1) Delivery_time -> Predict delivery time using sorting time
2) Salary_hike -> Build a prediction model for Salary_hike

Build a simple linear regression model by performing EDA and do necessary transformations and select the best model using R or Python.

1) Delivery_time -> Predict delivery time using sorting time

—----------Import Important Libraries—----------
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.formula.api as smf
```

—----------Read the Dataset—----------
```
data = pd.read_csv('Downloads/delivery_time.csv')
data
```

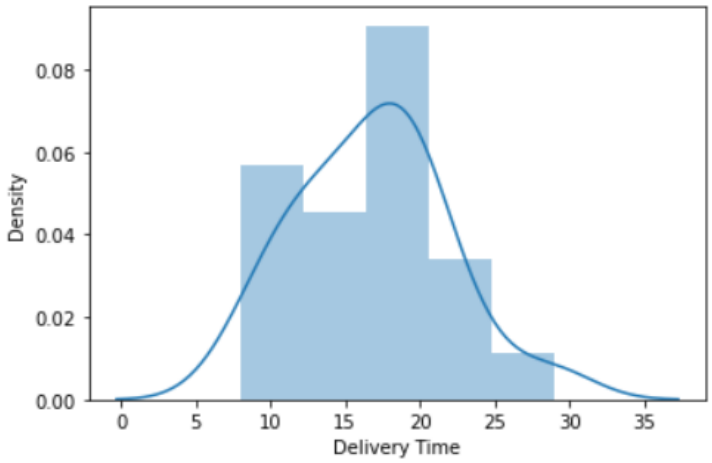
| | Delivery Time | Sorting Time |
|---|---|---|
| 0 | 21.00 | 10 |
| 1 | 13.50 | 4 |
| 2 | 19.75 | 6 |
| 3 | 24.00 | 9 |
| 4 | 29.00 | 10 |
| 5 | 15.35 | 6 |
| 6 | 19.00 | 7 |
| 7 | 9.50 | 3 |
| 8 | 17.90 | 10 |
| 9 | 18.75 | 9 |
| 10 | 19.83 | 8 |
| 11 | 10.75 | 4 |
| 12 | 16.68 | 7 |
| 13 | 11.50 | 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 |

—----------Get Information About Dataset—----------
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 21 entries, 0 to 20
Data columns (total 2 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Delivery Time  21 non-null     float64
 1   Sorting Time   21 non-null     int64
dtypes: float64(1), int64(1)
memory usage: 464.0 bytes
```

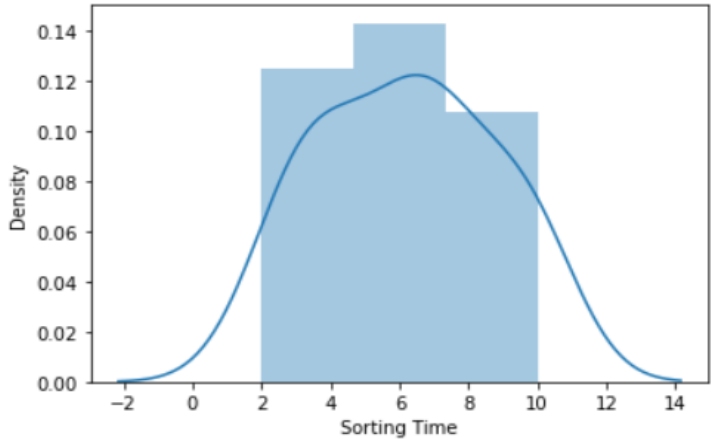
——————————Density Plot of Delivery Time Column——————————

```
sns.distplot(data['Delivery Time'])
```




——————————Density Plot of Sorting Time Column——————————

```
sns.distplot(data['Sorting Time'])
```




——————————Rename both the column names——————————

```
data = data.rename ({'Delivery Time':'DT','Sorting Time':'ST'},axis=1)
data
```

| | DT | ST |
|---|---|---|
| 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 |

