Linear Regression Salary by Years of Experiences

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import preprocessing, svm
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn import metrics
import warnings
warnings.filterwarnings('ignore')
import statsmodels.api as smt
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
```

```
df = pd.read_csv("Salary_Data.csv")
df.head()
```

```
   YearsExperience  Salary
0              1.1   39343
1              1.3   46205
2              1.5   37731
3              2.0   43525
4              2.2   39891
```

```
df.shape
```

```
(30, 2)
```

```
df.describe()
```

```
       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
```

```
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     int64
dtypes: float64(1), int64(1)
memory usage: 612.0 bytes
```

```
df.isnull().sum()
```

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

```
    for k, v in df.items():
        q1 = v.quantile(0.25)
        q3 = v.quantile(0.75)
        irq = q3 - q1
        v_col = v[(v <= q1 - 1.5 * irq) | (v >= q3 + 1.5 * irq)]
        perc = np.shape(v_col)[0] * 100.0 / np.shape(df)[0]
        print("Column %s outliers = %.2f%%" % (k, perc))
```

```
Column YearsExperience outliers = 0.00%
Column Salary outliers = 0.00%
```

```
df.skew()
```

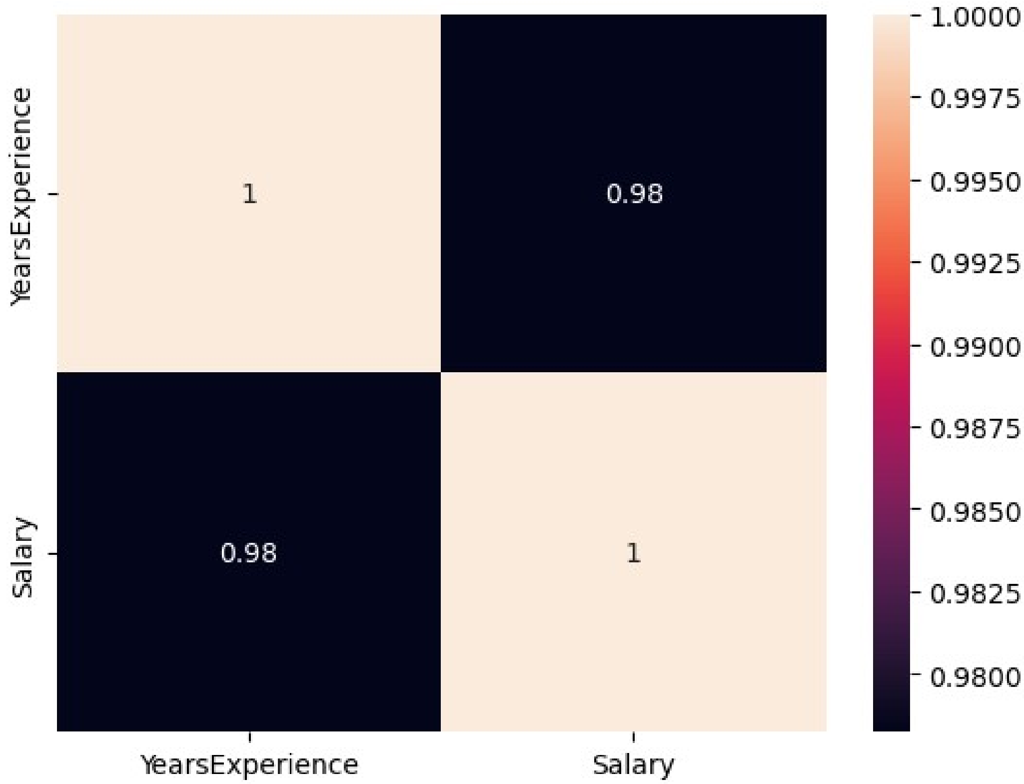
```
YearsExperience    0.37956
Salary             0.35412
dtype: float64
```

