Performing Analysis of Meteorological Data

June 3, 2021

1 Author : Pratik korat

2 Needed imports for analysis

[64]:
```
#Pandas for the reading csv files and numpy for numerical ops and matplotlib␣
 ↪for visualization of our data
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

2.0.1 1) Reading our dataset into Jupyter Notebook

[3]:
```
data = pd.read_csv("weatherHistory.csv", parse_dates = ['Formatted Date'] ,␣
 ↪index_col = ['Formatted Date'])
```

2.0.2 2) Inspecting our imported data

[4]:
```
data.head()
# This function will return top 5 records of our data to get sight of our␣
 ↪weather data
```

[4]:

Formatted Date	Summary	Precip Type	Temperature (C)	Apparent Temperature (C)	Humidity
2006-04-01 00:00:00+02:00	Partly Cloudy	rain	9.472222	7.388889	0.89
2006-04-01 01:00:00+02:00	Partly Cloudy	rain	9.355556	7.227778	0.86
2006-04-01 02:00:00+02:00	Mostly Cloudy	rain	9.377778	9.377778	0.89
2006-04-01 03:00:00+02:00	Partly Cloudy	rain	8.288889	5.944444	0.83
2006-04-01 04:00:00+02:00	Mostly Cloudy	rain	8.755556	6.977778	0.83

```
                           Wind Speed (km/h)  Wind Bearing (degrees)  \
Formatted Date
2006-04-01 00:00:00+02:00             14.1197                   251.0
2006-04-01 01:00:00+02:00             14.2646                   259.0
2006-04-01 02:00:00+02:00              3.9284                   204.0
2006-04-01 03:00:00+02:00             14.1036                   269.0
2006-04-01 04:00:00+02:00             11.0446                   259.0

                           Visibility (km)  Loud Cover  Pressure (millibars)  \
Formatted Date
2006-04-01 00:00:00+02:00          15.8263         0.0                1015.13
2006-04-01 01:00:00+02:00          15.8263         0.0                1015.63
2006-04-01 02:00:00+02:00          14.9569         0.0                1015.94
2006-04-01 03:00:00+02:00          15.8263         0.0                1016.41
2006-04-01 04:00:00+02:00          15.8263         0.0                1016.51

                                                Daily Summary
Formatted Date
2006-04-01 00:00:00+02:00  Partly cloudy throughout the day.
2006-04-01 01:00:00+02:00  Partly cloudy throughout the day.
2006-04-01 02:00:00+02:00  Partly cloudy throughout the day.
2006-04-01 03:00:00+02:00  Partly cloudy throughout the day.
2006-04-01 04:00:00+02:00  Partly cloudy throughout the day.
```

[5]:
```
# Inspecting features of our data
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 96453 entries, 2006-04-01 00:00:00+02:00 to 2016-09-09 23:00:00+02:00
Data columns (total 11 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Summary                   96453 non-null  object
 1   Precip Type               95936 non-null  object
 2   Temperature (C)           96453 non-null  float64
 3   Apparent Temperature (C)  96453 non-null  float64
 4   Humidity                  96453 non-null  float64
 5   Wind Speed (km/h)         96453 non-null  float64
 6   Wind Bearing (degrees)    96453 non-null  float64
 7   Visibility (km)           96453 non-null  float64
 8   Loud Cover                96453 non-null  float64
 9   Pressure (millibars)      96453 non-null  float64
 10  Daily Summary             96453 non-null  object
dtypes: float64(8), object(3)
memory usage: 8.8+ MB
```

By the above record we can see that most of our features get the float value and 3 of them get

string values

[6]:
```
# Describing the dataset
data.describe()
```

[6]:

	Temperature (C)	Apparent Temperature (C)	Humidity
count	96453.000000	96453.000000	96453.000000
mean	11.932678	10.855029	0.734899
std	9.551546	10.696847	0.195473
min	-21.822222	-27.716667	0.000000
25%	4.688889	2.311111	0.600000
50%	12.000000	12.000000	0.780000
75%	18.838889	18.838889	0.890000
max	39.905556	39.344444	1.000000

	Wind Speed (km/h)	Wind Bearing (degrees)	Visibility (km)	Loud Cover
count	96453.000000	96453.000000	96453.000000	96453.0
mean	10.810640	187.509232	10.347325	0.0
std	6.913571	107.383428	4.192123	0.0
min	0.000000	0.000000	0.000000	0.0
25%	5.828200	116.000000	8.339800	0.0
50%	9.965900	180.000000	10.046400	0.0
75%	14.135800	290.000000	14.812000	0.0
max	63.852600	359.000000	16.100000	0.0

	Pressure (millibars)
count	96453.000000
mean	1003.235956
std	116.969906
min	0.000000
25%	1011.900000
50%	1016.450000
75%	1021.090000
max	1046.380000

This gives us statistics of our data like mean , standard deviation , min value and max value

[14]:
```
# Now we have to check how many records are invalid in sense of whether any of␣
 ↪them having inapropriate value
# and any of them empty or not

# If there is null record or not
print(data.isnull().sum())
print("\n")
# If there is na value or not
print(data.isna().sum())
```

```
Summary                      0
Precip Type                517
```

```
Temperature (C)               0
Apparent Temperature (C)      0
Humidity                      0
Wind Speed (km/h)             0
Wind Bearing (degrees)        0
Visibility (km)               0
Loud Cover                    0
Pressure (millibars)          0
Daily Summary                 0
dtype: int64
```

```
Summary                       0
Precip Type                 517
Temperature (C)               0
Apparent Temperature (C)      0
Humidity                      0
Wind Speed (km/h)             0
Wind Bearing (degrees)        0
Visibility (km)               0
Loud Cover                    0
Pressure (millibars)          0
Daily Summary                 0
dtype: int64
```

Look likes we got some invalid data so we have to remove unneccesary data

[15]:
```
# New data
# Removing na data
data = data.dropna()
```

[17]:
```
data.isnull().sum()

# Looks like we get rid of our na values data and be having clean data for␣
 ↪further validation
```

[17]:
```
Summary                     0
Precip Type                 0
Temperature (C)             0
Apparent Temperature (C)    0
Humidity                    0
Wind Speed (km/h)           0
Wind Bearing (degrees)      0
Visibility (km)             0
Loud Cover                  0
Pressure (millibars)        0
Daily Summary               0
dtype: int64
```

[35]:
```
data.index = pd.to_datetime(data.index , utc =True)
```

[36]:
```
data.head()
```

[36]:

	Temperature (C)	Apparent Temperature (C)	\
2005-12-31 00:00:00+00:00	0.577778	-4.050000	
2006-01-31 00:00:00+00:00	-1.677942	-4.173708	
2006-02-28 00:00:00+00:00	-0.065394	-2.990716	
2006-03-31 00:00:00+00:00	4.559274	1.969780	
2006-04-30 00:00:00+00:00	12.635031	12.098827	

	Humidity	Wind Speed (km/h)	\
2005-12-31 00:00:00+00:00	0.890000	17.114300	
2006-01-31 00:00:00+00:00	0.834610	8.894211	
2006-02-28 00:00:00+00:00	0.843467	10.957008	
2006-03-31 00:00:00+00:00	0.778737	14.421488	
2006-04-30 00:00:00+00:00	0.728625	10.930670	

	Wind Bearing (degrees)	Visibility (km)	\
2005-12-31 00:00:00+00:00	140.000000	9.982000	
2006-01-31 00:00:00+00:00	161.018817	7.894064	
2006-02-28 00:00:00+00:00	197.886905	7.418794	
2006-03-31 00:00:00+00:00	195.059140	9.602590	
2006-04-30 00:00:00+00:00	191.877778	10.626760	

	Loud Cover	Pressure (millibars)	month	year
2005-12-31 00:00:00+00:00	0.0	1016.660000	12	2005
2006-01-31 00:00:00+00:00	0.0	1021.204960	1	2006
2006-02-28 00:00:00+00:00	0.0	995.183914	2	2006
2006-03-31 00:00:00+00:00	0.0	976.436263	3	2006
2006-04-30 00:00:00+00:00	0.0	1013.493694	4	2006

3 Using resampling function from pandas library

[21]:
```
data = data.resample('M').mean() # resample accroading to Month end ('M')
# This function will resample our dataframe by taking mean of every month in␣
 ↪our dataset
```

[27]:
```
data.head()
```

[27]:

Formatted Date	Temperature (C)	Apparent Temperature (C)	\
2005-12-31 00:00:00+00:00	0.577778	-4.050000	
2006-01-31 00:00:00+00:00	-1.677942	-4.173708	
2006-02-28 00:00:00+00:00	-0.065394	-2.990716	
2006-03-31 00:00:00+00:00	4.559274	1.969780	
2006-04-30 00:00:00+00:00	12.635031	12.098827	

```
                           Humidity  Wind Speed (km/h)  \
Formatted Date
2005-12-31 00:00:00+00:00  0.890000          17.114300
2006-01-31 00:00:00+00:00  0.834610           8.894211
2006-02-28 00:00:00+00:00  0.843467          10.957008
2006-03-31 00:00:00+00:00  0.778737          14.421488
2006-04-30 00:00:00+00:00  0.728625          10.930670

                           Wind Bearing (degrees)  Visibility (km)  \
Formatted Date
2005-12-31 00:00:00+00:00              140.000000         9.982000
2006-01-31 00:00:00+00:00              161.018817         7.894064
2006-02-28 00:00:00+00:00              197.886905         7.418794
2006-03-31 00:00:00+00:00              195.059140         9.602590
2006-04-30 00:00:00+00:00              191.877778        10.626760

                           Loud Cover  Pressure (millibars)
Formatted Date
2005-12-31 00:00:00+00:00         0.0           1016.660000
2006-01-31 00:00:00+00:00         0.0           1021.204960
2006-02-28 00:00:00+00:00         0.0            995.183914
2006-03-31 00:00:00+00:00         0.0            976.436263
2006-04-30 00:00:00+00:00         0.0           1013.493694
```

[28]:
```
data.tail()
```

[28]:
```
                           Temperature (C)  Apparent Temperature (C)  \
Formatted Date
2016-08-31 00:00:00+00:00        21.420296                 21.383094
2016-09-30 00:00:00+00:00        18.467924                 18.355833
2016-10-31 00:00:00+00:00        10.593141                  9.825775
2016-11-30 00:00:00+00:00         5.158800                  2.860089
2016-12-31 00:00:00+00:00         1.239158                 -2.017272

                           Humidity  Wind Speed (km/h)  \
Formatted Date
2016-08-31 00:00:00+00:00  0.674046           9.151378
2016-09-30 00:00:00+00:00  0.688833           6.849029
2016-10-31 00:00:00+00:00  0.827951          11.075846
2016-11-30 00:00:00+00:00  0.848847          10.507636
2016-12-31 00:00:00+00:00  0.887981          11.024860

                           Wind Bearing (degrees)  Visibility (km)  \
Formatted Date
2016-08-31 00:00:00+00:00              184.563172        13.948140
2016-09-30 00:00:00+00:00              177.738889        13.723260
```

```
2016-10-31 00:00:00+00:00              206.046914         9.208206
2016-11-30 00:00:00+00:00              163.690511         8.725824
2016-12-31 00:00:00+00:00              179.064603         7.460627

                           Loud Cover  Pressure (millibars)
Formatted Date
2016-08-31 00:00:00+00:00         0.0           1018.026398
2016-09-30 00:00:00+00:00         0.0           1017.969736
2016-10-31 00:00:00+00:00         0.0           1017.725457
2016-11-30 00:00:00+00:00         0.0           1019.215737
2016-12-31 00:00:00+00:00         0.0           1019.946339
```

[29]:
```
#Here we have got the data from month to month starting from december of 2005␣
↪to end of 2016
```

[32]:
```
# Adding addtional data for visualization
data['month'] = data.index.month
data['year'] = data.index.year
data.index = data.index.date
```

[33]:
```
data.head()
```

[33]:
```
            Temperature (C)  Apparent Temperature (C)  Humidity  \
2005-12-31         0.577778                 -4.050000  0.890000
2006-01-31        -1.677942                 -4.173708  0.834610
2006-02-28        -0.065394                 -2.990716  0.843467
2006-03-31         4.559274                  1.969780  0.778737
2006-04-30        12.635031                 12.098827  0.728625

