```
import pandas as pd
import numpy as np

dataset = pd.read_csv("Material Compressive Strength Experimental
Data.csv")

dataset.head(5)

   Material Quantity (gm)  Additive Catalyst (gm)  Ash Component (gm)
\
0                  486.42                  180.60               21.26

1                  133.32                  260.14              185.60

2                  559.97                    2.84              111.76

3                  391.43                  351.05               76.39

4                  394.78                  352.61              194.35


   Water Mix (ml)  Plasticizer (gm)  Moderate Aggregator  Refined
Aggregator  \
0          201.66             16.11              1151.17
708.50
1          175.99              6.27              1090.57
1010.25
2          295.23             11.95              1024.93
810.69
3          299.14             19.00              1134.88
881.34
4          235.54             17.02              1098.24
781.01

   Formulation Duration (hrs)  Compression Strength MPa
0                      344.43                     79.89
1                       28.86                     59.80
2                      237.68                     77.86
3                      208.81                     71.74
4                      266.84                     76.07

dataset.isnull().sum()

Material Quantity (gm)        109
Additive Catalyst (gm)        109
Ash Component (gm)            109
Water Mix (ml)                109
Plasticizer (gm)              109
Moderate Aggregator           109
Refined Aggregator            109
Formulation Duration (hrs)    109
```

```
Compression Strength MPa        0
dtype: int64
```

NOTE: The below code eliminates all the null values in integer, strings format and by default fills the missing values with mean.

code reference - stack overflow

```
def clean_dataset(dataset):
    assert isinstance(dataset, pd.DataFrame)
    dataset.dropna(inplace=True)
    indices_to_keep = ~dataset.isin([np.nan, np.inf, -
np.inf]).any(axis=1)
    return dataset[indices_to_keep].astype(np.float64)

clean_dataset(dataset)

      Material Quantity (gm)  Additive Catalyst (gm)  Ash Component
(gm)  \
0                     486.42                  180.60
21.26
1                     133.32                  260.14
185.60
2                     559.97                    2.84
111.76
3                     391.43                  351.05
76.39
4                     394.78                  352.61
194.35
...                      ...                     ...                .
..
6134                  188.78                  162.30
142.65
6135                  349.87                  291.45
77.82
6136                  358.29                   22.70
17.99
6137                  445.25                  275.59
178.86
6138                  560.23                  266.56
167.14

      Water Mix (ml)  Plasticizer (gm)  Moderate Aggregator  \
0             201.66             16.11              1151.17
1             175.99              6.27              1090.57
2             295.23             11.95              1024.93
3             299.14             19.00              1134.88
4             235.54             17.02              1098.24
...              ...               ...                  ...
6134          163.66             15.98              1003.82
```

```
6135          188.26             25.82              925.10
6136          208.58             34.91             1081.07
6137          191.77             18.07              865.15
6138          175.49             10.63             1165.87

      Refined Aggregator  Formulation Duration (hrs)  Compression
Strength MPa
0                 708.50                      344.43
79.89
1                1010.25                       28.86
59.80
2                 810.69                      237.68
77.86
3                 881.34                      208.81
71.74
4                 781.01                      266.84
76.07
...                  ...                         ...
...
6134             1002.47                      357.91
50.61
6135             1005.31                      104.20
54.24
6136              792.44                      302.76
56.57
6137              833.10                      374.63
58.21
6138              894.53                      360.96
58.96

[6030 rows x 9 columns]
```

```
dataset.isnull().sum()
```

```
Material Quantity (gm)        0
Additive Catalyst (gm)        0
Ash Component (gm)            0
Water Mix (ml)                0
Plasticizer (gm)              0
Moderate Aggregator           0
Refined Aggregator            0
Formulation Duration (hrs)    0
Compression Strength MPa      0
dtype: int64
```

OBSERVATION:

A clean dataset is achieved.

```
#scaling to increase the efficieny of the model
from sklearn.preprocessing import RobustScaler

transformer = RobustScaler().fit_transform(dataset)
transformer

array([[ 0.42668571, -0.05108089, -0.72481203, ..., -0.54009872,
         0.89630627,  0.88356314],
       [-0.95535094,  0.29558926,  0.54251012, ...,  1.06146171,
        -0.66358547,  0.00415846],
       [ 0.71456109, -0.82583682, -0.02691344, ...,  0.00228226,
         0.36863113,  0.79470344],
       ...,
       [-0.07481629, -0.73927824, -0.75002892, ..., -0.09458097,
         0.6903276 , -0.13722915],
       [ 0.26554596,  0.36292713,  0.49053403, ...,  0.12122499,
         1.04558767, -0.06544102],
       [ 0.71557874,  0.32357043,  0.40015423, ...,  0.44726925,
         0.9780156 , -0.03261107]])

scaled_dataset = pd.DataFrame(transformer, columns = dataset.columns)
scaled_dataset

      Material Quantity (gm)  Additive Catalyst (gm)  Ash Component 
(gm)  \
0                   0.426686               -0.051081            -
0.724812
1                  -0.955351                0.295589             
0.542510
2                   0.714561               -0.825837            -
0.026913
3                   0.054894                0.691815            -
0.299672
4                   0.068006                0.698614             
0.609987
...                      ...                     ...                  .
..
6025               -0.738280               -0.130840             
0.211297
6026               -0.107772                0.432052            -
0.288645
6027               -0.074816               -0.739278            -
0.750029
6028                0.265546                0.362927             
0.490534
6029                0.715579                0.323570             
0.400154

      Water Mix (ml)  Plasticizer (gm)  Moderate Aggregator  \
0          -0.358485         -0.011902             0.949160
```

```
1          -0.741276         -0.510256            0.573672
2           1.036833         -0.222588            0.166956
3           1.095139          0.134464            0.848225
4           0.146734          0.034186            0.621197
...              ...               ...                 ...
6025       -0.925142         -0.018486            0.036155
6026       -0.558306          0.479868           -0.451608
6027       -0.255294          0.940238            0.514809
6028       -0.505965          0.087364           -0.823068
6029       -0.748732         -0.289440            1.040244

      Refined Aggregator  Formulation Duration (hrs)  Compression
Strength MPa
0               -0.540099                    0.896306
0.883563
1                1.061462                   -0.663585
0.004158
2                0.002282                    0.368631
0.794703
3                0.377262                    0.225924
0.526811
4               -0.155247                    0.512772
0.716349
...                   ...                         ...
...
6025             1.020169                    0.962939                  -
0.398118
6026             1.035242                   -0.291173                  -
0.239221
6027            -0.094581                    0.690328                  -
0.137229
6028             0.121225                    1.045588                  -
0.065441
6029             0.447269                    0.978016                  -
0.032611

[6030 rows x 9 columns]
```

```python
from sklearn.model_selection import train_test_split

x = scaled_dataset[['Material Quantity (gm)','Additive Catalyst
(gm)','Ash Component (gm)','Water Mix (ml)','Plasticizer
(gm)','Moderate Aggregator','Refined Aggregator','Formulation Duration
(hrs)']].values

y = scaled_dataset['Compression Strength MPa'].values

#hyper-parameter used here is random_state
x_train,x_test,y_train,y_test = train_test_split(x, y, test_size =
0.2, random_state = 150)
```

OBSERVATION:

After tuning the combinations of test_size and random_state continuosly, highest r2_score was achieved at test_size = 0.2, random_state = 150

```
from sklearn.ensemble import RandomForestRegressor

#hyper-parameters n_estimators is tuned manually
regf = RandomForestRegressor(n_estimators=178)
regf.fit(x_train, y_train)
y_pred = regf.predict(x_test)

from sklearn.metrics import r2_score

r2_score(y_test,y_pred)

0.4632811125970906

r2_score(y_test,y_pred)*100

46.32811125970906
```