```
df.corr()
```

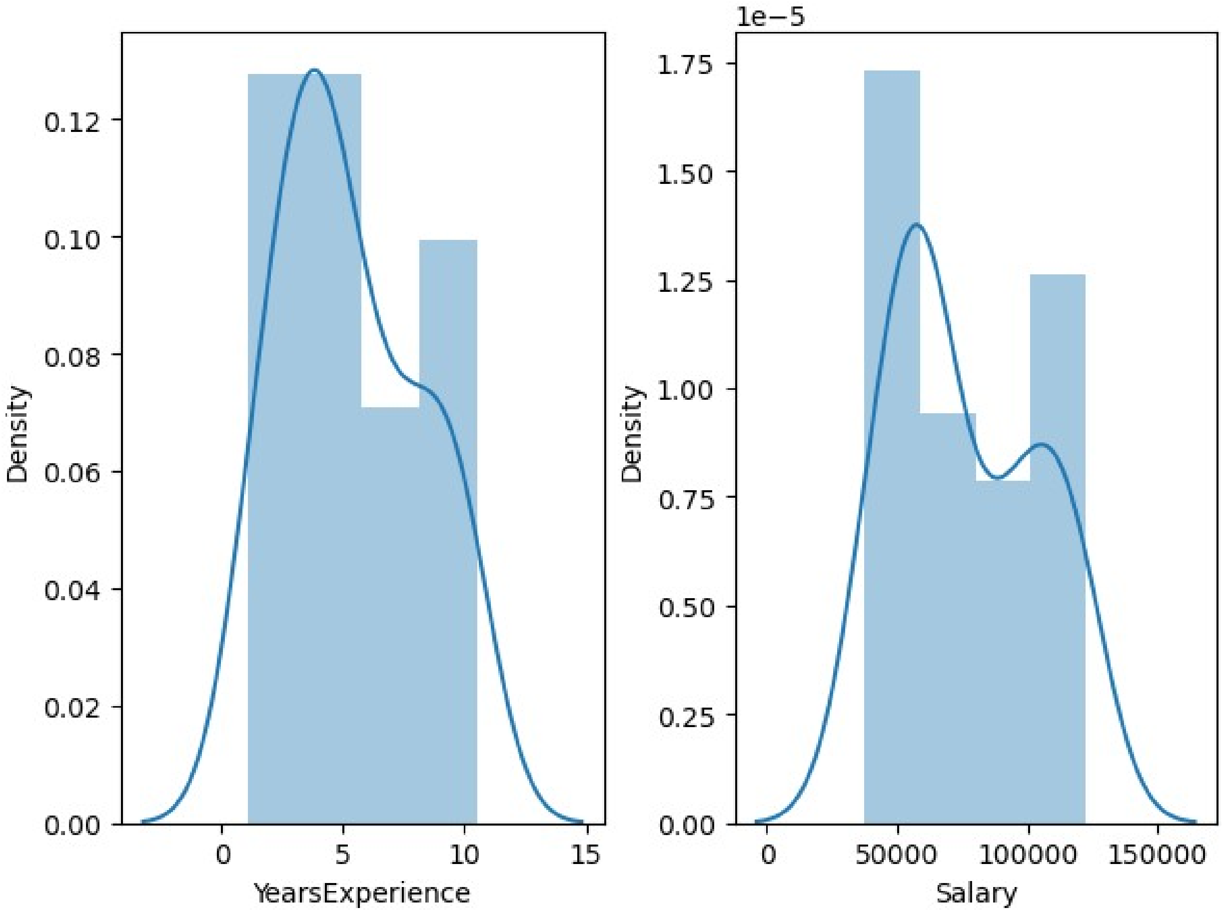
	YearsExperience	Salary
YearsExperience	1.000000	0.978242
Salary	0.978242	1.000000

