Simple Linear Regression Assignment

* Salary vs Experience

In [6]:

```
import pandas as pd

from matplotlib import pyplot as plt
import seaborn as sns

import warnings
warnings.filterwarnings('ignore')
```

In [6]:

```
salary_data = pd.read_csv('Salary_Data.csv')
salary_data
```

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
5	2.9	56642.0
6	3.0	60150.0
7	3.2	54445.0
8	3.2	64445.0
9	3.7	57189.0
10	3.9	63218.0
11	4.0	55794.0
12	4.0	56957.0
13	4.1	57081.0
14	4.5	61111.0
15	4.9	67938.0
16	5.1	66029.0
17	5.3	83088.0
18	5.9	81363.0
19	6.0	93940.0
20	6.8	91738.0
21	7.1	98273.0
22	7.9	101302.0
23	8.2	113812.0
24	8.7	109431.0
25	9.0	105582.0
26	9.5	116969.0
27	9.6	112635.0
28	10.3	122391.0
29	10.5	121872.0

In [3]:

```
1 salary_data.shape
```

Out[3]:

```
(30, 2)
```

In [4]:

```
1 salary_data.isna().sum()
```

Out[4]:

```
YearsExperience    0
Salary             0
dtype: int64
```

In [5]:

```
1 salary_data.dtypes
```

Out[5]:

```
YearsExperience    float64
Salary             float64
dtype: object
```

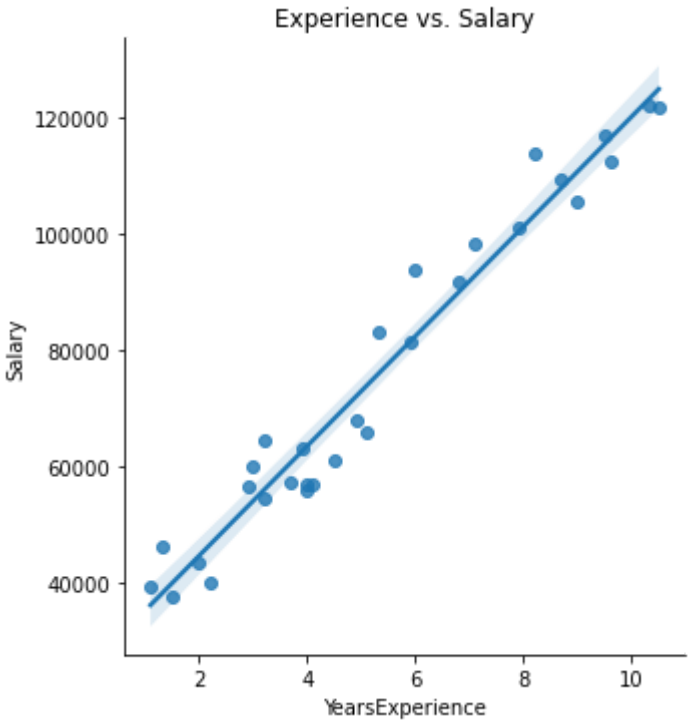
Linearity test

In [6]:

```
1 sns.scatterplot(x='YearsExperience',y='Salary',data=salary_data)
2 plt.title('Experience vs. Salary')
3 plt.show()
```

In [7]:

```
1 sns.lmplot(x='YearsExperience',y='Salary',data=salary_data)
2 plt.title('Experience vs. Salary')
3 plt.show()
```

