Importing Required Libraries

In [15]:
```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns
import statsmodels.formula.api as smf
```

Importing Data

In [2]:
```
employee = pd.read_csv("emp_data.csv")
employee
```

Out[2]:

	Salary_hike	Churn_out_rate
0	1580	92
1	1600	85
2	1610	80
3	1640	75
4	1660	72
5	1690	70
6	1706	68
7	1730	65
8	1800	62
9	1870	60

Data Understanding

In [3]:
```
employee.head()
```

Out[3]:

	Salary_hike	Churn_out_rate
0	1580	92
1	1600	85
2	1610	80
3	1640	75
4	1660	72

In [4]:
```
employee.shape
```

Out[4]:
```
(10, 2)
```

In [5]:
```
employee.isna().sum()
```

Out[5]:
```
Salary_hike       0
Churn_out_rate    0
dtype: int64
```

In [6]:
```
employee.info
```

Out[6]:
```
<bound method DataFrame.info of    Salary_hike  Churn_out_rate
0         1580              92
1         1600              85
2         1610              80
3         1640              75
4         1660              72
5         1690              70
6         1706              68
7         1730              65
8         1800              62
9         1870              60>
```

In [7]:
```
employee.dtypes
```

Out[7]:
```
Salary_hike       int64
Churn_out_rate    int64
dtype: object
```

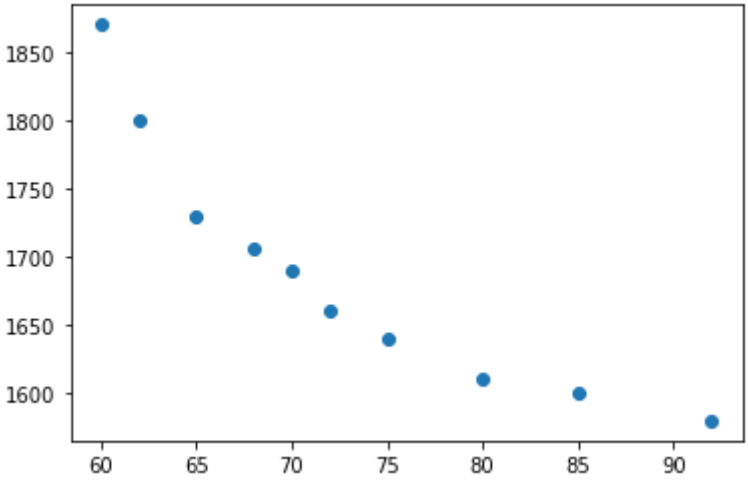
In [8]:
```
employee.describe()
```

Out[8]:

	Salary_hike	Churn_out_rate
count	10.000000	10.000000
mean	1688.600000	72.900000
std	92.096809	10.257247
min	1580.000000	60.000000
25%	1617.500000	65.750000
50%	1675.000000	71.000000
75%	1724.000000	78.750000
max	1870.000000	92.000000

Checking The Assumptions Are Matching or Not

In [10]:
```
plt.scatter(x = 'Churn_out_rate' , y = 'Salary_hike' , data = employee)
plt.show()
```

