```
# sklearn basics
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
import pandas as pd

# read data
mobile = pd.read_csv("/content/Cellphone.csv")
mobile.head()
```

	Product_id	Price	Sale	weight	resoloution	ppi	cpu core	cpu freq	internal mem	ram	RearCam	Front_Cam	ba
0	203	2357	10	135.0	5.2	424	8	1.35	16.0	3.000	13.00	8.0	
1	880	1749	10	125.0	4.0	233	2	1.30	4.0	1.000	3.15	0.0	
2	40	1916	10	110.0	4.7	312	4	1.20	8.0	1.500	13.00	5.0	
3	99	1315	11	118.5	4.0	233	2	1.30	4.0	0.512	3.15	0.0	
4	880	1749	11	125.0	4.0	233	2	1.30	4.0	1.000	3.15	0.0	

```
mobile.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 161 entries, 0 to 160
Data columns (total 14 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Product_id    161 non-null    int64  
 1   Price         161 non-null    int64  
 2   Sale          161 non-null    int64  
 3   weight        161 non-null    float64
 4   resoloution   161 non-null    float64
 5   ppi           161 non-null    int64  
 6   cpu core      161 non-null    int64  
 7   cpu freq      161 non-null    float64
 8   internal mem  161 non-null    float64
 9   ram           161 non-null    float64
 10  RearCam       161 non-null    float64
 11  Front_Cam     161 non-null    float64
 12  battery       161 non-null    int64  
 13  thickness     161 non-null    float64
dtypes: float64(8), int64(6)
memory usage: 17.7 KB
```

```
mobiles = mobile.dropna()
mobiles.isna().sum()
```

```
Product_id      0
Price           0
Sale            0
weight          0
resoloution     0
ppi             0
cpu core        0
cpu freq        0
internal mem    0
ram             0
RearCam         0
Front_Cam       0
battery         0
thickness       0
dtype: int64
```

Split Data

```
# prepare data ,
X = mobiles.drop(["Product_id","Price","Sale"], axis=1)
y = mobiles["Price"]

# split data
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size = 0.25, random_state = 42 #set.seed()
)
```

```
print("X_train.shape = ", {X_train.shape})
print("X_test.shape = ", {X_test.shape})
print("y_train.shape = ", {y_train.shape})
print("y_train.shape = ", {y_test.shape})  # ( row, column )
```

```
X_train.shape =  {(120, 11)}
X_test.shape =  {(41, 11)}
y_train.shape =  {(120,)}
y_train.shape =  {(41,)}
```

Linear Regression

```
# train model
model = LinearRegression()
model.fit(X_train, y_train)

# test model
p = model.predict(X_test)
print(p)
```

```
[ 896.7931879  1867.68797876 2630.57706878 1522.53876659 1615.81719647
 1874.93053669  896.7931879  1332.36219066 2362.11236461 2630.57706878
 1379.13090935 1576.69957704 1746.03612054 2860.68585851 2218.9966377
 2196.25352333 2474.24526178 3033.54434954 1458.16863568 2196.25352333
 2324.81703092 1746.03612054 1912.81417903 2433.30241493 3092.52684098
 4166.07747612 2417.10671487 3378.66524903 2668.82318113 2507.39348484
 3050.17525413 2570.27591585 2551.71520088 2776.34201504 2021.67599338
 1884.33555415 2055.19153313 1573.70989261 2455.08700187 1576.69957704
 1871.61702742]
```

```
# Model Evaluation
model.score(X_test, y_test) # R2
```

```
0.9543086771912769
```

Desicion Tree Regression

```
from sklearn.tree import DecisionTreeRegressor
# train model
tree_model = DecisionTreeRegressor()
tree_model.fit(X_train, y_train)

# test model
p = tree_model.predict(X_test)
print(p)
```

```
[ 791. 1676. 2536. 1777. 1511. 2266.  791. 1347. 2343. 2536. 1238.  754.
 1734. 2746. 1942. 2137. 2491. 3055. 1522. 2137. 2276. 1734. 1831. 2580.
 2977. 4361. 2562. 3658. 2466. 2744. 3116. 2714. 2824. 2858. 2266. 2001.
 1989. 1843. 2323.  754. 2006.]
```

```
# Model Evaluation
tree_model.score(X_test, y_test)
```

```
0.8995021663099808
```