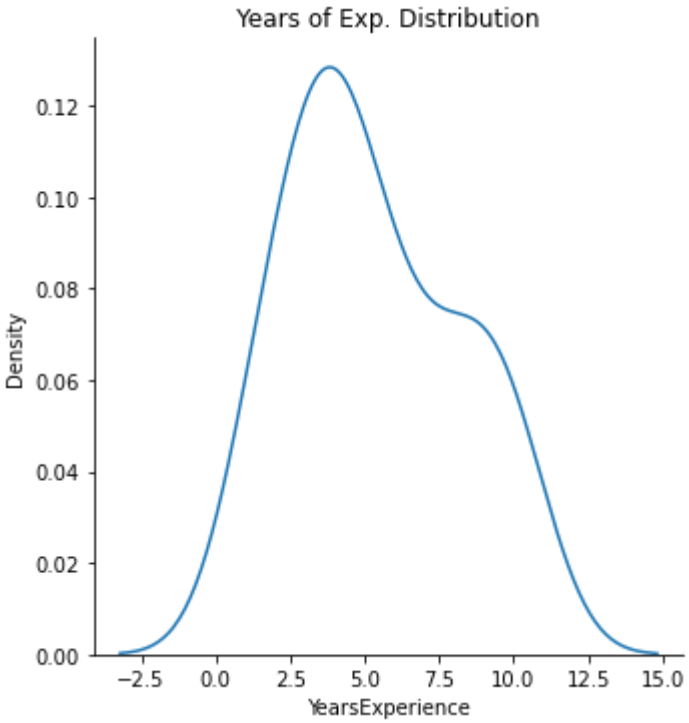


Linearity test is passed

Normality test

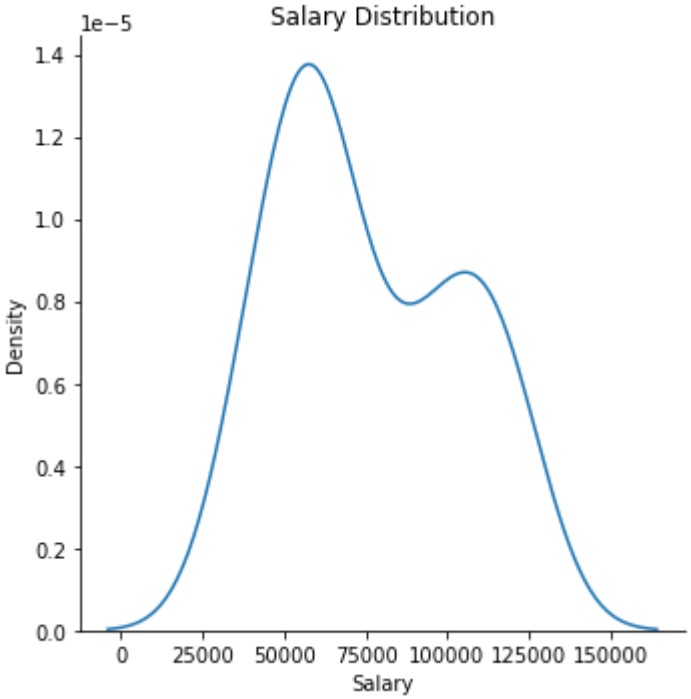
In [9]:

```
sns.displot(data=salary_data,x='YearsExperience',kind='kde')
plt.title('Years of Exp. Distribution')
plt.show()
```



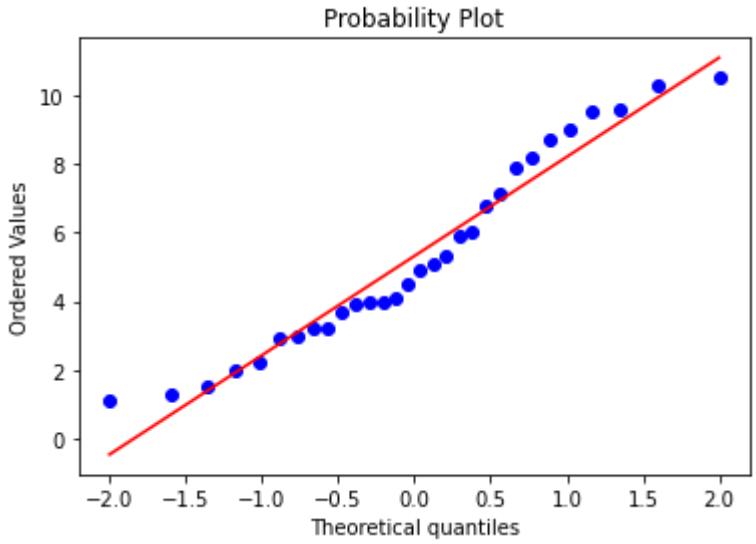
In [10]:

```
1 sns.displot(data=salary_data,x='Salary',kind='kde')
2 plt.title('Salary Distribution')
3 plt.show()
```