```
sns.heatmap(data=df.corr(),annot=True)
```

```
<Axes: >
```

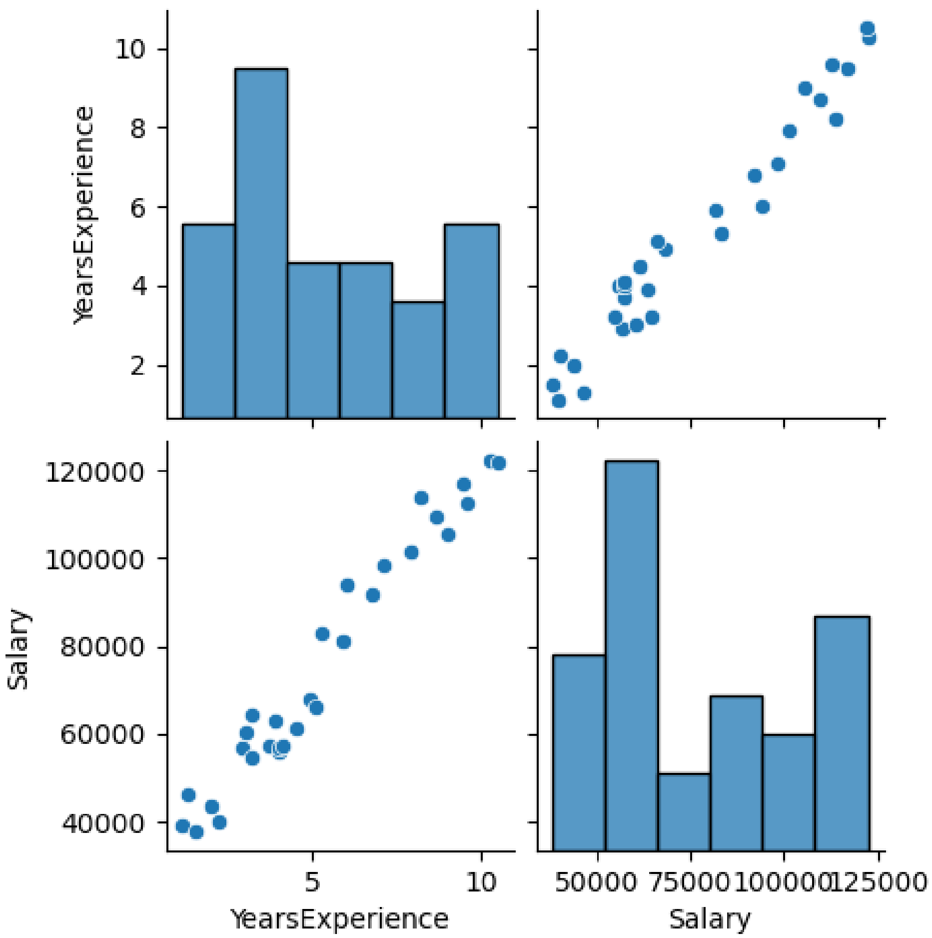



```
fig, axs = plt.subplots(ncols=2, nrows=1)
index = 0
axs = axs.flatten()
for k,v in df.items():
    sns.distplot(v, ax=axs[index])
    index += 1
plt.tight_layout(pad=0.4, w_pad=0.5, h_pad=5.0)
```



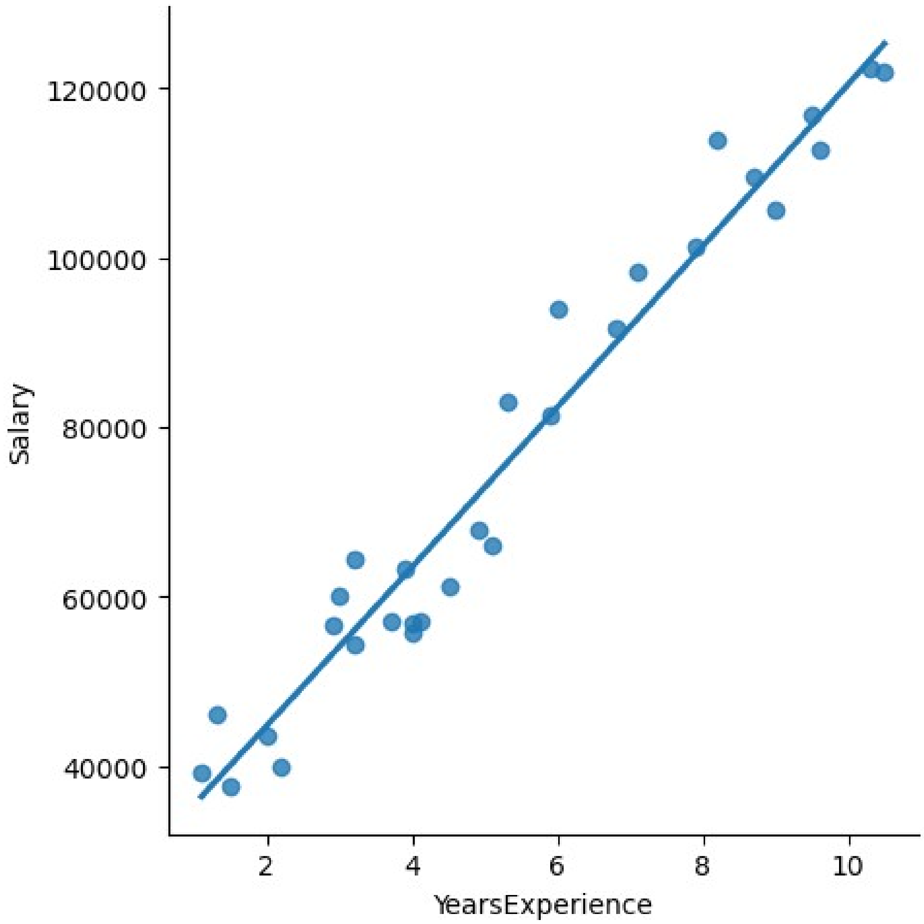