            Wind Speed (km/h)  Wind Bearing (degrees)  Visibility (km)  \
2005-12-31          17.114300              140.000000         9.982000
2006-01-31           8.894211              161.018817         7.894064
2006-02-28          10.957008              197.886905         7.418794
2006-03-31          14.421488              195.059140         9.602590
2006-04-30          10.930670              191.877778        10.626760

            Loud Cover  Pressure (millibars)  month  year
2005-12-31         0.0           1016.660000     12  2005
2006-01-31         0.0           1021.204960      1  2006
2006-02-28         0.0            995.183914      2  2006
2006-03-31         0.0            976.436263      3  2006
2006-04-30         0.0           1013.493694      4  2006
```

[37]:
```
# YEar wise sampling
data_yearwise = data.resample("Y").mean()
```

[39]:
```
data_yearwise
```

[39]:

	Temperature (C)	Apparent Temperature (C)	\
2005-12-31 00:00:00+00:00	0.577778	-4.050000	
2006-12-31 00:00:00+00:00	11.157677	10.091090	
2007-12-31 00:00:00+00:00	12.087294	11.008692	
2008-12-31 00:00:00+00:00	12.139005	11.029242	
2009-12-31 00:00:00+00:00	12.218067	11.008698	
2010-12-31 00:00:00+00:00	11.125243	10.009624	
2011-12-31 00:00:00+00:00	11.460123	10.511277	
2012-12-31 00:00:00+00:00	11.941184	10.696427	
2013-12-31 00:00:00+00:00	11.893297	10.770698	
2014-12-31 00:00:00+00:00	12.492318	11.585657	
2015-12-31 00:00:00+00:00	12.253649	11.275878	
2016-12-31 00:00:00+00:00	12.032331	10.889949	

	Humidity	Wind Speed (km/h)	\
2005-12-31 00:00:00+00:00	0.890000	17.114300	
2006-12-31 00:00:00+00:00	0.767717	10.199795	
2007-12-31 00:00:00+00:00	0.690331	10.833914	
