Import libraries:

In [81]:
```
import csv
import pandas as pd
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import confusion_matrix
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import accuracy_score
```

QUE-1

Load data:

Univariate

In [82]:
```
univariate=pd.read_csv("/content/Akanksha Dewangan - chocolates-univariate.csv")
```

Unanmed columns having all NaN values hence not require so drop it.

In [83]:
```
univariate
```

Out[83]:

	UGive	Unnamed: 1	IGive
0	1	NaN	2
1	2	NaN	4
2	3	NaN	6
3	4	NaN	8
4	5	NaN	10
...	...	...	...
998	999	NaN	1819
999	1000	NaN	1820
1000	1001	NaN	1822
1001	1002	NaN	1824
1002	1003	NaN	1826

1003 rows × 3 columns

In [84]:
```
univariate.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1003 entries, 0 to 1002
Data columns (total 3 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   UGive       1003 non-null   int64  
 1   Unnamed: 1  0 non-null      float64
 2   IGive       1003 non-null   int64  
dtypes: float64(1), int64(2)
memory usage: 23.6 KB
```

In [85]:
```
univariate=univariate.drop(['Unnamed: 1'],axis=1)
```

In [86]:
```
univariate
```

Out[86]:

	UGive	IGive
0	1	2
1	2	4
2	3	6
3	4	8
4	5	10
...	...	...
998	999	1819
999	1000	1820
1000	1001	1822
1001	1002	1824
1002	1003	1826

1003 rows × 2 columns

split into test and train:

In [87]:
```
X_train, X_test, y_train, y_test = train_test_split(np.array(univariate['UGive']).reshape(-1, 1) , np.array(univariate['IGive']).reshape(-1, 1), test_size=0.2,shuffle= True ,random_state=4)
```

univariate Linear Regression:

In [88]:
```
model=LinearRegression().fit(X_train,y_train)
acc=model.score(X_test,y_test)
print("Accuracy : ",acc)
x=model.coef_
print("w0: ",model.intercept_,"w1: ",x[0])
```

```
Accuracy :  0.9999997042964314
w0:  [0.49147333] w1:  [1.82000039]
```

Multivariate

In [89]:
```
multivariate=pd.read_csv("/content/Akanksha Dewangan - chocolates-multivariate-csv.csv")
```

In [90]:
```
multivariate
```

Out[90]:

	UGive	Age	IGive
0	1	10	2
1	2	15	4
2	3	20	6
3	4	25	8
4	5	30	10
...	...	...	...
1095	1096	35	1996
1096	1097	40	1997
1097	1098	45	1999
1098	1099	50	2001
1099	1100	55	2003

1100 rows × 3 columns

In [91]:
```
X_train, X_test, y_train, y_test = train_test_split(multivariate.iloc[:,:2], multivariate.iloc[:,2], test_size=0.2,shuffle= True ,random_state=4)
```

In [92]:
```
model=LinearRegression().fit(X_train,y_train)
```

Multivariate linear regression:

In [93]:
```
acc=model.score(X_test,y_test)
print("Accuracy: ",acc)
x=model.coef_
print("w0: ",model.intercept_,"w1 is: ", x[0]," w2: ", x[1])
```

```
Accuracy:  0.9999997705958328
w0:  0.46026231168173126 w1 is:  1.8199822432695036  w2:  0.0109332259012924
```

QUE-2

Load dataset:

In [94]:
```
iris=pd.read_csv("Iris.csv.txt")
```

In [95]:
```
iris.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   sepal_length  150 non-null    float64
 1   sepal_width   150 non-null    float64
 2   petal_length  150 non-null    float64
 3   petal_width   150 non-null    float64
 4   species       150 non-null    object 
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

In [96]:
```
iris['species']=iris['species'].astype('category').cat.codes
```

In [97]:
```
iris.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   sepal_length  150 non-null    float64
 1   sepal_width   150 non-null    float64
 2   petal_length  150 non-null    float64
 3   petal_width   150 non-null    float64
 4   species       150 non-null    int8   
dtypes: float64(4), int8(1)
memory usage: 5.0 KB
```

In [98]:
```
X_train, X_test, y_train, y_test = train_test_split(iris.iloc[:,:4],iris.iloc
[:,4], test_size=0.1,shuffle= True ,random_state=4)
```

Iris dataset with MLP Classifier:

In [99]:
```
clf = MLPClassifier(solver='lbfgs',hidden_layer_sizes=(10, 7), random_state=3)
.fit(X_train, y_train)
pred=clf.predict(X_test)
acc=accuracy_score(y_test,pred)
print("Accuracy ",acc)
```

```
Accuracy  0.9333333333333333
```

Now its time to label encode of target variable: setosa to 1 and other 0

In [100]:
```
iris=pd.read_csv("Iris.csv.txt")
```

In [101]:
```
label=[1 if i=="setosa" else 0 for i in iris["species"]]
```

In [102]:
```
iris["species"]=label
```

In [103]:
```
X_train, X_test, y_train, y_test = train_test_split(iris.iloc[:,:4], iris.iloc
[:,4], test_size=0.1,shuffle= True ,random_state=4)
```

Applying MLP classifier after encoding setosa to 0 and rest to 1 in species column:

In [104]:
```
clf = MLPClassifier(solver='lbfgs',hidden_layer_sizes=(10,3), random_state=3).
fit(X_train, y_train)
pred=clf.predict(X_test)
acc=accuracy_score(y_test,pred)
print("Accuracy: ",acc)
```

```
Accuracy:  0.6
```

In [104]:
```

```