```
sns.pairplot(df)
```

```
<seaborn.axisgrid.PairGrid at 0x2c7959746d0>
```



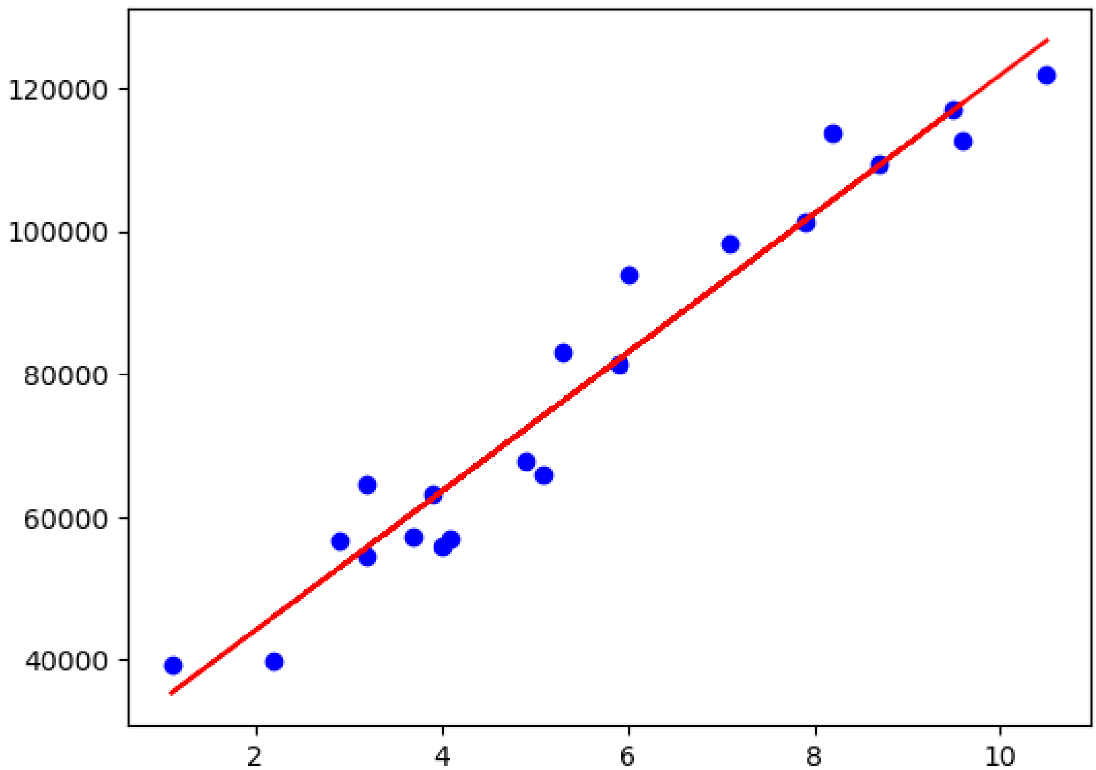

```
sns.lmplot(x='YearsExperience', y='Salary',data=df,order=2,ci=None)
```

```
<seaborn.axisgrid.FacetGrid at 0x2c795f83390>
```




```
x =np.array(df['YearsExperience']).reshape(-1,1)
y =np.array(df['Salary']).reshape(-1,1)

x_train, x_test, y_train,y_test = train_test_split(x,y,test_size=0.30)
reg = LinearRegression()
reg.fit(x_train,y_train)
print(y_train.flatten())

[ 93940  81363  83088 116969  56642  57081  39343  98273 113812  55794
 109431  57189 112635  64445 101302  66029  63218  54445  39891  67938
 121872]

reg.intercept_

array([24648.40962547])

reg.coef_

array([[9718.66152021]])

