air-quality

August 26, 2023

0.1 Problem I: If you have a limited number of monitors for a city (say only six-eight for any million-pluspopulation city), where would you place them? In other words, how would you choose sites for deploying the sensors?

0.1.1 A. Collecting information about the city

Before deciding on the locations of monitoring stations in the city, we need to gather some information about the city in following areas:

1. Source of Emission For the industrial areas, we need to get the information regarding - Type and count of industries, - kind of fuel used by them, - their distribution on geographical plane.

For residential areas, - Count and type of vehicles - Fuel used by them

2. Health and demographics In order to capture how the air quality is affecting the health of citizen of city, we need to get the information about the population first. A survey form can be circulated among the citizen to record their response on how they feel about the air quality in their area. This will help in shortlisting the locations in the city, especially in the residential areas for the establishing a monitoring station. Generally, the stations are put where the population density is high. However, other socio-economic factors are taken into consideration. The data recorded by these stations can help in epidemiological studies of air pollutants on human health. **3. Metereological Information** This involves data collection of meterological parameters like : temperature, relative humidity, wind speed, wind direction. Monitors are fixed in the areas which are downwind from the source at mixing height as Indian Meterological Department (IMD). **4. Topographical Information** Mountains, hills, valleys and large water bodies can affect the dispersion of pollutants. Thus it is suggested to put monitors in the area where spatial variation in concentration is large. **5. Previous Air Quality** Earlier air quality data can also act as a basis for shortlisting locations for establishing monitors.

0.1.2 B. Selection of location for monitors

The main requirements for the selection of location of monitors: **1. Representative Site**

There are certain conditions that needs to be fulfilled before finalizing the location: - It should be representation site. - It should be away from *absorbing surfaces*. - Location should be represention site for a long time i.e. no interference in the surrounding e.g. construction. - There should be free air flow available all the time. Thus, station should not be fixed in balcony, corner etc. - Should be at least 25 m away form the pollution source like domestic chimney (WHO 1977) especially if chiminey height is below the height of sampling point.

2. Comparability It is important that the data collected from different stations is comparable. Thus, one needs to follow the guidelines provided in IS 5182 (part 14) 2000: - It should be open from all sides, - For monitoring traffic pollution, sampling intake should be greater than 3 m above street level. While for unpaved roads, the samplers should be kept 200 m above from them for the protection from dust.

3. Physical requirements of monitoring site - Location should be able to be a representative site for a long time. - Easily accessible throughout year, - Should be protected from extreme weather conditions and vandals. - Site should have sheltering with facilities like water and electricity

4. Topographical and Meterological Factors Hills, mountains, valleys and large water bodies can affect the pollutant distribution. Also, wind pattern can change because of heating up of wind and day and its cooling in nights. This is could also change how pollutant flows. Thus these factors must be kept in mind while fixing monitors.

The next step is to decide on pollutants or *pollutant selection* and to deploy sensors to record for the same. Then, we check on *sample duration and requirement, measurement methods, lab requirements, quality assurance,* and finally *data handling and presentation.*

0.2 Distribution of monitors in the city

As per WHO 1977, the distribution of stations: - 5 stations : 3 (Industrial / City Centre), 2 (residential) - 10 stations : 6 (Industrial / City Centre), 4 (residential)

Upon interpolation, we get that for stations 6 or 8 distribution should be as follows: - 6 stations: 4 (Industrial / City Centre), 2 (residential) - 8 stations : 5 (Industrial / City Centre),3 (residential)

0.3 Problem II : Using data from one of the nearest CAAQMS (continuous ambient air quality monitoring station) in your city or the one at Sri Aurobindo Marg, New Delhi, summarize PM2.5 levels in your area for July 2023 (01 July - 31 July). Support your observations with meteorology parameters like wind speed, wind direction, temperature, relative humidity, and rainfall from the same source. Using programming languages like R (openair) or Python will be highly appreciated. Ensure to write a detailed description of the observations

Data was collected from the website of Central Pollution Control Board (https://app.cpcbccr.com/ccr/#/login) for the location of **Sri Aurobindo Marg, New Delhi** for the duration of *July 2023*. The main objective of this analysis is to make observations about the change in PM2.5 levels with different meterological parameters.

Columns description - PM2.5 : Particulate Matter (2.5) (ug/m3) - WS : Wind Speed (m/s) - SR : Solar Radiations (W/mt2) - RF : Rainfall (mm) - TOT-RF : Total Rainfall(mm)

[1]:

```python
# importing libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import datetime as dt

from sklearn.impute import KNNImputer
```

[2]:
```
# important functions

# function that will assign wind direction a quadrant of magnetic compass
def wind_dir(x):
    if x >=0 and x < 90:
        return "NE"
    elif x >= 90 and x < 180:
        return "NW"
    elif x>= 180 and x < 270:
        return "SW"
    else:
        return "SE"

# function to change the labelling of weeks
def week_num(x):
    if x == 26:
        return 1
    elif x == 27:
        return 2
    elif x == 28:
        return 3
    else:
        return 4
```

[3]:
```
# reading data
df =  pd.read_excel('./data.xlsx')
```

[4]:
```
# viewing data
df.tail()
```

[4]:
```
            From Date           To Date  PM2.5   RF     SR      WD    WS  \
739  31-07-2023 19:00  31-07-2023 20:00   17.0  0.0  10.78  126.75  0.83
740  31-07-2023 20:00  31-07-2023 21:00   19.0  0.0   8.47  139.05  0.48
741  31-07-2023 21:00  31-07-2023 22:00   23.0  0.0   8.50  104.83  0.57
742  31-07-2023 22:00  31-07-2023 23:00   20.0  0.0   8.50  124.38  0.68
743  31-07-2023 23:00  31-07-2023 23:19   24.0  0.0   8.50  155.55  1.20

        RH  Temp  TOT-RF
739  69.55   NaN     0.0
740  80.95   NaN     0.0
741  84.15   NaN     0.0
742  85.63   NaN     0.0
743  80.45   NaN     0.0
```

[5]:
```
# changing the datatype of timestamp to datetime
df["from_date"] = pd.to_datetime(df['From Date'], dayfirst = True)
df['to_date'] = pd.to_datetime(df['To Date'], dayfirst = True)
```

[6]:
```
# creating separate columns for week, day, hour, weekday
df['hour'] = df['to_date'].dt.hour
df['hour'] = df['hour'].apply(lambda x: 24 if x == 0 else x) # converting 0th␣
 ↪hour to 24th hour


df['day'] = df['to_date'].dt.day
df['weekday'] = df['to_date'].dt.weekday

# assigning week number
df['week'] = (df['to_date'].dt.isocalendar().week).apply(week_num)
```

[7]:
```
# total available columns
df.columns
```

[7]:
```
Index(['From Date', 'To Date', 'PM2.5', 'RF', 'SR', 'WD', 'WS', 'RH', 'Temp',
       'TOT-RF', 'from_date', 'to_date', 'hour', 'day', 'weekday', 'week'],
      dtype='object')
```

% of missing values in each feature

[8]:
```
# checking for null values
df.isna().sum()*100/df.shape[0]
```

[8]:
```
From Date       0.000000
To Date         0.000000
PM2.5          24.193548
RF             27.553763
SR             27.688172
WD             27.822581
WS             27.822581
RH             27.553763
Temp          100.000000
TOT-RF          0.000000
from_date       0.000000
to_date         0.000000
hour            0.000000
day             0.000000
weekday         0.000000
week            0.000000
dtype: float64
```

[9]:
```
# unique values in features
print('Total unique values in feature TOT-RF:',df['TOT-RF'].nunique())
```

```
Total unique values in feature TOT-RF: 11
```

```
[10]: # deleting the redundant columns
      df.drop(['From Date','To Date','from_date',"to_date", 'Temp', 'TOT-RF'], axis =␣
       ↪1, inplace = True)
```

Missing value imputation using KNN

```
[11]: # KNN imputer
      imputer = KNNImputer(n_neighbors = 4, weights ='distance')
```

```
[12]: df1 = pd.DataFrame(imputer.fit_transform(df),columns = df.columns)
      df1.head()
```

```
[12]:    PM2.5   RF    SR      WD    WS     RH  hour  day  weekday  week
      0   20.0  0.0  8.50  131.38  0.57  93.40   1.0  1.0      5.0   1.0
      1   17.0  0.0  8.50   83.75  0.68  93.40   2.0  1.0      5.0   1.0
      2   20.0  0.0  8.50   48.60  0.65  93.40   3.0  1.0      5.0   1.0
      3   18.0  0.0  8.55  206.85  0.52  93.47   4.0  1.0      5.0   1.0
      4   14.0  0.0  8.53  136.15  0.40  93.47   5.0  1.0      5.0   1.0
```

```
[13]: # checking for missing values
      df1.isna().sum()
```

```
[13]: PM2.5      0
      RF         0
      SR         0
      WD         0
      WS         0
      RH         0
      hour       0
      day        0
      weekday    0
      week       0
      dtype: int64
```

```
[14]: # creating new feature by assigning quadrant to wind speed
      df1['WD2'] = df1['WD'].apply(wind_dir)
```

```
[15]: df1['hour'] = df1['hour'].astype('int')
      df1['day'] = df1['day'].astype('int')
      df1['week'] = df1['week'].astype('int')

      df1['RF'] = df1['RF'].round(2)
      df1['SR'] = df1['SR'].round(2)
      df1['WD'] = df1['WD'].round(2)
      df1['WS'] = df1['WS'].round(2)
      df1['RH'] = df1['RH'].round(2)
```

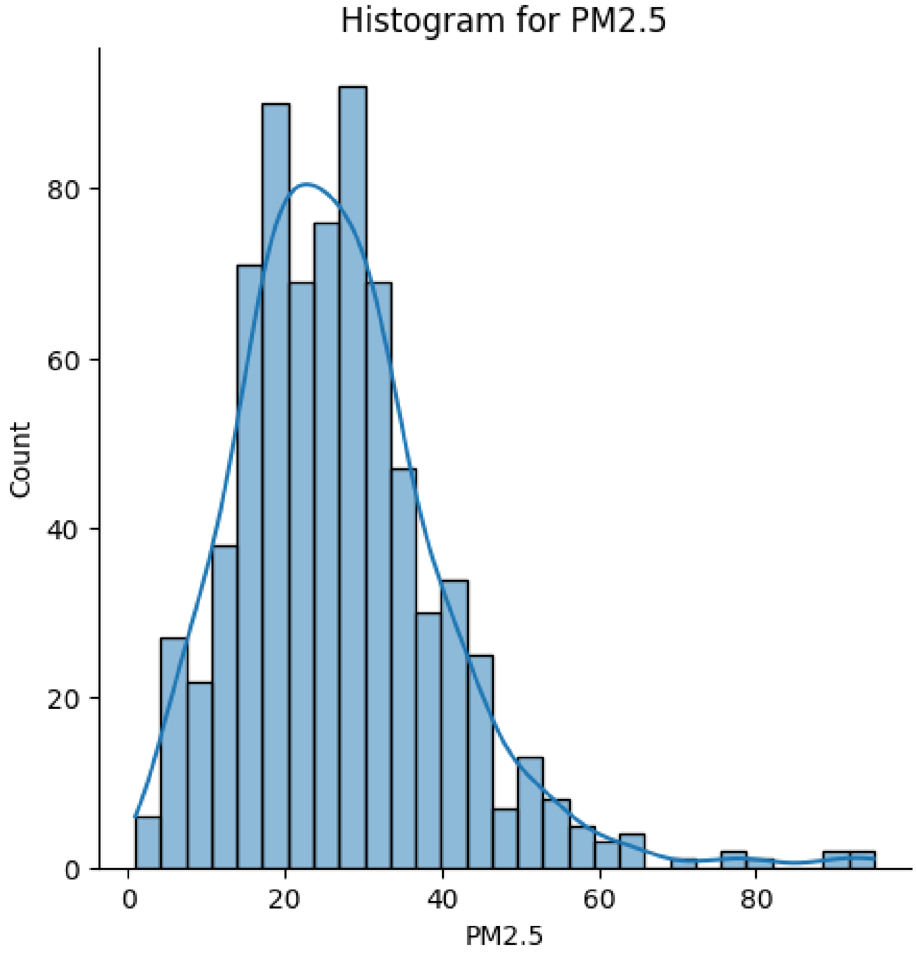
0.4 Exploratory Data Analysis

0.4.1 Univariate Analysis

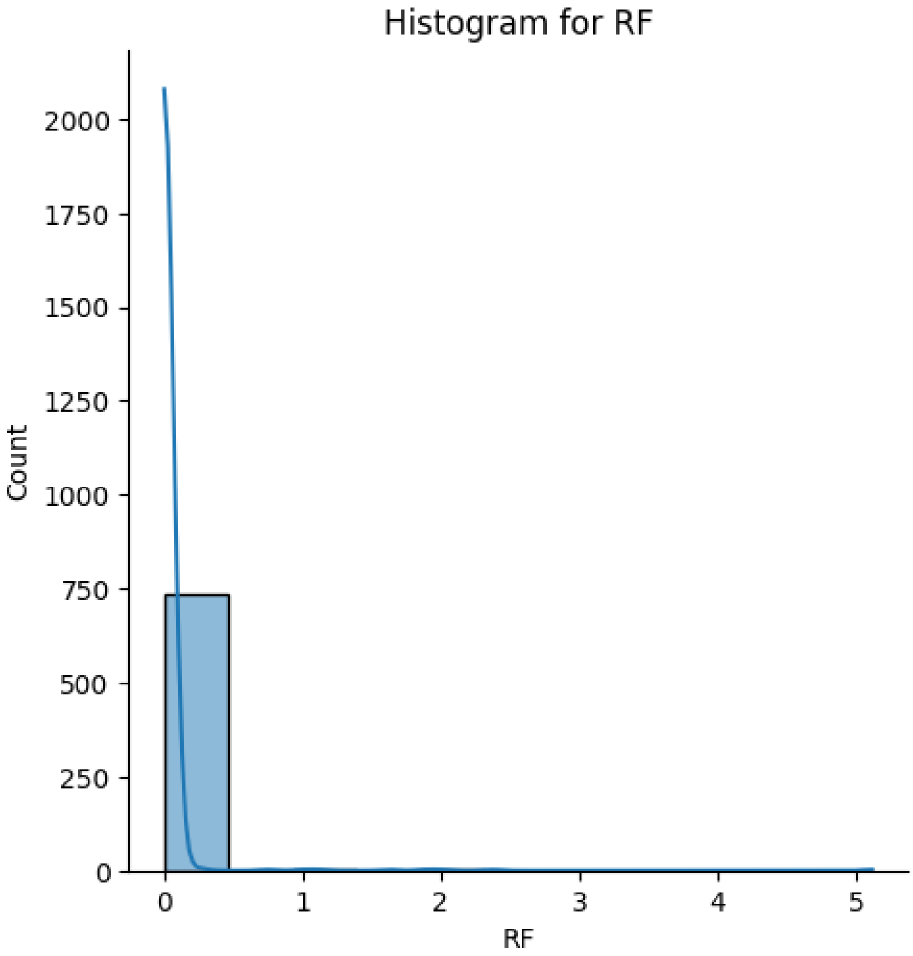
[16]:

```python
# histogram for PM2.5
cols = ['PM2.5','RF','WD','WS','RH']
for i in cols:
    print(f"mean of {i} is : {np.mean(df1[i])}")
    sns.displot(df1[i], kde = True)
    tit = 'Histogram for '+ i
    plt.title(tit)
    plt.show()
```

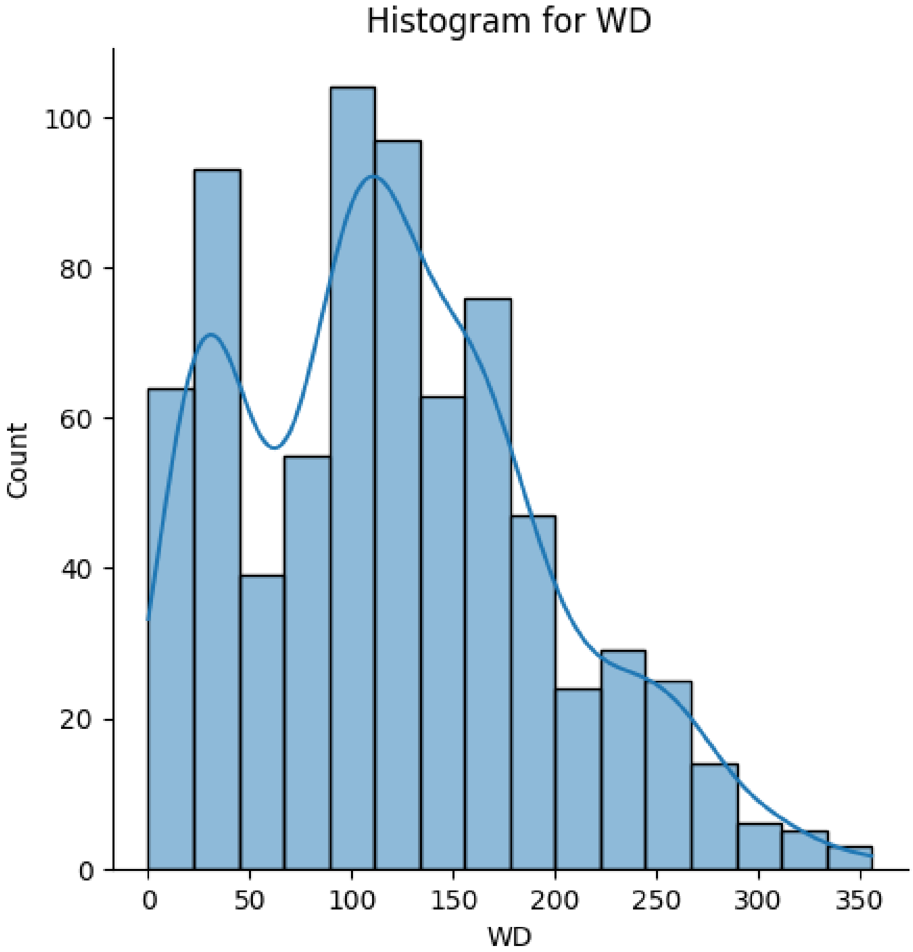
```
mean of PM2.5 is : 26.879671296815154
```



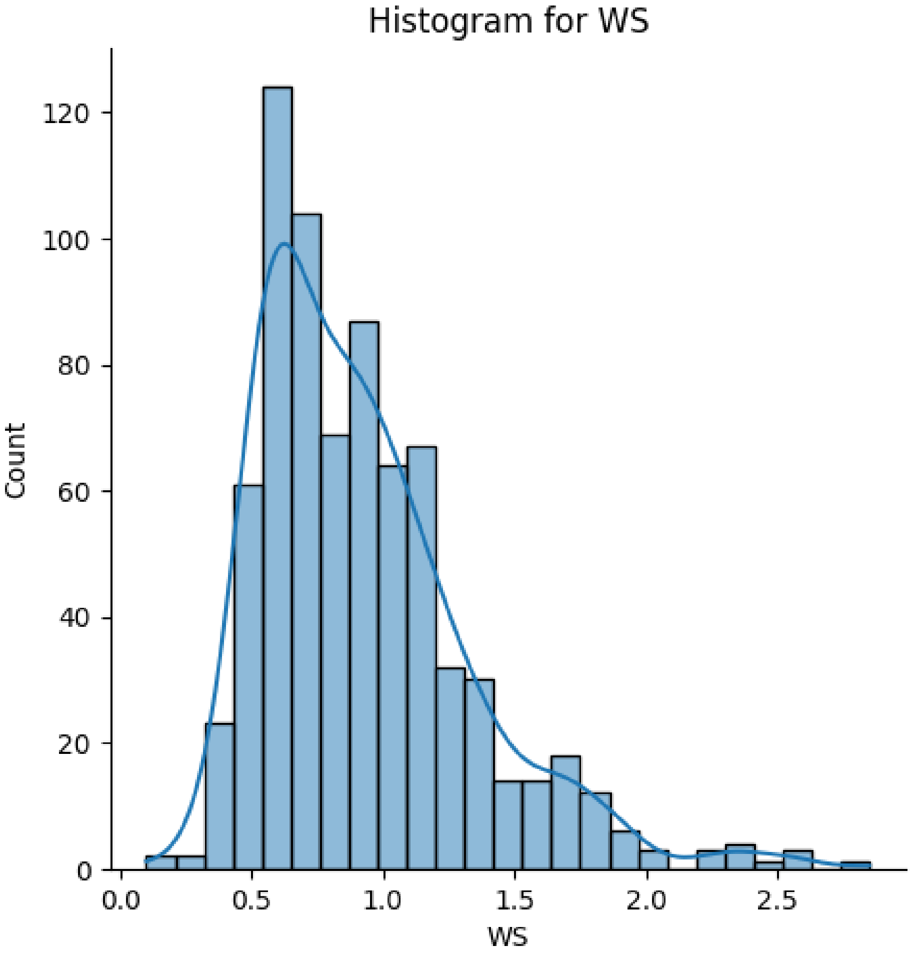
```
mean of RF is : 0.022513440860215055
```



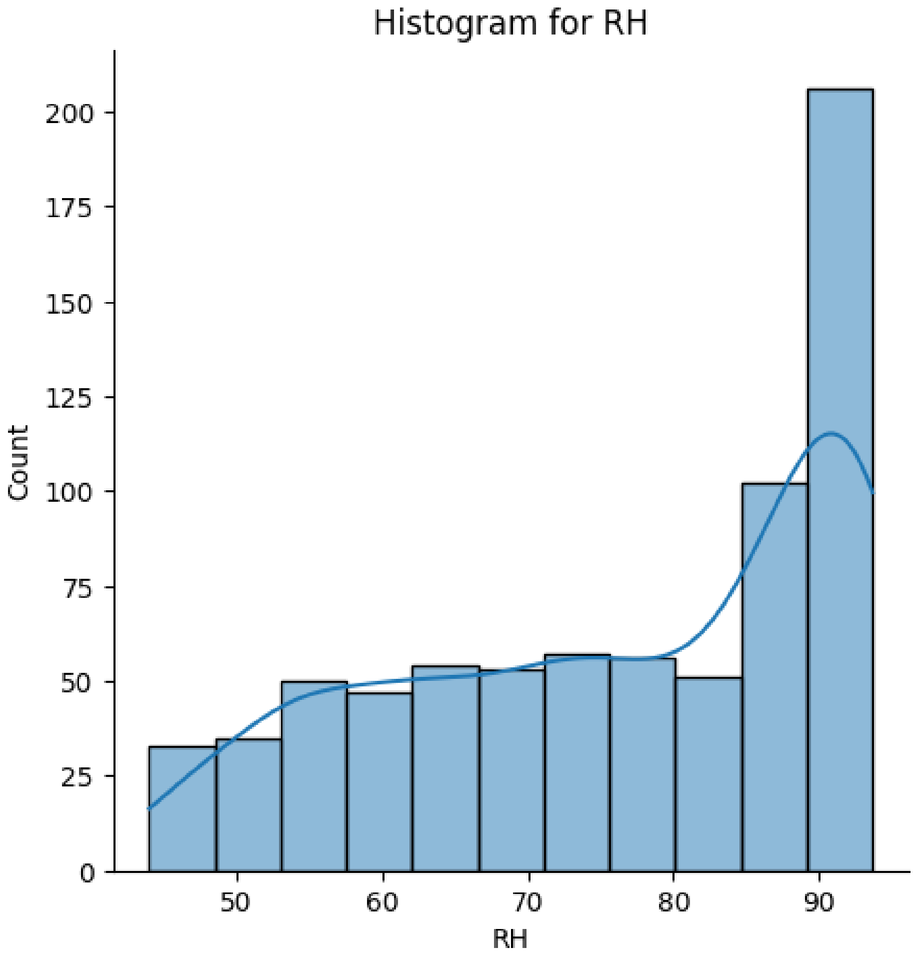
```
mean of WD is : 121.24185483870968
```



```
mean of WS is : 0.9230241935483872
```



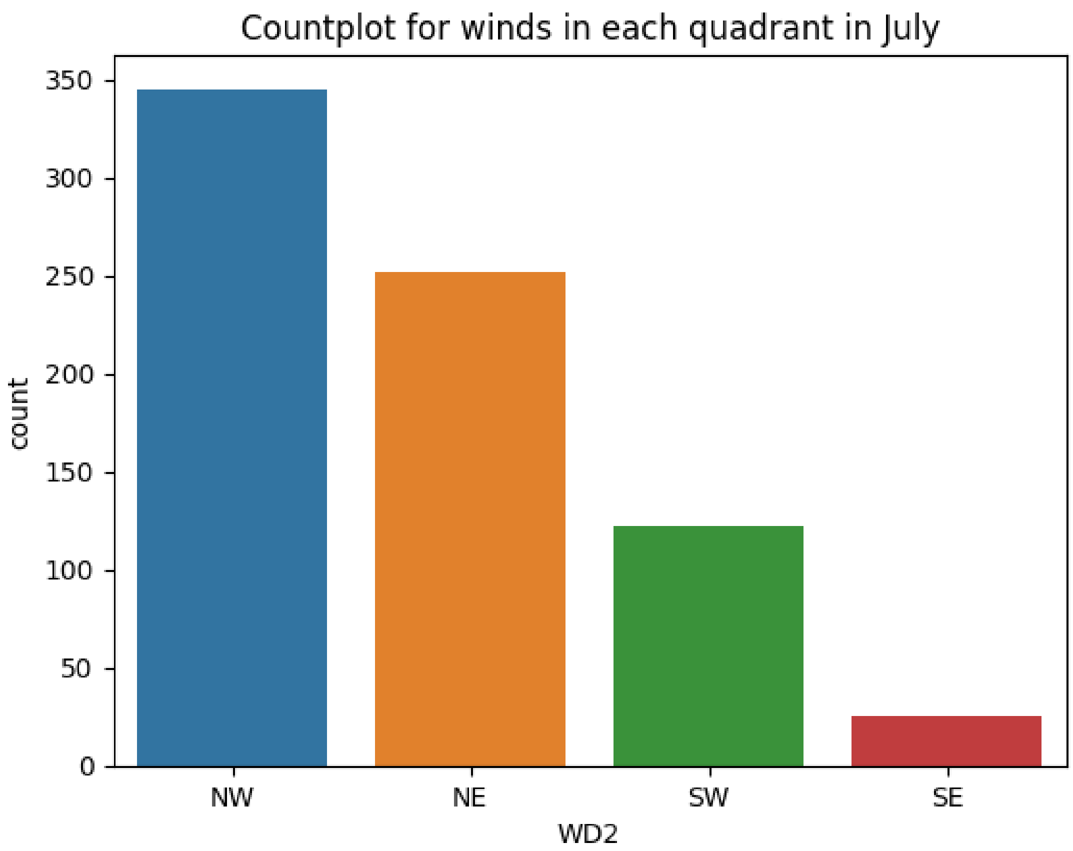
```
mean of RH is : 76.25048387096774
```



Observation - PM2.5 is somewhat normal however statistical test like KS test or QQ plot can be used to check normality. - There hasn't been much rainfall occured in July 2023 with mean of only 0.022 mm. - Most of the time wind has blow in II quadrant i.e. in NW (North - West) quadrant. - The mean Relative humidity in this month is 76.25%. - The mean wind speed is 0.923 m/sec.

[61]:
```
sns.countplot(data = df1, x  = 'WD2' )
plt.title('Countplot for winds in each quadrant in July')
plt.show()
```

[61]:
```
Text(0.5, 1.0, 'Countplot for winds in each quadrant in July')
```

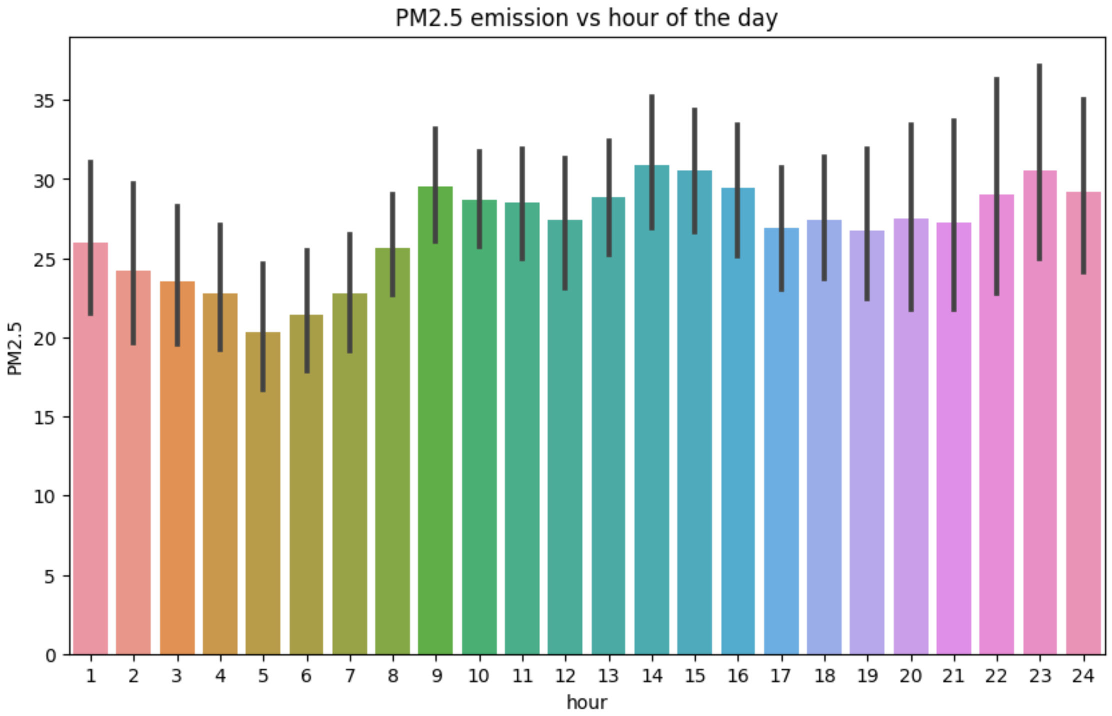


Observation - This shows that the wind has blow in NW (North-West) quadrant the most followed by NE (North-East) quadrant. - Least amount of winds are blowing in IV quadrant of SE (South-East).

0.4.2 Bivariate Analysis

At what time of the day PM2.5 is max?

[19]:
```
plt.figure(figsize = (10,6))
sns.barplot(data = df1, x = 'hour',y = 'PM2.5')
plt.title('PM2.5 emission vs hour of the day')
plt.show()
```



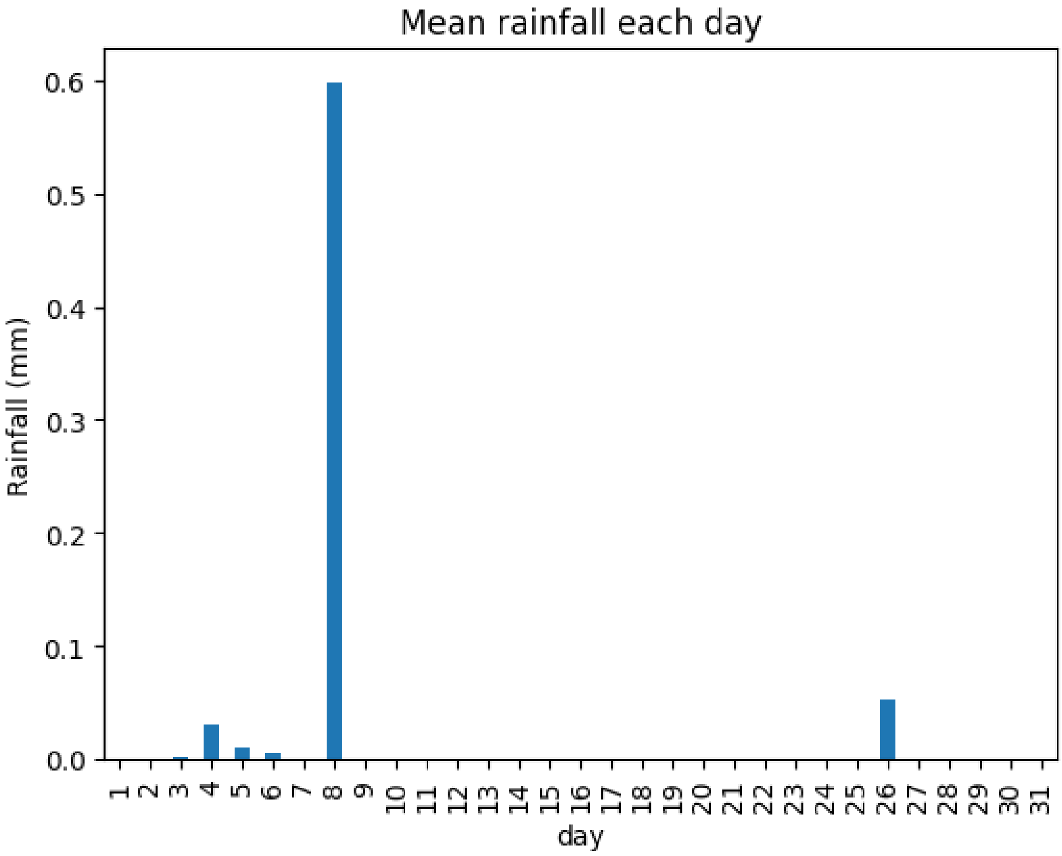
Observation - The PM 2.5 levels starts to rise from the morning and after 9 AM keep on fluctuating by small changes. The early morning rise may be caused by the school buses and after 9:00 AM PM 2.5 rise increase can be due to public transport and personal automobiles as people start their businesses like going office, running shops, opening of shopping malls etc. - By the evening, people tend to return from their work, or go for an outing. That is why the traffic in evening is high and hence the PM2.5 levels.

How many days did the rainfall occur in the month of July?

[58]:
```
print(f"Number of days rainfall occured: {(((((df1[['day','RF']]).groupby('day'))['RF'].mean()) != 0).sum()}")

(((df1[['day','RF']]).groupby('day'))['RF'].mean()).plot(kind= 'bar', title = 'Mean rainfall each day')
plt.ylabel('Rainfall (mm)')
plt.show()
```

```
Number of days rainfall occured: 6
```



Only 6 days there was rainfall in the month of July 2023 (3,4,5,6,8, and 26). On 8th July max rainfall was observed.

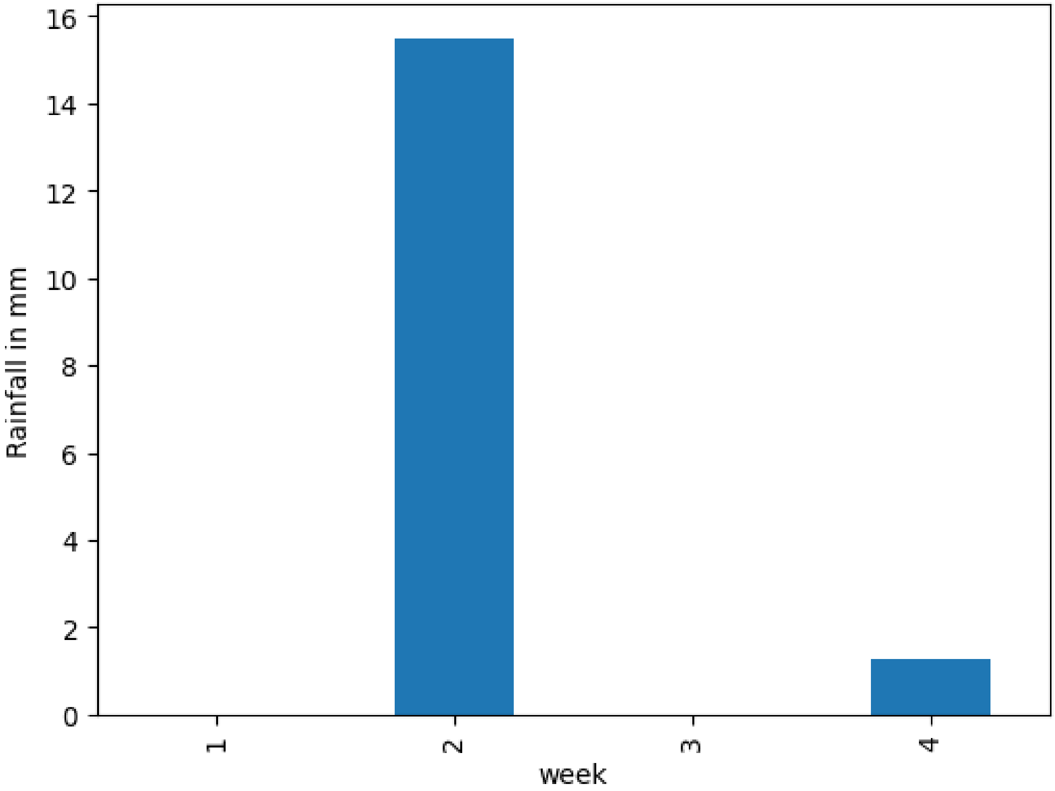
Which week did the rainfall occur in the month of July?

```
[20]: print("Rainfall in mm: \n",((df1[['RF','week']]).groupby('week')['RF'].sum()))

# visualization
((df1[['RF','week']]).groupby('week')['RF'].sum()).plot(kind = 'bar')
plt.ylabel('Rainfall in mm')
```

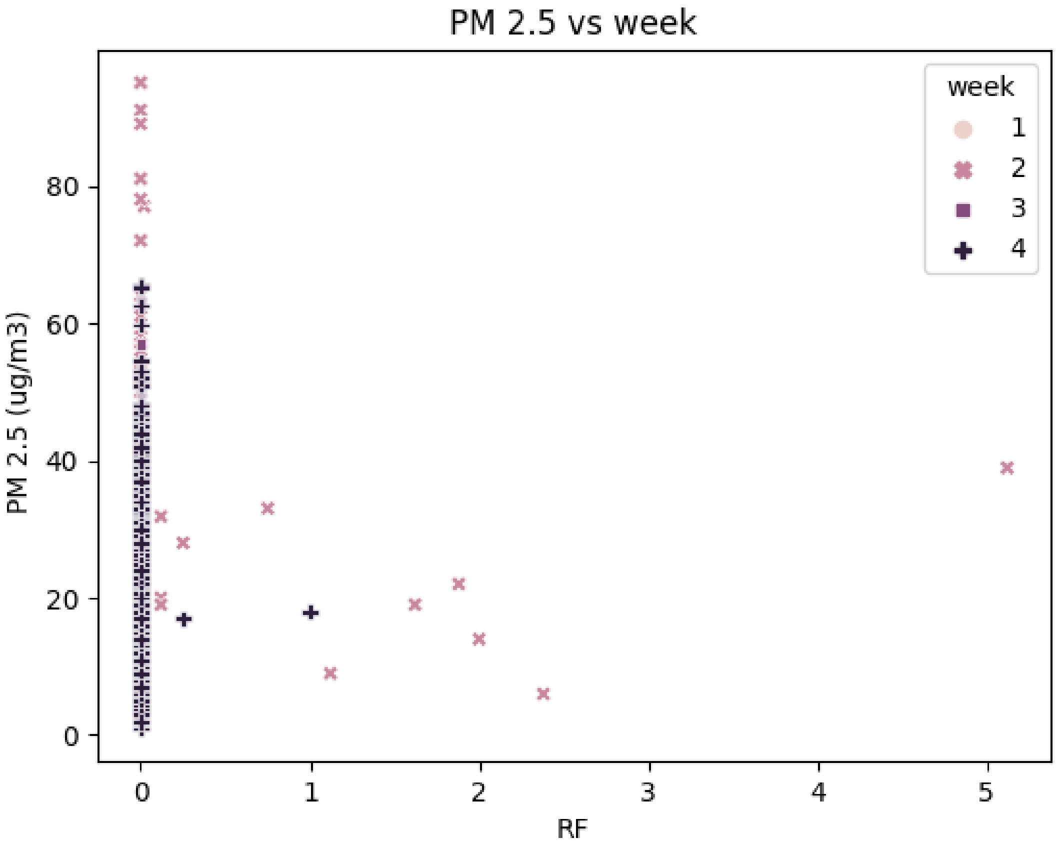
```
Rainfall in mm:
 week
1     0.00
2    15.50
3     0.00
4     1.25
Name: RF, dtype: float64
```

```
[20]: Text(0, 0.5, 'Rainfall in mm')
```



Observation - There was rainfall in the week 2 and last week, with max rainfall occuring in week 2

[63]:
```
sns.scatterplot(data = df1, x = 'RF', y = 'PM2.5', hue = 'week', style = 'week')
plt.ylabel('PM 2.5 (ug/m3) ')
plt.title('PM 2.5 vs week')
plt.show()
```

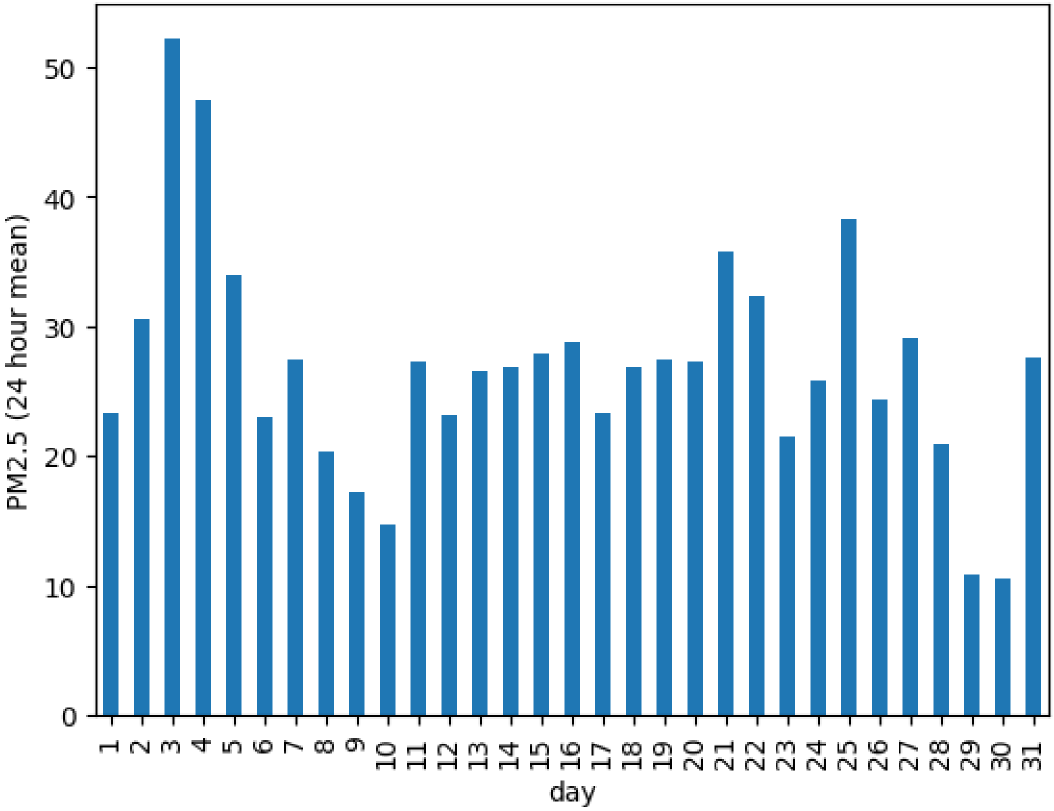


Observation - No significant drop was observed in the PM2.5 even when rainfall occured. - Since, most of the days were without rainfall most of the values are concentrated in a single vertical line. - In week 2, PM2.5 levels reach their maximum values.

What is the 24 hour mean of PM2.5 ?

```
[39]: print("24 hour mean  of PM 2.5\n",((df1[['day','PM2.5']]).groupby('day')['PM2.
      ↪5'].mean()).values.mean())
      # visualization
      ((df1[['day','PM2.5']]).groupby('day')['PM2.5'].mean()).plot(kind = 'bar')
      plt.ylabel('PM2.5 (24 hour mean)')
      plt.show()
```

```
24 hour mean  of PM 2.5
 26.87395600929763
```



As per ambient air quality standards are based on the authority of the Environment Act (1986), for residential/industrial/rural and other area (other than ecologically sensitive areas), PM2.5 levels should not be greated than 60 ug/m3. And values in not observed greater than 60 ug/m3, which is a good sign.

Source: https://www.transportpolicy.net/standard/india-air-quality-standards/#:~:text=Under%20the%20authority%20of%20the,planning%20to%20meet%20such%20standards.

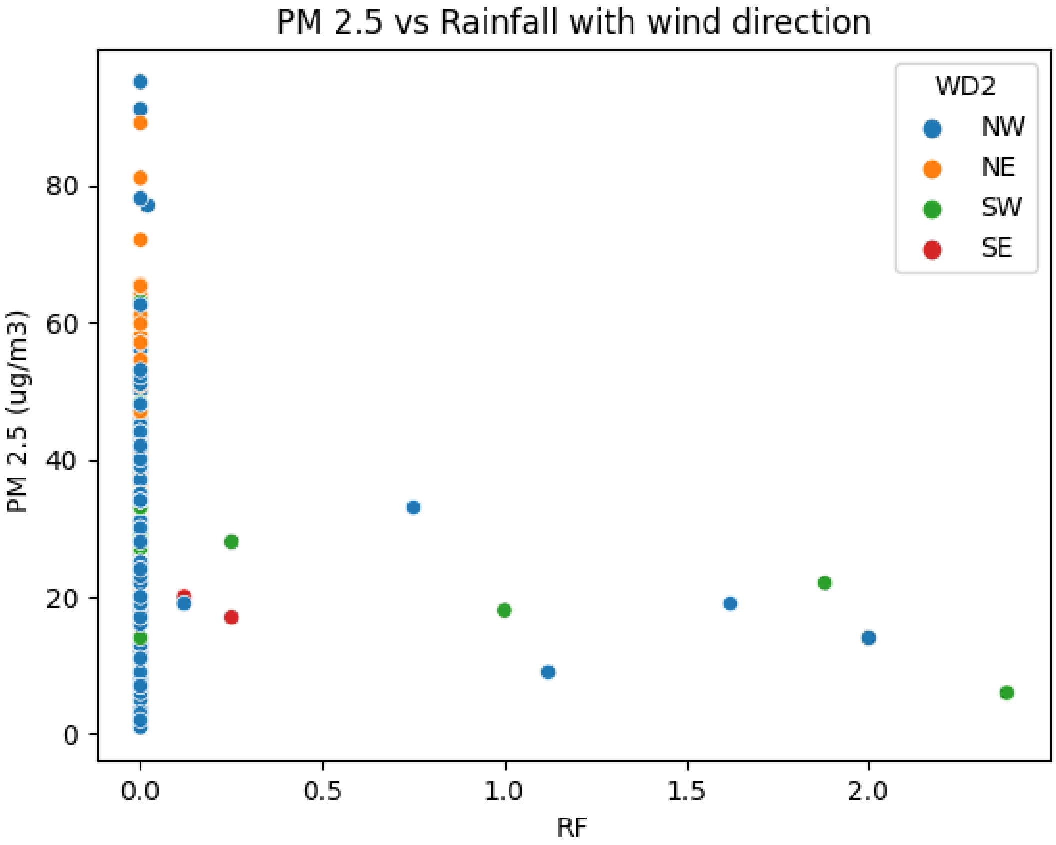
```
[65]: # reading of wind as per quadrant each hour
      df1['WD2'].value_counts(normalize = True)*100
```

```
[65]: WD2
      NW    46.370968
      NE    33.870968
      SW    16.397849
      SE     3.360215
      Name: proportion, dtype: float64
```

Observation - There 46.37% of total hours in the month of July wind has blown in the quandrant of NW (North-West) followed by NE (North-East) with 33.87% of total hours.

[66]:

```
sns.scatterplot(data = df1, x = df1['RF'], y = df['PM2.5'], hue = 'WD2')
plt.ylabel('PM 2.5 (ug/m3) ')
plt.title("PM 2.5 vs Rainfall with wind direction")
plt.show()
```



Observation - Days of rainfall, the wind direction was in SW and NW direction and very little rainfall occured when the wind direction was in IV quadrand SE (South-East).

How is the wind speed in each quadrant ?

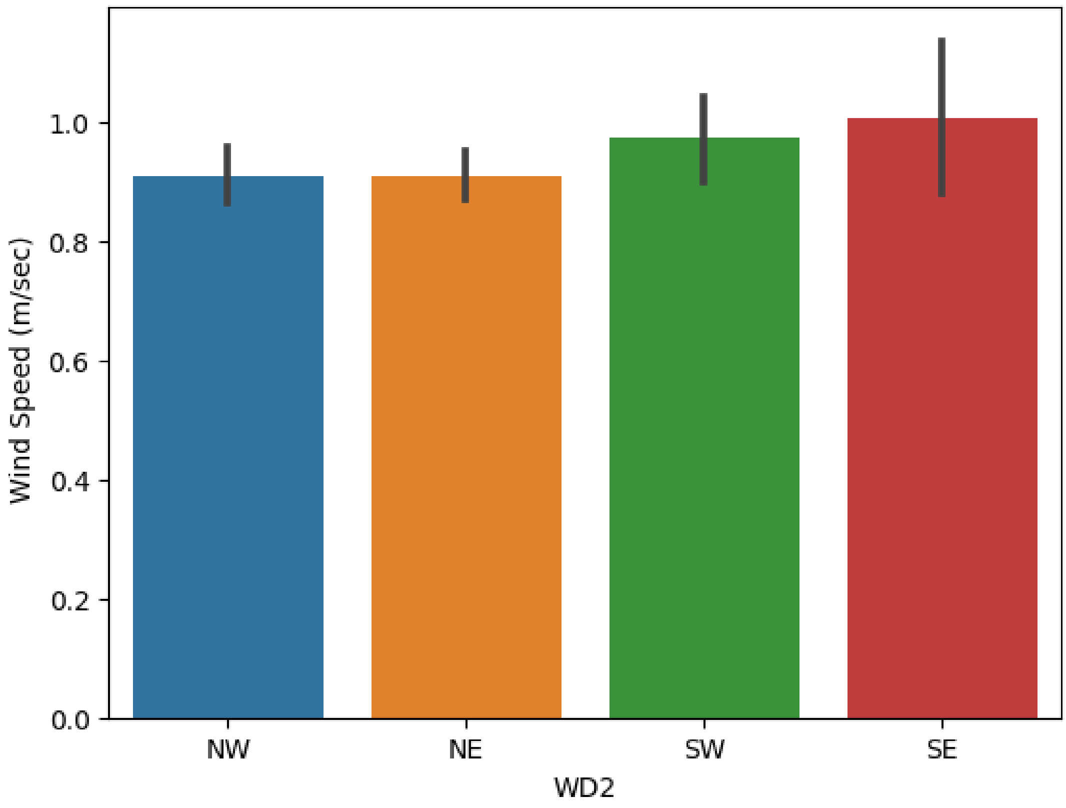
[72]:

```
print('Mean of wind speed in each quadrant:')
print((df1[['WD2','WS']].groupby('WD2'))['WS'].mean())
sns.barplot(data = df1, x = 'WD2',y = 'WS')
plt.ylabel('Wind Speed (m/sec) ')
```

```
Mean of wind speed in each quadrant:
WD2
NE    0.909484
NW    0.908986
SE    1.005200
SW    0.973852
```

```
Name: WS, dtype: float64
```

```
[72]: Text(0, 0.5, 'Wind Speed (m/sec) ')
```

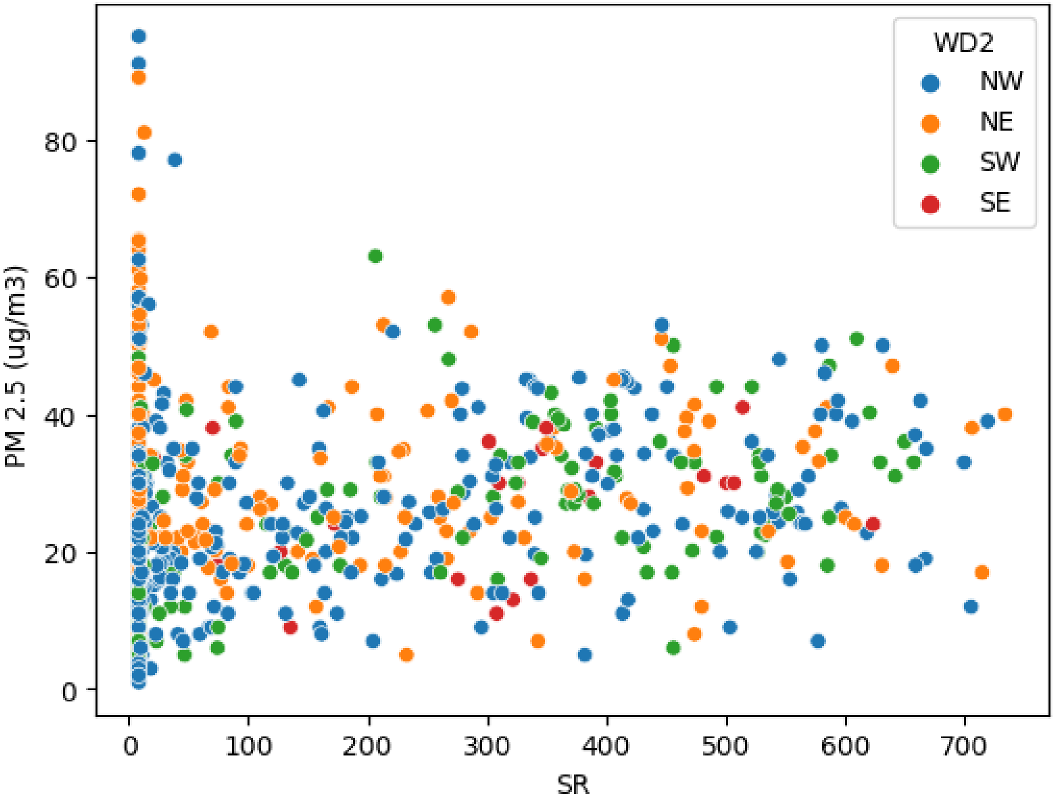


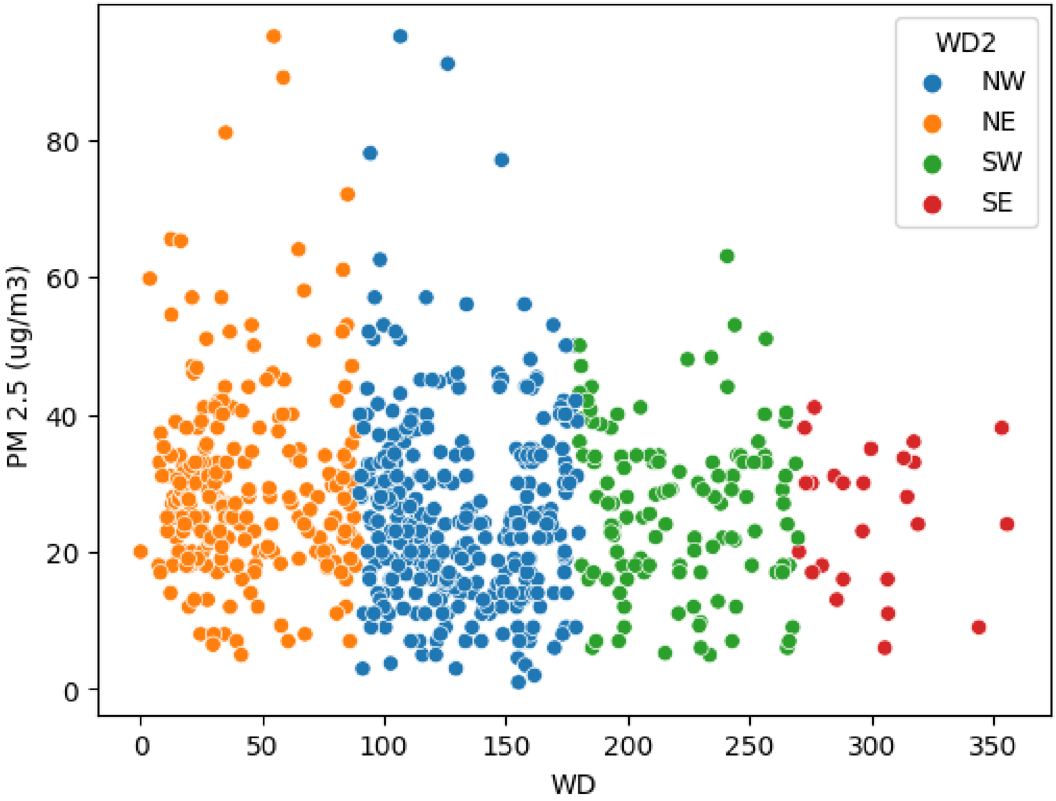
Observation - Winds are fastest when they are flowing in IV quandrant SE (south-east). - winds are slowest when they are in NW (North-West) quadrant.

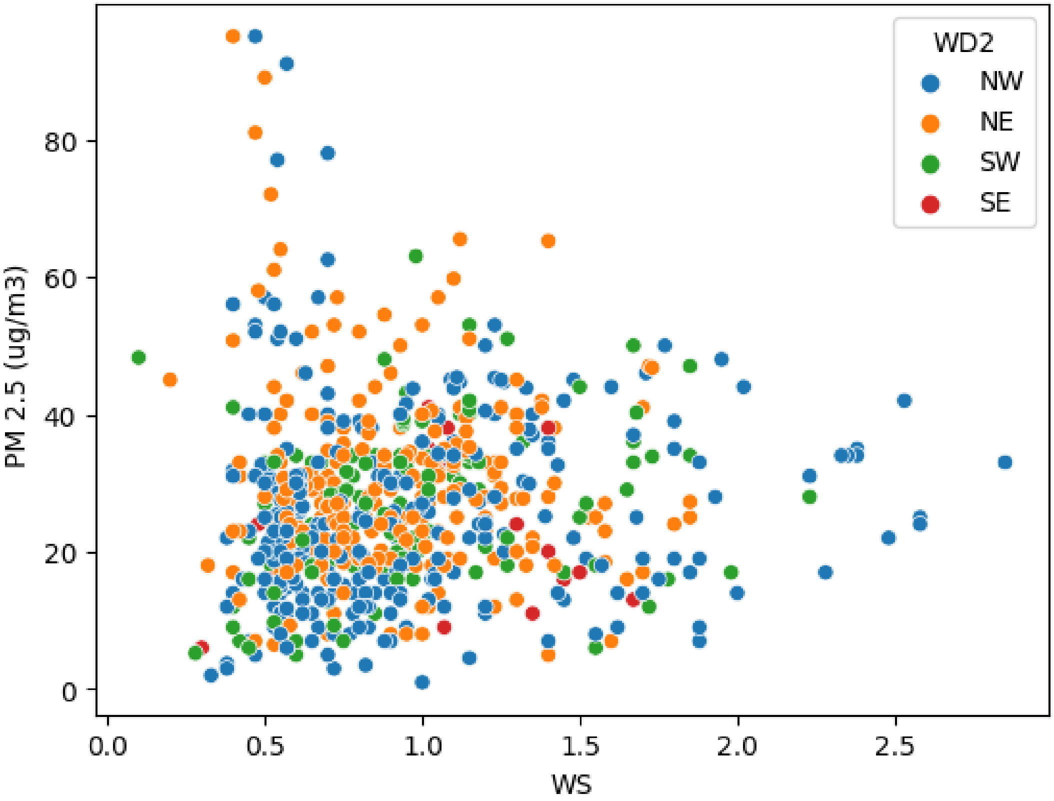
Scatter plot of PM 2.5 with other meterological features

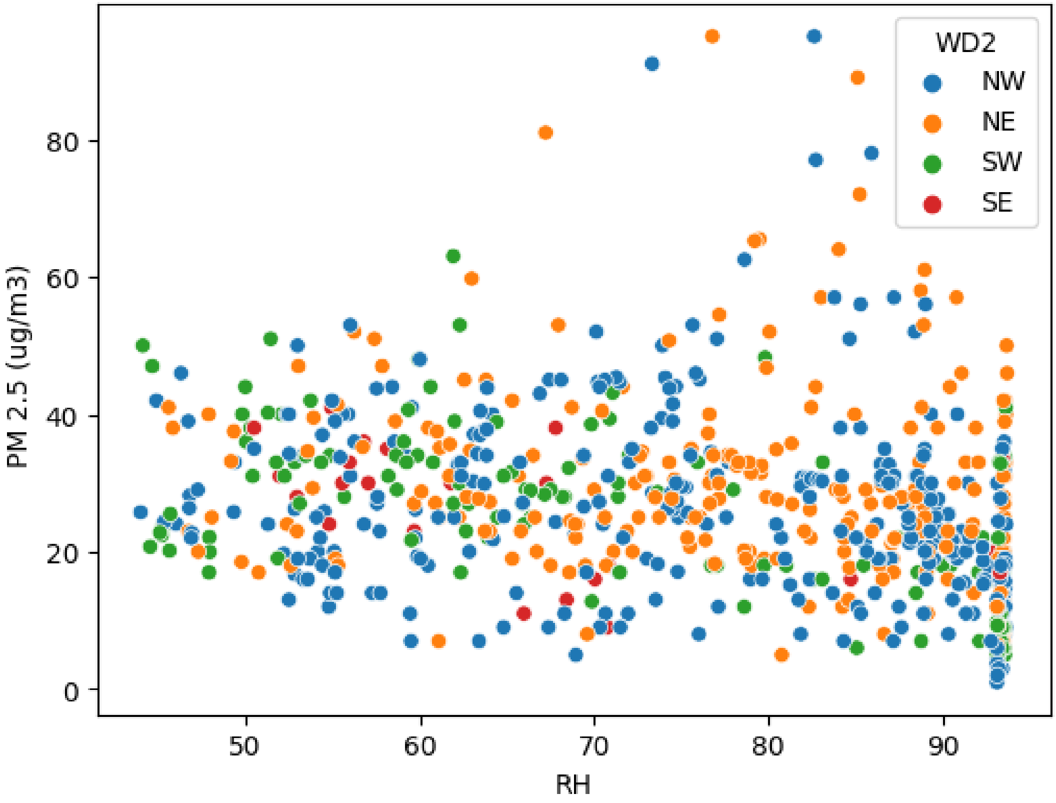
```
[29]: cols = ['SR', 'WD', 'WS', 'RH']

      for i in cols:
          sns.scatterplot(data = df1, x = i, y = 'PM2.5', hue = 'WD2')
          plt.xlabel(i)
          plt.ylabel('PM 2.5 (ug/m3) ')
          plt.show()
```


WD2
NW
NE
SW
SE
PM 2.5 (ug/m3)
0
20
40
60
80
SR
0
100
200
300
400
500
600
700


WD2
NW
NE
SW
SE
PM 2.5 (ug/m3)
0
20
40
60
80
WD
0
50
100
150
200
250
300
350


WD2
NW
NE
SW
SE
PM 2.5 (ug/m3)
0
20
40
60
80
0.0
0.5
1.0
1.5
2.0
2.5
WS



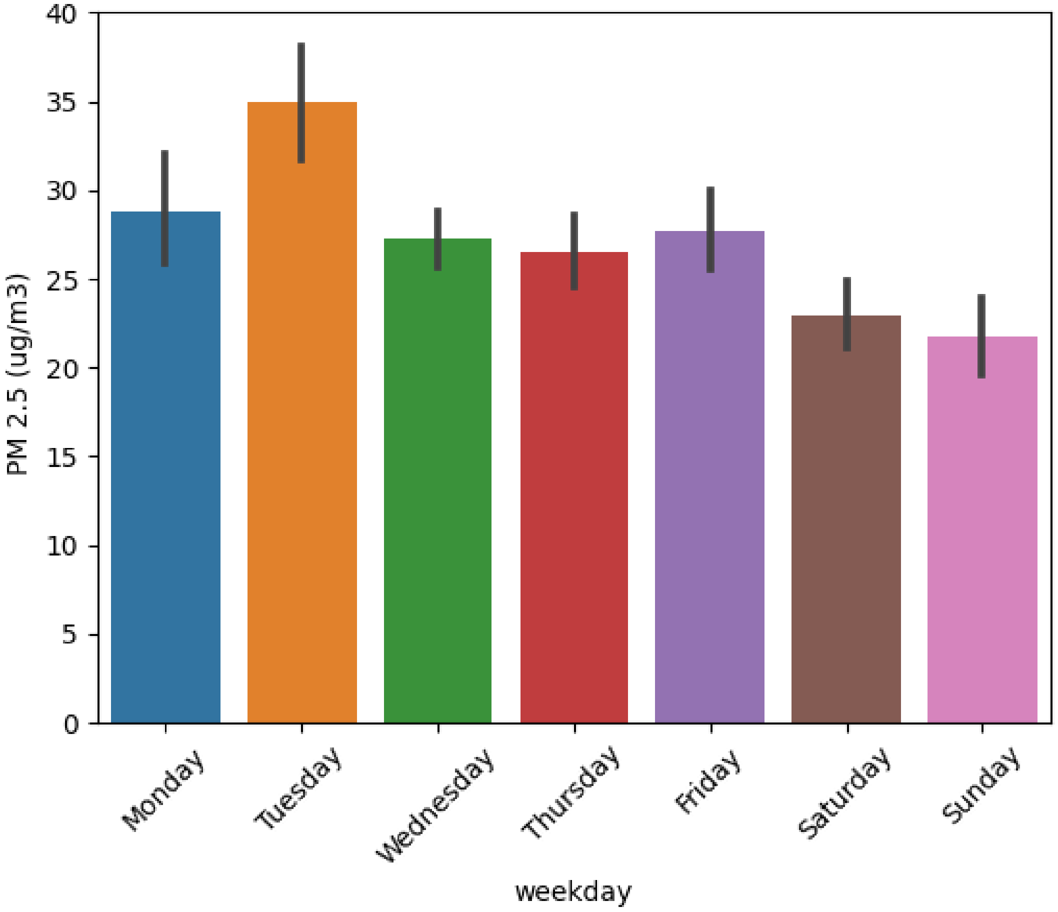
Observation - Hours when wind speed is low, the spectrum of possible values of PM2.5 is greater as compare to the other hours when the wind speed was large. - As wind speed decreases, PM 2.5 levels increases (i.e. inverse relationship) - Change in the RH is not causing a significant change in the PM2.5 values.

[30]:
```
print('THe mean PM2.5 on each week day:')
print((df1[['PM2.5','weekday']]).groupby('weekday')['PM2.5'].mean())

sns.barplot(data = df1, x = 'weekday',y = 'PM2.5').set_xticklabels(␣
 ↪['Monday','Tuesday','Wednesday','Thursday','Friday','Saturday',"Sunday"],␣
 ↪rotation = 45)
plt.ylabel('PM 2.5 (ug/m3) ')
plt.title('')
plt.show()
```

```
THe mean PM2.5 on each week day:
weekday
0.0    28.732333
1.0    34.988935
2.0    27.238912
3.0    26.502156
4.0    27.754471
```

```
5.0    22.963381
6.0    21.722593
Name: PM2.5, dtype: float64
```



Observation - The max average PM2.5 is observed on Tuesdays and least on Sundays.

[64]:
```
# statistical values
df1.describe()
```

[64]:
```
            PM2.5          RF          SR          WD          WS          RH
count  744.000000  744.000000  744.000000  744.000000  744.000000  744.000000  \
mean    26.879671    0.022513  163.405228  121.241855    0.923024   76.250484
std     13.145625    0.245130  201.770021   74.387613    0.404378   15.047548
min      1.000000    0.000000    8.400000    0.200000    0.100000   44.000000
25%     18.000000    0.000000    8.530000   61.082500    0.610000   63.475000
50%     25.000000    0.000000   37.790000  114.795000    0.840000   79.015000
75%     33.013294    0.000000  310.325000  165.387500    1.120000   90.125000
max     95.000000    5.120000  733.900000  355.700000    2.850000   93.680000
```

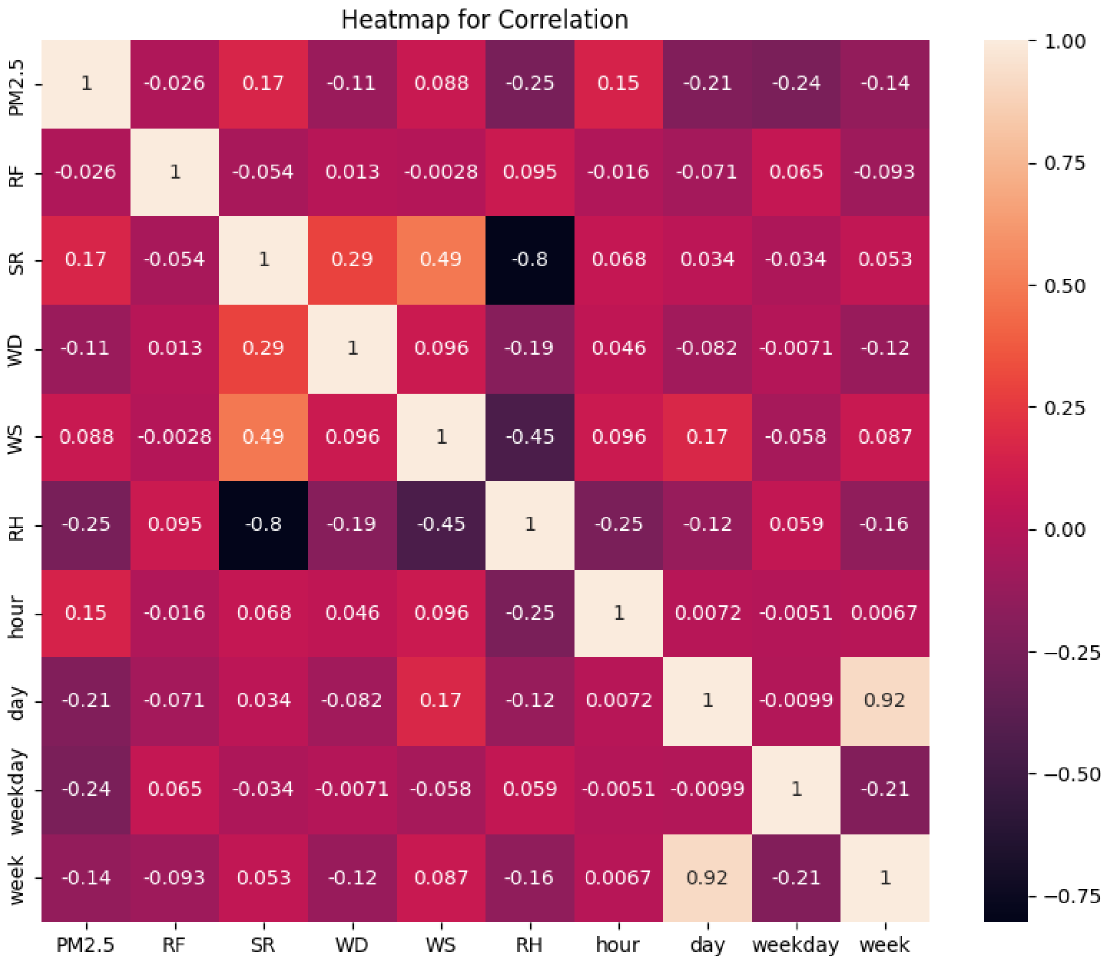
```
            hour         day     weekday        week
count 744.000000  744.000000  744.000000  744.000000
mean   12.498656   16.040323    3.057796    3.133065
std     6.924705    8.950198    2.081294    0.973281
min     1.000000    1.000000    0.000000    1.000000
25%     6.750000    8.000000    1.000000    2.000000
50%    12.500000   16.000000    3.000000    3.000000
75%    18.250000   24.000000    5.000000    4.000000
max    24.000000   31.000000    6.000000    4.000000
```

Observation - The median PM2.5 level, Rainfall, wind speed and Relative humidity is 25 ug/m3, 0 mm, 0.84 m/sec, and 79.01% respectively.

0.4.3 Correlation

[38]:

```
plt.figure(figsize = (10,8))
sns.heatmap(df1[['PM2.5', 'RF', 'SR', 'WD', 'WS', 'RH', 'hour', 'day',␣
 ↪'weekday', 'week']].corr(),annot = True)
plt.title('Heatmap for Correlation')
plt.show()
```



	PM2.5	RF	SR	WD	WS	RH	hour	day	weekday	week
PM2.5	1	-0.026	0.17	-0.11	0.088	-0.25	0.15	-0.21	-0.24	-0.14
RF	-0.026	1	-0.054	0.013	-0.0028	0.095	-0.016	-0.071	0.065	-0.093
SR	0.17	-0.054	1	0.29	0.49	-0.8	0.068	0.034	-0.034	0.053
WD	-0.11	0.013	0.29	1	0.096	-0.19	0.046	-0.082	-0.0071	-0.12
WS	0.088	-0.0028	0.49	0.096	1	-0.45	0.096	0.17	-0.058	0.087
RH	-0.25	0.095	-0.8	-0.19	-0.45	1	-0.25	-0.12	0.059	-0.16
hour	0.15	-0.016	0.068	0.046	0.096	-0.25	1	0.0072	-0.0051	0.0067
day	-0.21	-0.071	0.034	-0.082	0.17	-0.12	0.0072	1	-0.0099	0.92
weekday	-0.24	0.065	-0.034	-0.0071	-0.058	0.059	-0.0051	-0.0099	1	-0.21
week	-0.14	-0.093	0.053	-0.12	0.087	-0.16	0.0067	0.92	-0.21	1

Observation - Only relative humidity (RH) and solar radiation (SR) are showing a strong negative correlation. - Solar radiation and Wind speed are showing a significant positive correlation between them. - Wind direction and solar radiation have weak positve correlation between them.

This relationship was also observed in the scatter plots above.

[ ]: