Raw data to clean data conversion using python EDA

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

In [2]:
```
emp=pd.read_excel(r"D:\Data Science with AI\Data Science With AI\18th, 19th, 21s
```

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.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
```

In [14]:
```
emp['Name']
```

Out[14]:
```
0      Mike
1    Teddy^
2     Uma#r
3      Jane
4    Uttam*
5       Kim
Name: Name, dtype: object
```

In [15]:
```
emp['Name']=emp['Name'].str.replace(r'\W','',regex=True)
```

In [16]:
```
emp['Name']
```

```
Out[16]: 0     Mike
         1    Teddy
         2     Umar
         3     Jane
         4    Uttam
         5      Kim
         Name: Name, dtype: object
```

```
In [17]: emp['Domain']=emp['Domain'].str.replace(r'\W','',regex=True)
```

```
In [18]: emp['Domain']
```

```
Out[18]: 0    Datascience
         1        Testing
         2    Dataanalyst
         3      Analytics
         4     Statistics
         5            NLP
         Name: Domain, dtype: object
```

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

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

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

```
In [23]: emp['Salary']=emp['Salary'].str.replace(r'\W','',regex=True)
```

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

```
Out[24]: 0     5000
         1    10000
         2    15000
         3    20000
         4    30000
         5    60000
         Name: Salary, dtype: object
```

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

Out[25]:

	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 [26]:
```
emp['Exp']=emp['Exp'].str.extract('(\\d+)')
```

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

Out[27]:
```
0      2
1      3
2      4
3    NaN
4      5
5     10
Name: Exp, dtype: object
```

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

Out[28]:

	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 [29]:
```
clean_data=emp.copy()
```

In [30]:
```
clean_data
```

Out[30]:

	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 [31]:
```
clean_data.isnull().sum()
```

Out[31]:
```
Name        0
Domain      0
Age         2
Location    2
Salary      0
Exp         1
dtype: int64
```

In [32]:
```
import numpy as np
```

In [33]:
```
clean_data['Age']=clean_data['Age'].fillna(np.mean(pd.to_numeric(clean_data['Age
```

In [34]:
```
clean_data['Age']
```

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

In [35]:
```
clean_data['Exp']=clean_data['Exp'].fillna(np.mean(pd.to_numeric(clean_data['Exp
```

In [36]:
```
clean_data['Exp']
```

Out[36]:
```
0      2
1      3
2      4
3    4.8
4      5
5     10
Name: Exp, dtype: object
```

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	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 [38]:
```
clean_data['Location']=clean_data['Location'].fillna(clean_data['Location'].mode
```

In [39]:
```
clean_data['Location']
```

Out[39]:
```
0       Mumbai
1    Bangalore
2    Bangalore
3     Hyderbad
4    Bangalore
5        Delhi
Name: Location, dtype: object
```

In [40]:
```
clean_data
```

Out[40]:

	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 [41]:
```
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 [42]:
```
clean_data['Age']=clean_data['Age'].astype(int)
```

In [43]:
```
clean_data
```

Out[43]:

	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 [44]:
```
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      object
 5   Exp       6 non-null      object
dtypes: int64(1), object(5)
memory usage: 420.0+ bytes
```

In [45]:
```
clean_data['Salary']=clean_data['Salary'].astype(int)
```

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

Out[46]:
```
0     5000
1    10000
2    15000
3    20000
4    30000
5    60000
Name: Salary, dtype: int64
```