y_predict= reg.predict(x_train)
plt.scatter(x_train,y_train,color='b')
```

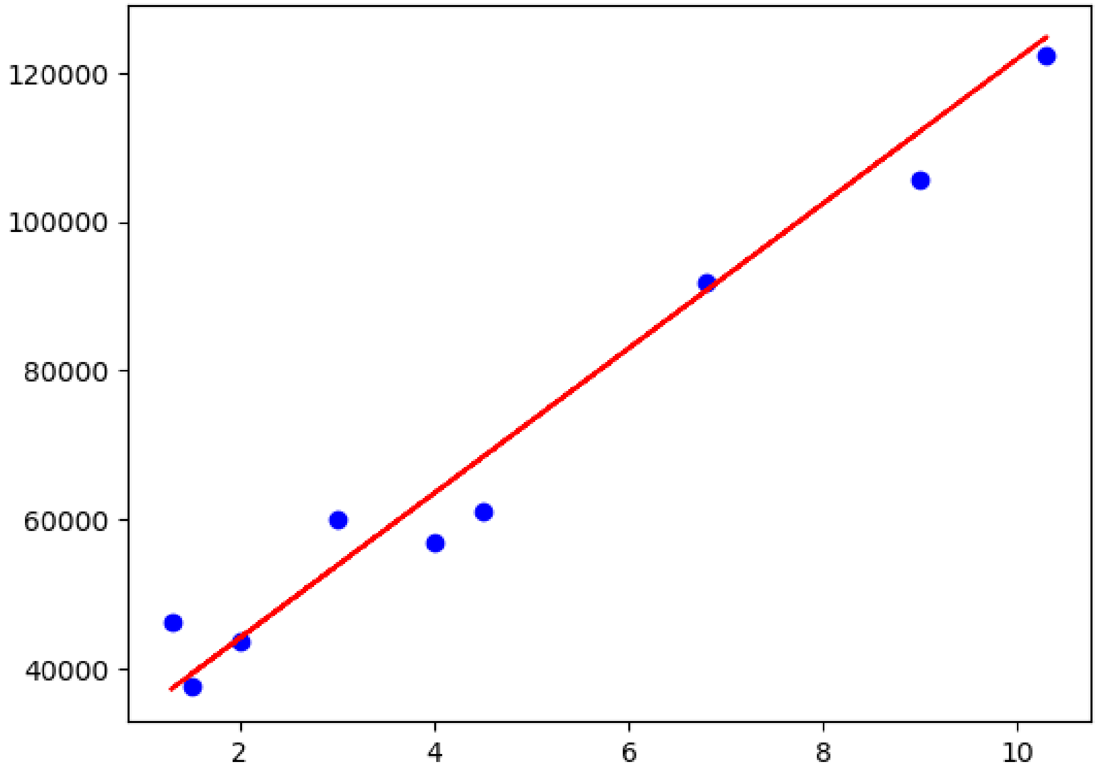
```
plt.plot(x_train,y_predict,color='r')
plt.show()
```




```
print(reg.score(x_test,y_test))
```

```
0.9624865713263218
```

```
y_predict= reg.predict(x_test)
plt.scatter(x_test,y_test,color='b')
plt.plot(x_test,y_predict,color='r')
plt.show()
```




```
R2=metrics.r2_score(y_test,y_predict)
R2
```

```
0.9624865713263218
```

```
print(metrics.mean_absolute_error(y_test,y_predict))
```

```
4562.02128719395
```

```
print(metrics.mean_squared_error(y_test,y_predict))
```

```
29742511.860138074
```

```
print(np.sqrt(metrics.mean_squared_error(y_test,y_predict)))
```

```
5453.669577462323
```

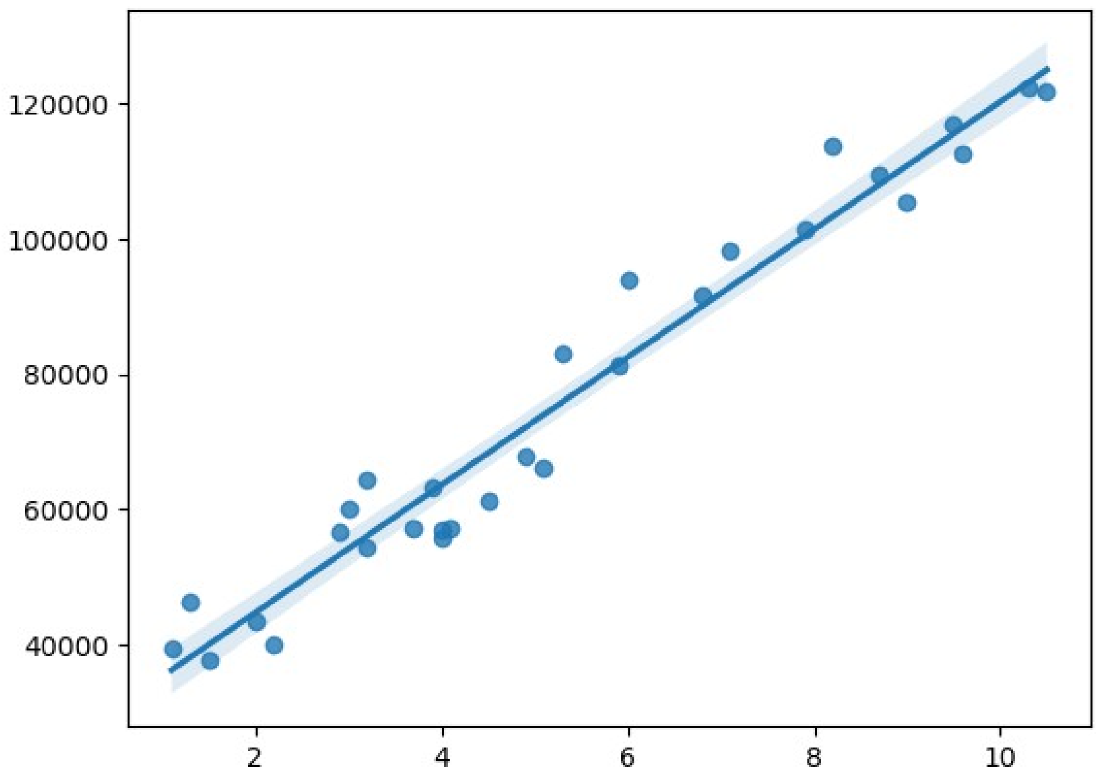
```
print(reg.predict([[5]]))
```