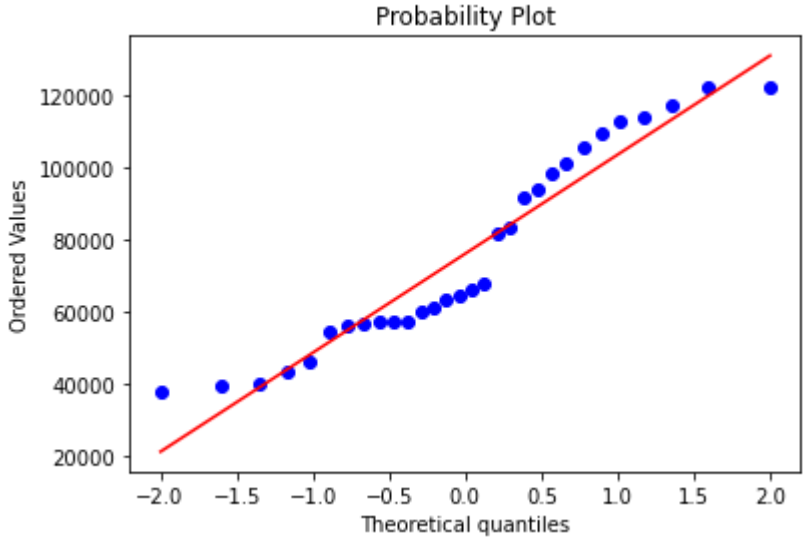
In [11]:

```
from scipy import stats
stats.probplot(x = salary_data['YearsExperience'],dist='norm',plot=plt)
plt.show()
```



In [12]:

```
from scipy import stats
stats.probplot(x = salary_data['Salary'],dist='norm',plot=plt)
plt.show()
```



Normality test is failed

Multi-colinearity test

In [13]:

```
salary_corr = salary_data.corr().round(2)
salary_corr
```

Out[13]:

	YearsExperience	Salary
YearsExperience	1.00	0.98
Salary	0.98	1.00

There is multicolinearity in this data, hence multicolinearity test is failed

Using Stats Model

In [18]:

```
import statsmodels.formula.api as smf
```

In [30]:

```
linear_model_1 = smf.ols(formula = 'YearsExperience~Salary', data = salary_data).fit()
linear_model_1
```

Out[30]:

```
<statsmodels.regression.linear_model.RegressionResultsWrapper at 0x215ca0874
c0>
```

In [31]:

```
linear_model_1.params
```

Out[31]:

```
Intercept   -2.383161
Salary       0.000101
dtype: float64
```

In [21]:

```
salary_data.head()
```

Out[21]:

	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

In [77]:

```
X_test = pd.DataFrame(data={'YearsExperience':[1.1,1.3,2,1.5,3]})
X_test
```

Out[77]:

	YearsExperience
0	1.1
1	1.3
2	2.0
3	1.5
4	3.0

In [78]:

```
linear_model.predict(X_test)
```

Out[78]:

```
array([[36187.15875227],
       [38077.15121656],
       [44692.12484158],
       [39967.14368085],
       [54142.08716303]])
```

Using Sk Learn Model

In [79]:

```
X = salary_data.drop('Salary',axis=1)
y = salary_data[['Salary']]
```

In [80]:

```
from sklearn.linear_model import LinearRegression
linear_model = LinearRegression()
linear_model.fit(X,y)
```

Out[80]:

```
LinearRegression()
```

In [81]:

```
linear_model.coef_
```

Out[81]:

```
array([[9449.96232146]])
```

In [82]:

```
1 linear_model.intercept_
```

Out[82]:

```
array([25792.20019867])
```

In [83]:

```
1 y_pred = linear_model.predict(X)
```

In [76]:

```
error = y - y_pred
error
```

Out[76]:

	Salary
0	3155.841248
1	8127.848783
2	-2236.143681
3	-1167.124842
4	-6691.117306
5	3444.909069
6	6007.912837
7	-1587.079627
8	8412.920373
9	-3568.060788
10	570.946748
11	-7798.049484
12	-6635.049484
13	-7456.045717
14	-7206.030645
15	-4159.015574
16	-7958.008038
17	7210.999498
18	-183.977895
19	11448.025873
20	1686.056015
21	5386.067319
22	855.097462
23	10530.108765
24	1424.127605
25	-5259.861092
26	1402.157748
27	-3876.838485
28	-735.812110
29	-3144.804574

* Delivery time

In [7]:

```
time_data = pd.read_csv('delivery_time.csv')
time_data
```

Out[7]:

	Delivery Time	Sorting Time
0	21.00	10
1	13.50	4
2	19.75	6
3	24.00	9
4	29.00	10
5	15.35	6
6	19.00	7
7	9.50	3
8	17.90	10
9	18.75	9
10	19.83	8
11	10.75	4
12	16.68	7
13	11.50	3
14	12.03	3
15	14.88	4
16	13.75	6
17	18.11	7
18	8.00	2
19	17.83	7
20	21.50	5