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      int64 
 3   Location  6 non-null      object
 4   Salary    6 non-null      int64 
 5   Exp       6 non-null      object
dtypes: int64(2), object(4)
memory usage: 420.0+ bytes
```

In [48]:
```
clean_data['Exp']=clean_data['Exp'].astype(int)
```

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

Out[49]:
```
0     2
1     3
2     4
3     4
4     5
5    10
Name: Exp, dtype: int64
```

In [50]:
```
clean_data['Exp']
```

Out[50]:
```
0     2
1     3
2     4
3     4
4     5
5    10
Name: Exp, dtype: int64
```

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      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 [52]:
```
clean_data
```

Out[52]:

	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 [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.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 [55]:
```
clean_data
```

Out[55]:

	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 [56]:
```
clean_data.to_csv('clean_data.csv')
```

In [57]:
```
import os
os.getcwd()
```

Out[57]:
```
'C:\\Users\\DELL\\FSDS'
```

In [58]:
```
clean_data.columns
```

Out[58]:
```
Index(['Name', 'Domain', 'Age', 'Location', 'Salary', 'Exp'], dtype='object')
```

In [59]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [60]:
```
import warnings
warnings.filterwarnings('ignore')
```

In [61]:
```
clean_data
```

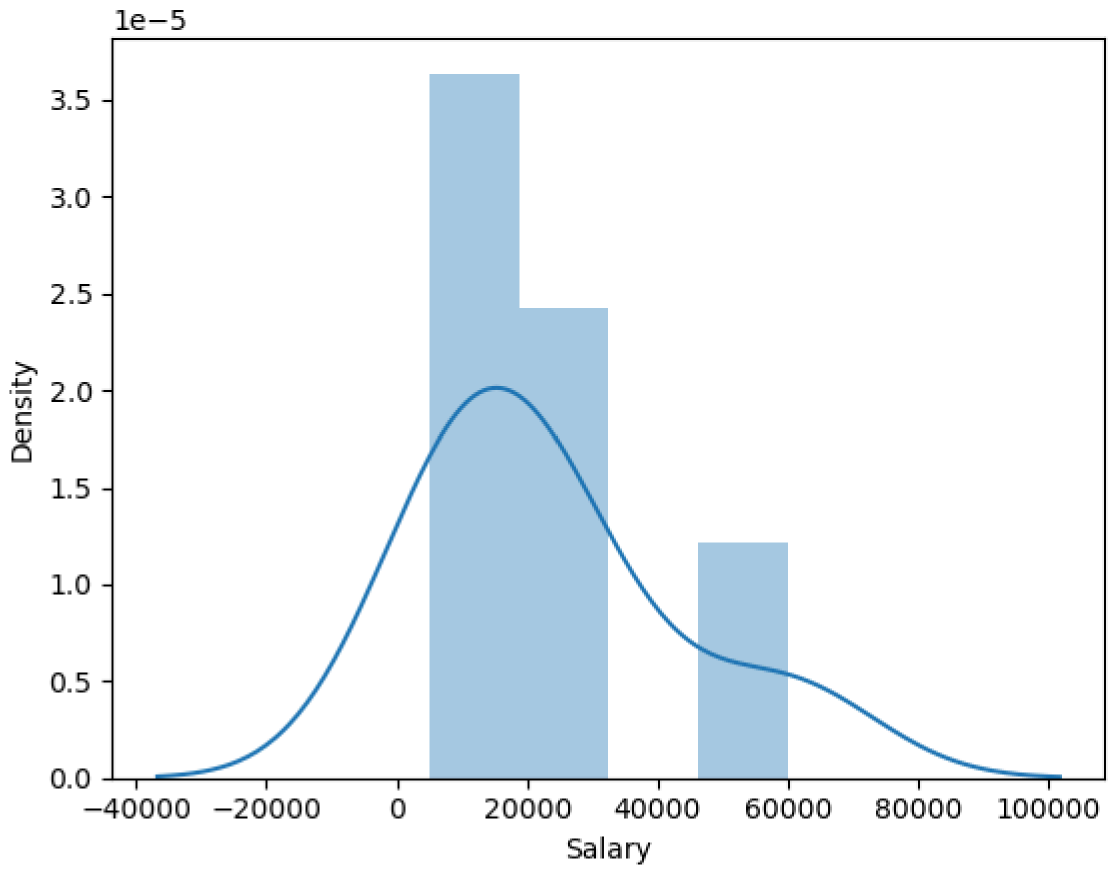
Out[61]:

	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 [62]:
```
clean_data['Salary']
```

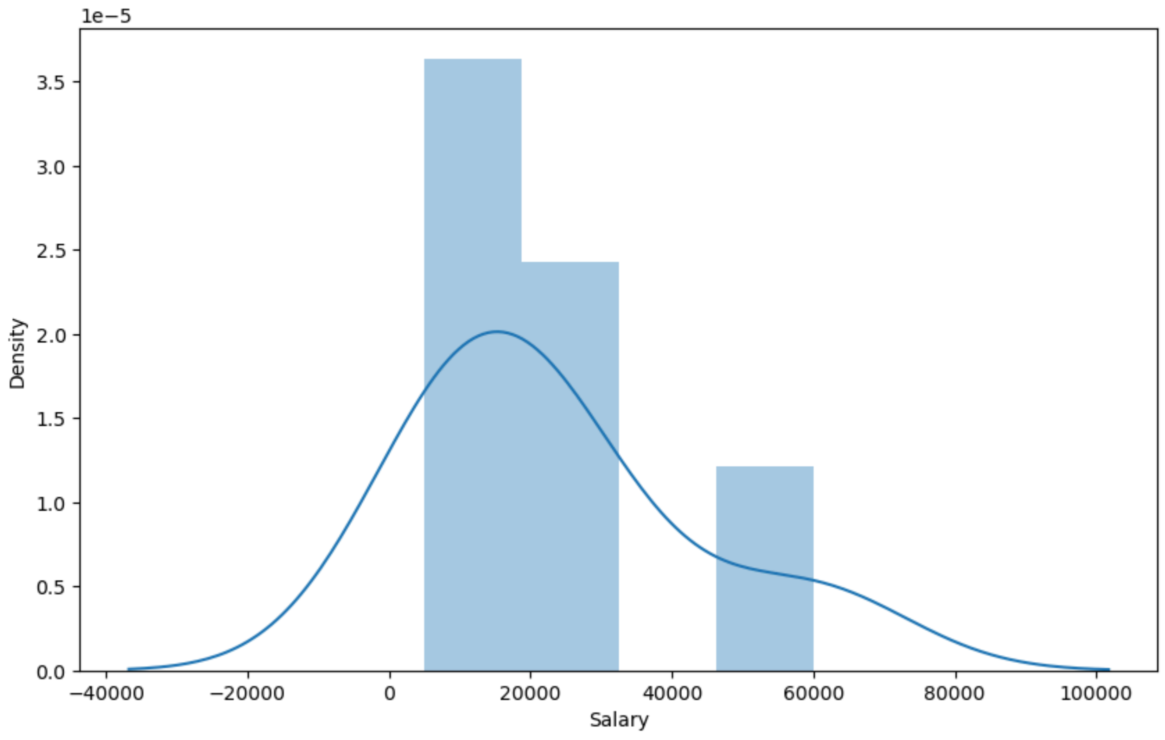
Out[62]:
```
0     5000
1    10000
2    15000
3    20000
4    30000
5    60000
Name: Salary, dtype: int64
```