```
[[73241.71722654]]
```

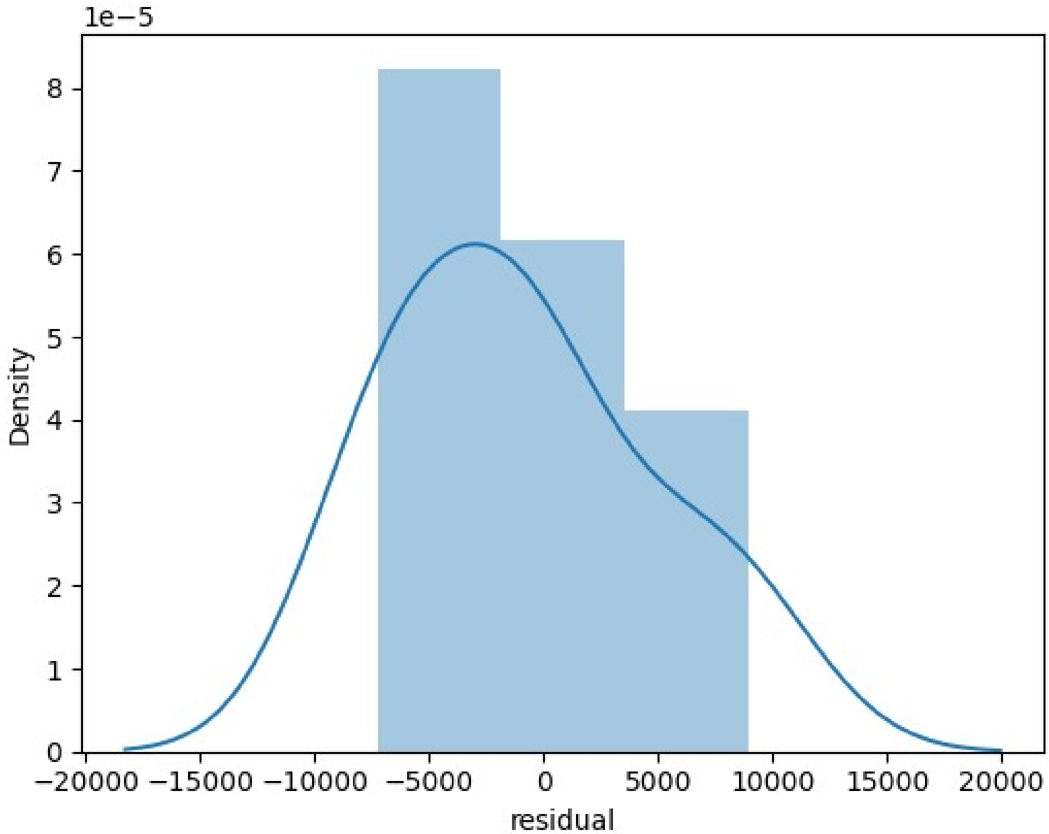
Linear Regression Assumptions

```
error= y_test-y_predict
error
```

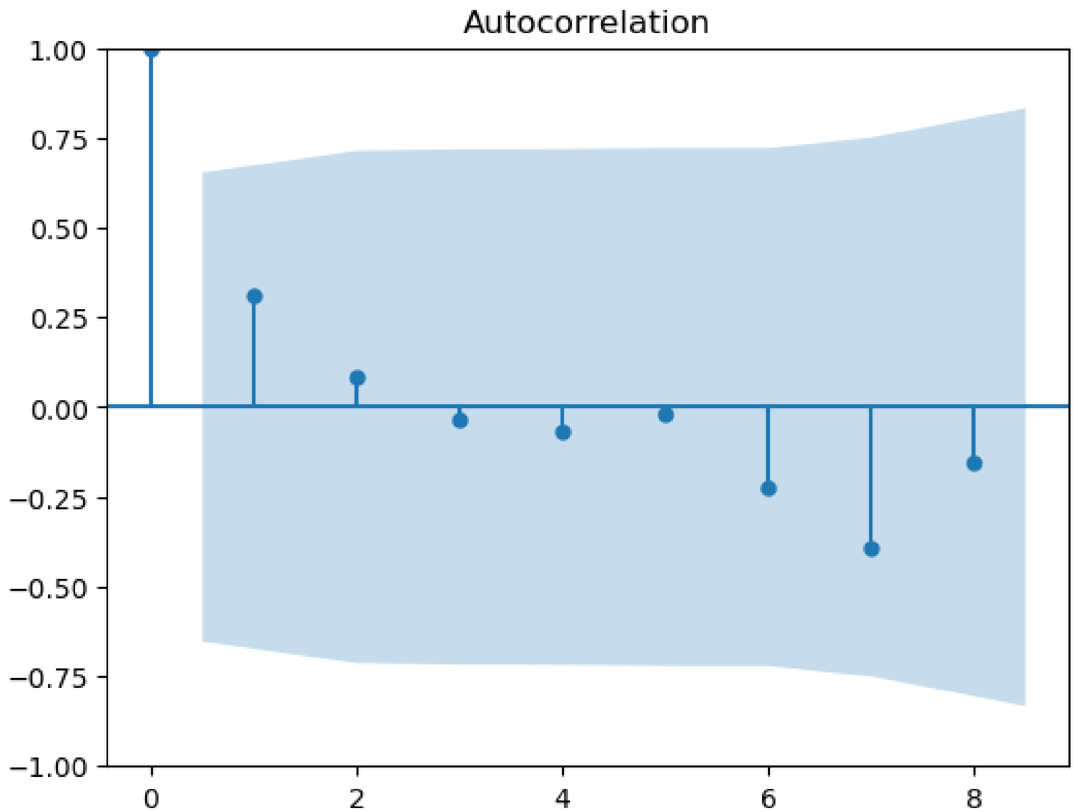
```
array([[-6534.3633074 ],
       [-7271.38646644],
       [-1495.4019058 ],
       [ -560.7326659 ],
       [-6566.05570633],
       [ 1002.69203707],
       [-2359.62328368],
       [ 8922.33039825],
       [ 6345.60581388]])

#Linearity
sns.regplot(x=x,y=y)
plt.show()
```



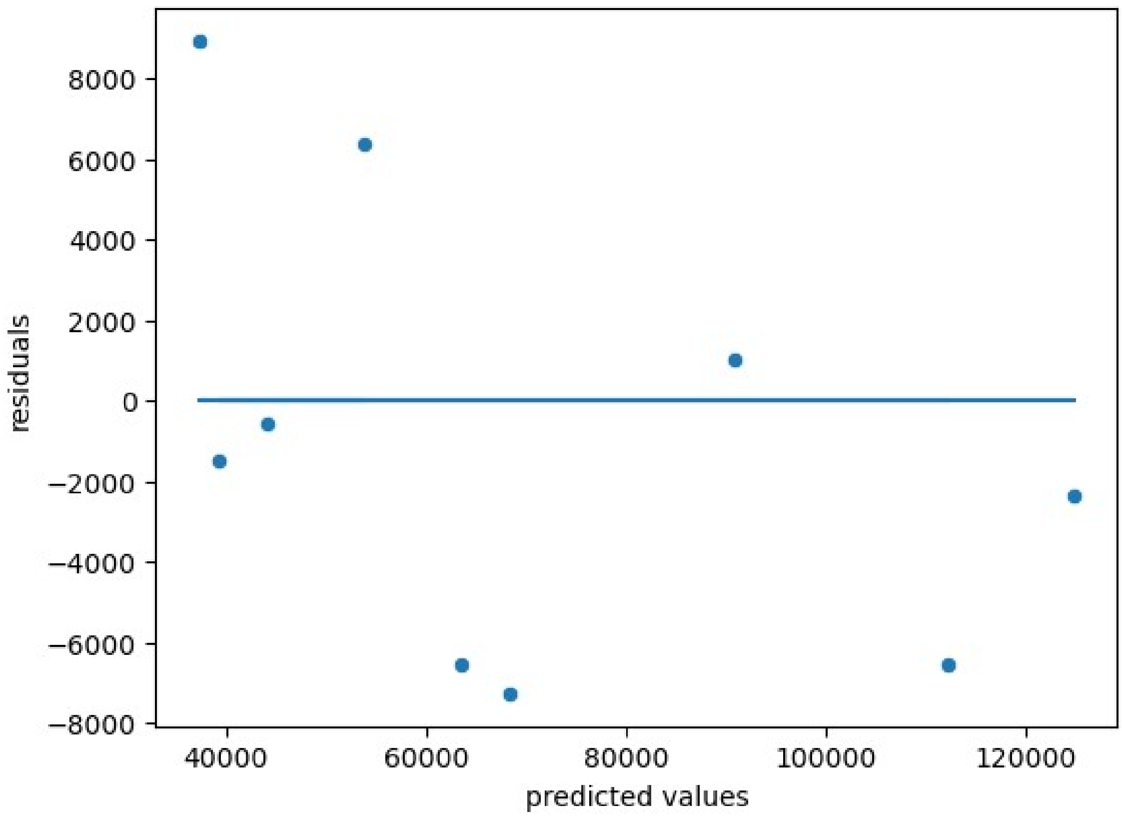

```
#Normality
sns.distplot(error)
plt.xlabel('residual')
plt.show()
```



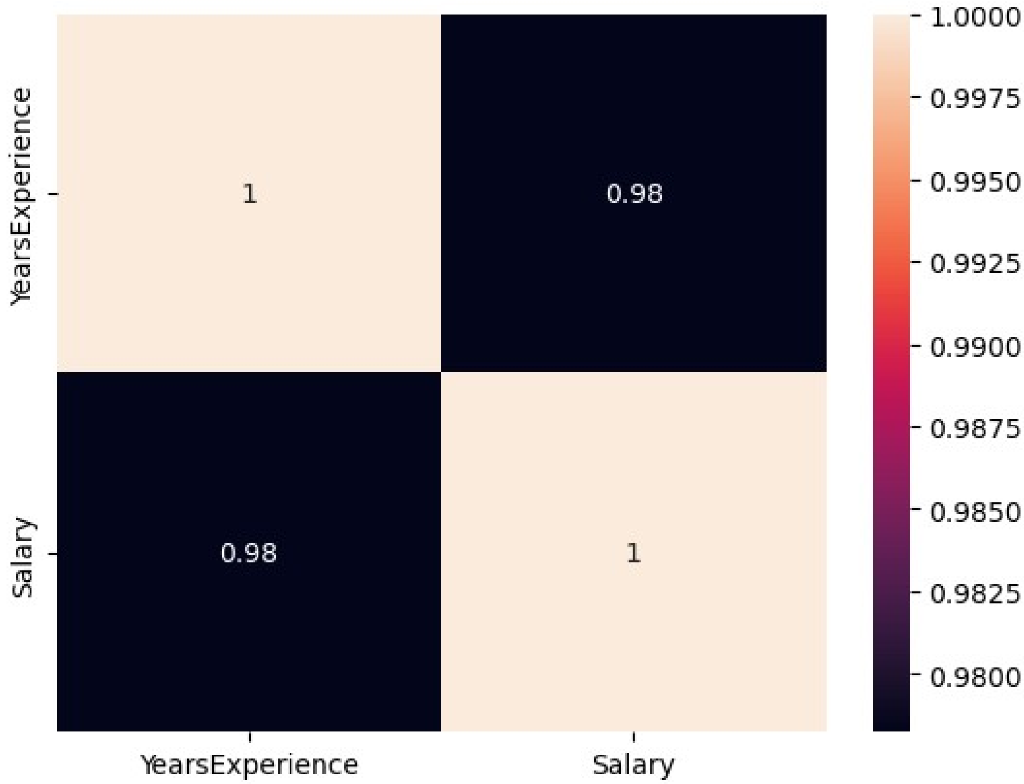

```
#No Autocorrelation
acf=plot_acf(error)
plt.show()
```




```
#Homoscedasticity
print(type(error))
sns.scatterplot(x=y_predict.flatten(),y=error.flatten())
plt.xlabel('predicted values')
plt.ylabel('residuals')
plt.plot(y_predict, [0]*len(y_predict))
plt.show()

<class 'numpy.ndarray'>
```




```
#Multicollinearity
sns.heatmap(df.corr(),annot=True)
plt.show()
```


YearsExperience
Salary
1
0.98
0.98
1
1.0000
0.9975
0.9950
0.9925
0.9900
0.9875
0.9850
0.9825
0.9800