In [29]:

```
time_data.shape
```

Out[29]:

```
(21, 2)
```

In [30]:

```
1 time_data.isna().sum()
```

Out[30]:

```
Delivery Time    0
Sorting Time     0
dtype: int64
```

In [31]:

```
1 time_data.dtypes
```

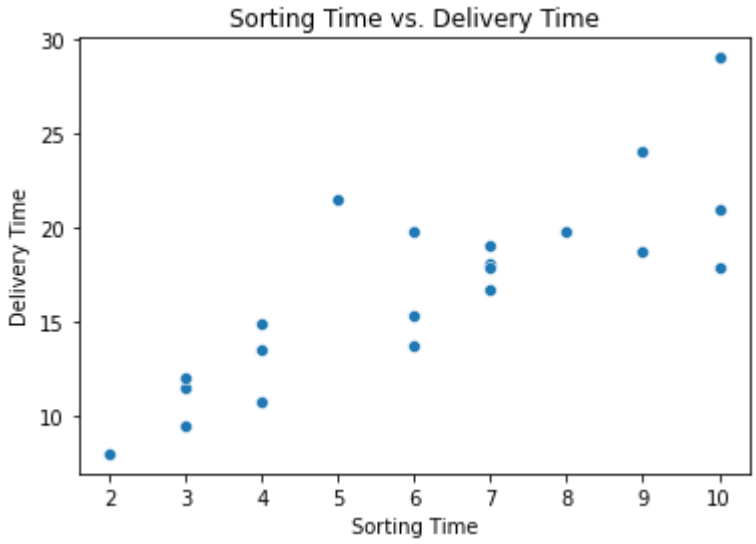
Out[31]:

```
Delivery Time    float64
Sorting Time       int64
dtype: object
```

Linearity test

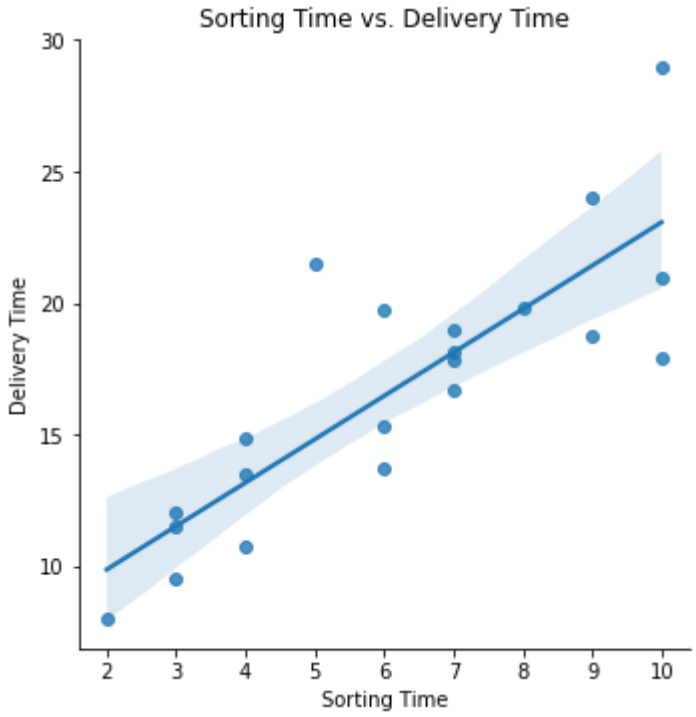
In [32]:

```
1 sns.scatterplot(x='Sorting Time',y='Delivery Time',data=time_data)
2 plt.title('Sorting Time vs. Delivery Time')
3 plt.show()
```



In [33]:

```
1 sns.lmplot(x='Sorting Time',y='Delivery Time',data=time_data)
2 plt.title('Sorting Time vs. Delivery Time')
3 plt.show()
```