```
In [63]: vis1=sns.distplot(clean_data['Salary'])
```

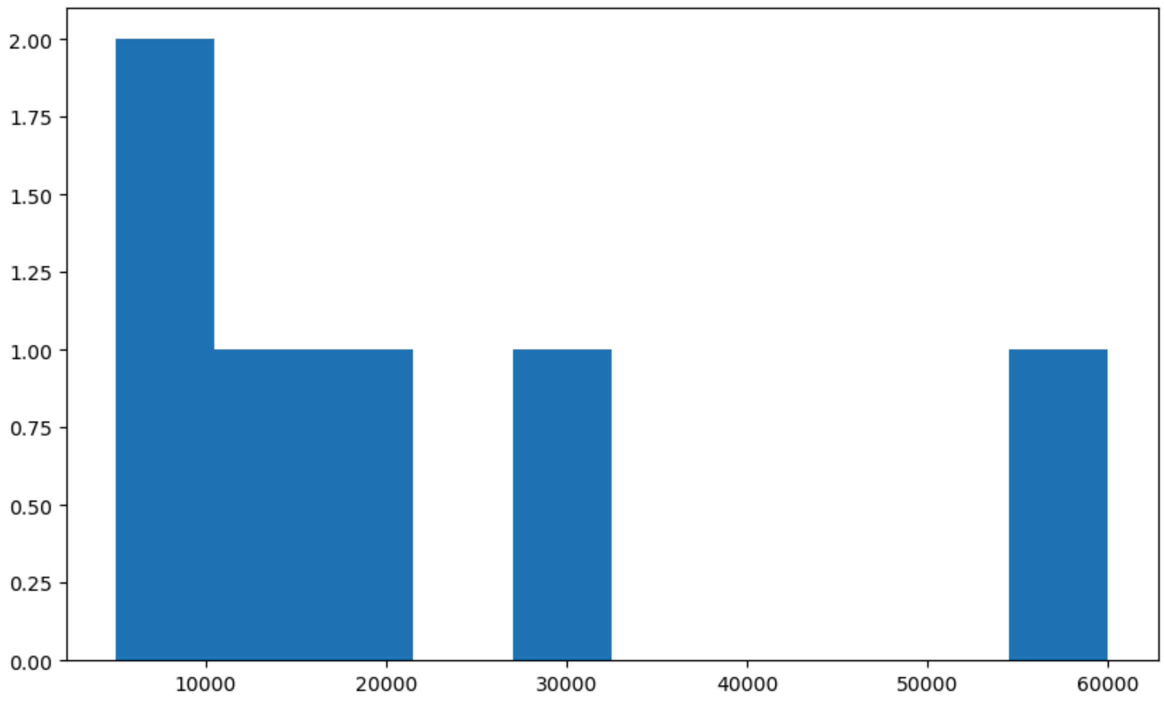