2008-12-31 00:00:00+00:00	0.701208	11.298354	
2009-12-31 00:00:00+00:00	0.707834	11.521911	
2010-12-31 00:00:00+00:00	0.797404	11.033093	
2011-12-31 00:00:00+00:00	0.736254	9.900793	
2012-12-31 00:00:00+00:00	0.689626	11.255930	
2013-12-31 00:00:00+00:00	0.754963	10.977535	
2014-12-31 00:00:00+00:00	0.748870	10.504392	
2015-12-31 00:00:00+00:00	0.732399	10.742030	
2016-12-31 00:00:00+00:00	0.763356	10.691133	

	Wind Bearing (degrees)	Visibility (km)	\
2005-12-31 00:00:00+00:00	140.000000	9.982000	
2006-12-31 00:00:00+00:00	189.383810	9.744841	
2007-12-31 00:00:00+00:00	194.206888	10.405626	
2008-12-31 00:00:00+00:00	193.897882	10.266580	
2009-12-31 00:00:00+00:00	180.557158	10.000805	
2010-12-31 00:00:00+00:00	191.907332	9.122681	
2011-12-31 00:00:00+00:00	189.262773	9.490687	
2012-12-31 00:00:00+00:00	187.365760	10.348863	
2013-12-31 00:00:00+00:00	187.972975	10.883147	
2014-12-31 00:00:00+00:00	178.066042	11.320014	
2015-12-31 00:00:00+00:00	184.157350	10.907546	
2016-12-31 00:00:00+00:00	186.365187	11.390070	

	Loud Cover	Pressure (millibars)	month	year
2005-12-31 00:00:00+00:00	0.0	1016.660000	12.0	2005.0
2006-12-31 00:00:00+00:00	0.0	992.468348	6.5	2006.0
2007-12-31 00:00:00+00:00	0.0	1001.776882	6.5	2007.0
2008-12-31 00:00:00+00:00	0.0	1007.788840	6.5	2008.0

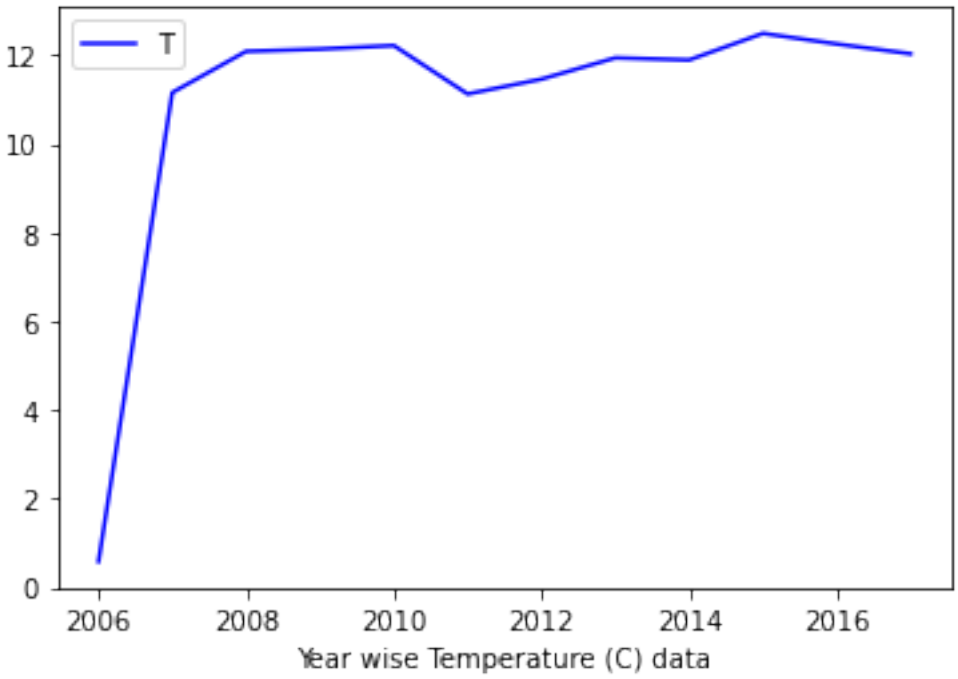
```
2009-12-31 00:00:00+00:00         0.0          1002.735507    6.5  2009.0
2010-12-31 00:00:00+00:00         0.0          1004.896113    6.5  2010.0
2011-12-31 00:00:00+00:00         0.0          1014.195619    6.5  2011.0
2012-12-31 00:00:00+00:00         0.0           999.072946    6.5  2012.0
2013-12-31 00:00:00+00:00         0.0          1004.380017    6.5  2013.0
