```
import numpy as np
import pandas as pd
import matplotlib as mpl
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error
```

```
df_car = pd.read_csv('C:\Users\user\Downloads\archive (1)\ccpp_data.csv')
```

```
df_car.head()
```

```
   Temp     AP     RH     PE  EP
0 14.96 1010.8  94.80  473.90  438.08
1 25.18 1011.6  80.10  443.67  431.23
2 5.11 1029.3  90.80  475.96  477.50
3 20.86 1013.3  80.60  444.37  428.62
4 10.82 1018.7  83.70  488.56  475.98
```

```
df_car.tail()
```

```
      Temp     AP     RH     PE  EP
9563 16.65 1015.1  79.51  436.90  435.29
9564 24.77 1020.1  60.97  435.27  429.69
9565 33.29 1012.9  35.47  429.69  432.14
9566 24.30 1016.5  60.797  435.75  430.24
9567 10.47 1026.7  86.69  477.50  473.26
```

```
df_car.shape
```

```
(9568, 5)
```

```
df_car.columns
```

```
Index(['Temp', 'AP', 'RH', 'PE', 'EP'], dtype='object')
```

```
df_info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9568 entries, 0 to 9567
Data columns (total 5 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Temp    9568 non-null   float64
 1   AP      9568 non-null   float64
 2   RH      9568 non-null   float64
 3   PE      9568 non-null   float64
 4   EP      9568 non-null   float64
dtypes: float64(5)
memory usage: 374.0 KB
None
```

```
df_describ()
```

```
       Temp           AP           RH           PE           EP
count  9568  9568.000000  9568.000000  9568.000000  9568.000000
mean    19.  1013.259078    73.308978   454.365009   451.556692
std      7.    6.463555    14.600275    17.066995    16.933554
min      1.  992.890000    25.560000   420.26000   417.320000
25%     13.  1009.100000    63.327000   435.81000   432.958000
50%     20.  1012.830000    74.970000   451.19000   449.696000
75%     25.  1017.260000    84.830000   468.43000   465.962000
max     37.  1033.030000   100.160000   495.76000   493.210000
```

```
df_isnull().sum()
```

```
Temp    0
AP      0
RH      0
PE      0
EP      0
dtype: int64
```

```
df_fillna(df_mean())
```

```
       Temp     AP     RH     PE  EP
9563 16.65 1015.1  79.51  436.90  435.29
9564 24.77 1020.1  60.97  435.27  429.69
9565 33.29 1012.9  35.47  429.69  432.14
9566 24.30 1016.5  60.797  435.75  430.24
9567 10.47 1026.7  86.69  477.50  473.26
```

```
df_corr()
```

```
       Temp     AP     RH     PE  EP
Temp  1.000 -0.127 -0.543  0.844  0.872
AP   -0.127  1.000  0.099 -0.186 -0.158
RH   -0.543  0.099  1.000 -0.389 -0.341
PE    0.844 -0.186 -0.389  1.000  0.893
EP    0.872 -0.158 -0.341  0.893  1.000
```

```
df_corr()['EP']
```

```
Temp    0.872
AP     -0.158
RH     -0.341
PE      0.893
EP      1.000
Name: EP, dtype: float64
```

```
df_corr()['EP'].sort_values
```

```
<bound_method> of Temp    0.872
AP     -0.158
RH     -0.341
PE      0.893
EP      1.000
Name: EP, dtype: float64>
```

checking positive and negative correlations

```
corr=df_corr()
corr
```

```
       Temp     AP     RH     PE  EP
Temp  1.000 -0.127 -0.543  0.844  0.872
AP   -0.127  1.000  0.099 -0.186 -0.158
RH   -0.543  0.099  1.000 -0.389 -0.341
PE    0.844 -0.186 -0.389  1.000  0.893
EP    0.872 -0.158 -0.341  0.893  1.000
```

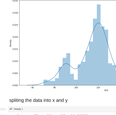
```
corr['Temp']
```

```
Temp    1.000000
AP     -0.126555
RH     -0.542837
PE      0.843710
EP      0.871846
Name: Temp, dtype: float64
```

```
corr['AP']
```

```
Temp    -0.126555
AP       1.000000
RH       0.098574
PE      -0.185851
EP      -0.158192
Name: AP, dtype: float64
```

```
C:\Users\User\AppDataLocal\Local\Programs\Spyder\Lib\site-packages\spyder\utils\misc\rsetup.py:204: FutureWarning: Version 1.5 is deprecated functions and will be removed in a future version. Please select pin versions in your requirements file. Consider using compatible-release or pinned versions instead. For instance, use '~=1.5' instead of '==1.5'.
  warnings.warn(msg, FutureWarning)
Collecting matplotlib==1.5.1
  Downloading matplotlib-1.5.1.tar.gz (1.2MB)
```



splitting the data into x and y

```
df_head()
```

```
   Temp     AP     RH     PE  EP
0 14.96 1010.8  94.80  473.90  438.08
1 25.18 1011.6  80.10  443.67  431.23
2 5.11 1029.3  90.80  475.96  477.50
3 20.86 1013.3  80.60  444.37  428.62
4 10.82 1018.7  83.70  488.56  475.98
```

```
df_tail()
```

```
      Temp     AP     RH     PE  EP
9563 16.65 1015.1  79.51  436.90  435.29
9564 24.77 1020.1  60.97  435.27  429.69
9565 33.29 1012.9  35.47  429.69  432.14
9566 24.30 1016.5  60.797  435.75  430.24
9567 10.47 1026.7  86.69  477.50  473.26
```

```
df_drop('EP',axis=1)
```

```
      Temp     AP     RH     PE
0 14.96 1010.8  94.80  473.90
1 25.18 1011.6  80.10  443.67
2 5.11 1029.3  90.80  475.96
3 20.86 1013.3  80.60  444.37
4 10.82 1018.7  83.70  488.56
```

```
x=df_drop('EP',axis=1)
x.shape
```

```
(9568, 4)
```

```
x
```

```
      Temp     AP     RH     PE
0 14.96 1010.8  94.80  473.90
1 25.18 1011.6  80.10  443.67
2 5.11 1029.3  90.80  475.96
3 20.86 1013.3  80.60  444.37
4 10.82 1018.7  83.70  488.56
9563 16.65 1015.1  79.51  436.90
9564 24.77 1020.1  60.97  435.27
9565 33.29 1012.9  35.47  429.69
9566 24.30 1016.5  60.797  435.75
9567 10.47 1026.7  86.69  477.50
[9568 rows x 4 columns]
```

```
y=df_drop('Temp',axis=1)
y.shape
```

```
(9568, 4)
```

```
y
```

```
      AP     RH     PE  EP
0 1010.8  94.80  473.90  438.08
1 1011.6  80.10  443.67  431.23
2 1029.3  90.80  475.96  477.50
3 1013.3  80.60  444.37  428.62
4 1018.7  83.70  488.56  475.98
9563 1015.1  79.51  436.90  435.29
9564 1020.1  60.97  435.27  429.69
9565 1012.9  35.47  429.69  432.14
9566 1016.5  60.797  435.75  430.24
9567 1026.7  86.69  477.50  473.26
[9568 rows x 4 columns]
```

train test split

```
x_train,x_test,y_train,y_test=train_test_split(x,y, test_size=0.3,random_state=1)
```

model training

```
lr=LinearRegression()lr.fit(x_train,y_train)
```

```
NameEr:name 'LinearRegression' is not defined
```

```
mr_fit(x_train,y_train)
```

```
NameEr:name'mr' is not defined
```

```
lr=LinearRegression()
```

```
lr.fit(x_train,y_train)
```

```
LinearRegression()
```

model prediction

```
prediction=lr.predict(x_test)
print(
```

```
array([438.12199, 438.09944, 437.07000000, 429.17410000, 432.06000000,
       432.20000000, 432.07000000, 432.06000000, 432.06000000,
       432.06000000, 432.06000000, 432.06000000, 432.06000000,
       432.06000000, 432.06000000, 432.06000000, 432.06000000,
       432.06000000, 432.06
```