```
In [64]: plt.rcParams['figure.figsize']=10,6
```

```
In [65]: vis1=sns.distplot(clean_data['Salary'])
```



```
In [66]: vis2=plt.hist(clean_data['Salary'])
```

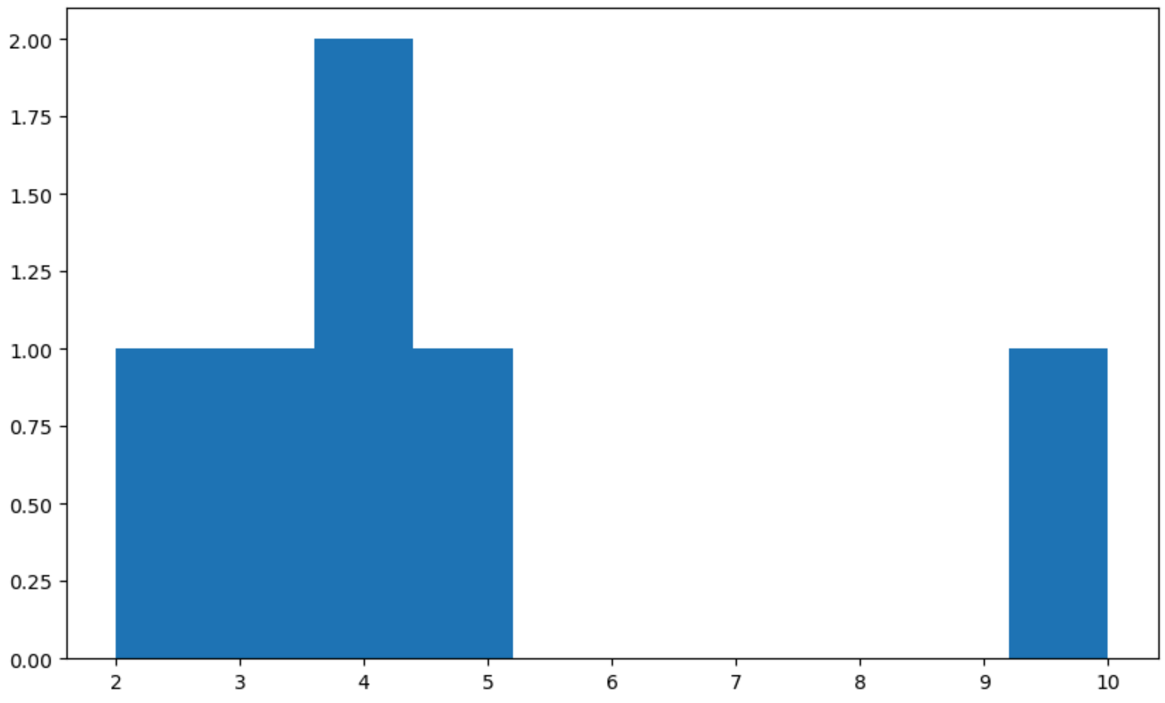


```
In [67]: vis3=sns.hist(clean_data['Exp'])
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[67], line 1
----> 1 vis3=sns.hist(clean_data['Exp'])

AttributeError: module 'seaborn' has no attribute 'hist'
```

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



```
In [69]: clean_data['Exp']
```

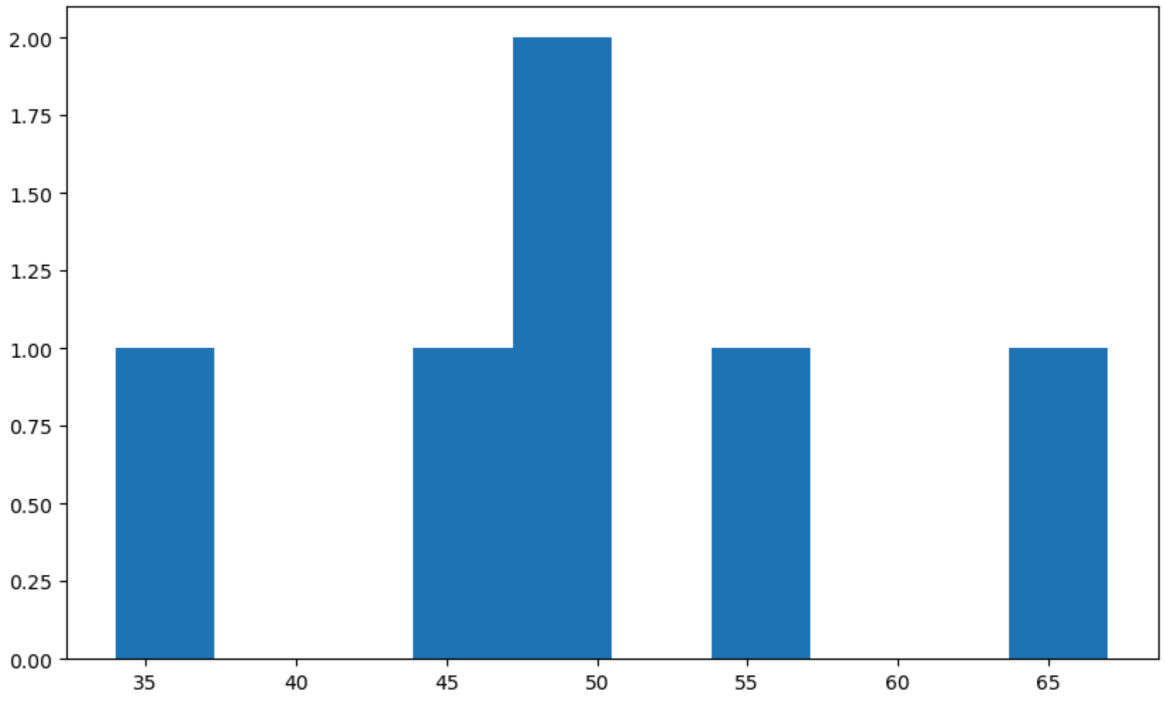
```
Out[69]: 0     2
         1     3
         2     4
         3     4
         4     5
         5    10
         Name: Exp, dtype: int64
```