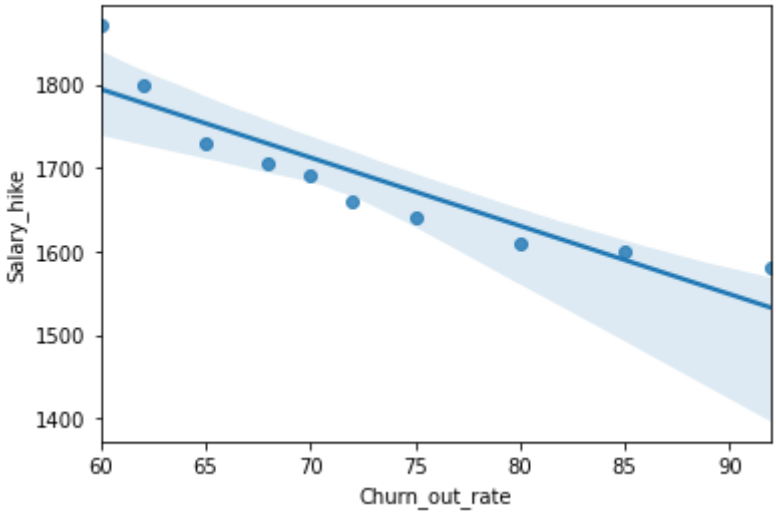


In [11]:
```
employee.corr()
```

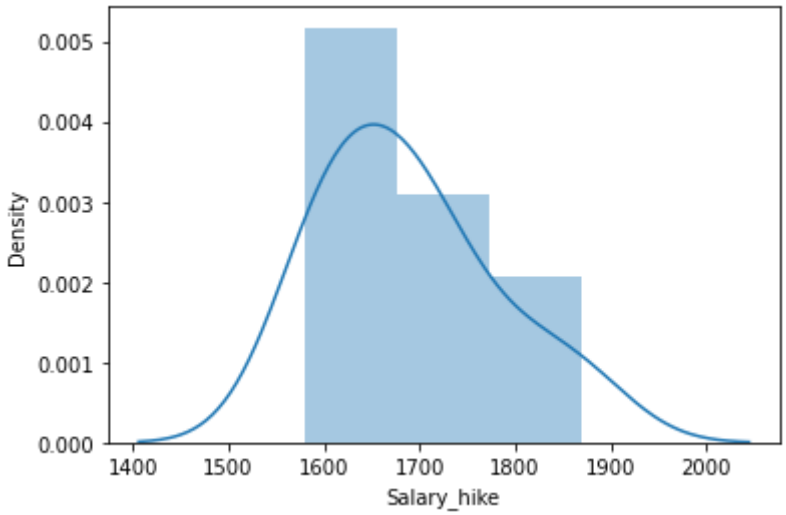
Out[11]:

	Salary_hike	Churn_out_rate
Salary_hike	1.000000	-0.911722
Churn_out_rate	-0.911722	1.000000

In [14]:
```
sns.regplot(x = 'Churn_out_rate', y = 'Salary_hike', data = employee )
plt.show()
```

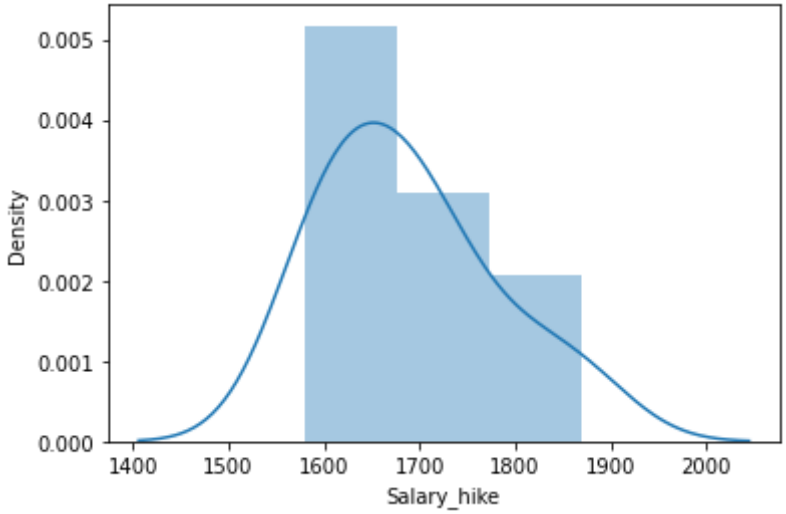


In [40]:
```
sns.distplot(employee ['Salary_hike'])
plt.show()
import warnings#this library imported becaouse removal of warnings
warnings.filterwarnings('ignore')
```



In [39]:
```
sns.distplot(employee['Salary_hike'])
```

Out[39]: <AxesSubplot:xlabel='Salary_hike', ylabel='Density'>



Model Building

In [17]:
```
linear_model = smf.ols(formula = 'Salary_hike~Churn_out_rate',data = employee).fit()
linear_model
```

Out[17]: <statsmodels.regression.linear_model.RegressionResultsWrapper at 0x185ea6a4b20>

Model Testing

In [18]:
```
linear_model.params
```

Out[18]:
```
Intercept         2285.365297
Churn_out_rate      -8.186081
dtype: float64
```

In [19]:
```
linear_model.tvalues , linear_model.pvalues
```

Out[19]:
```
(Intercept         23.827849
 Churn_out_rate    -6.277226
```

```
dtype: float64,
Intercept         1.024852e-08
Churn_out_rate    2.385777e-04
dtype: float64)
```

In [20]:
```
linear_model.rsquared , linear_model.rsquared_adj
```

Out[20]:
```
(0.8312363099883748, 0.8101408487369217)
```

Model Prediction

Sample calculation

In [23]:
```
## y = mx+c
emp_data = ( 2285.365297 +-8.186081 )*(5)#manual prediction say sorting time is 5
emp_data
```

Out[23]:
```
11385.89608
```

In [26]:
```
#machine prediction
pred_data = {'Churn_out_rate':[30,40,50]}
pred_data
```

Out[26]:
```
{'Churn_out_rate': [30, 40, 50]}
```

In [29]:
```
new_data = pd.DataFrame(data = pred_data)
new_data
```

Out[29]:

	Churn_out_rate
0	30
1	40
2	50

In [30]:
```
linear_model.predict(new_data)
```

Out[30]:
```
0    2039.782870
1    1957.922061
2    1876.061253
dtype: float64
```

In [ ]:
```

```