1) ***Calories consumed: predict weight gained using calories consumed***

Here
Independent variable is calories consumed (X)
Dependent variable is weight gain (Y)

Solution:

For S. L. R : $\mathbf{Y = \beta_0 + \beta_1 X}$

Codes to build a Linear regression model:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

cc=pd.read_csv("E:\Data\Assignments\i made\SLR\calories_consumed.csv")
```

To find the correlation between the variables:

```
cc.Calories_Consumed.corr(cc.Weight_gained_grams)
```

Correlation: 0.9469910088554458
The 0.94 indiactes that the variables inside the data are strongly correlated.

Building a Linear regression model: (using ordinary least square technique)

```
import statsmodels.formula.api as smf
model = smf.ols("Calories_Consumed~Weight_gained_grams",data=cc).fit()
model
```

$\boldsymbol{\beta_0}$ = 1577.200702
$\boldsymbol{\beta_1}$ = 2.134423

To get the model summary:

```
model.summary()
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:       Calories_Consumed   R-squared:                       0.897
Model:                             OLS   Adj. R-squared:                  0.888
Method:                  Least Squares   F-statistic:                     104.3
Date:                 Sat, 28 Mar 2020   Prob (F-statistic):           2.86e-07
Time:                         15:11:25   Log-Likelihood:                -96.170
No. Observations:                   14   AIC:                             196.3
Df Residuals:                       12   BIC:                             197.6
Df Model:                            1
Covariance Type:             nonrobust
=======================================================================================
                          coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------------------
Intercept            1577.2007    100.541     15.687      0.000    1358.141    1796.260
Weight_gained_grams     2.1344      0.209     10.211      0.000       1.679       2.590
==============================================================================
Omnibus:                        0.254   Durbin-Watson:                   2.308
Prob(Omnibus):                  0.881   Jarque-Bera (JB):                0.425
Skew:                          -0.098   Prob(JB):                        0.808
Kurtosis:                       2.169   Cond. No.                         719.
==============================================================================
```

To Predict values of Calories_Consumed using the model

pred = model.predict(cc.iloc[:,0])
pred

0   1807.718381
1   2004.085294
2   3498.181364
3   2004.085294
4   2217.527589
5   1811.987227
6   1850.406841
7   1709.534925
8   2857.854477
9   3925.065955
10   1790.642998
11   1897.364146
12   2324.248737
13   3071.296772

2) *Delivery time : Predict delivery time using sorting time*

Independent variable is Sorting_Time (X)
Dependent variable is Delivery_Time (Y)

Solution:

For S. L. R : $\mathbf{Y = \beta_0 + \beta_1 X}$

Codes to build a Linear regression model:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

dt=pd.read_csv("E:\Data\Assignments\i made\SLR\Delivery_Time.csv")
```

To find the correlation between the variables:

```
dt. Sorting_Time.corr(dt. Delivery_Time)
```

Correlation: 0.82599726
The 0.826 indicates that the variables inside the data are moderately correlated. Since the r value here is lesser than 0.85 we need improve the correlation, so that we use R^2 (R squared value - coefficient of determination)
Building a Linear regression model: (using ordinary least square technique)

```
import statsmodels.formula.api as smf
model = smf.ols("Sorting_Time~ Delivery_Time",data=dt).fit()
model
```

$\boldsymbol{\beta_0}$ = -0.7567
$\boldsymbol{\beta_1}$ = 0.4137

To get the model summary:

model.summary()

```
==============================================================================
Dep. Variable:           Sorting_Time   R-squared:                       0.682
Model:                            OLS   Adj. R-squared:                  0.666
Method:                 Least Squares   F-statistic:                     40.80
Date:                Sun, 29 Mar 2020   Prob (F-statistic):           3.98e-06
Time:                        18:30:20   Log-Likelihood:                -36.839
No. Observations:                  21   AIC:                             77.68
Df Residuals:                      19   BIC:                             79.77
Df Model:                           1
Covariance Type:            nonrobust
=================================================================================
                    coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------------
Intercept        -0.7567      1.134     -0.667      0.513      -3.130       1.617
Delivery_Time     0.4137      0.065      6.387      0.000       0.278       0.549
==============================================================================
Omnibus:                        1.409   Durbin-Watson:                   1.346
Prob(Omnibus):                  0.494   Jarque-Bera (JB):                0.371
Skew:                           0.255   Prob(JB):                        0.831
Kurtosis:                       3.405   Cond. No.                         62.1
==============================================================================
```

To Predict values of Delivery _Time using the model

```
pred = model.predict(dt.iloc[:,0])
pred
```

```
0     7.931943
1     4.828866
2     7.414763
3     9.173174
4    11.241892
5     5.594291
6     7.104456
7     3.173891
8     6.649338
9     7.001020
10    7.447863
11    3.691071
12    6.144570
13    4.001378
14    4.220662
15    5.399832
16    4.932302
17    6.736224
18    2.553276
19    6.620376
20    8.138815
```

3) Emp_data: Build a prediction model for Churn_out_rate

Independent variable is Salary_hike (X)
Dependent variable is Churn_out_rate (Y)

Solution:

For S. L. R : $\mathbf{Y = \beta_0 + \beta_1 X}$

Codes to build a Linear regression model:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

ed=pd.read_csv("E:\Data\Assignments\i made\SLR\emp_data.csv")
```

To find the correlation between the variables:

```
ed.Salary_hike.corr(ed.Churn_out_rate)
```

Correlation: -0.91172161

To find the value of β_0 and β_1 :

```
import statsmodels.formula.api as smf
model = smf.ols("Churn_out_rate~Salary_hike",data=ed).fit()
model
```

$\boldsymbol{\beta_0}$ = 244.364911
$\boldsymbol{\beta_1}$ = -0.101543

To get the model summary:

model.summary()

```
                            OLS Regression Results
==============================================================================
Dep. Variable:         Churn_out_rate   R-squared:                       0.831
Model:                            OLS   Adj. R-squared:                  0.810
Method:                 Least Squares   F-statistic:                     39.40
Date:                Sun, 29 Mar 2020   Prob (F-statistic):           0.000239
Time:                        18:39:38   Log-Likelihood:                -28.046
No. Observations:                  10   AIC:                             60.09
Df Residuals:                       8   BIC:                             60.70
Df Model:                           1
Covariance Type:            nonrobust
===============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
-------------------------------------------------------------------------------
Intercept    244.3649     27.352      8.934      0.000     181.291     307.439
Salary_hike   -0.1015      0.016     -6.277      0.000      -0.139      -0.064
==============================================================================
Omnibus:                        2.201   Durbin-Watson:                   0.562
Prob(Omnibus):                  0.333   Jarque-Bera (JB):                1.408
Skew:                           0.851   Prob(JB):                        0.495
Kurtosis:                       2.304   Cond. No.                     3.27e+04
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 3.27e+04. This might indicate that there are
strong multicollinearity or other numerical problems.
```

To Predict values of Delivery _Time using the model

```
pred = model.predict(ed.iloc[:,0])
pred
```

```
0   83.927531
1   81.896678
2   80.881252
3   77.834973
4   75.804120
5   72.757840
6   71.133158
7   68.696134
8   61.588149
9   54.480164
```

4) Salary_hike: Build a prediction model for Salary_hike

5)

Independent variable is YearsExperiance (X)
Dependent variable is Salary (Y)

Solution:

For S. L. R : $\mathbf{Y = \beta_0 + \beta_1 X}$

Codes to build a Linear regression model:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

sd=pd.read_csv("E:\Data\Assignments\i made\SLR\Salary_Data.csv")
```

To find the correlation between the variables:

```
sd. YearsExperiance corr(ed.Salary)
```

Correlation: 0.978241

The 0.978 indicates that the variables inside the data are strongly correlated.
Since the r value here is lesser than 0.85 we need improve the correlation, so that we use R^2 (R squared value - coefficient of determination)
Building a Linear regression model: (using ordinary least square technique)

To find the value of β_0 and β_1 :

```
import statsmodels.formula.api as smf
model = smf.ols("YearsExperiance ~Salary ",data=sd).fit()
model
```

$\boldsymbol{\beta_0}$ = 25792.200199
$\boldsymbol{\beta_1}$ = 9449.962321

To get the model summary:

model.summary()

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                 Salary   R-squared:                       0.957
Model:                            OLS   Adj. R-squared:                  0.955
Method:                 Least Squares   F-statistic:                     622.5
Date:                Sun, 29 Mar 2020   Prob (F-statistic):           1.14e-20
Time:                        20:23:48   Log-Likelihood:                -301.44
No. Observations:                  30   AIC:                             606.9
Df Residuals:                      28   BIC:                             609.7
Df Model:                           1
Covariance Type:            nonrobust
===================================================================================
                      coef    std err          t      P>|t|      [0.025      0.975]
-----------------------------------------------------------------------------------
Intercept        2.579e+04   2273.053     11.347      0.000    2.11e+04    3.04e+04
YearsExperience  9449.9623    378.755     24.950      0.000    8674.119    1.02e+04
==============================================================================
Omnibus:                        2.140   Durbin-Watson:                   1.648
Prob(Omnibus):                  0.343   Jarque-Bera (JB):                1.569
Skew:                           0.363   Prob(JB):                        0.456
Kurtosis:                       2.147   Cond. No.                         13.2
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
"""
```

To Predict values of Delivery _Time using the model

pred = model.predict(sd.iloc[:,0])
pred

0 36187.158752

1 38077.151217

2 39967.143681

3 44692.124842

4 46582.117306

5 53197.090931

6 54142.087163

7 56032.079627

8 56032.079627

9 60757.060788

10	62647.053252
11	63592.049484
12	63592.049484
13	64537.045717
14	68317.030645
15	72097.015574
16	73987.008038
17	75877.000502
18	81546.977895
19	82491.974127
20	90051.943985
21	92886.932681
22	100446.902538
23	103281.891235
24	108006.872395
25	110841.861092
26	115566.842252
27	116511.838485
28	123126.812110
29	125016.804574