```
In [70]: clean_data
```

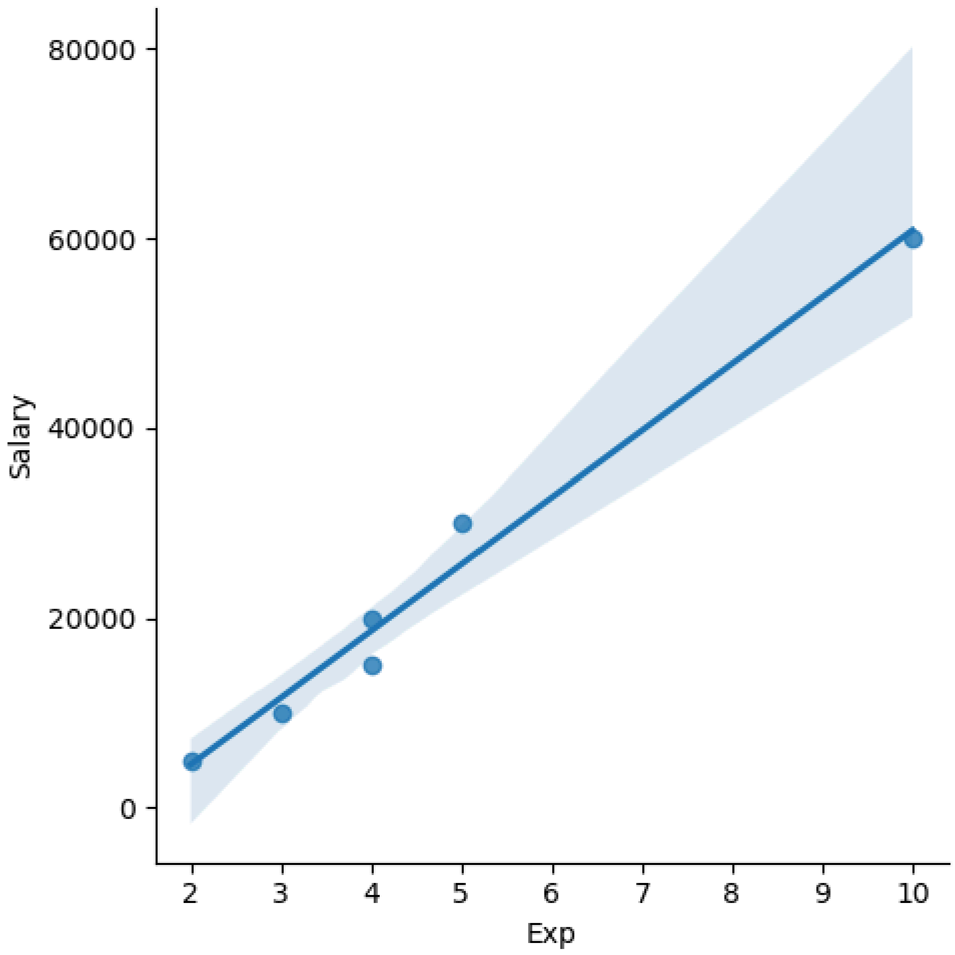
Out[70]:

	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

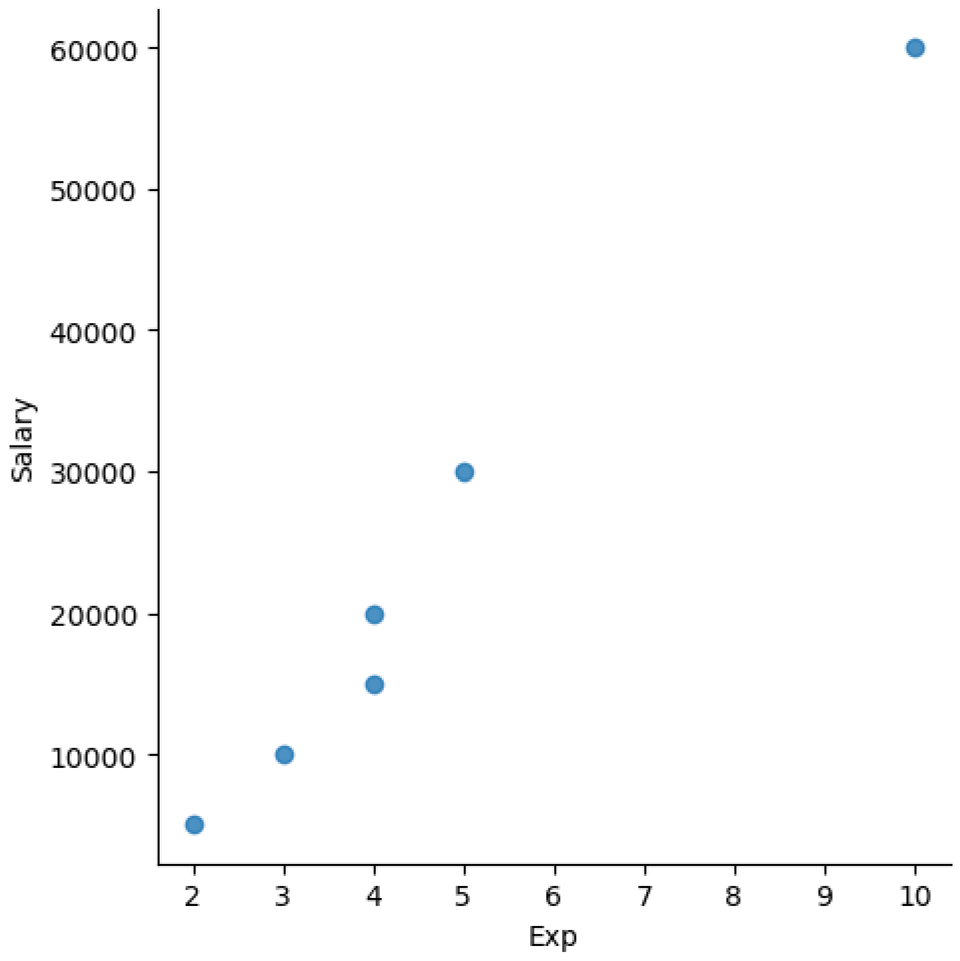
```
In [75]: vis9=plt.hist(clean_data['Age'])