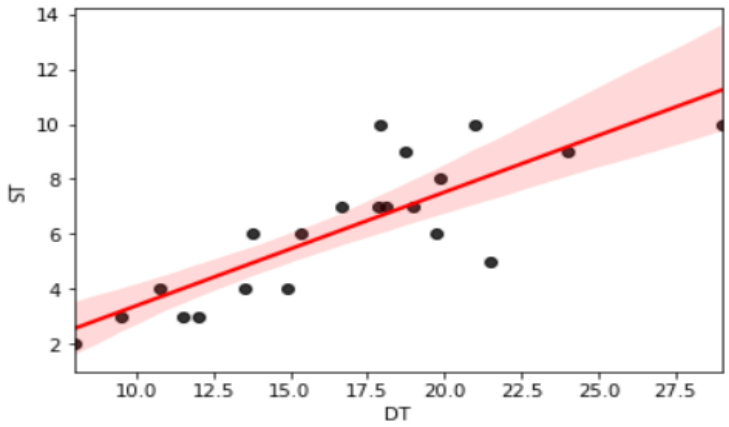
——————Correlation Analysis——————————————

data.corr()

| | DT | ST |
|---|---|---|
| **DT** | 1.000000 | 0.825997 |
| **ST** | 0.825997 | 1.000000 |

————————————Plotting of Regression Plot————————————

sns.regplot(data['DT'],data['ST'],line_kws={'color':'red'},scatter_kws={'color':'k'})




————————————Model Building——————————————

import statsmodels.formula.api as smf

model = smf.ols ('DT~ST',data=data).fit()

————————————Model Testing——————————————

Model.params

```
Intercept    6.582734
ST           1.649020
dtype: float64
```

—----------Finding Tvalues and Pvalues—----------

```
print (model.tvalues, '\n', model.pvalues)
```

```
Intercept    3.823349
ST           6.387447
dtype: float64
 Intercept    0.001147
ST           0.000004
dtype: float64
```

—----------Finding Rsquared values—----------

```
print (model.rsquared, model.rsquared_adj)
```

```
0.6822714748417231 0.6655489208860244
```

—----------Model Predictions—----------

```
delivery_time = (6.582734)+(1.649020)*(5)
```

—---sorting time as 5—----------

```
delivery_time
```

```
14.827834
```

```
newdata = pd.Series([5,8])
```

—----------sorting time as 5 & 8—------

```
newdata
```

```
0    5
1    8
dtype: int64
```

```
predict = pd.DataFrame (newdata, columns = ['ST'])
predict
```

| | ST |
|---|---|
| **0** | 5 |
| **1** | 8 |

```
model.predict(predict)
```

```
0    14.827833
1    19.774893
dtype: float64
```

2) Salary_hike -> Build a prediction model for Salary_hike

—----------Import Important Libraries—----------

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.formula.api as smf
```

—----------Read the Dataset—----------

```
data = pd.read_csv('Downloads/Salary_Data.csv')
data
```

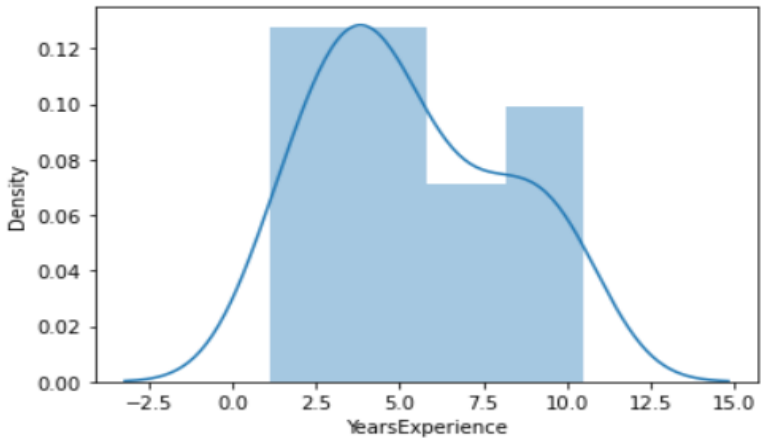
| | YearsExperience | Salary |
|---|---|---|
| 0 | 1.1 | 39343.0 |
| 1 | 1.3 | 46205.0 |
| 2 | 1.5 | 37731.0 |
| 3 | 2.0 | 43525.0 |
| 4 | 2.2 | 39891.0 |
| 5 | 2.9 | 56642.0 |
| 6 | 3.0 | 60150.0 |
| 7 | 3.2 | 54445.0 |
| 8 | 3.2 | 64445.0 |
| 9 | 3.7 | 57189.0 |
| 10 | 3.9 | 63218.0 |
| 11 | 4.0 | 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 |

—----------Necessary information about dataset—----------

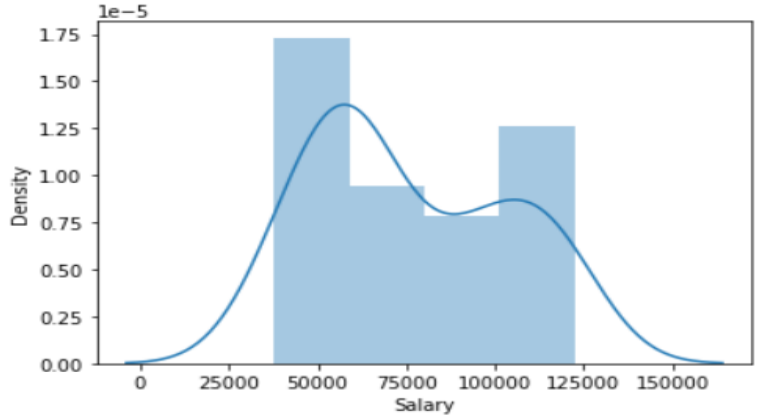
```
data.info()
```

```
<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: 608.0 bytes
```

——————Density plot for YearsExperience column——————————
sns.distplot(data['YearsExperience'])



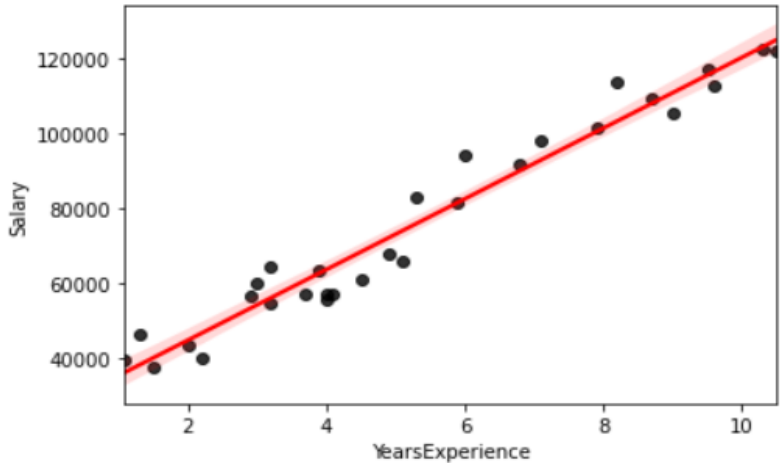

——————Density plot for Salary column——————————
sns.distplot(data['Salary'])




——————Correlation Analysis——————————
data.corr()

| | YearsExperience | Salary |
|---|---|---|
| **YearsExperience** | 1.000000 | 0.978242 |
| **Salary** | 0.978242 | 1.000000 |

——————Regression Plot——————————
sns.regplot(data['YearsExperience'],data['Salary'],line_kws={'color':'red'},scatter_kws={'color':'k'})




——————Model Building——————————————

```
import statsmodels.formula.api as smf
model = smf.ols ('Salary~YearsExperience', data = data).fit()
```

————————————Find Coefficient Parameters——————————

```
Model.params
Intercept          25792.200199
YearsExperience     9449.962321
dtype: float64
```

——————————Find Tvalues and Pvalues———————————

```
print (model.tvalues, '\n', model.pvalues)
Intercept          11.346940
YearsExperience    24.950094
dtype: float64
 Intercept          5.511950e-12
YearsExperience    1.143068e-20
dtype: float64
```

——————————Find Rsquared values——————————————

```
print (model.rsquared, model.rsquared_adj)
0.9569566641435086 0.9554194021486339
```

——————————Model Predictions———————————————

```
salary = (25792.200199)+(9449.962321)*3        ——-say 3 years of experience———————
salary
54142.087162
```

```
newdata = pd.Series([3,5])                  ————-say 3 and 5 years of experience———————
newdata
```

```
0    3
1    5
dtype: int64
```

predict = pd.DataFrame (newdata,columns=['YearsExperience'])
predict

| | YearsExperience |
|---|---|
| **0** | 3 |
| **1** | 5 |

model.predict(predict)

```
0    54142.087163
1    73042.011806
dtype: float64
```