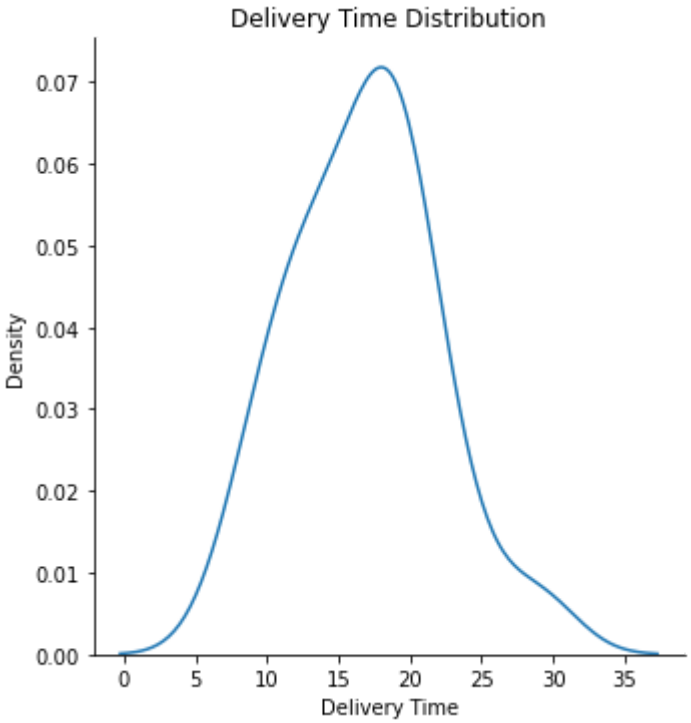


Linearity test failed

Normality test

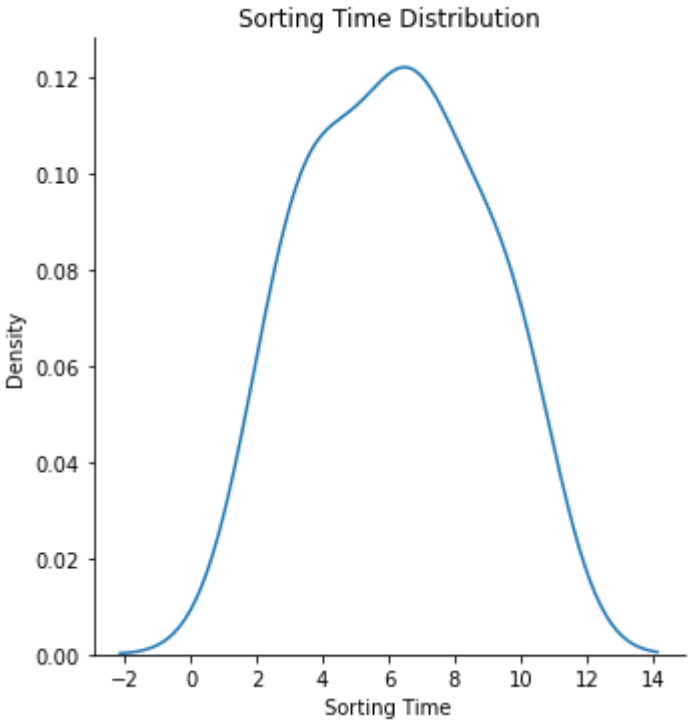
In [5]:

```
sns.displot(data=time_data,x='Delivery Time',kind='kde')
plt.title('Delivery Time Distribution')
plt.show()
```



In [7]:

```
sns.displot(data=time_data,x='Sorting Time',kind='kde')
plt.title('Sorting Time Distribution')
plt.show()
```



Normality test is passed

Multi-colinearity test

In [8]:

```
time_corr = time_data.corr().round(2)
time_corr
```

Out[8]:

	Delivery Time	Sorting Time
Delivery Time	1.00	0.83
Sorting Time	0.83	1.00

In [9]:

```
sns.heatmap(time_corr,annot=True)
plt.show()
```

There is multicolinearity in this data so, multicolinearity test is failed

Using Sk Learn Model

In [8]:

```
X = time_data.drop('Sorting Time',axis=1)
y = time_data[['Sorting Time']]
```

In [9]:

```
from sklearn.linear_model import LinearRegression
linear_model = LinearRegression()
linear_model.fit(X,y)
```

Out[9]:

```
LinearRegression()
```

In [10]:

```
1 linear_model.coef_
```

Out[10]:

```
array([[0.41374363]])
```

In [11]:

```
1 linear_model.intercept_
```

Out[11]:

```
array([-0.75667337])
```

In [12]:

```
1 import statsmodels.formula.api as smf
```

In [13]:

```
1 time_data.head()
```

Out[13]:

	Delivery Time	Sorting Time
0	21.00	10
1	13.50	4
2	19.75	6
3	24.00	9
4	29.00	10

In [44]:

```
1 y_pred = linear_model.predict(X)
```

In [45]:

```
error = y - y_pred
error
```

Out[45]:

	Sorting Time
0	2.068057
1	-0.828866
2	-1.414763
3	-0.173174
4	-1.241892
5	0.405709
6	-0.104456
7	-0.173891
8	3.350662
9	1.998980
10	0.552137
11	0.308929
12	0.855430
13	-1.001378
14	-1.220662
15	-1.399832
16	1.067698
17	0.263776
18	-0.553276
19	0.379624
20	-3.138815

In [49]:

```
error.head()
```

Out[49]:

	Sorting Time
0	2.068057
1	-0.828866
2	-1.414763
3	-0.173174
4	-1.241892

In [ ]:

```
1
```