```



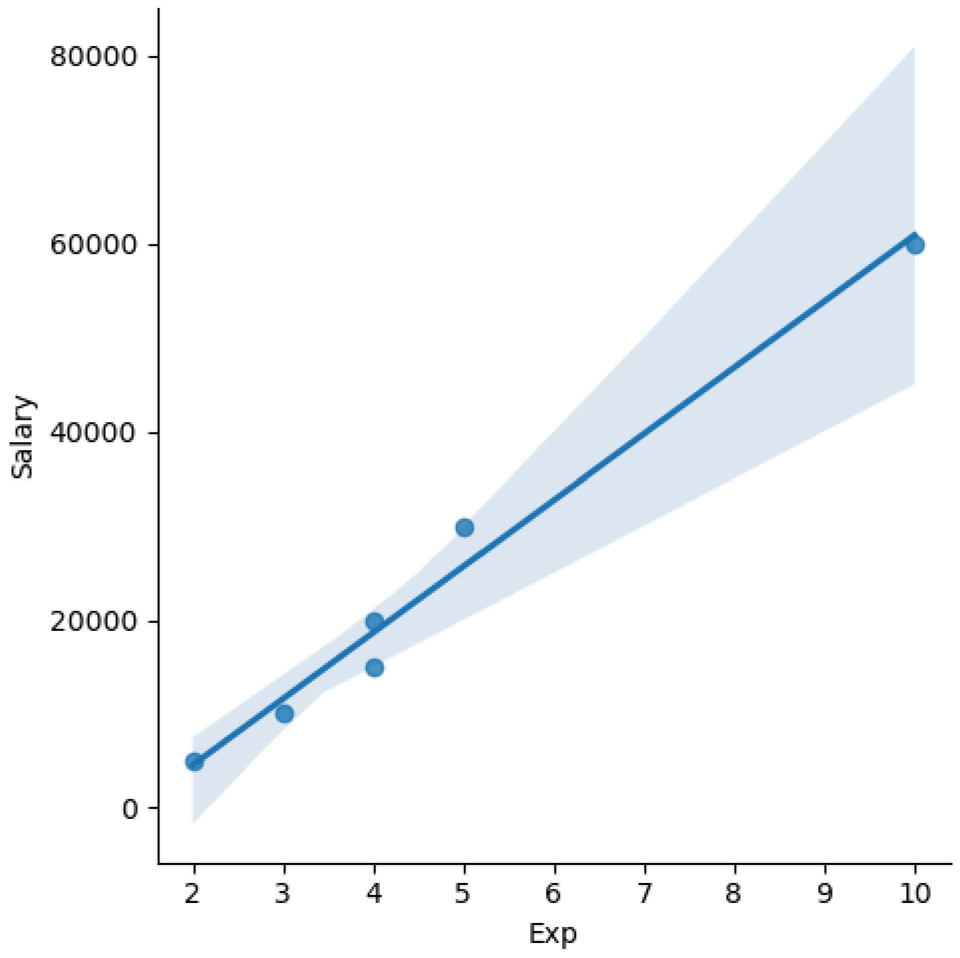
```
In [76]: vis4=sns.lmplot(data=clean_data,x='Exp',y='Salary')
```



In [77]:
```
vis5=sns.lmplot(data=clean_data,x='Exp',y='Salary',fit_reg=False)
```



```
In [78]: vis6=sns.lmplot(data=clean_data,x='Exp',y='Salary',fit_reg=True)
```



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	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[:]
```

Out[80]:

	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 [81]:
```
clean_data[:2]
```

Out[81]:

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Datascience	34	Mumbai	5000	2
1	Teddy	Testing	45	Bangalore	10000	3

In [82]:
```
clean_data[2:]
```

Out[82]:

	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 [83]:
```
clean_data[:]
```

Out[83]:

	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 [84]:
```
clean_data[0:1]
```

Out[84]:

	Name	Domain	Age	Location	Salary	Exp
0	Mike	Datascience	34	Mumbai	5000	2

In [85]:
```
clean_data[0:3][2:4]
```

Out[85]:

	Name	Domain	Age	Location	Salary	Exp
2	Umar	Dataanalyst	50	Bangalore	15000	4

In [86]:
```
clean_data[0,3]
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
File D:\New folder\Lib\site-packages\pandas\core\indexes\base.py:3805, in Index.g
et_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.Py
ObjectHashTable.get_item()

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

KeyError: (0, 3)

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

KeyError                                  Traceback (most recent call last)
Cell In[86], line 1
----> 1 clean_data[0,3]

File D:\New folder\Lib\site-packages\pandas\core\frame.py:4102, in DataFrame.__ge
titem__(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 D:\New folder\Lib\site-packages\pandas\core\indexes\base.py:3812, in Index.g
et_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: (0, 3)
```

In [87]:
```
clean_data
```

Out[87]:

	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 [88]:
```
x_iv=clean_data.drop(['Salary'],axis=1)
```

In [89]:
```
clean_data
```

Out[89]:

	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 [90]:
```
x_iv
```

Out[90]:

	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 [91]:
```
clean_data
```

Out[91]:

	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]:
```
x_iv
```

Out[92]:

	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 [93]:
```
x_iv=clean_data['Salary'],axis=0)
```

```
  Cell In[93], line 1
    x_iv=clean_data['Salary'],axis=0)
                                    ^
SyntaxError: unmatched ')'
```

In [94]:
```
x_iv=clean_data(['salary'],axis=0)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[94], line 1
----> 1 x_iv=clean_data(['salary'],axis=0)

TypeError: 'DataFrame' object is not callable
```

In [103…
```
x_iv=clean_data.drop(['Salary'],axis=2)
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
D:\New folder\Lib\site-packages\pandas\core\generic.py in ?(cls, axis)
    576             return cls._AXIS_TO_AXIS_NUMBER[axis]
    577         except KeyError:
--> 578             raise ValueError(f"No axis named {axis} for object type {cls.
__name__}")

KeyError: 2

During handling of the above exception, another exception occurred:

ValueError                                Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_5480\3457634926.py in ?()
----> 1 x_iv=clean_data.drop(['Salary'],axis=2)

D:\New folder\Lib\site-packages\pandas\core\frame.py in ?(self, labels, axis, ind
ex, columns, level, inplace, errors)
   5577                 weight  250.0   150.0
   5578         falcon  speed   320.0   250.0
   5579                 weight  1.0     0.8
   5580         """
-> 5581         return super().drop(
   5582             labels=labels,
   5583             axis=axis,
   5584             index=index,

D:\New folder\Lib\site-packages\pandas\core\generic.py in ?(self, labels, axis, i
ndex, columns, level, inplace, errors)
   4769
   4770         if labels is not None:
   4771             if index is not None or columns is not None:
   4772                 raise ValueError("Cannot specify both 'labels' and 'inde
x'/'columns'")
-> 4773             axis_name = self._get_axis_name(axis)
   4774             axes = {axis_name: labels}
   4775         elif index is not None or columns is not None:
   4776             axes = {"index": index}

D:\New folder\Lib\site-packages\pandas\core\generic.py in ?(cls, axis)
    580     @final
    581     @classmethod
    582     def _get_axis_name(cls, axis: Axis) -> Literal["index", "columns"]:
--> 583         axis_number = cls._get_axis_number(axis)
    584         return cls._AXIS_ORDERS[axis_number]

D:\New folder\Lib\site-packages\pandas\core\generic.py in ?(cls, axis)
    574     def _get_axis_number(cls, axis: Axis) -> AxisInt:
    575         try:
    576             return cls._AXIS_TO_AXIS_NUMBER[axis]
    577         except KeyError:
--> 578             raise ValueError(f"No axis named {axis} for object type {cls.
__name__}")

ValueError: No axis named 2 for object type DataFrame
```

In [102…

```
x_iv=clean_data.drop(['Salary'],axis=1)
```

In [97]:

```
clean_data
```

Out[97]:

	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 [98]:
```
x_iv.columns
```

Out[98]:
```
Index(['Name', 'Domain', 'Age', 'Location', 'Exp'], dtype='object')
```

In [99]:
```
clean_data.columns
```

Out[99]:
```
Index(['Name', 'Domain', 'Age', 'Location', 'Salary', 'Exp'], dtype='object')
```

In [100…
```
y_dv=clean_data(['Name','Domain','Age','Location','Exp'],axis=2)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[100], line 1
----> 1 y_dv=clean_data(['Name','Domain','Age','Location','Exp'],axis=2)

TypeError: 'DataFrame' object is not callable
```

In [101…
```
y_dv=clean_data(['Name','Domain','Age','Location','Exp'],axis=0)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[101], line 1
----> 1 y_dv=clean_data(['Name','Domain','Age','Location','Exp'],axis=0)

TypeError: 'DataFrame' object is not callable
```

In [106…
```
y_dv=clean_data.drop(['Name','Domain','Age','Location','Exp'],axis=0)
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[106], line 1
----> 1 y_dv=clean_data.drop(['Name','Domain','Age','Location','Exp'],axis=0)

File D:\New folder\Lib\site-packages\pandas\core\frame.py:5581, in DataFrame.drop
(self, labels, axis, index, columns, level, inplace, errors)
   5433 def drop(
   5434     self,
   5435     labels: IndexLabel | None = None,
   (...)
   5442     errors: IgnoreRaise = "raise",
   5443 ) -> DataFrame | None:
   5444     """
   5445     Drop specified labels from rows or columns.
   5446 
   (...)
   5579             weight  1.0     0.8
   5580     """
-> 5581     return super().drop(
   5582         labels=labels,
   5583         axis=axis,
   5584         index=index,
   5585         columns=columns,
   5586         level=level,
   5587         inplace=inplace,
   5588         errors=errors,
   5589     )

File D:\New folder\Lib\site-packages\pandas\core\generic.py:4788, in NDFrame.drop
(self, labels, axis, index, columns, level, inplace, errors)
   4786 for axis, labels in axes.items():
   4787     if labels is not None:
-> 4788         obj = obj._drop_axis(labels, axis, level=level, errors=errors)
   4790 if inplace:
   4791     self._update_inplace(obj)

File D:\New folder\Lib\site-packages\pandas\core\generic.py:4830, in NDFrame._dro
p_axis(self, labels, axis, level, errors, only_slice)
   4828         new_axis = axis.drop(labels, level=level, errors=errors)
   4829     else:
-> 4830         new_axis = axis.drop(labels, errors=errors)
   4831     indexer = axis.get_indexer(new_axis)
   4833 # Case for non-unique axis
   4834 else:

File D:\New folder\Lib\site-packages\pandas\core\indexes\base.py:7070, in Index.d
rop(self, labels, errors)
   7068 if mask.any():
   7069     if errors != "ignore":
-> 7070         raise KeyError(f"{labels[mask].tolist()} not found in axis")
   7071     indexer = indexer[~mask]
   7072 return self.delete(indexer)

KeyError: "['Name', 'Domain', 'Age', 'Location', 'Exp'] not found in axis"
```

In [107…
```
y_dv=clean_data.drop(['Name','Domain','Age','Location','Exp'],axis=1)
```

In [108…
```
y_dv
```

Out[108…

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

In [109…

```
clean_data
```

Out[109…

	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 [110…

```
x_iv
```

Out[110…

	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 [111…

```
y_dv
```

Out[111…

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

In [112…

```
clean_data
```

Out[112…

	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 [113…

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

In [114…

```
imputation
```

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

In [115…

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

Out[115…

	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 [116…
```
clean_data[::-1]
```

Out[116…

	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 [117…
```
x_iv=clean_data[['Name','Domain','Age','Location','Exp']]
```

In [118…
```
x_iv
```

Out[118…

	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 [119…
```
y_dv=clean_data[['Salary']]
```

In [120…
```
y_dv
```

Out[120…

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

In [121…
```
emp
```

Out[121…

	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 [122…

```
clean_data
```

Out[122…

	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 [123…

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

In [124…

```
imputation
```

Out[124…

	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 [125…

```
clean_data
```

Out[125…

	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 [126…

```
imputation
```

Out[126…

	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 [ ]: