```
import pandas as pd
df=pd.read_excel('/content/filtered_data.xlsx')
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1044506 entries, 0 to 1044505
Data columns (total 8 columns):
 #   Column                Non-Null Count    Dtype
---  ------                --------------    -----
 0   Unnamed: 0            1044506 non-null  int64
 1   Date                  1044506 non-null  datetime64[ns]
 2   Time                  1044506 non-null  object
 3   Global_active_power   1044506 non-null  float64
 4   Sub_metering_1        1044506 non-null  int64
 5   Sub_metering_2        1044506 non-null  int64
 6   Sub_metering_3        1044506 non-null  int64
 7   Total_power_consumed  1044506 non-null  float64
dtypes: datetime64[ns](1), float64(2), int64(4), object(1)
memory usage: 63.8+ MB
```

```
df.drop(columns=df.columns[0],axis=1,inplace=True)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1044506 entries, 0 to 1044505
Data columns (total 7 columns):
 #   Column                Non-Null Count    Dtype
---  ------                --------------    -----
 0   Date                  1044506 non-null  datetime64[ns]
 1   Time                  1044506 non-null  object
 2   Global_active_power   1044506 non-null  float64
 3   Sub_metering_1        1044506 non-null  int64
 4   Sub_metering_2        1044506 non-null  int64
 5   Sub_metering_3        1044506 non-null  int64
 6   Total_power_consumed  1044506 non-null  float64
dtypes: datetime64[ns](1), float64(2), int64(3), object(1)
memory usage: 55.8+ MB
```

```
df.drop(columns=df.columns[6],axis=1,inplace=True)
```

```
df['Timeinstr']=df['Time'].astype(str)
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1044506 entries, 0 to 1044505
Data columns (total 7 columns):
 #   Column               Non-Null Count    Dtype
```

```
 ---  ------                --------------    -----
  0   Date                  1044506 non-null  datetime64[ns]
  1   Time                  1044506 non-null  object
  2   Global_active_power   1044506 non-null  float64
  3   Sub_metering_1        1044506 non-null  int64
  4   Sub_metering_2        1044506 non-null  int64
  5   Sub_metering_3        1044506 non-null  int64
  6   Timeinstr             1044506 non-null  object
dtypes: datetime64[ns](1), float64(1), int64(3), object(2)
memory usage: 55.8+ MB
```

```
df.loc[(df['Timeinstr']>="06:00:00") &
       (df['Timeinstr'] < "12:00:00") ,
       'TimeinCat'] = "Morning"

df.loc[(df['Timeinstr']>="12:00:00") &
       (df['Timeinstr'] < "15:00:00") ,
       'TimeinCat'] = "Noon"

df.loc[(df['Timeinstr']>="15:00:00") &
       (df['Timeinstr'] < "18:00:00") ,
       'TimeinCat'] = "Evening"

df.loc[(df['Timeinstr']>="18:00:00") &
       (df['Timeinstr'] <= "23:59:00") ,
       'TimeinCat'] = "Night"

df.loc[(df['Timeinstr']>="00:00:00") &
       (df['Timeinstr'] < "06:00:00") ,
       'TimeinCat'] = "Early Morning"
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1044506 entries, 0 to 1044505
Data columns (total 8 columns):
 #   Column                Non-Null Count    Dtype
---  ------                --------------    -----
 0   Date                  1044506 non-null  datetime64[ns]
 1   Time                  1044506 non-null  object
 2   Global_active_power   1044506 non-null  float64
 3   Sub_metering_1        1044506 non-null  int64
 4   Sub_metering_2        1044506 non-null  int64
 5   Sub_metering_3        1044506 non-null  int64
 6   Timeinstr             1044506 non-null  object
 7   TimeinCat             1044506 non-null  object
dtypes: datetime64[ns](1), float64(1), int64(3), object(3)
memory usage: 63.8+ MB
```

```
df.drop(columns=df.columns[1],axis=1,inplace=True)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1044506 entries, 0 to 1044505
Data columns (total 7 columns):
 #   Column               Non-Null Count    Dtype         
---  ------               --------------    -----         
 0   Date                 1044506 non-null  datetime64[ns]
 1   Global_active_power  1044506 non-null  float64       
 2   Sub_metering_1       1044506 non-null  int64         
 3   Sub_metering_2       1044506 non-null  int64         
 4   Sub_metering_3       1044506 non-null  int64         
 5   Timeinstr            1044506 non-null  object        
 6   TimeinCat            1044506 non-null  object        
dtypes: datetime64[ns](1), float64(1), int64(3), object(2)
memory usage: 55.8+ MB
```

```
df.drop(columns=df.columns[5],axis=1,inplace=True)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1044506 entries, 0 to 1044505
Data columns (total 6 columns):
 #   Column               Non-Null Count    Dtype         
---  ------               --------------    -----         
 0   Date                 1044506 non-null  datetime64[ns]
 1   Global_active_power  1044506 non-null  float64       
 2   Sub_metering_1       1044506 non-null  int64         
 3   Sub_metering_2       1044506 non-null  int64         
 4   Sub_metering_3       1044506 non-null  int64         
 5   TimeinCat            1044506 non-null  object        
dtypes: datetime64[ns](1), float64(1), int64(3), object(1)
memory usage: 47.8+ MB
```

```
x=df.drop(columns=df.columns[1],axis=1)
```

```
y=df['Global_active_power']
```

```
x
```

	Date	Sub_metering_1	Sub_metering_2	Sub_metering_3	TimeinCat
0	2006-12-16	0	1	17	Evening
1	2006-12-16	0	1	16	Evening
2	2006-12-16	0	2	17	Evening
3	2006-12-16	0	1	17	Evening
4	2006-12-16	0	1	17	Evening
...	...	...	...	...	...
1044501	2008-12-13	0	0	0	Night

```
x.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1044506 entries, 0 to 1044505
Data columns (total 5 columns):
 #   Column          Non-Null Count    Dtype         
---  ------          --------------    -----         
 0   Date            1044506 non-null  datetime64[ns]
 1   Sub_metering_1  1044506 non-null  int64         
 2   Sub_metering_2  1044506 non-null  int64         
 3   Sub_metering_3  1044506 non-null  int64         
 4   TimeinCat       1044506 non-null  object        
dtypes: datetime64[ns](1), int64(3), object(1)
memory usage: 39.8+ MB
```

```
import datetime
x['Date']=pd.to_datetime(x['Date'])
x['Date']=x['Date'].dt.strftime("%d.%m.%y")
x['year']=pd.DatetimeIndex(x['Date']).year
x['month']=pd.DatetimeIndex(x['Date']).month
x['day']=pd.DatetimeIndex(x['Date']).day
x['dayofyear']=pd.DatetimeIndex(x['Date']).dayofyear
x['weekofyear']=pd.DatetimeIndex(x['Date']).weekofyear
x['weekday']=pd.DatetimeIndex(x['Date']).weekday
x['quarter']=pd.DatetimeIndex(x['Date']).quarter
x['is_month_start']=pd.DatetimeIndex(x['Date']).is_month_start
x['is_month_end']=pd.DatetimeIndex(x['Date']).is_month_end
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:8: FutureWarning: weekofyea
```

```
# handle categorical variable
x=pd.get_dummies(x,columns=['TimeinCat'],drop_first=True)
# # dropping extra column
# x= x.drop(columns=x.columns[1],axis=1)
# # concatation of independent variables and new cateorical variable.
```

```
# x=pd.concat([x,time],axis=1)
```

```
x
```

	Date	Sub_metering_1	Sub_metering_2	Sub_metering_3	TimeinCat_Evening	Timei
0	2006-12-16	0	1	17	1	
1	2006-12-16	0	1	16	1	
2	2006-12-16	0	2	17	1	
3	2006-12-16	0	1	17	1	
4	2006-12-16	0	1	17	1	
...	...	...	...	...	...	
1044501	2008-12-13	0	0	0	0	
1044502	2008-12-13	0	0	0	0	

```
x.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1044506 entries, 0 to 1044505
Data columns (total 17 columns):
 #   Column             Non-Null Count    Dtype
---  ------             --------------    -----
 0   Date               1044506 non-null  object
 1   Sub_metering_1     1044506 non-null  int64
 2   Sub_metering_2     1044506 non-null  int64
 3   Sub_metering_3     1044506 non-null  int64
 4   TimeinCat_Evening  1044506 non-null  uint8
 5   TimeinCat_Morning  1044506 non-null  uint8
 6   TimeinCat_Night    1044506 non-null  uint8
 7   TimeinCat_Noon     1044506 non-null  uint8
 8   year               1044506 non-null  int64
 9   month              1044506 non-null  int64
 10  day                1044506 non-null  int64
 11  dayofyear          1044506 non-null  int64
 12  weekofyear         1044506 non-null  int64
 13  weekday            1044506 non-null  int64
 14  quarter            1044506 non-null  int64
 15  is_month_start     1044506 non-null  bool
 16  is_month_end       1044506 non-null  bool
dtypes: bool(2), int64(10), object(1), uint8(4)
memory usage: 93.6+ MB
```

```
x.drop(columns=x.columns[0],axis=1,inplace=True)
```

```
x.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1044506 entries, 0 to 1044505
Data columns (total 16 columns):
 #   Column             Non-Null Count    Dtype
---  ------             --------------    -----
 0   Sub_metering_1     1044506 non-null  int64
 1   Sub_metering_2     1044506 non-null  int64
 2   Sub_metering_3     1044506 non-null  int64
 3   TimeinCat_Evening  1044506 non-null  uint8
 4   TimeinCat_Morning  1044506 non-null  uint8
 5   TimeinCat_Night    1044506 non-null  uint8
 6   TimeinCat_Noon     1044506 non-null  uint8
 7   year               1044506 non-null  int64
 8   month              1044506 non-null  int64
 9   day                1044506 non-null  int64
 10  dayofyear          1044506 non-null  int64
 11  weekofyear         1044506 non-null  int64
 12  weekday            1044506 non-null  int64
 13  quarter            1044506 non-null  int64
 14  is_month_start     1044506 non-null  bool
 15  is_month_end       1044506 non-null  bool
dtypes: bool(2), int64(10), uint8(4)
memory usage: 85.7 MB
```

```
# importing train_test_split from sklearn
from sklearn.model_selection import train_test_split
# splitting the data
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.3, random_state = 42)
```

```
# importing module
from sklearn.linear_model import LinearRegression
# creating an object of LinearRegression class
LR = LinearRegression()
# fitting the training data
LR.fit(x_train,y_train)
```

```
LinearRegression()
```

```
y_prediction=LR.predict(x_test)
y_prediction
```

```
array([0.91717919, 1.63782724, 0.41323844, ..., 0.88374066, 2.47710531,
       0.7893088 ])
```

```
# importing r2_score module
```

```
from sklearn.metrics import r2_score,classification_report
from sklearn.metrics import mean_squared_error
import numpy as np
# predicting the accuracy score
score=r2_score(y_test,y_prediction)
print('r2 socre is ',score)
print('mean_sqrd_error is==',mean_squared_error(y_test,y_prediction))
print('root_mean_squared error of is==',np.sqrt(mean_squared_error(y_test,y_prediction)))
#print(classification_report(y_test,y_prediction))
```

```
r2 socre is  0.7396906952151788
mean_sqrd_error is== 0.33406303087355976
root_mean_squared error of is== 0.577981860332623
```