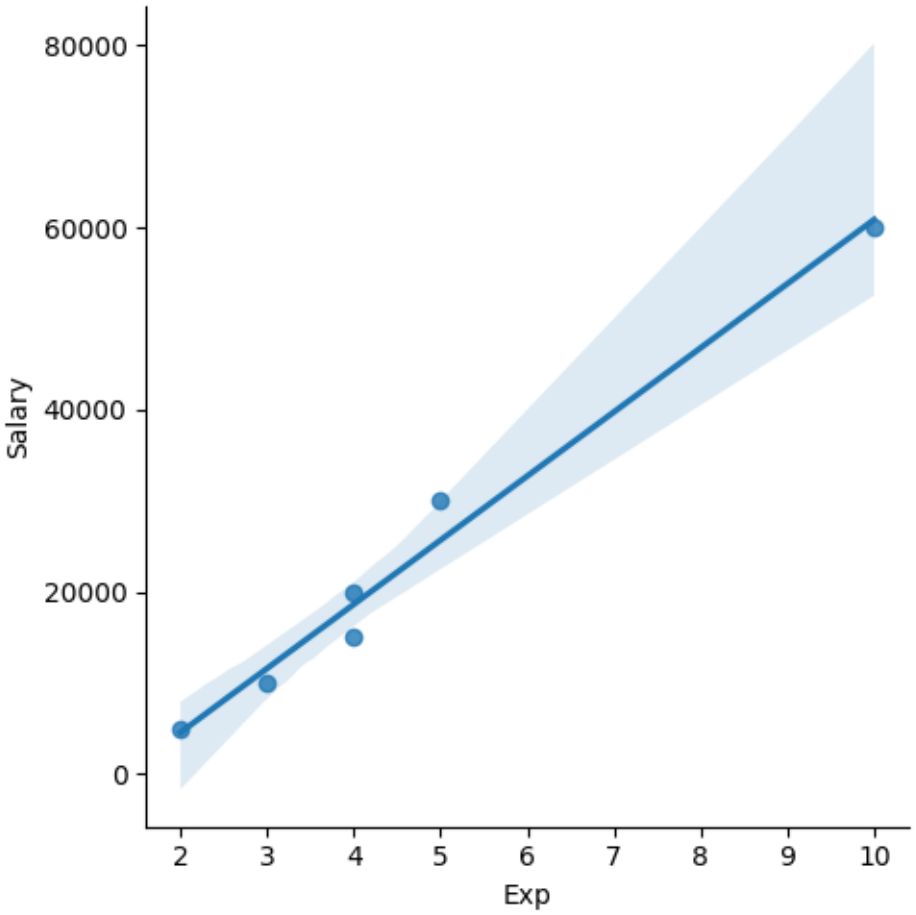
[62]:
```
#univariate analysis--Plot the graph using one variable
vis1 = sns.distplot(clean_data['Salary'])
```



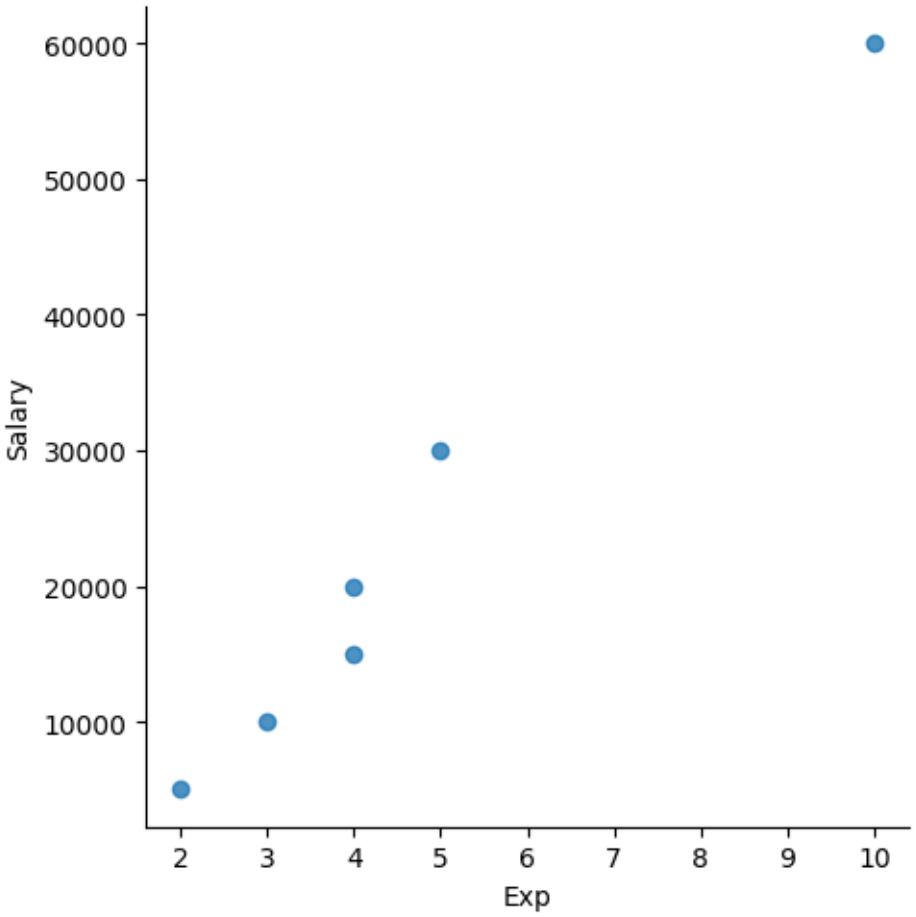
[63]:
```
vis2 = plt.hist(clean_data['Salary'])
```



[64]:
```
#Bivariate analysis
vis4 = sns.lmplot(clean_data,x ='Exp',y='Salary')
```



[65]:
```
vis5 = sns.lmplot(data=clean_data,x = 'Exp', y='Salary', fit_reg = False)␣
 ↪#remove the regrassion line
```



[66]:
```
clean_data[:] # Slicing
```

[66]:

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Mike	34	Mike	5000	2
1	Teddy	Teddy	45	Teddy	10000	3
2	Umar	Umar	50	Umar	15000	4
3	Jane	Jane	50	Jane	20000	4
4	Uttam	Uttam	67	Uttam	30000	5
5	Kim	Kim	55	Kim	60000	10

[67]:
```
clean_data[0:6:2]
```

[67]:

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Mike	34	Mike	5000	2
2	Umar	Umar	50	Umar	15000	4
4	Uttam	Uttam	67	Uttam	30000	5

[68]:
```
clean_data[::-1]
```

[68]:

	Name	Domain	Age	Location	Salary	Exp
5	Kim	Kim	55	Kim	60000	10
4	Uttam	Uttam	67	Uttam	30000	5
3	Jane	Jane	50	Jane	20000	4
2	Umar	Umar	50	Umar	15000	4
1	Teddy	Teddy	45	Teddy	10000	3
0	Mike	Mike	34	Mike	5000	2

[69]:
```
#Variable identification -- 2 type-- Dependent & Independent
clean_data.columns #see the columns
```

[69]:
```
Index(['Name', 'Domain', 'Age', 'Location', 'Salary', 'Exp'], dtype='object')
```

[70]:
```
X_iv = clean_data[['Name', 'Domain', 'Age', 'Location', 'Exp']] #Displayes the␣
↪independent variables
```

[71]:
```
X_iv
```

[71]:

	Name	Domain	Age	Location	Exp
0	Mike	Mike	34	Mike	2
1	Teddy	Teddy	45	Teddy	3
2	Umar	Umar	50	Umar	4
3	Jane	Jane	50	Jane	4
4	Uttam	Uttam	67	Uttam	5
5	Kim	Kim	55	Kim	10

[72]:
```
y_dv = clean_data[['Salary']] #dependent variables
```

[73]:
```
y_dv
```

[73]:

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

[74]:
```
emp
```

[74]:

	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

[75]:
```
clean_data
```

[75]:

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Mike	34	Mike	5000	2
1	Teddy	Teddy	45	Teddy	10000	3
2	Umar	Umar	50	Umar	15000	4
3	Jane	Jane	50	Jane	20000	4
4	Uttam	Uttam	67	Uttam	30000	5
5	Kim	Kim	55	Kim	60000	10

[76]:
```
X_iv
```

[76]:

	Name	Domain	Age	Location	Exp
0	Mike	Mike	34	Mike	2
1	Teddy	Teddy	45	Teddy	3
2	Umar	Umar	50	Umar	4
3	Jane	Jane	50	Jane	4
4	Uttam	Uttam	67	Uttam	5
5	Kim	Kim	55	Kim	10

[77]:
```
y_dv
```

[77]:

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

[78]:
```
clean_data
```

[78]:

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Mike	34	Mike	5000	2
1	Teddy	Teddy	45	Teddy	10000	3
2	Umar	Umar	50	Umar	15000	4
3	Jane	Jane	50	Jane	20000	4
4	Uttam	Uttam	67	Uttam	30000	5
5	Kim	Kim	55	Kim	60000	10

[79]:
```
#Variable creation and variable transformation
imputation = pd.get_dummies(clean_data) #Here we create the dummy variable for␣
 ↪transfering and creating the variable
```

[80]:
```
imputation  #display it
```

[80]:
```
   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

   Name_Uttam  Domain_Jane  …  Domain_Mike  Domain_Teddy  Domain_Umar  \
0       False        False  …         True         False        False
1       False        False  …        False          True        False
2       False        False  …        False         False         True
3       False         True  …        False         False        False
4        True        False  …        False         False        False
5       False        False  …        False         False        False

   Domain_Uttam  Location_Jane  Location_Kim  Location_Mike  Location_Teddy  \
0         False          False         False           True           False
1         False          False         False          False            True
2         False          False         False          False           False
3         False           True         False          False           False
4          True          False         False          False           False
5         False          False          True          False           False

   Location_Umar  Location_Uttam
0          False           False
1          False           False
2           True           False
3          False           False
4          False            True
5          False           False

[6 rows x 21 columns]
```

[81]:
```
clean_data
```

[81]:
```
    Name Domain  Age Location  Salary  Exp
0   Mike   Mike   34     Mike    5000    2
1  Teddy  Teddy   45    Teddy   10000    3
2   Umar   Umar   50     Umar   15000    4
3   Jane   Jane   50     Jane   20000    4
4  Uttam  Uttam   67    Uttam   30000    5
5    Kim    Kim   55      Kim   60000   10
```

[82]:
```
imputation #get the output as boolen type--true / false
```

[82]:
```
   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

   Name_Uttam  Domain_Jane  …  Domain_Mike  Domain_Teddy  Domain_Umar  \
0       False        False  …         True         False        False
1       False        False  …        False          True        False
2       False        False  …        False         False         True
3       False         True  …        False         False        False
4        True        False  …        False         False        False
5       False        False  …        False         False        False

   Domain_Uttam  Location_Jane  Location_Kim  Location_Mike  Location_Teddy  \
0         False          False         False           True           False
1         False          False         False          False            True
2         False          False         False          False           False
3         False           True         False          False           False
4          True          False         False          False           False
5         False          False          True          False           False

   Location_Umar  Location_Uttam
0          False           False
1          False           False
2           True           False
3          False           False
4          False            True
5          False           False

[6 rows x 21 columns]
```

[83]:
```
#lets convert it as integer type
imputation=imputation.astype(int)
```

[84]:
```
imputation
```

[84]:
```
   Age  Salary  Exp  Name_Jane  Name_Kim  Name_Mike  Name_Teddy  Name_Umar  \
0   34    5000    2          0         0          1           0          0
1   45   10000    3          0         0          0           1          0
2   50   15000    4          0         0          0           0          1
3   50   20000    4          1         0          0           0          0
4   67   30000    5          0         0          0           0          0
5   55   60000   10          0         1          0           0          0
```

```
   Name_Uttam  Domain_Jane  …  Domain_Mike  Domain_Teddy  Domain_Umar  \
0           0            0  …            1             0            0
1           0            0  …            0             1            0
2           0            0  …            0             0            1
3           0            1  …            0             0            0
4           1            0  …            0             0            0
5           0            0  …            0             0            0

   Domain_Uttam  Location_Jane  Location_Kim  Location_Mike  Location_Teddy  \
0             0              0             0              1               0
1             0              0             0              0               1
2             0              0             0              0               0
3             0              1             0              0               0
4             1              0             0              0               0
5             0              0             1              0               0

   Location_Umar  Location_Uttam
0              0               0
1              0               0
2              1               0
3              0               0
4              0               1
5              0               0

[6 rows x 21 columns]
```

[85]:
```
clean_data
```

[85]:
```
     Name Domain  Age Location  Salary  Exp
0   Mike   Mike   34     Mike    5000    2
1  Teddy  Teddy   45    Teddy   10000    3
2   Umar   Umar   50     Umar   15000    4
3   Jane   Jane   50     Jane   20000    4
4  Uttam  Uttam   67    Uttam   30000    5
5    Kim    Kim   55      Kim   60000   10
```