2014-12-31 00:00:00+00:00         0.0           987.813694    6.5  2014.0
2015-12-31 00:00:00+00:00         0.0          1005.369326    6.5  2015.0
2016-12-31 00:00:00+00:00         0.0          1014.897484    6.5  2016.0
```

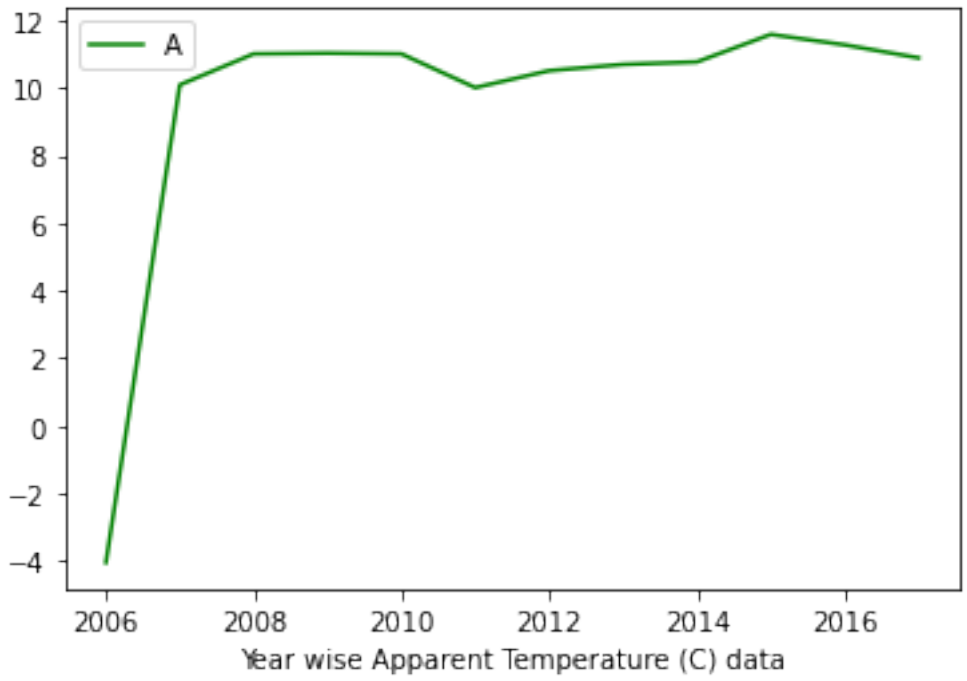
4 Visualization

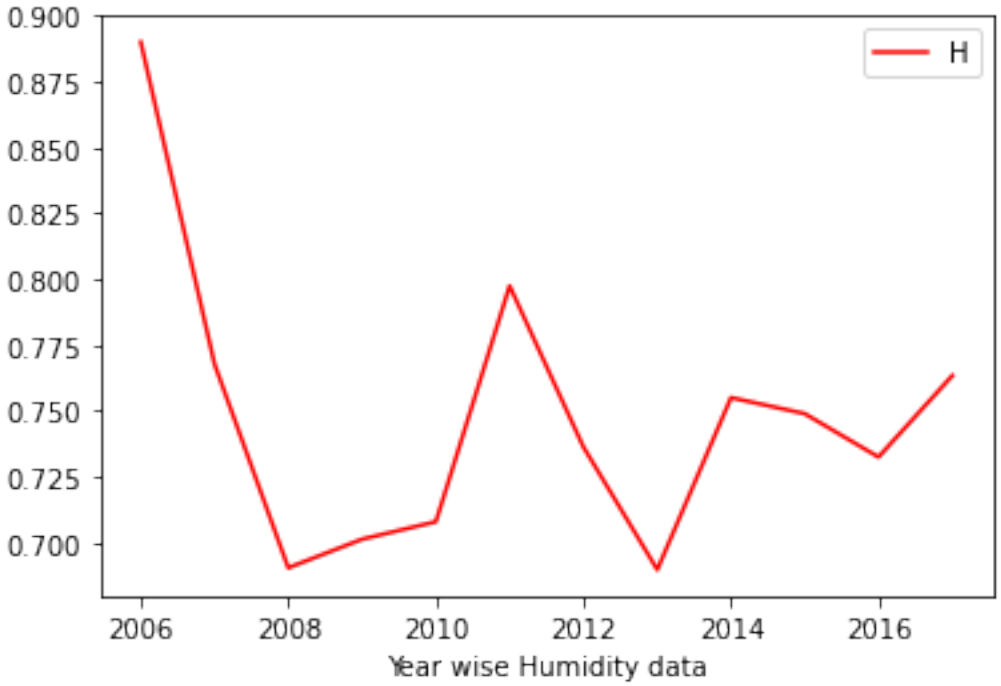
[40]:
```
# 1) line plot to get the hint how this data varying  from year to year
```

[58]:
```
def line_plotting_features():
    colors = ["blue", "green","red","cyan", "magenta","yellow","black" , "blue"]
    for i,column in enumerate(data_yearwise.columns[:-2]):
        plt.plot(data_yearwise[column] , color = colors[i])
        plt.legend(column)
        plt.xlabel(f"Year wise {column} data")
        plt.show()
```

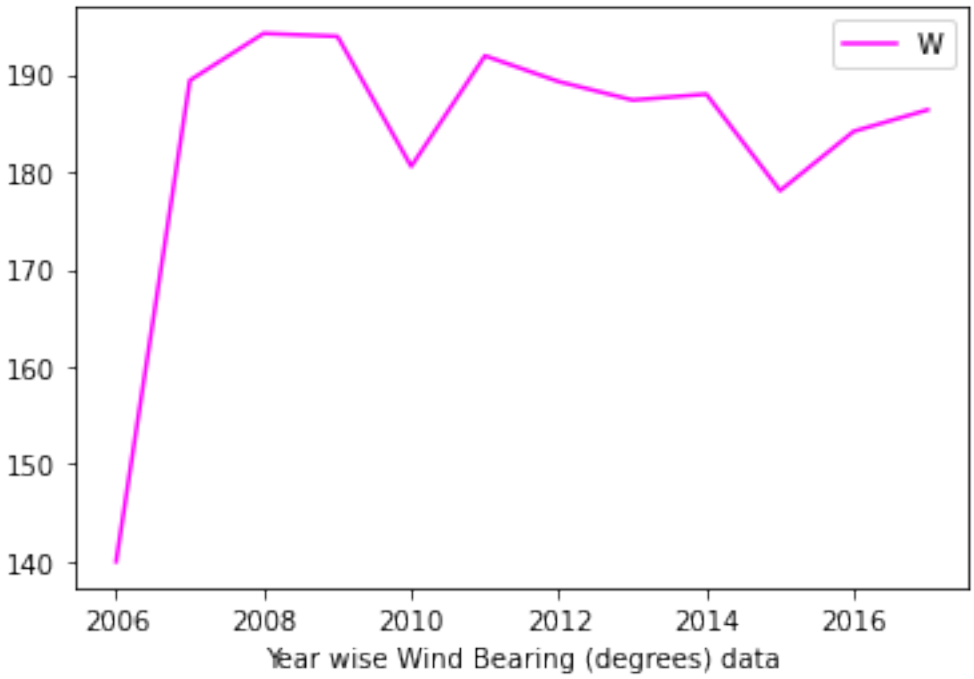
[60]:
```
# Time series visualization , i mean how they changes over the times
line_plotting_features()
```



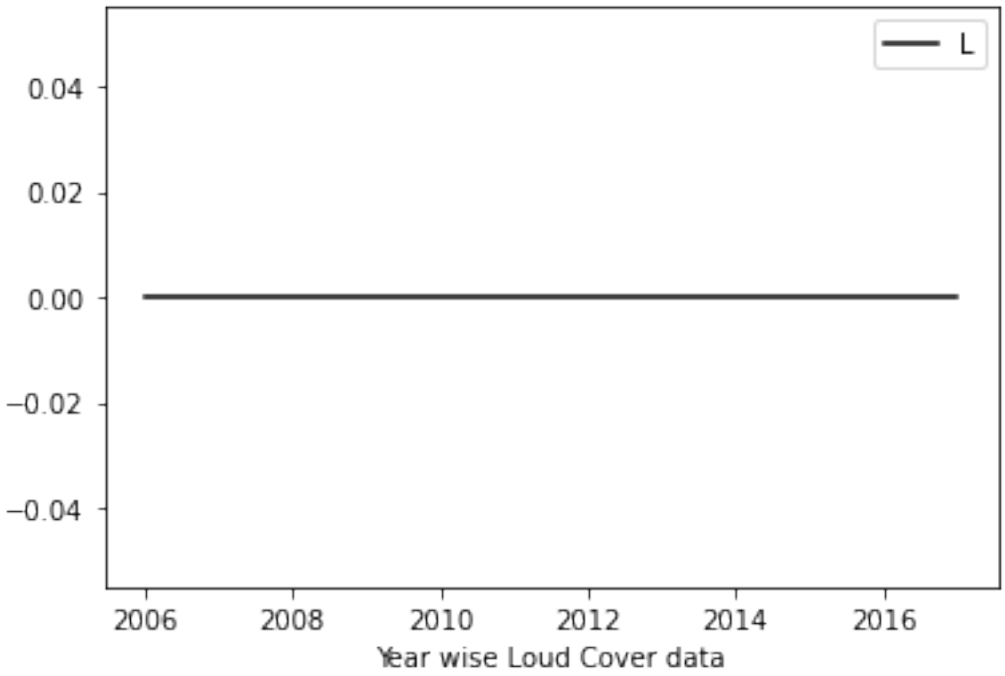

A
12
10
8
6
4
2
0
−2
−4
2006
2008
2010
2012
2014
2016
Year wise Apparent Temperature (C) data

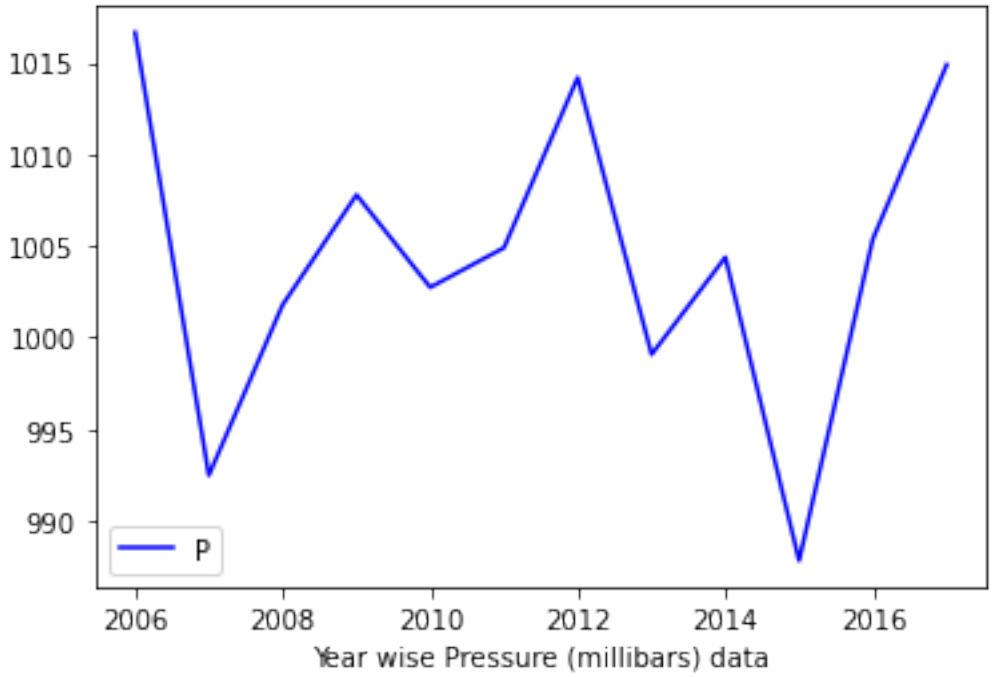

H
0.900
0.875
0.850
0.825
0.800
0.775
0.750
0.725
0.700
2006
2008
2010
2012
2014
2016
Year wise Humidity data


W
17
16
15
14
13
12
11
10
2006
2008
2010
2012
2014
2016
Year wise Wind Speed (km/h) data


W
190
180
170
160
150
140
2006
2008
2010
2012
2014
2016
Year wise Wind Bearing (degrees) data


V
11.5
11.0
10.5
10.0
9.5
2006
2008
2010
2012
2014
2016
Year wise Visibility (km) data

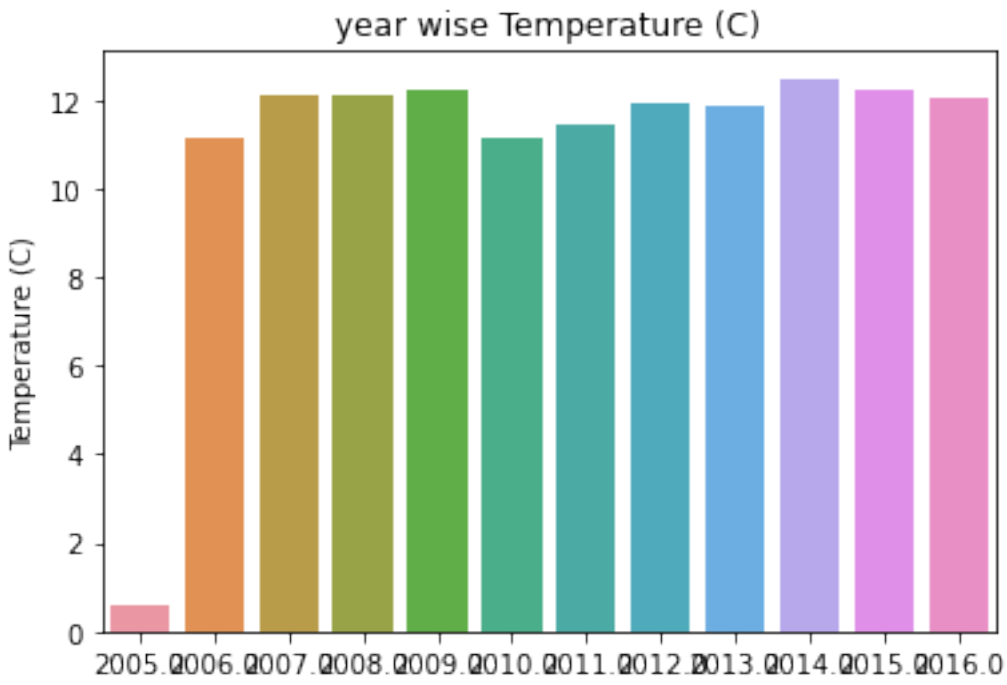

L
0.04
0.02
0.00
−0.02
−0.04
2006
2008
2010
2012
2014
2016
Year wise Loud Cover data

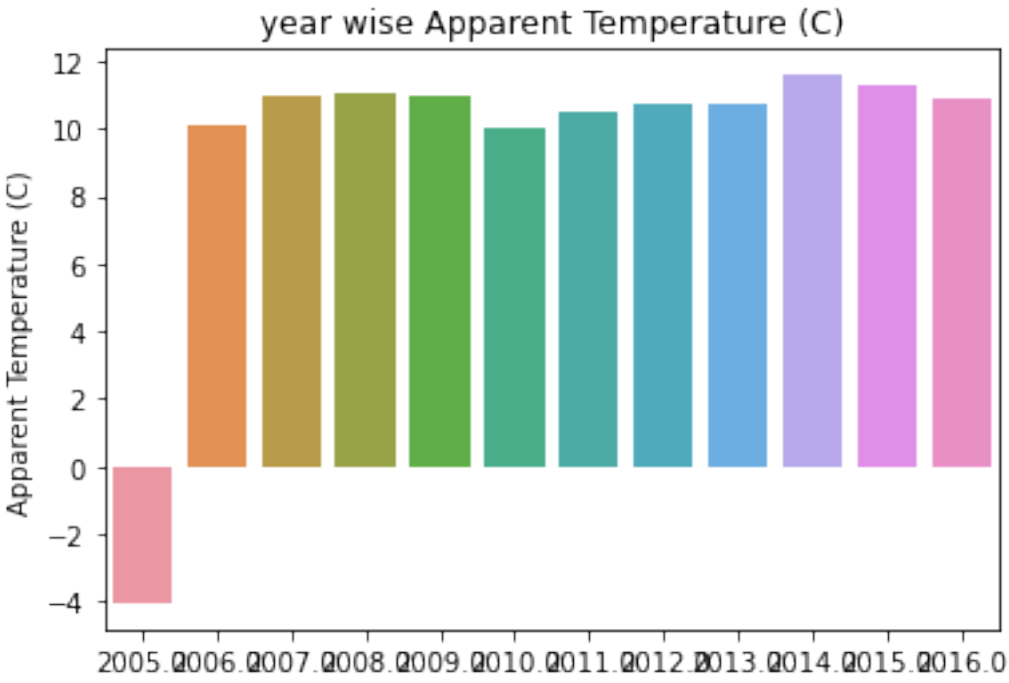


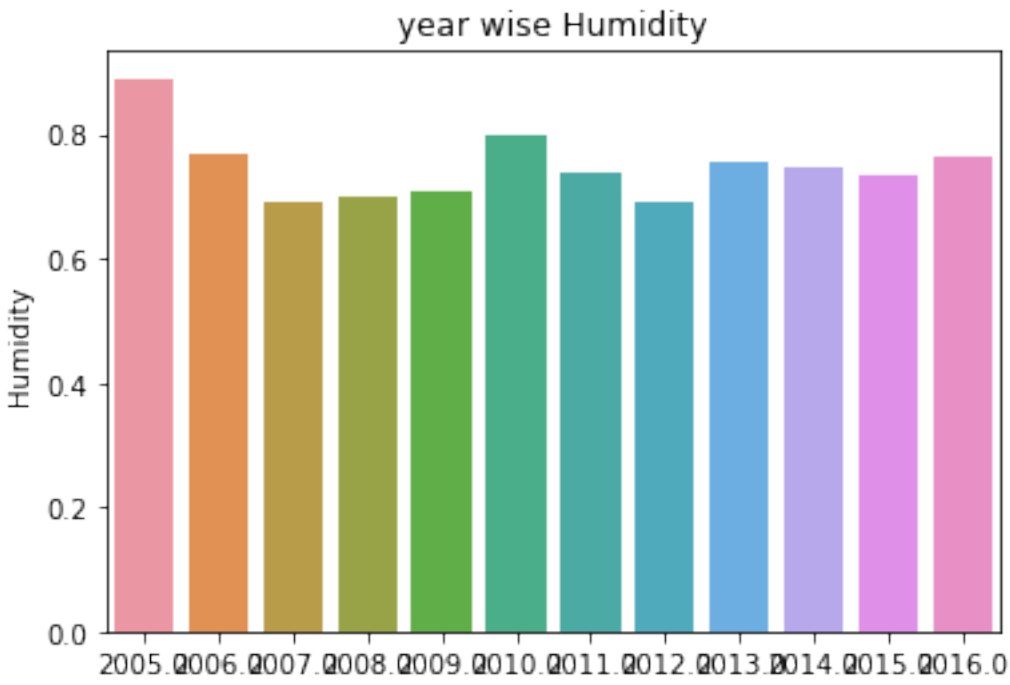
[61]:
```
# Bar plot for the same reason how they showing up in the given data
```

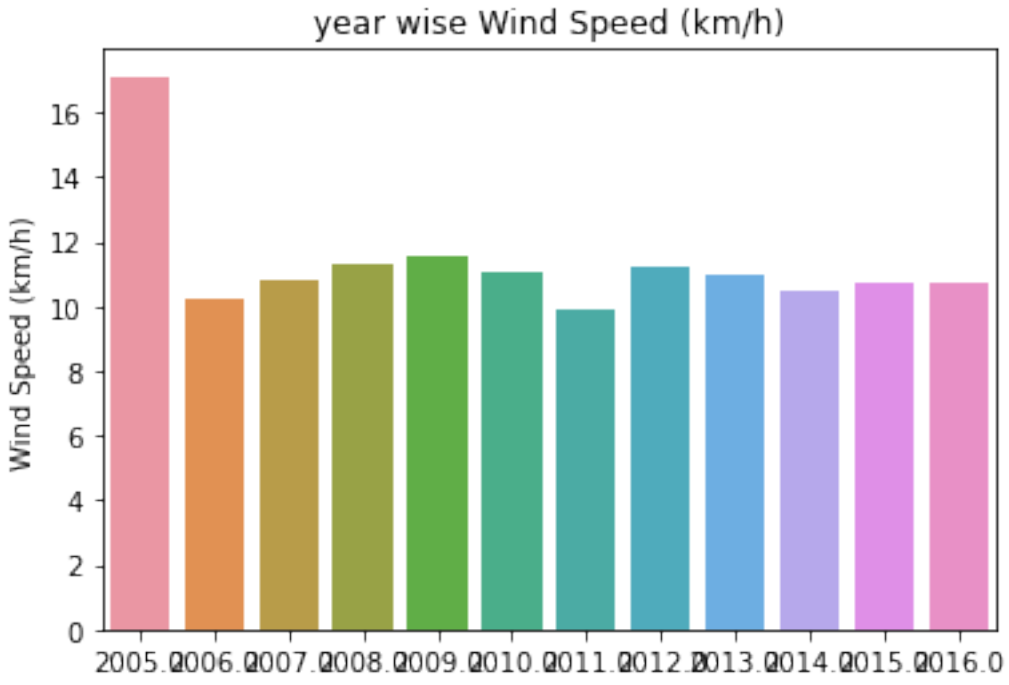
[65]:
```
def bar_plot_visualization():
    years = data_yearwise["year"].values
    for column in data_yearwise.columns[:-2]:
        sns.barplot(years , data_yearwise[column])
        plt.title(f"year wise {column}")
        plt.show()
```

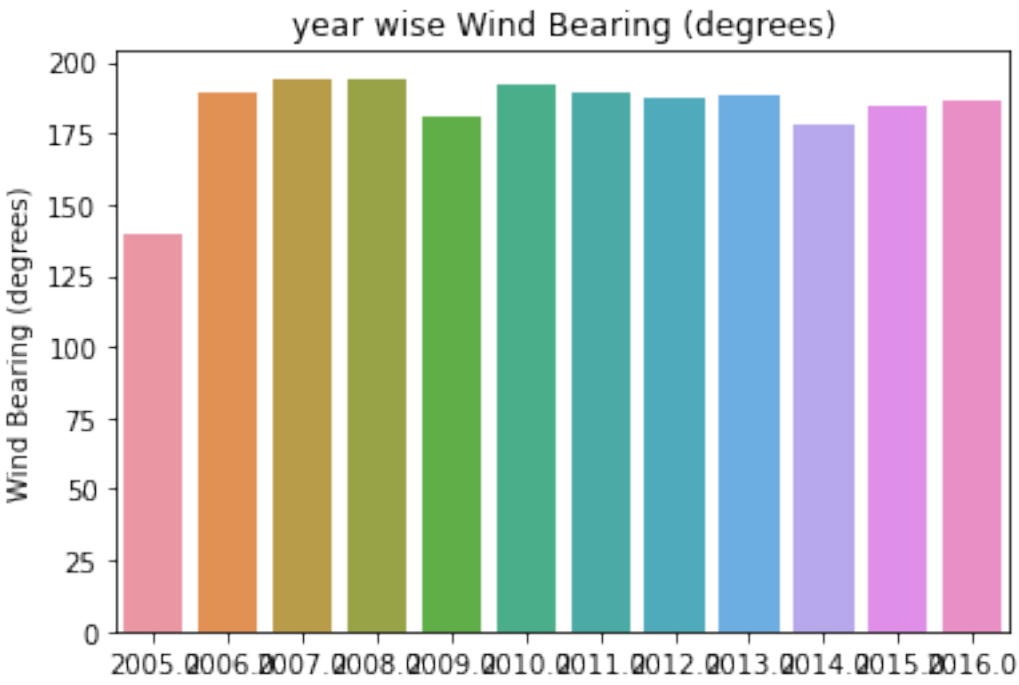
[67]:
```
bar_plot_visualization()
```

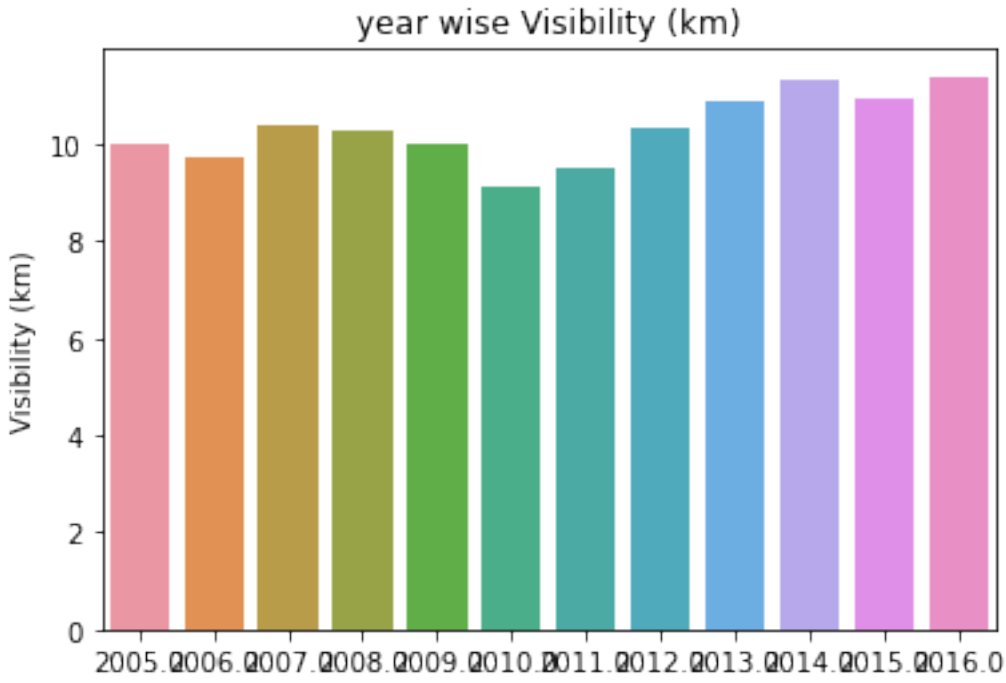

year wise Temperature (C)
Temperature (C)
0
2
4
6
8
10
12
2005.0
2006.0
2007.0
2008.0
2009.0
2010.0
2011.0
2012.0
2013.0
2014.0
2015.0
2016.0

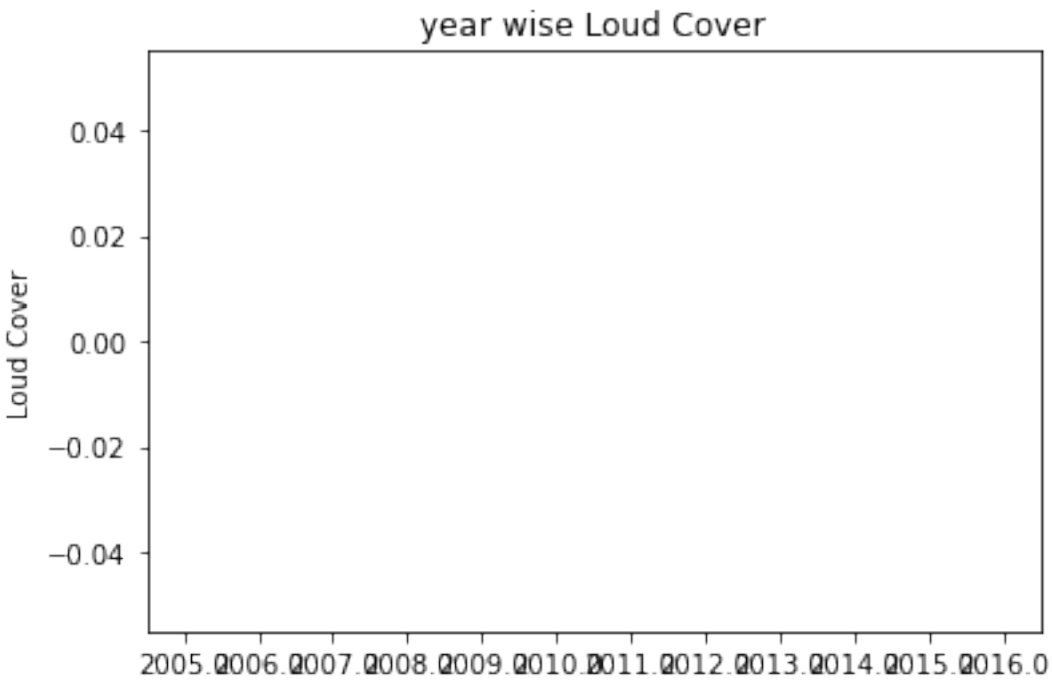

year wise Apparent Temperature (C)
Apparent Temperature (C)
−4
−2
0
2
4
6
8
10
12
2005.0
2006.0
2007.0
2008.0
2009.0
2010.0
2011.0
2012.0
2013.0
2014.0
2015.0
2016.0

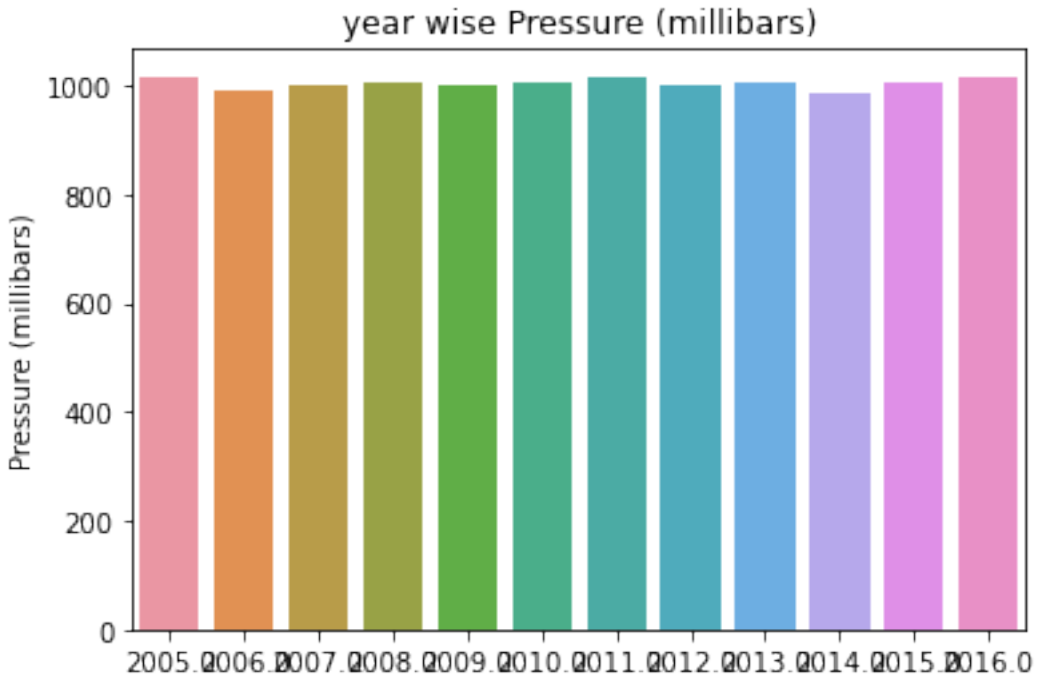

year wise Humidity
Humidity
0.0
0.2
0.4
0.6
0.8
2005.0
2006.0
2007.0
2008.0
2009.0
2010.0
2011.0
2012.0
2013.0
2014.0
2015.0
2016.0


year wise Wind Speed (km/h)
Wind Speed (km/h)
0
2
4
6
8
10
12
14
16
2005.0
2006.0
2007.0
2008.0
2009.0
2010.0
2011.0
2012.0
2013.0
2014.0
2015.0
2016.0


year wise Wind Bearing (degrees)
Wind Bearing (degrees)
200
175
150
125
100
75
50
25
0
2005.0
2006.0
2007.0
2008.0
2009.0
2010.0
2011.0
2012.0
2013.0
2014.0
2015.0
2016.0


year wise Visibility (km)
Visibility (km)
10
8
6
4
2
0
2005.0
2006.0
2007.0
2008.0
2009.0
2010.0
2011.0
2012.0
2013.0
2014.0
2015.0
2016.0

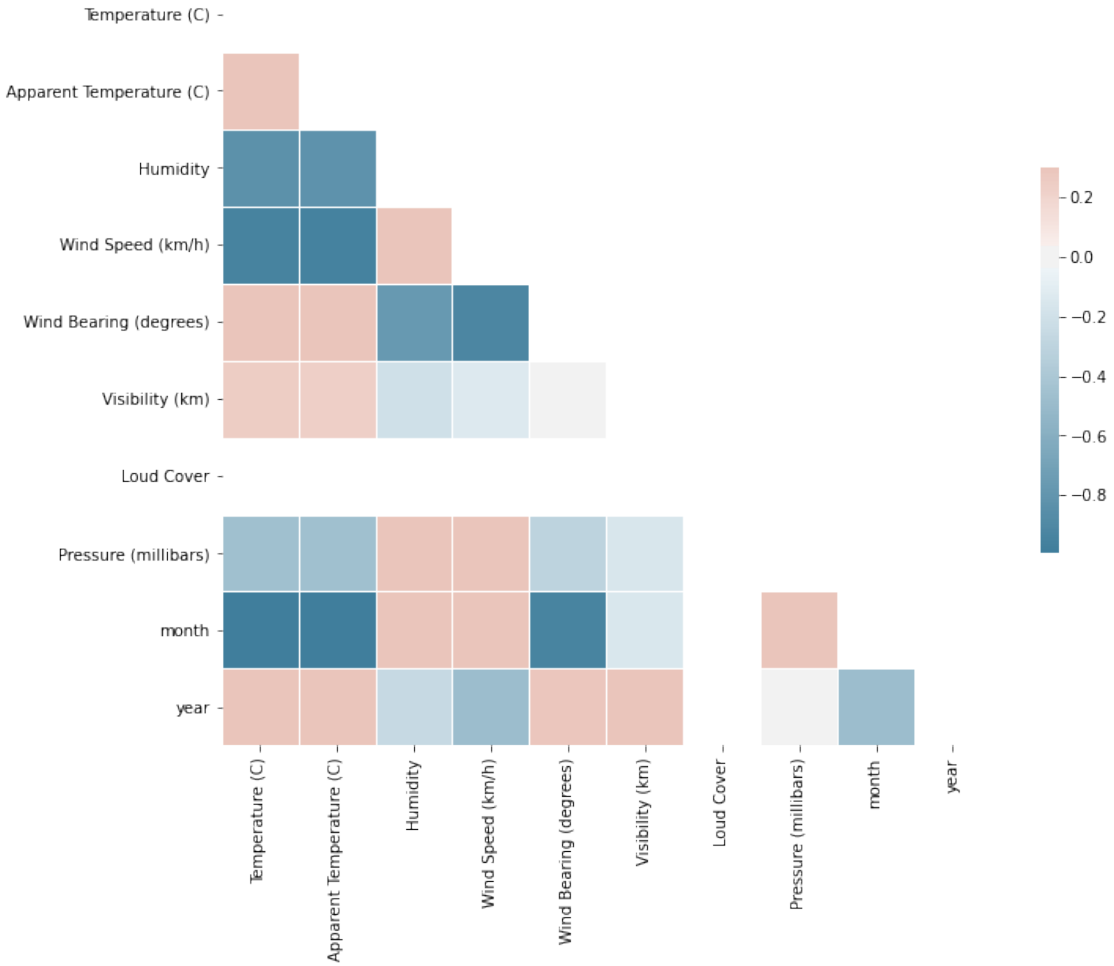




[68]:
```
# We'll have look at correlation for the all the features
```

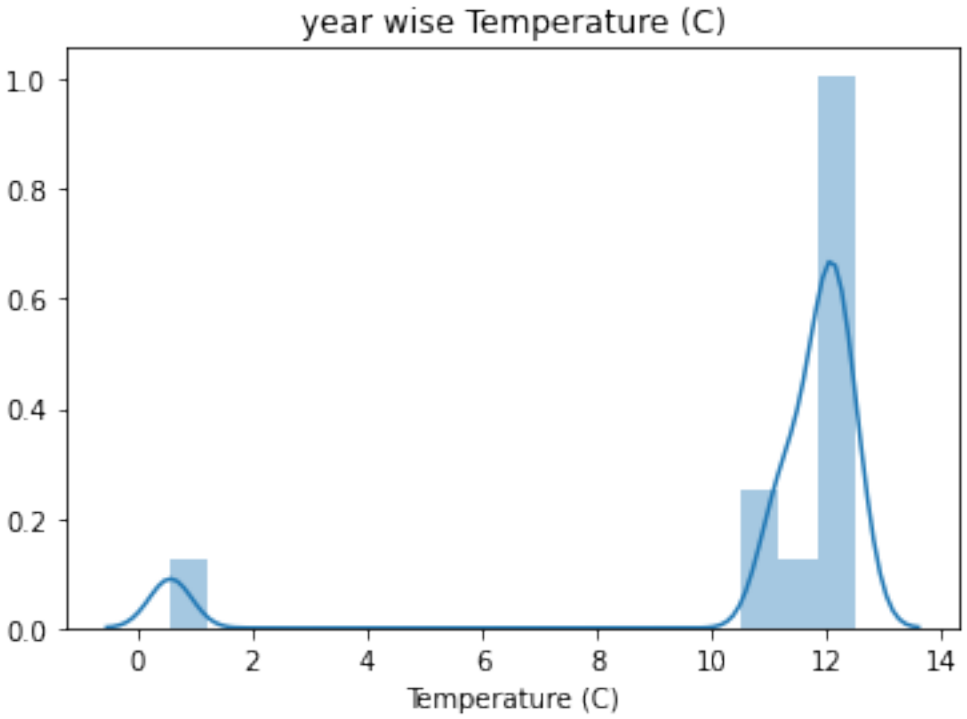
[69]:
```
# Compute the correlation matrix
corr = data_yearwise.corr()

# Generate a mask for the upper triangle
mask = np.triu(np.ones_like(corr, dtype=bool))

# Set up the matplotlib figure
f, ax = plt.subplots(figsize=(11, 9))

# Generate a custom diverging colormap
cmap = sns.diverging_palette(230, 20, as_cmap=True)

# Draw the heatmap with the mask and correct aspect ratio
sns.heatmap(corr, mask=mask, cmap=cmap, vmax=.3, center=0,
            square=True, linewidths=.5, cbar_kws={"shrink": .5})
```

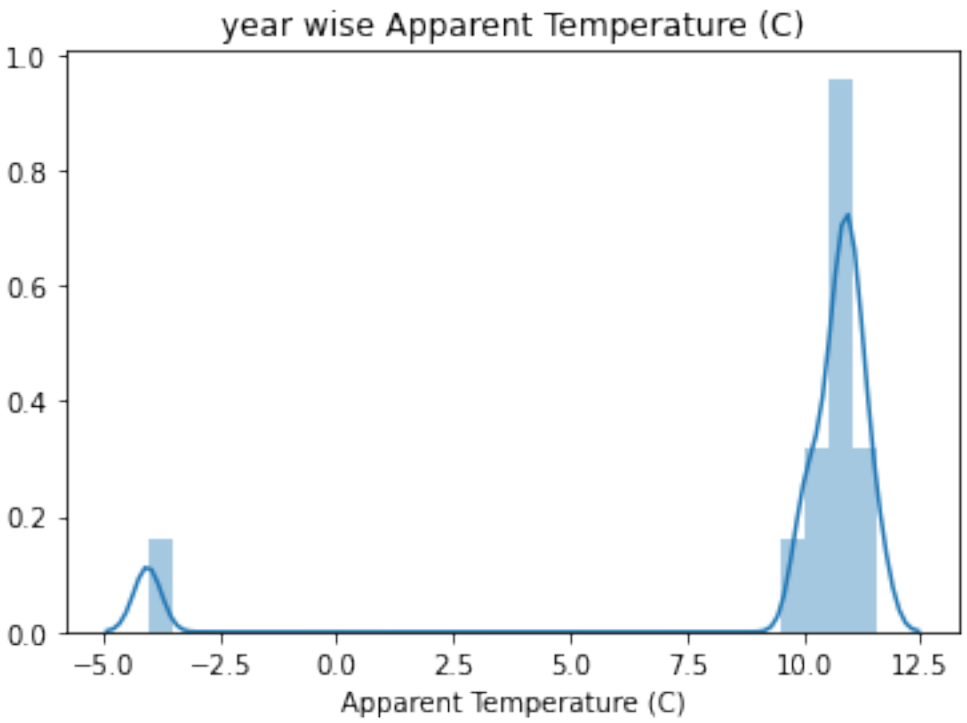
[69]: <AxesSubplot:>

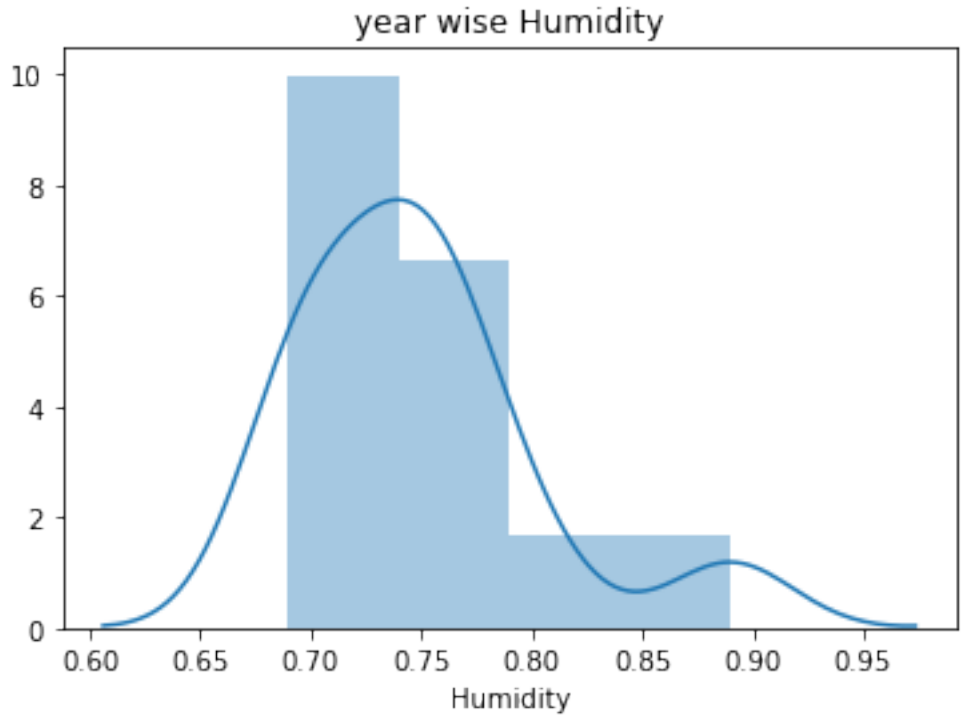


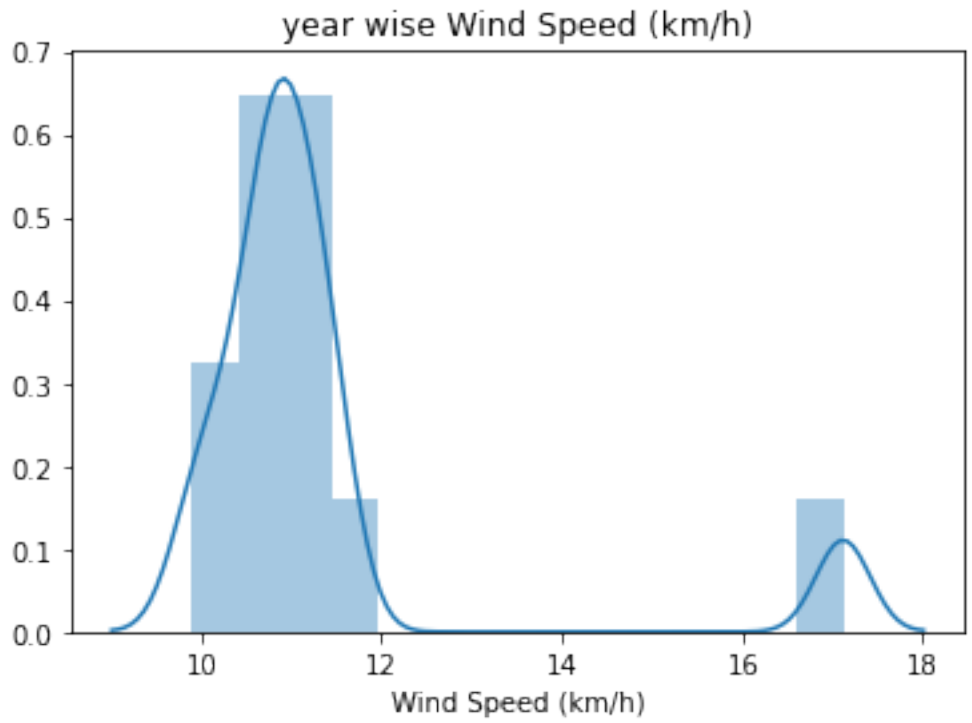
[72]:
```
#Distribution plot

def dist_plot_visualization():
    years = data_yearwise["year"].values
    for column in data_yearwise.columns[:-2]:
        sns.distplot(data_yearwise[column] , kde = True)
        plt.title(f"year wise {column}")
        plt.show()
```

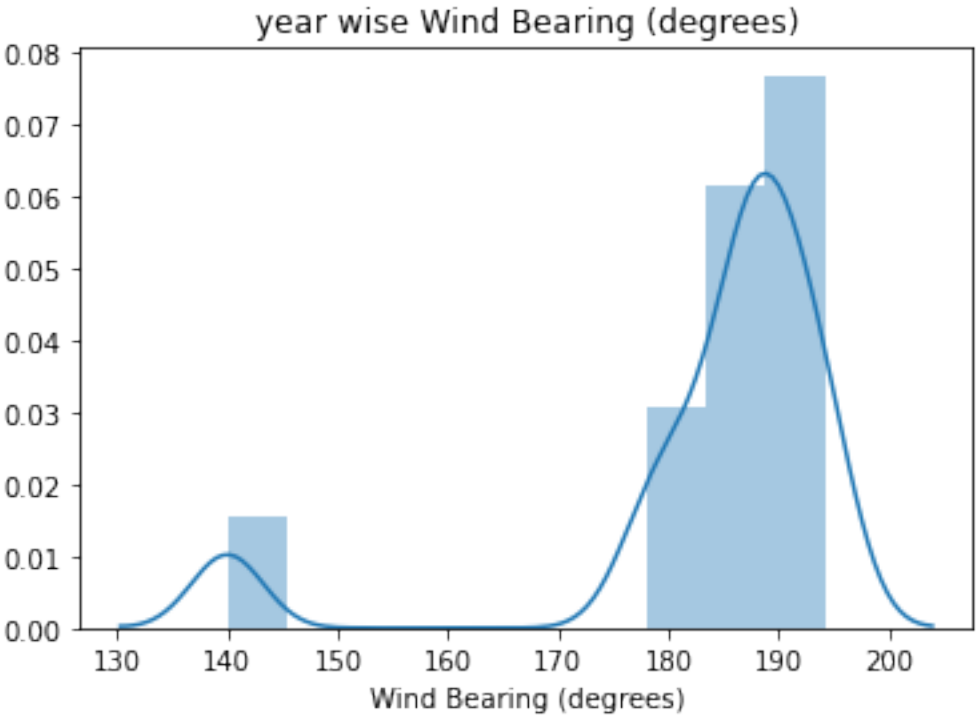
[74]:
```
dist_plot_visualization()
```

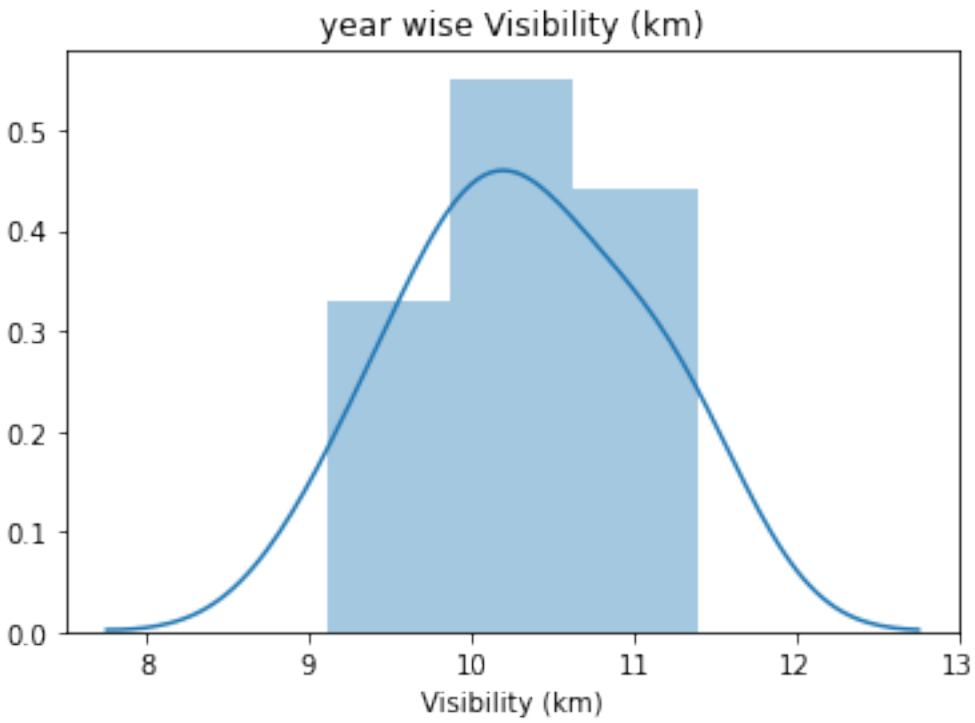



year wise Apparent Temperature (C)
1.0
0.8
0.6
0.4
0.2
0.0
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0
12.5
Apparent Temperature (C)

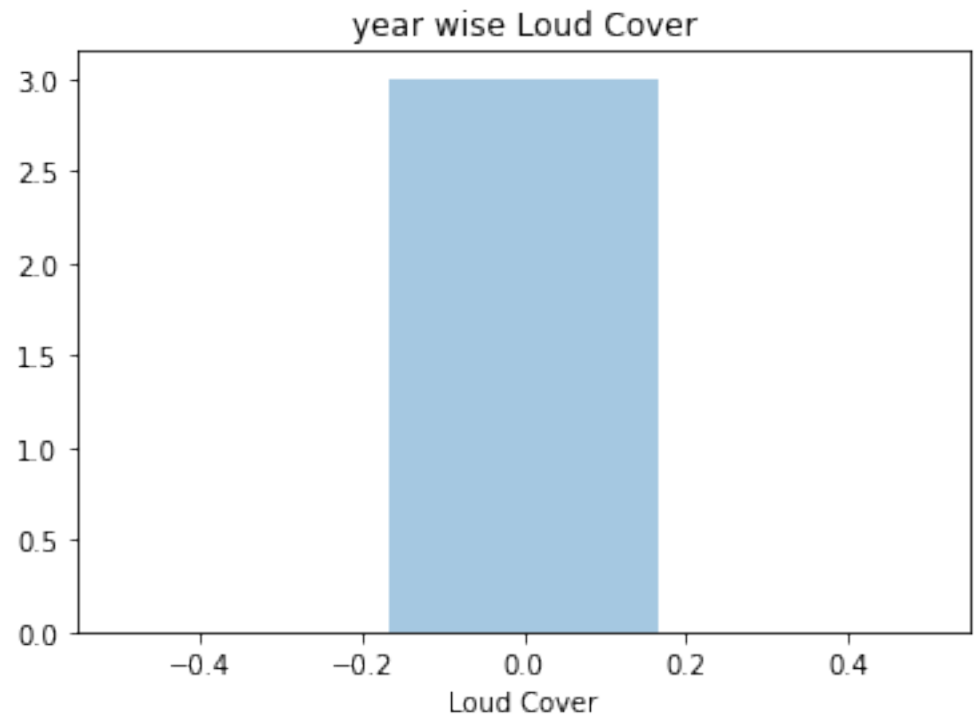

year wise Humidity
10
8
6
4
2
0
0.60
0.65
0.70
0.75
0.80
0.85
0.90
0.95
Humidity

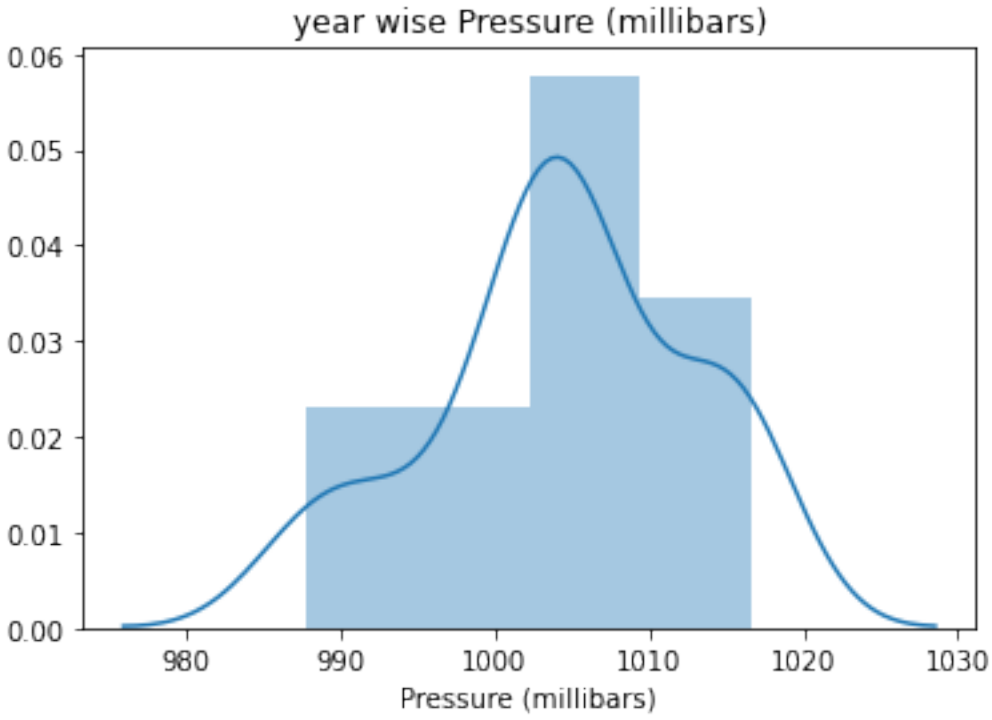

year wise Wind Speed (km/h)
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
10
12
14
16
18
Wind Speed (km/h)


year wise Wind Bearing (degrees)
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0.00
130
140
150
160
170
180
190
200
Wind Bearing (degrees)




```
C:\Users\prati\anaconda3\lib\site-packages\seaborn\distributions.py:288:
UserWarning: Data must have variance to compute a kernel density estimate.
  if set(self.variables) - {"x", "y"}:
```


year wise Loud Cover
3.0
2.5
2.0
1.5
1.0
0.5
0.0
−0.4
−0.2
0.0
0.2
0.4
Loud Cover


year wise Pressure (millibars)
0.06
0.05
0.04
0.03
0.02
0.01
0.00
980
990
1000
1010
1020
1030
Pressure (millibars)

[75]:
```
# Seems like we got rough distribution of some of the features
```

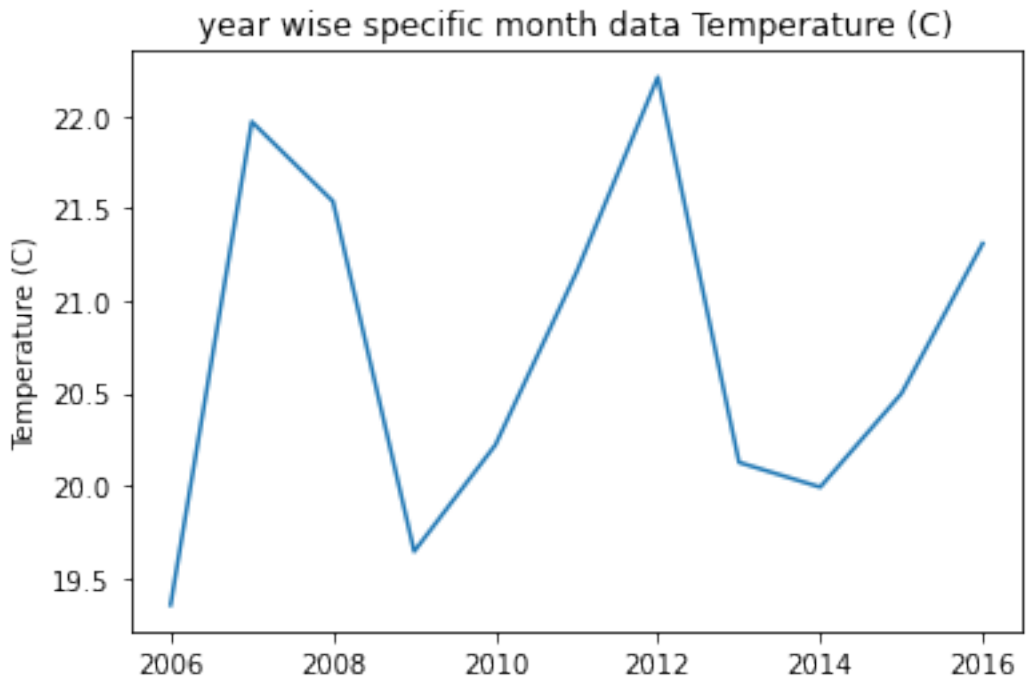
[77]:
```
data[data["month"] == 4]["Temperature (C)"]
```

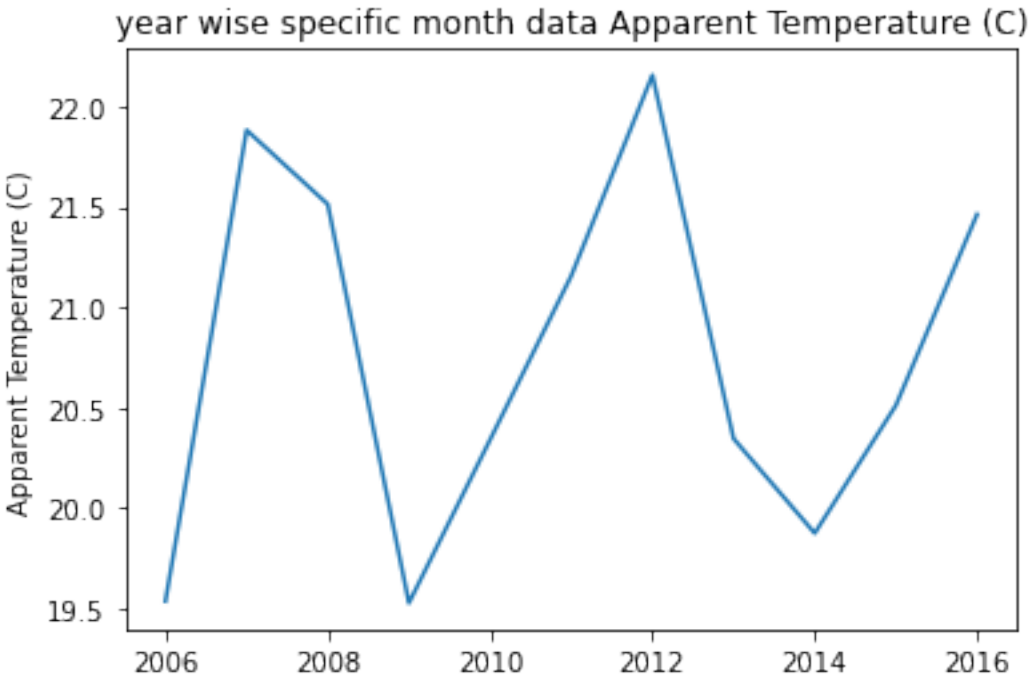
[77]:
```
2006-04-30 00:00:00+00:00    12.635031
2007-04-30 00:00:00+00:00    12.348071
2008-04-30 00:00:00+00:00    11.804622
2009-04-30 00:00:00+00:00    14.559159
2010-04-30 00:00:00+00:00    12.194329
2011-04-30 00:00:00+00:00    13.378665
2012-04-30 00:00:00+00:00    12.639399
2013-04-30 00:00:00+00:00    13.014205
2014-04-30 00:00:00+00:00    12.978812
2015-04-30 00:00:00+00:00    11.612160
2016-04-30 00:00:00+00:00    13.345062
Name: Temperature (C), dtype: float64
```

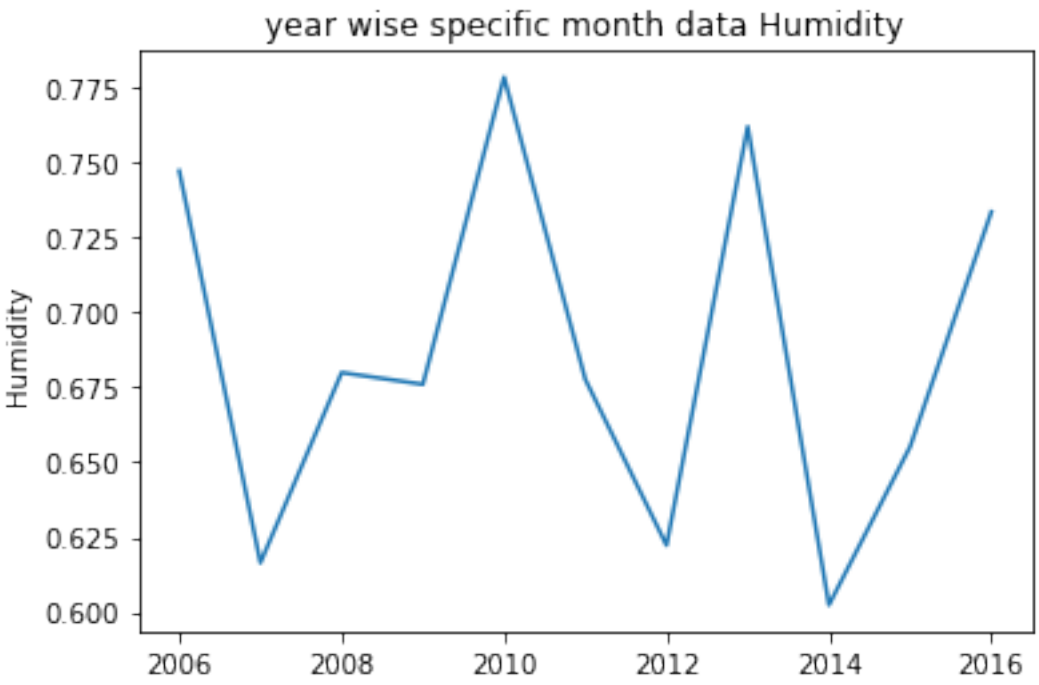
[79]:
```
# We'll be ploting about something specific month and that'll be june
```

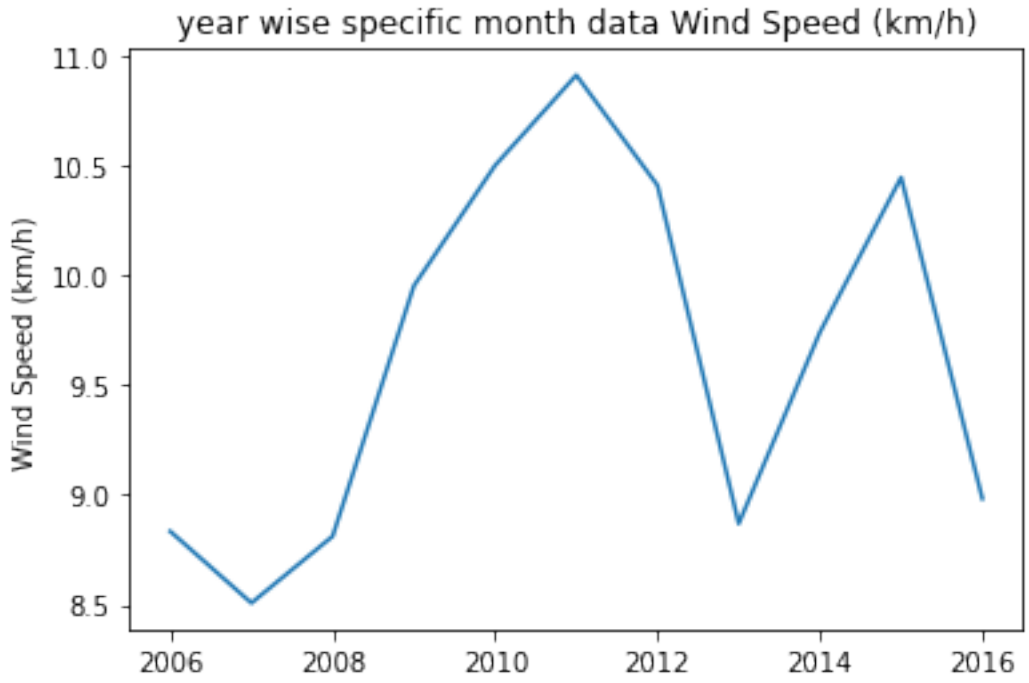
[107]:
```
# 1) line plot for it

def line_plot_visualization(month_indx):
    years = data_yearwise["year"].values
    years = years[1:]
    for column in data_yearwise.columns[:-2]:
        sns.lineplot(x = years , y = data[data["month"] == month_indx][column])
        plt.title(f"year wise specific month data {column}")
        plt.show()
```

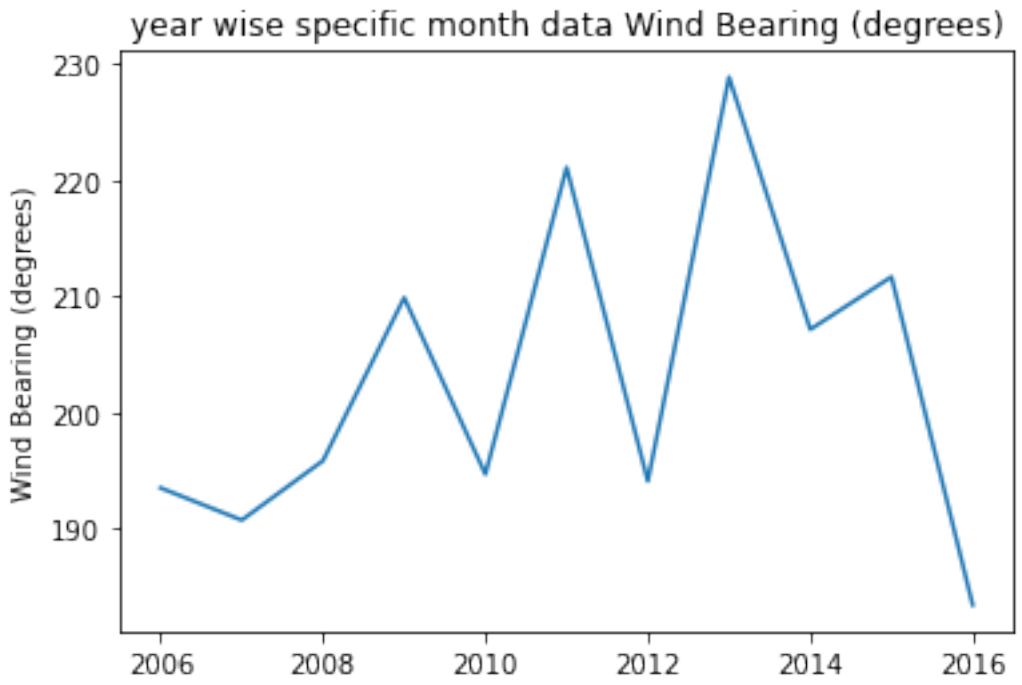
[108]:
```
line_plot_visualization(6)
```

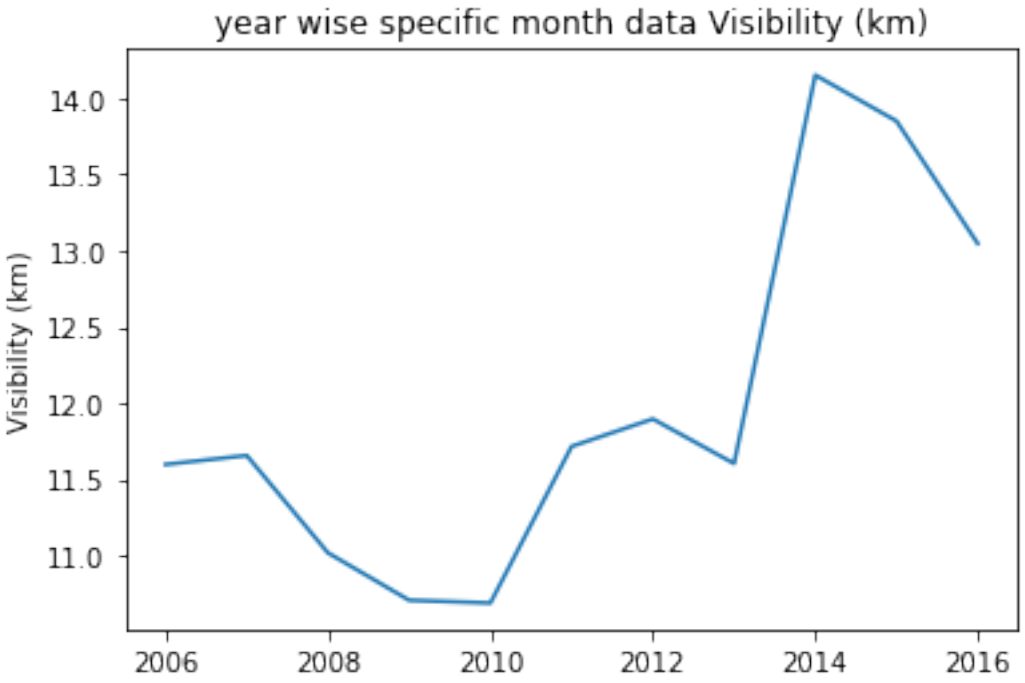

year wise specific month data Temperature (C)
Temperature (C)
22.0
21.5
21.0
20.5
20.0
19.5
2006
2008
2010
2012
2014
2016

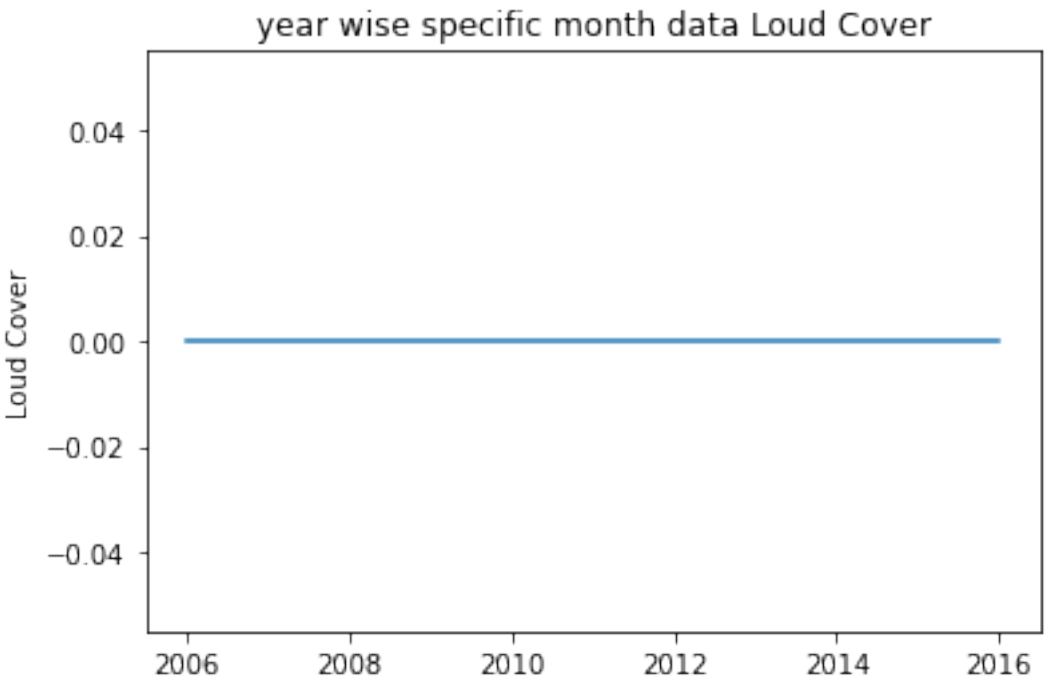

year wise specific month data Apparent Temperature (C)
Apparent Temperature (C)
22.0
21.5
21.0
20.5
20.0
19.5
2006
2008
2010
2012
2014
2016

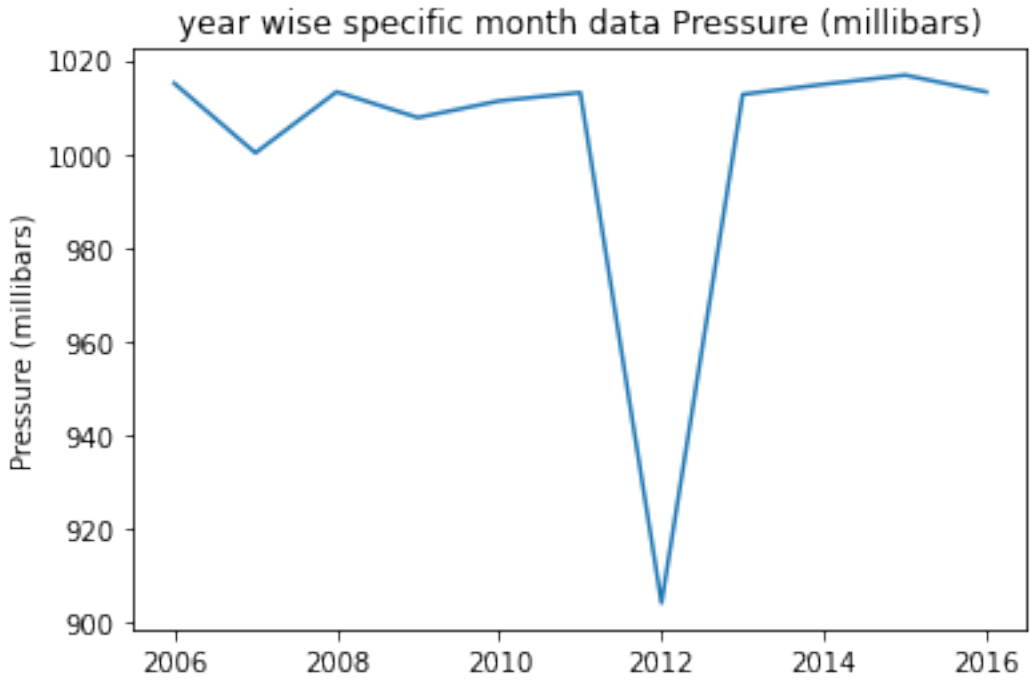

year wise specific month data Humidity
Humidity
0.775
0.750
0.725
0.700
0.675
0.650
0.625
0.600
2006
2008
2010
2012
2014
2016


year wise specific month data Wind Speed (km/h)
Wind Speed (km/h)
11.0
10.5
10.0
9.5
9.0
8.5
2006
2008
2010
2012
2014
2016


year wise specific month data Wind Bearing (degrees)
Wind Bearing (degrees)
230
220
210
200
190
2006
2008
2010
2012
2014
2016


year wise specific month data Visibility (km)
Visibility (km)
14.0
13.5
13.0
12.5
12.0
11.5
11.0
2006
2008
2010
2012
2014
2016


year wise specific month data Loud Cover
Loud Cover
0.04
0.02
0.00
−0.02
−0.04
2006
2008
2010
2012
2014
2016


year wise specific month data Pressure (millibars)
Pressure (millibars)
1020
1000
980
960
940
920
900
2006
2008
2010
2012
2014
2016

[ ]: