In [4]:

```python
# Importing all important Libraries

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib as plt

import warnings

warnings.filterwarnings('ignore')

from IPython import display

pd.set_option('display.max_columns',None)

pd.set_option('display.max_rows',None)

```

Data Preprocessing :

In [6]:

```python
df = pd.read_csv('Salary_Data (1).csv')
df.head()
```

Out[6]:

	YearsExperience	Salary
0	1.1	39343.0
1	1.3	46205.0
2	1.5	37731.0
3	2.0	43525.0
4	2.2	39891.0

Data Preprocessing :

In [11]:

```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30 entries, 0 to 29
Data columns (total 2 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   YearsExperience  30 non-null     float64
 1   Salary           30 non-null     float64
dtypes: float64(2)
memory usage: 612.0 bytes
```

In [8]:
```
df.describe()
```

Out[8]:

	YearsExperience	Salary
count	30.000000	30.000000
mean	5.313333	76003.000000
std	2.837888	27414.429785
min	1.100000	37731.000000
25%	3.200000	56720.750000
50%	4.700000	65237.000000
75%	7.700000	100544.750000
max	10.500000	122391.000000

In [9]:
```
df.nunique()
```

Out[9]:
```
YearsExperience    28
Salary             30
dtype: int64
```

In [12]:
```
df.axes
```

Out[12]:
```
[RangeIndex(start=0, stop=30, step=1),
 Index(['YearsExperience', 'Salary'], dtype='object')]
```

In [13]:
```
df.shape
```

Out[13]:
```
(30, 2)
```

In [14]:
```
df.columns
```

Out[14]:
```
Index(['YearsExperience', 'Salary'], dtype='object')
```

In [15]:
```
df.shape
```

Out[15]:
```
(30, 2)
```

In [16]:
```
df.dtypes
```

Out[16]:
```
YearsExperience    float64
Salary             float64
dtype: object
```

In [17]:
```
df.isna().sum()
```

Out[17]:
```
YearsExperience    0
Salary             0
dtype: int64
```

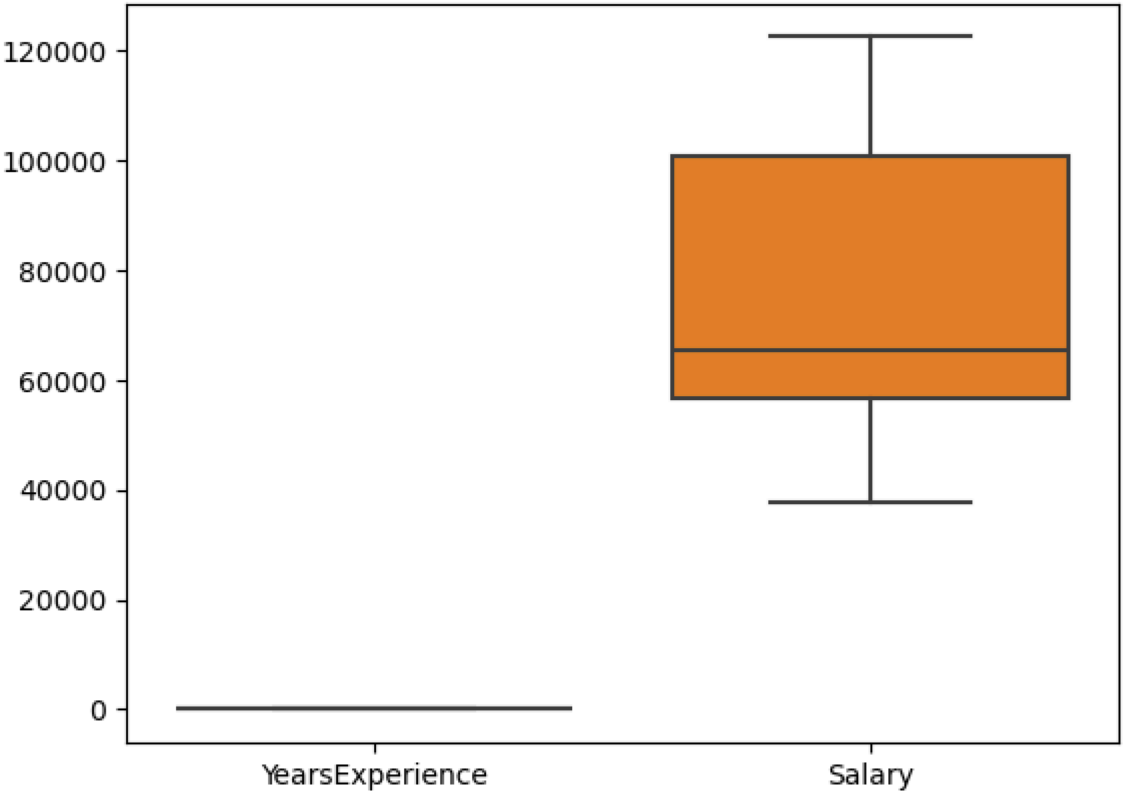
Data Visualization :

In [18]:

```
sns.boxplot(df)
```

Out[18]:

```
<Axes: >
```

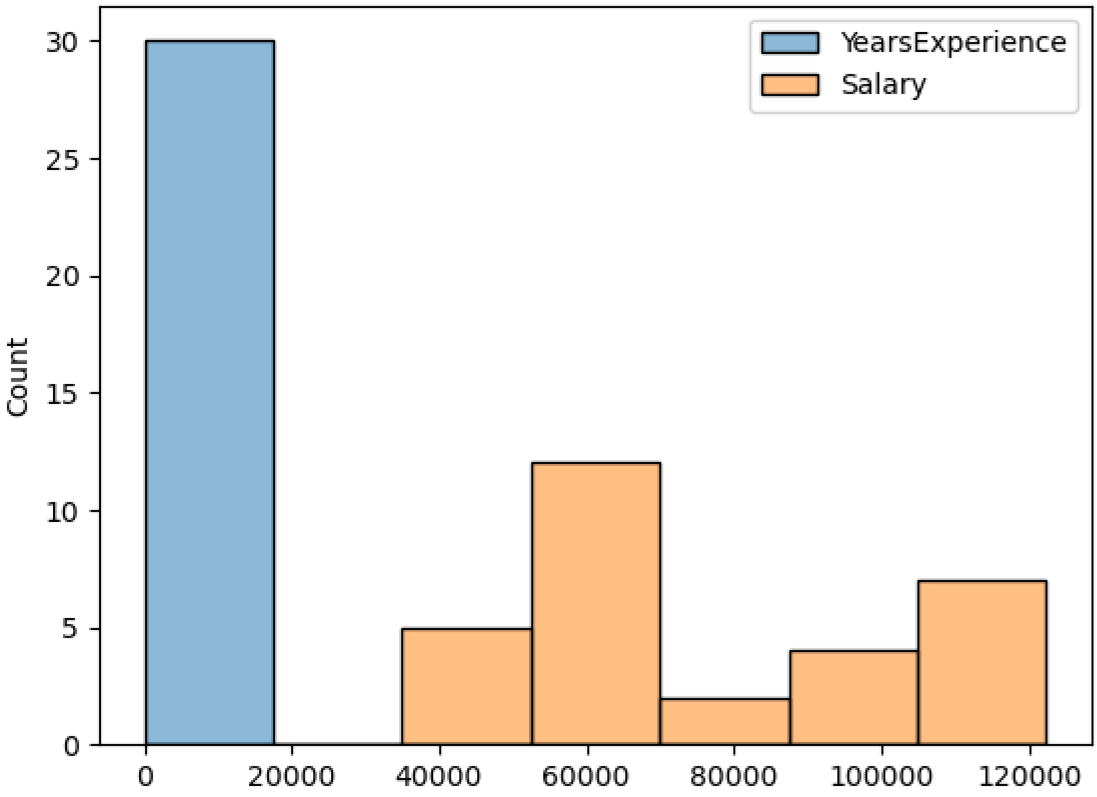


In [20]:

```
sns.histplot(df)
```

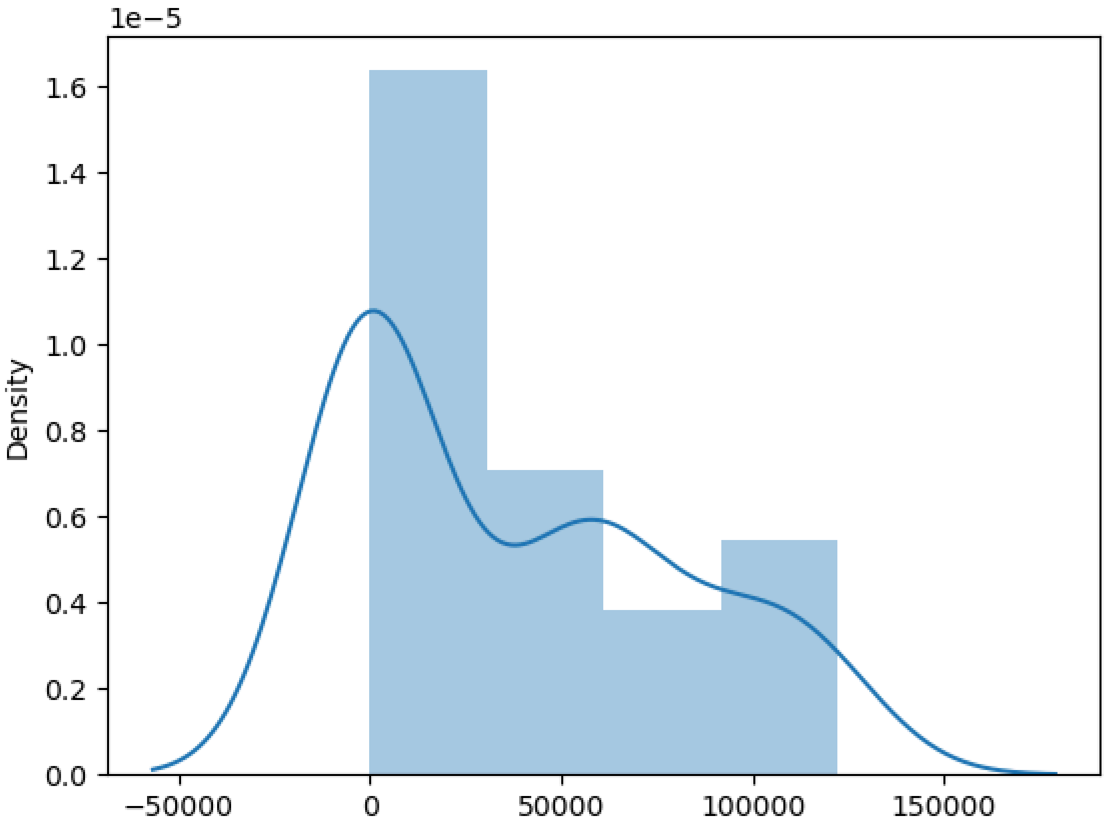
Out[20]:

```
<Axes: ylabel='Count'>
```



In [21]:
```
sns.distplot(df)
```

Out[21]:
```
<Axes: ylabel='Density'>
```

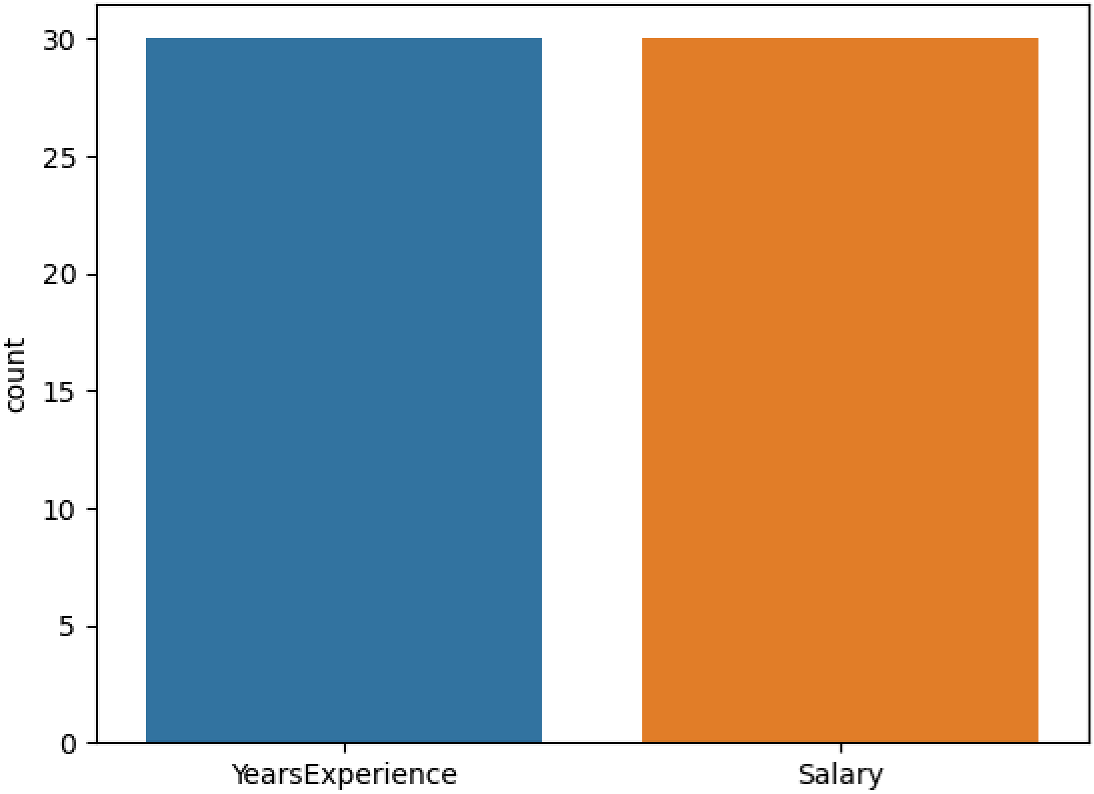


In [23]:

```
sns.countplot(df)
```

Out[23]:

```
<Axes: ylabel='count'>
```

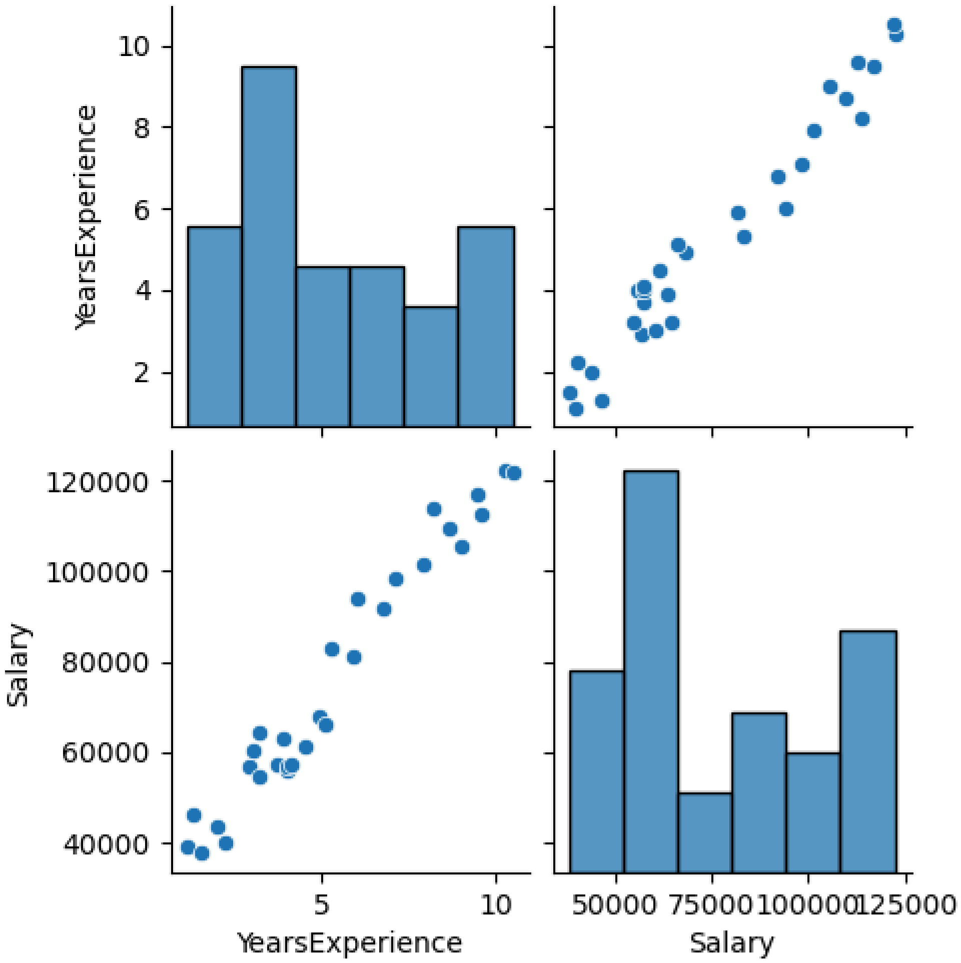


In [24]:

```
sns.pairplot(df)
```

Out[24]:

```
<seaborn.axisgrid.PairGrid at 0x25534717dd0>
```



Compairing Numerical Feature with Categorical Features :

In [26]:

```
df.select_dtypes(include = 'object').head()
```

Out[26]:

```
0
1
2
3
4
```

In [27]:

```
df.select_dtypes(exclude = 'object').head()
```

Out[27]:

	YearsExperience	Salary
0	1.1	39343.0
1	1.3	46205.0
2	1.5	37731.0
3	2.0	43525.0
4	2.2	39891.0

Using groupby :

In [29]:

```
df.groupby('Salary').first().head()
```

Out[29]:

	YearsExperience
Salary	
37731.0	1.5
39343.0	1.1
39891.0	2.2
43525.0	2.0
46205.0	1.3

Compairing two features :

In [31]:

```
pd.crosstab(df['Salary'],df['YearsExperience']).head()
```

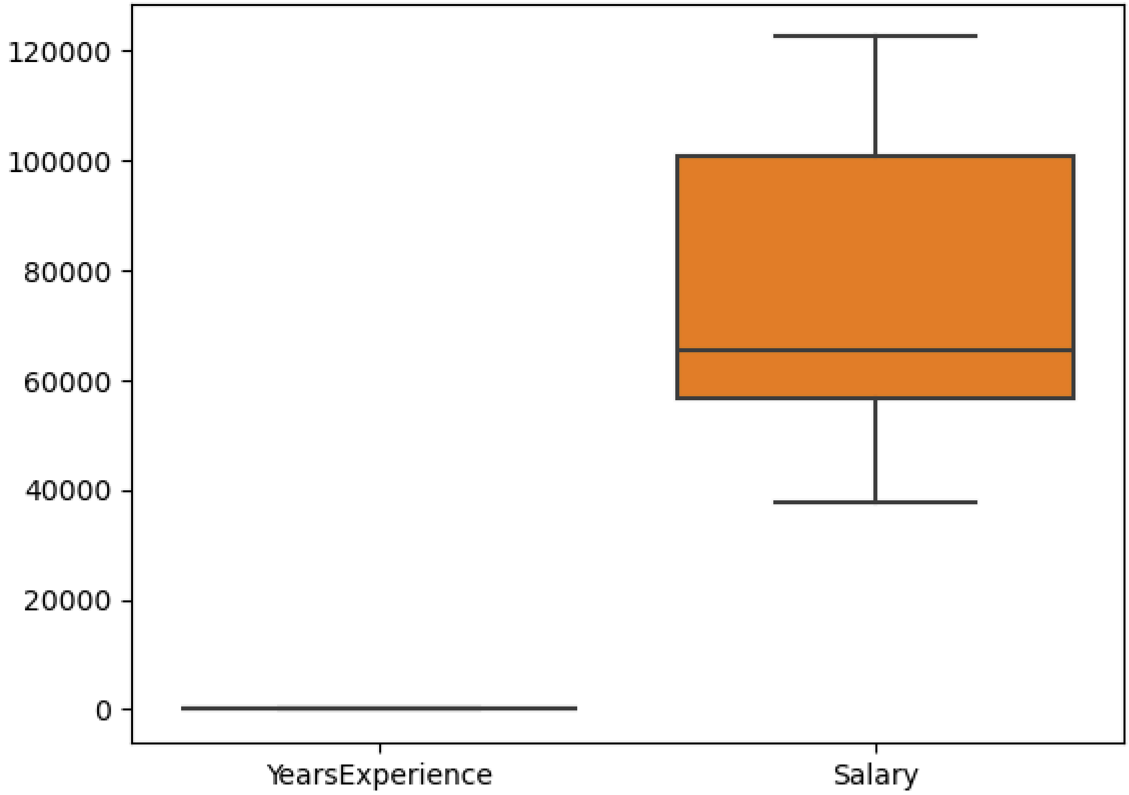
Out[31]:

YearsExperience	1.1	1.3	1.5	2.0	2.2	2.9	3.0	3.2	3.7	3.9	4.0	4.1	4.5	4.9	5.1	5.3	5.9
Salary																	
37731.0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
39343.0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
39891.0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
43525.0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
46205.0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Handling Outliers :

In [35]:
```
sns.boxplot(df)
```

Out[35]:
```
<Axes: >
```



In [34]:
```
df.columns
```

Out[34]:
```
Index(['YearsExperience', 'Salary'], dtype='object')
```

In [ ]:
```
There is no outliers present in Datasets to handle.
```

Feature Selection :

Linearity :

In [85]:
```
import matplotlib
import seaborn
import matplotlib.pyplot as plt
import seaborn as sns

```

In [86]:

```
r = df.corr()[['Salary']]
r
```

Out[86]:

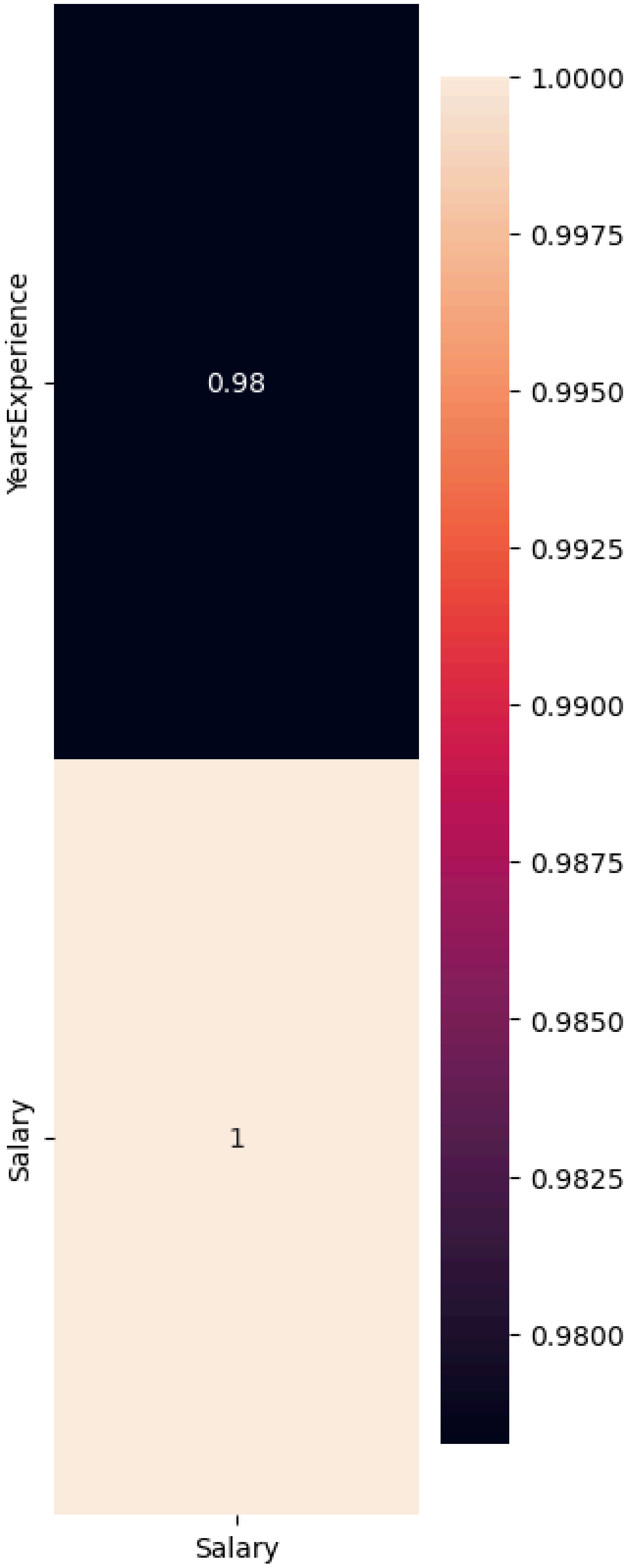
```
                    Salary
YearsExperience   0.978242
         Salary   1.000000
```

In [61]:

```
plt.figure(figsize = (3,10))
sns.heatmap(r,annot = True)
```

Out[61]:

```
<Axes: >
```



Multicollinearity :

In [62]:

```
df1 = df.drop('Salary',axis = 1)
df1.head()
```

Out[62]:

	YearsExperience
0	1.1
1	1.3
2	1.5
3	2.0
4	2.2

In [63]:

```
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

In [64]:

```
vif_list = []
for i in range(df.shape[1]) :
 vif = variance_inflation_factor(df,i)
 vif_list.append(vif)
print(vif_list)
```

```
[37.14597194848691, 37.14597194848691]
```

Model Building :

In [71]:

```
from sklearn.linear_model import LinearRegression
```

In [73]:

```
x = df.drop('Salary',axis = 1)
y = df['Salary']
```

In [74]:

```
from sklearn.model_selection import train_test_split
```

In [80]:

```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size =0.2 , rand
```

In [81]:

```
model = LinearRegression()
```

In [82]:

```
model.fit(x_train,y_train)
```

Out[82]:

```
LinearRegression()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [83]:

```
y_pred = model.predict(x_test)
```

In [84]:

```
y_pred_train = model.predict(x_train)
```

In [87]:

```
df3 = df.to_csv('third.csv')
df3
```

In [ ]:

```

```