Indian Summer Weather Analysis

April 25, 2022

[10]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

[11]:
```
sd=pd.read_csv("indian_summer.csv")
```

[12]:
```
sd.head()
```

[12]:

	City	Date	tempmax	tempmin	temp	feelslikemax	feelslikemin
0	New Delhi	01-04-2021	34.0	19.0	27.1	31.6	19.0
1	New Delhi	02-04-2021	33.9	16.0	25.8	31.8	16.0
2	New Delhi	03-04-2021	34.8	14.6	26.0	32.2	14.6
3	New Delhi	04-04-2021	36.8	16.9	27.1	34.2	16.9
4	New Delhi	05-04-2021	38.8	21.0	29.9	37.1	21.0

	feelslike	dew	humidity	windspeed	winddir	sealevelpressure	cloudcover
0	26.1	3.1	22.60	22.8	272.9	1002.8	0.0
1	24.9	4.5	27.62	12.4	275.0	1006.2	0.0
2	25.1	1.3	23.18	16.5	127.5	1008.8	1.4
3	26.0	4.8	28.00	18.3	157.6	1009.5	2.6
4	28.9	8.1	28.85	13.5	100.4	1007.8	38.4

	visibility	sunrise	sunset	moonphase	conditions
0	3.1	01-04-2021 6:11	01-04-2021 18:39	0.60	Clear
1	3.5	02-04-2021 6:10	02-04-2021 18:39	0.65	Clear
2	3.5	03-04-2021 6:08	03-04-2021 18:40	0.70	Clear
3	3.2	04-04-2021 6:07	04-04-2021 18:40	0.76	Clear
4	3.1	05-04-2021 6:06	05-04-2021 18:41	0.81	Partially cloudy

	description
0	Clear conditions throughout the day.
1	Clear conditions throughout the day.
2	Clear conditions throughout the day.
3	Clear conditions throughout the day.
4	Partly cloudy throughout the day.

[13]:
```
sd.tail()
```

[13]:
```
            City        Date  tempmax  tempmin  temp  feelslikemax  \
13645  Hyderabad  26-06-2012     32.1     22.1  25.8          35.9
13646  Hyderabad  27-06-2012     31.8     21.1  25.5          33.3
13647  Hyderabad  28-06-2012     31.8     23.1  26.8          33.3
13648  Hyderabad  29-06-2012     32.8     23.1  26.7          35.1
13649  Hyderabad  30-06-2012     32.9     23.1  27.7          34.5

       feelslikemin  feelslike   dew  humidity  windspeed  winddir  \
13645          22.1       26.7  19.9     71.60       31.3    248.8
13646          21.1       26.1  19.0     68.40       29.5    262.4
13647          23.1       27.6  19.1     63.67       31.3    264.5
13648          23.1       27.5  19.5     65.54       27.7    265.1
13649          23.1       28.6  18.8     59.46       27.7    264.2

       sealevelpressure  cloudcover  visibility           sunrise  \
13645               NaN        85.3         4.6  26-06-2012 05:44
13646               NaN        67.9         5.5  27-06-2012 05:44
13647               NaN        69.5         5.6  28-06-2012 05:44
13648               NaN        85.0         5.6  29-06-2012 05:44
13649               NaN        64.9         5.6  30-06-2012 05:45

                  sunset  moonphase              conditions  \
13645  26-06-2012 18:53       0.21  Rain, Partially cloudy
13646  27-06-2012 18:54       0.27  Rain, Partially cloudy
13647  28-06-2012 18:54       0.33       Partially cloudy
13648  29-06-2012 18:54       0.38  Rain, Partially cloudy
13649  30-06-2012 18:54       0.43       Partially cloudy

                                          description
13645  Partly cloudy throughout the day with afternoo…
13646        Partly cloudy throughout the day with rain.
13647                  Partly cloudy throughout the day.
13648  Partly cloudy throughout the day with afternoo…
13649                  Partly cloudy throughout the day.
```

[14]:
```
sd.shape
```

[14]:
```
(13650, 20)
```

[15]:
```
sd.columns
```

[15]:
```
Index(['City', 'Date', 'tempmax', 'tempmin', 'temp', 'feelslikemax',
       'feelslikemin', 'feelslike', 'dew', 'humidity', 'windspeed', 'winddir',
       'sealevelpressure', 'cloudcover', 'visibility', 'sunrise', 'sunset',
       'moonphase', 'conditions', 'description'],
```

```
      dtype='object')
```

[16]:
```
sd.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13650 entries, 0 to 13649
Data columns (total 20 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   City              13650 non-null  object
 1   Date              13650 non-null  object
 2   tempmax           13615 non-null  float64
 3   tempmin           13615 non-null  float64
 4   temp              13605 non-null  float64
 5   feelslikemax      13614 non-null  float64
 6   feelslikemin      13614 non-null  float64
 7   feelslike         13604 non-null  float64
 8   dew               13605 non-null  float64
 9   humidity          13605 non-null  float64
 10  windspeed         13605 non-null  float64
 11  winddir           13600 non-null  float64
 12  sealevelpressure  10631 non-null  float64
 13  cloudcover        13605 non-null  float64
 14  visibility        13605 non-null  float64
 15  sunrise           13650 non-null  object
 16  sunset            13650 non-null  object
 17  moonphase         13650 non-null  float64
 18  conditions        13605 non-null  object
 19  description       13605 non-null  object
dtypes: float64(14), object(6)
memory usage: 2.1+ MB
```

[8]:
```
sd.describe()
```

[8]:

	tempmax	tempmin	temp	feelslikemax	feelslikemin	\
count	13615.000000	13615.000000	13605.000000	13614.000000	13614.000000	
mean	36.728248	25.821160	31.151510	40.212605	27.221324	
std	4.115452	3.212167	3.074874	5.389016	4.907125	
min	0.000000	0.000000	19.900000	0.000000	0.000000	
25%	34.000000	23.700000	29.200000	36.500000	23.700000	
50%	37.000000	26.000000	31.100000	40.000000	26.000000	
75%	39.800000	28.100000	33.200000	43.700000	31.100000	
max	50.000000	37.000000	40.500000	79.200000	43.300000	

	feelslike	dew	humidity	windspeed	winddir	\
count	13604.000000	13605.000000	13605.000000	13605.000000	13600.000000	
mean	33.704535	19.049607	54.638537	20.078552	205.236559	
std	4.666616	5.966341	19.521510	9.886468	64.181345	

```
min       19.900000    -10.300000      7.410000      0.000000      0.000000
25%       30.200000     15.000000     38.190000     14.800000    159.800000
50%       33.500000     20.300000     56.120000     19.500000    218.300000
75%       37.200000     24.000000     71.410000     24.100000    258.625000
max       48.500000     29.100000     99.040000    263.200000    360.000000

       sealevelpressure    cloudcover    visibility     moonphase
count      10631.000000  13605.000000  13605.000000  13650.000000
mean        1004.302446     37.120235      4.666645      0.500692
std            4.183785     24.684504      1.382413      0.308204
min          908.500000      0.000000      1.300000      0.000000
25%         1001.600000     16.700000      3.700000      0.250000
50%         1004.700000     36.700000      4.300000      0.500000
75%         1007.300000     54.000000      5.600000      0.760000
max         1026.200000    100.000000     12.300000      1.000000
```

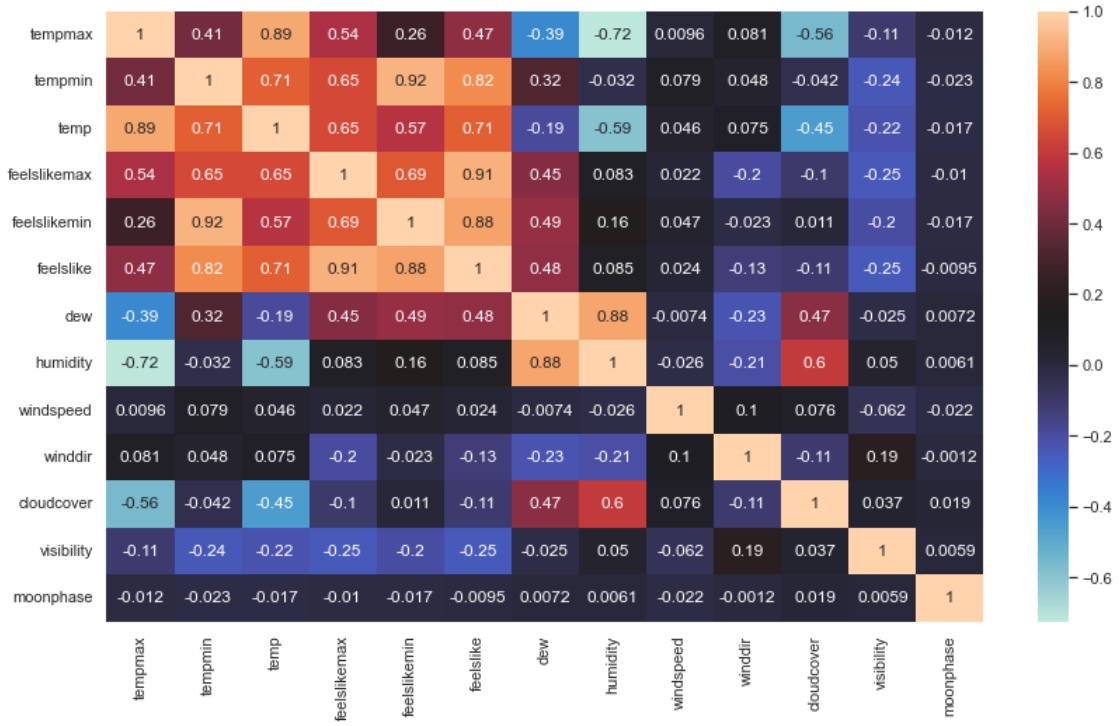
[17]:
```
sd.corr()
```

[17]:
```
                   tempmax   tempmin      temp  feelslikemax  feelslikemin  \
tempmax           1.000000  0.440258  0.889703      0.563109      0.288470
tempmin           0.440258  1.000000  0.713366      0.667459      0.918118
temp              0.889703  0.713366  1.000000      0.653402      0.567280
feelslikemax      0.563109  0.667459  0.653402      1.000000      0.699609
feelslikemin      0.288470  0.918118  0.567280      0.699609      1.000000
feelslike         0.473192  0.819431  0.713988      0.911220      0.876071
dew              -0.390626  0.320189 -0.189022      0.448867      0.492071
humidity         -0.723530 -0.033183 -0.589720      0.083387      0.163914
windspeed         0.010232  0.077965  0.046101      0.023002      0.046573
winddir           0.080809  0.047670  0.075168     -0.197638     -0.022646
sealevelpressure -0.256635 -0.407958 -0.382111     -0.390121     -0.313366
cloudcover       -0.558667 -0.041898 -0.450869     -0.104461      0.011275
visibility       -0.110979 -0.241262 -0.222571     -0.250963     -0.200458
moonphase        -0.013205 -0.024386 -0.017021     -0.011759     -0.018024

                  feelslike       dew  humidity  windspeed   winddir  \
tempmax            0.473192 -0.390626 -0.723530   0.010232  0.080809
tempmin            0.819431  0.320189 -0.033183   0.077965  0.047670
temp               0.713988 -0.189022 -0.589720   0.046101  0.075168
feelslikemax       0.911220  0.448867  0.083387   0.023002 -0.197638
feelslikemin       0.876071  0.492071  0.163914   0.046573 -0.022646
feelslike          1.000000  0.481024  0.085207   0.024289 -0.131108
dew                0.481024  1.000000  0.883934  -0.007274 -0.228871
humidity           0.085207  0.883934  1.000000  -0.026176 -0.207767
windspeed          0.024289 -0.007274 -0.026176   1.000000  0.101213
winddir           -0.131108 -0.228871 -0.207767   0.101213  1.000000
sealevelpressure  -0.381733 -0.159239  0.002652  -0.168941  0.019980
cloudcover        -0.106787  0.465260  0.597681   0.075356 -0.110428
```

```
visibility         -0.247751 -0.025038  0.050051  -0.062159  0.194713
moonphase          -0.009700  0.006798  0.005930  -0.022054 -0.001061

                   sealevelpressure  cloudcover  visibility  moonphase
tempmax                  -0.256635   -0.558667   -0.110979  -0.013205
tempmin                  -0.407958   -0.041898   -0.241262  -0.024386
temp                     -0.382111   -0.450869   -0.222571  -0.017021
feelslikemax             -0.390121   -0.104461   -0.250963  -0.011759
feelslikemin             -0.313366    0.011275   -0.200458  -0.018024
feelslike                -0.381733   -0.106787   -0.247751  -0.009700
dew                      -0.159239    0.465260   -0.025038   0.006798
humidity                  0.002652    0.597681    0.050051   0.005930
windspeed                -0.168941    0.075356   -0.062159  -0.022054
winddir                   0.019980   -0.110428    0.194713  -0.001061
sealevelpressure          1.000000   -0.209841    0.277592   0.027553
cloudcover               -0.209841    1.000000    0.037111   0.018903
visibility                0.277592    0.037111    1.000000   0.006121
moonphase                 0.027553    0.018903    0.006121   1.000000
```

[45]:
```
plt.figure(figsize=(14,8))
sns.heatmap(sd.corr(), annot=True,cmap='icefire')
```

[45]: <AxesSubplot:>



[20]:
```
sd.isnull().sum()
```

[20]:
```
City                  0
Date                  0
tempmax              35
tempmin              35
temp                 45
feelslikemax         36
feelslikemin         36
feelslike            46
dew                  45
humidity             45
windspeed            45
winddir              50
sealevelpressure   3019
cloudcover           45
visibility           45
sunrise               0
sunset                0
moonphase             0
conditions           45
description          45
dtype: int64
```

[21]:
```
sd=sd.drop(['sealevelpressure'],axis=1)
```

[22]:
```
sd.columns
```

[22]:
```
Index(['City', 'Date', 'tempmax', 'tempmin', 'temp', 'feelslikemax',
       'feelslikemin', 'feelslike', 'dew', 'humidity', 'windspeed', 'winddir',
       'cloudcover', 'visibility', 'sunrise', 'sunset', 'moonphase',
       'conditions', 'description'],
      dtype='object')
```

[23]:
```
sd.dropna(inplace=True)
```

[24]:
```
sd.shape
```

[24]:
```
(13599, 19)
```

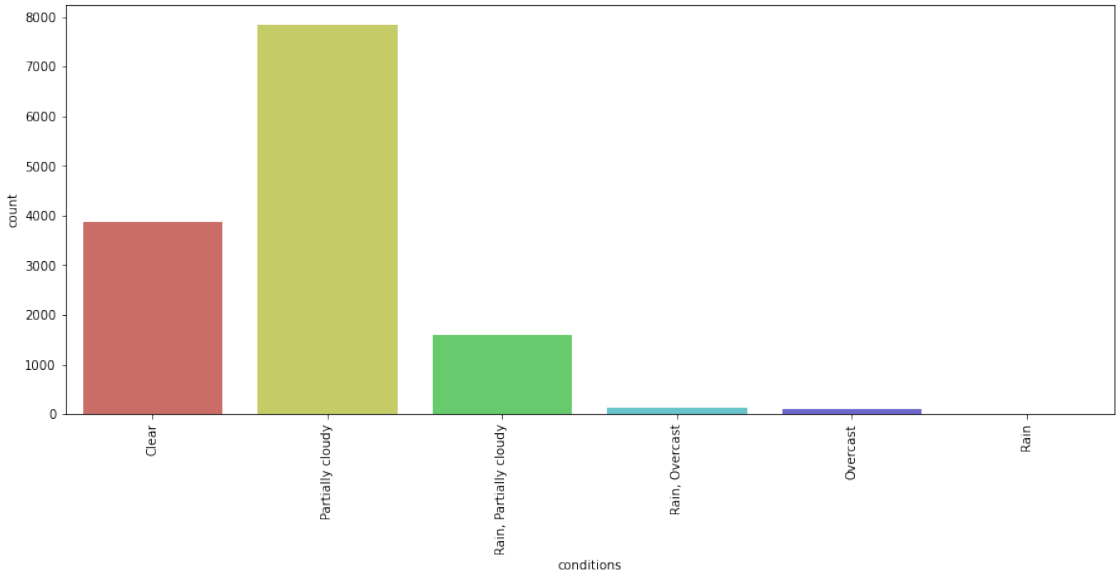
[25]:
```
sd['conditions'].unique()
```

[25]:
```
array(['Clear', 'Partially cloudy', 'Rain, Partially cloudy',
       'Rain, Overcast', 'Overcast', 'Rain'], dtype=object)
```

[26]:
```
sd['conditions'].value_counts()
```

[26]:
```
Partially cloudy          7852
Clear                     3880
Rain, Partially cloudy    1609
Rain, Overcast             136
Overcast                   113
Rain                         9
Name: conditions, dtype: int64
```

[29]:
```
plt.figure(figsize=(15,6))
sns.countplot('conditions', data=sd,palette='hls')
plt.xticks(rotation=90)
plt.show()
```



[30]:
```
count_clear=len(sd[sd.conditions=='Clear'])
count_pcloudy=len(sd[sd.conditions=='Partially Cloudy'])
count_rpcloudy=len(sd[sd.conditions=='Rain, Partially Cloudy'])
count_ro=len(sd[sd.conditions=='Rain, Overcast'])
count_overcast=len(sd[sd.conditions=='Overcast'])
count_rain=len(sd[sd.conditions=='Rain'])
```

[32]:
```
print("Percent of Clear:{:2f}%".format((count_clear/(len(sd.conditions))*100)))
print("Percent of Partial Cloudy:{:2f}%".format((count_pcloudy/(len(sd.
 ↪conditions))*100)))
print("Percent of Rain Partial Cloudy:{:2f}%".format((count_rpcloudy/(len(sd.
 ↪conditions))*100)))
print("Percent of Rain Overcast:{:2f}%".format((count_ro/(len(sd.
 ↪conditions))*100)))
```

```
print("Percent of Overcast:{:2f}%".format((count_overcast/(len(sd.
 ↪conditions))*100)))
print("Percent of Rain:{:2f}%".format((count_rain/(len(sd.conditions))*100)))
```

```
Percent of Clear:28.531510%
Percent of Partial Cloudy:0.000000%
Percent of Rain Partial Cloudy:0.000000%
Percent of Rain Overcast:1.000074%
Percent of Overcast:0.830943%
Percent of Rain:0.066181%
```

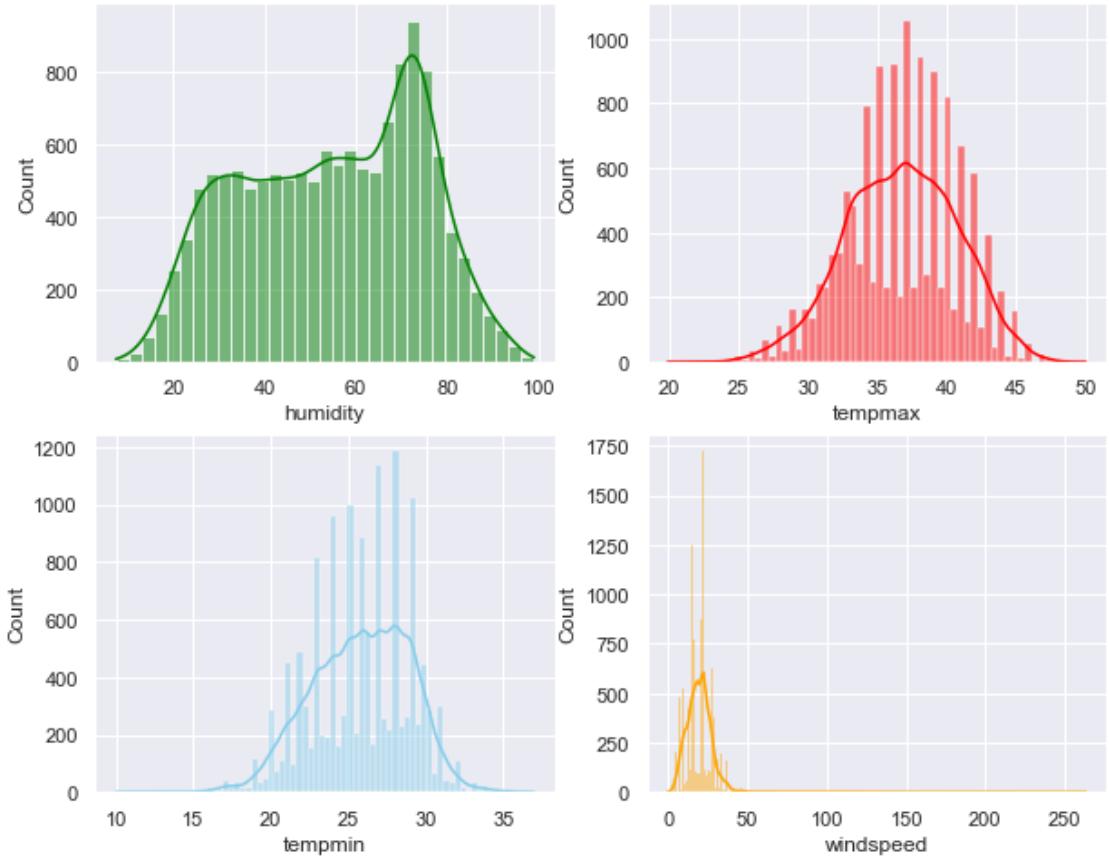
[33]:
```
sd[["humidity","tempmax","tempmin","windspeed"]].describe()
```

[33]:

	humidity	tempmax	tempmin	windspeed
count	13599.000000	13599.000000	13599.000000	13599.000000
mean	54.643784	36.756828	25.837937	20.083293
std	19.519355	3.993890	3.133552	9.885514
min	7.410000	19.900000	10.000000	0.000000
25%	38.195000	34.000000	23.700000	14.800000
50%	56.120000	37.000000	26.000000	19.500000
75%	71.420000	39.800000	28.100000	24.100000
max	99.040000	50.000000	37.000000	263.200000

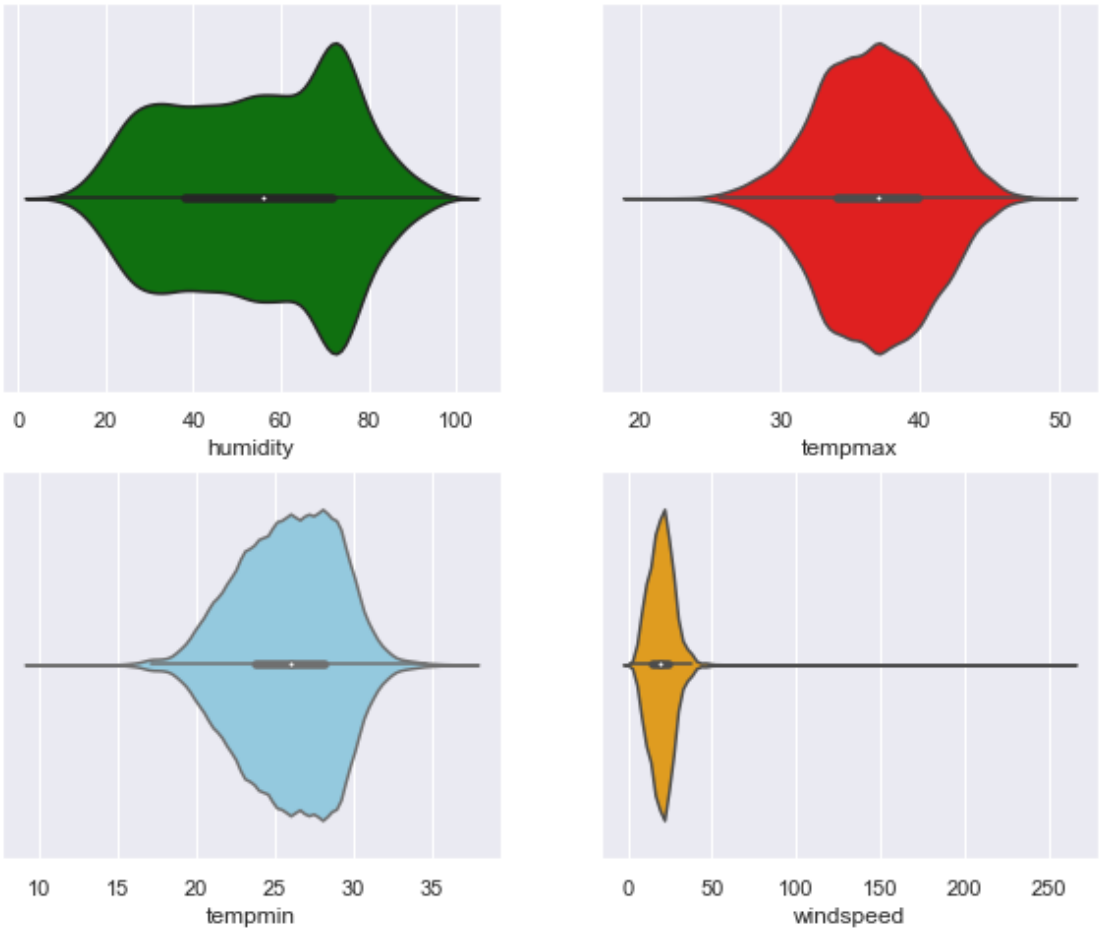
[35]:
```
sns.set(style="darkgrid")
fig,axs=plt.subplots(2,2,figsize=(10,8))
sns.histplot(data=sd,x='humidity',kde=True,ax=axs[0,0],color='green')
sns.histplot(data=sd,x='tempmax',kde=True,ax=axs[0,1],color='red')
sns.histplot(data=sd,x='tempmin',kde=True,ax=axs[1,0],color='skyblue')
sns.histplot(data=sd,x='windspeed',kde=True,ax=axs[1,1],color='orange')
```

[35]:
```
<AxesSubplot:xlabel='windspeed', ylabel='Count'>
```



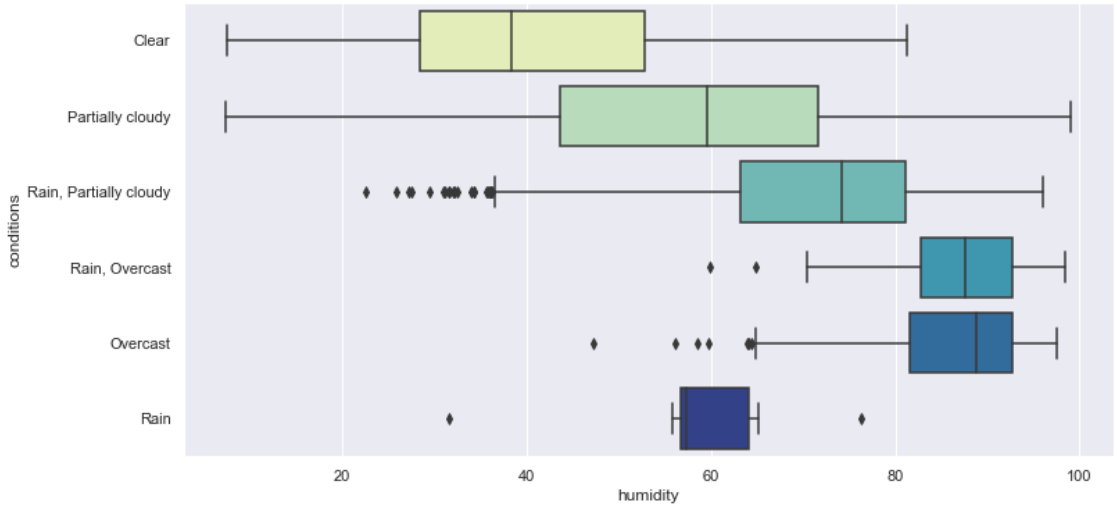
[36]:
```
sns.set(style="darkgrid")
fig,axs=plt.subplots(2,2,figsize=(10,8))
sns.violinplot(data=sd,x='humidity',kde=True,ax=axs[0,0],color='green')
sns.violinplot(data=sd,x='tempmax',kde=True,ax=axs[0,1],color='red')
sns.violinplot(data=sd,x='tempmin',kde=True,ax=axs[1,0],color='skyblue')
sns.violinplot(data=sd,x='windspeed',kde=True,ax=axs[1,1],color='orange')
```

[36]: <AxesSubplot:xlabel='windspeed'>



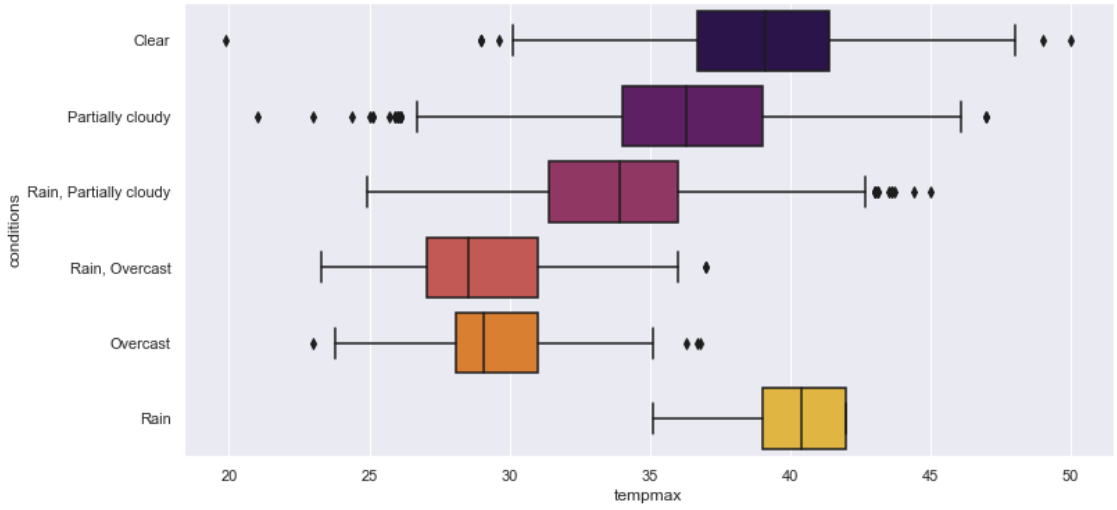
[39]:
```
plt.figure(figsize=(12,6))
sns.boxplot("humidity","conditions",data=sd,palette="YlGnBu")
```

[39]: <AxesSubplot:xlabel='humidity', ylabel='conditions'>



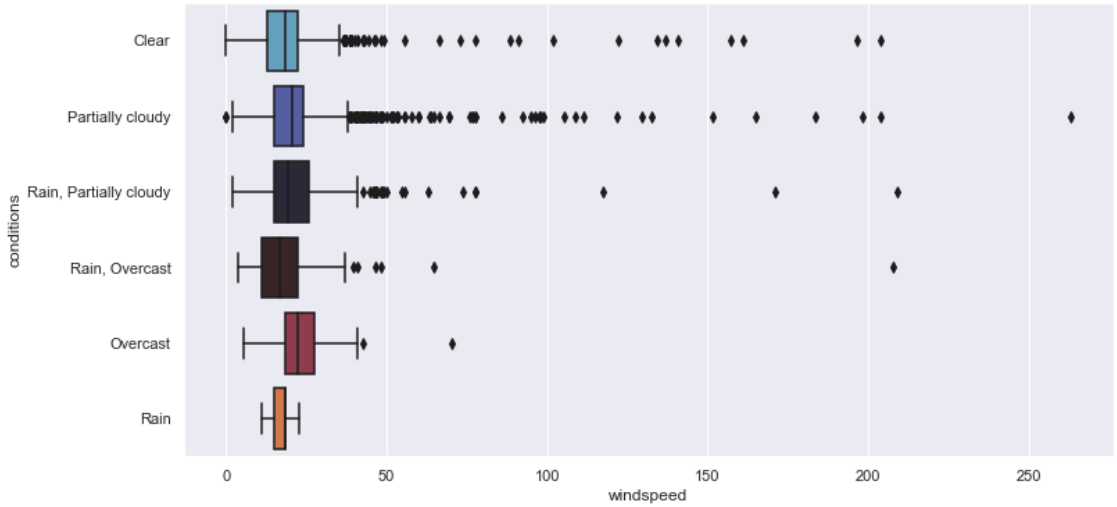
[40]:
```
plt.figure(figsize=(12,6))
sns.boxplot("tempmax","conditions",data=sd,palette="inferno")
```

[40]: <AxesSubplot:xlabel='tempmax', ylabel='conditions'>



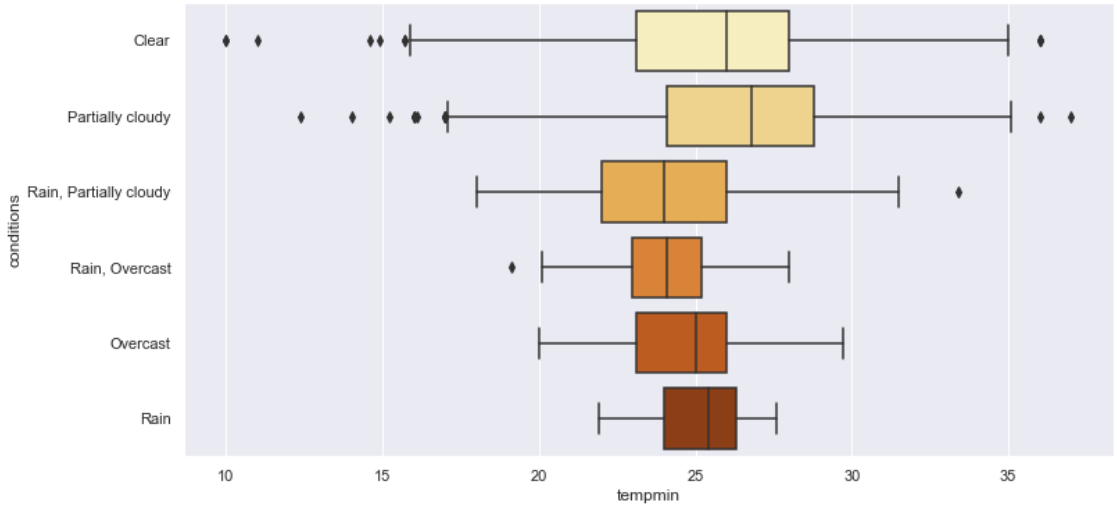
[41]:
```
plt.figure(figsize=(12,6))
sns.boxplot("windspeed","conditions",data=sd,palette="icefire")
```

[41]: <AxesSubplot:xlabel='windspeed', ylabel='conditions'>



[43]:
```
plt.figure(figsize=(12,6))
sns.boxplot("tempmin","conditions",data=sd,palette="YlOrBr")
```

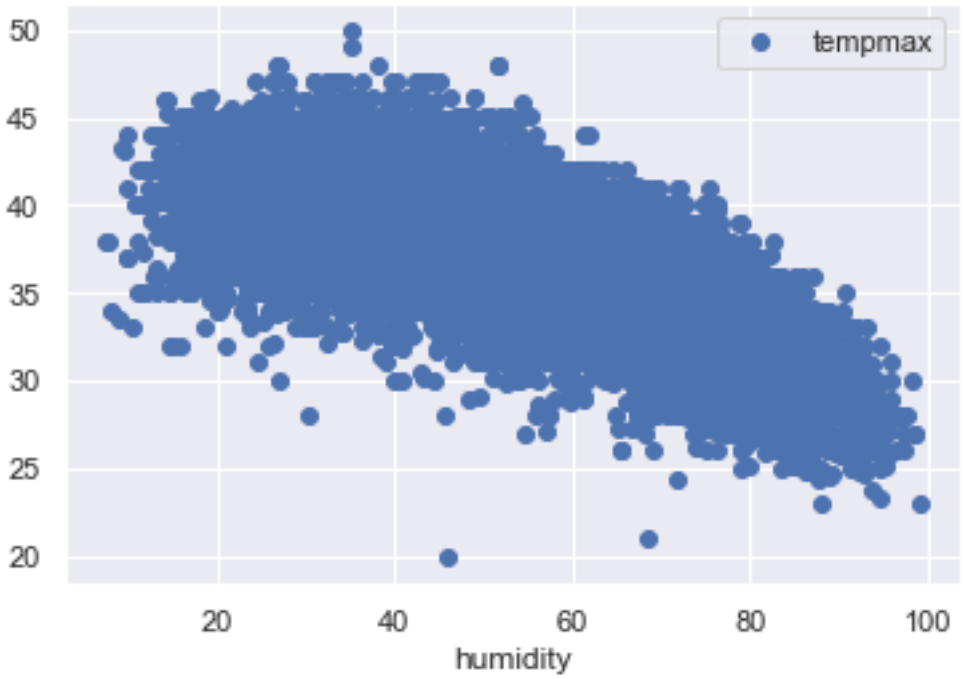
[43]: <AxesSubplot:xlabel='tempmin', ylabel='conditions'>



[46]:
```
from scipy import stats
```

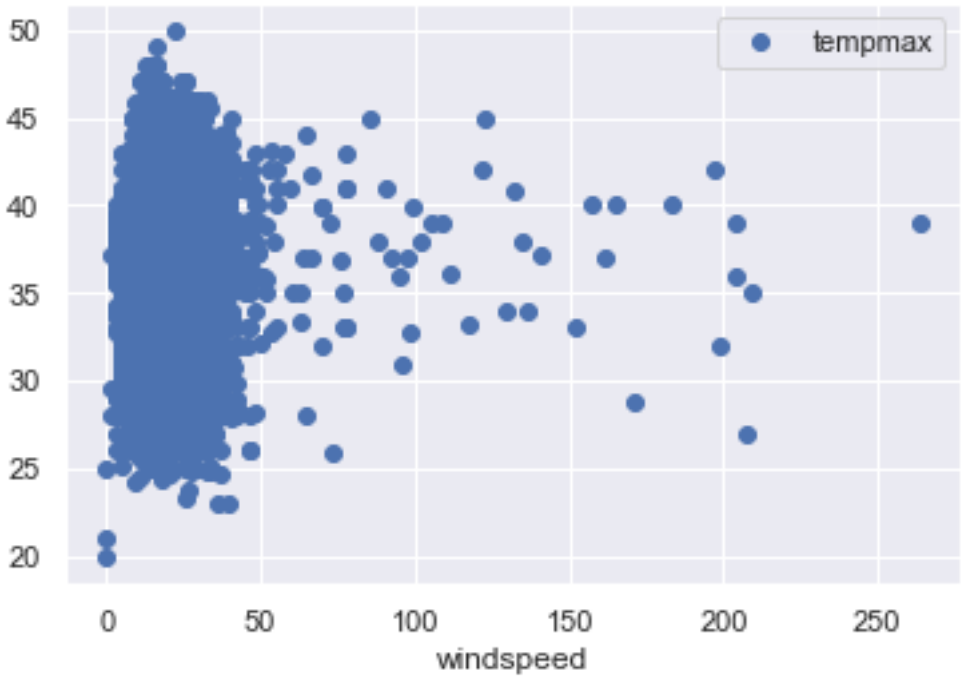
[47]:
```
sd.plot("humidity","tempmax",style='o')
print("Pearson correlation:",sd['humidity'].corr(sd['tempmax']))
print("T Test and P value:",stats.ttest_ind(sd['humidity'],sd['tempmax']))
```

```
Pearson correlation: -0.724123660887335
T Test and P value: Ttest_indResult(statistic=104.69321537002257, pvalue=0.0)
```



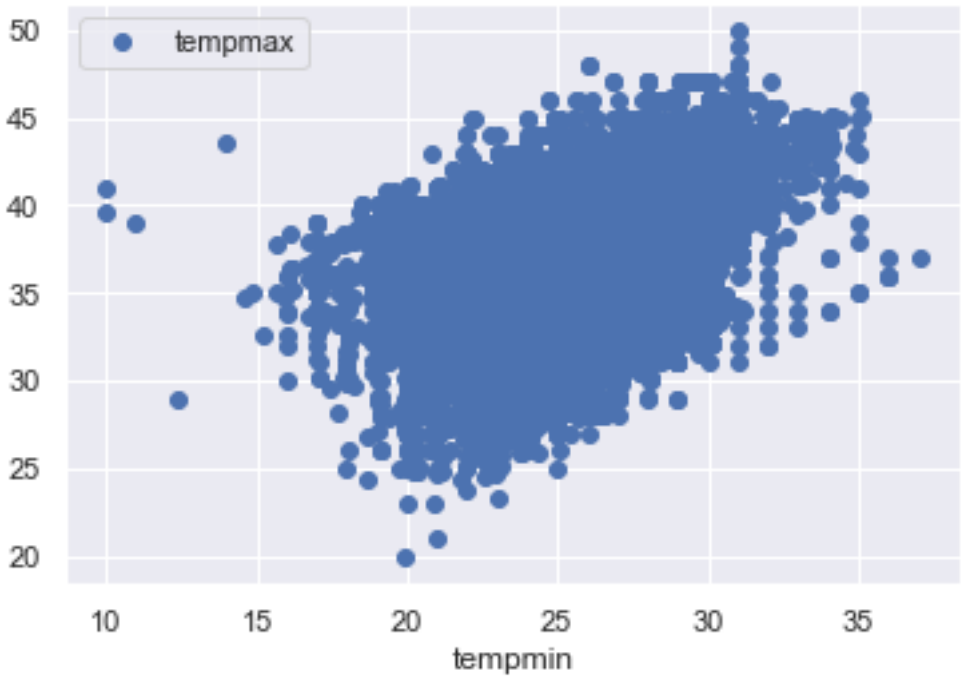
[49]:
```
sd.plot("windspeed","tempmax",style='o')
print("Pearson correlation:",sd['windspeed'].corr(sd['tempmax']))
print("T Test and P value:",stats.ttest_ind(sd['windspeed'],sd['tempmax']))
```

```
Pearson correlation: 0.009607384829183984
T Test and P value: Ttest_indResult(statistic=-182.368393165443, pvalue=0.0)
```



[50]:
```
sd.plot("tempmin","tempmax",style='o')
print("Pearson correlation:",sd['tempmin'].corr(sd['tempmax']))
print("T Test and P value:",stats.ttest_ind(sd['tempmin'],sd['tempmax']))
```

```
Pearson correlation: 0.410267019733804
T Test and P value: Ttest_indResult(statistic=-250.82583693772827, pvalue=0.0)
```



[51]:
```
df=sd.drop(['Date','sunrise','sunset','description'],axis=1)
```

[52]:
```
Q1=df.quantile(0.25)
Q3=df.quantile(0.75)
IQR=Q3-Q1
df=df[~((df<(Q1-1.5*IQR))|(df>(Q3+1.5*IQR))).any(axis=1)]
```

[54]:
```
df.humidity=np.sqrt(df.humidity)
df.windspeed=np.sqrt(df.windspeed)
```

[55]:
```
df.head()
```

[55]:

	City	tempmax	tempmin	temp	feelslikemax	feelslikemin	feelslike	\
0	New Delhi	34.0	19.0	27.1	31.6	19.0	26.1	
4	New Delhi	38.8	21.0	29.9	37.1	21.0	28.9	
5	New Delhi	38.0	22.6	30.4	37.2	22.6	29.5	
6	New Delhi	36.0	23.4	29.6	34.6	23.4	28.7	
7	New Delhi	34.9	20.9	27.6	32.6	20.9	26.7	

	dew	humidity	windspeed	winddir	cloudcover	visibility	moonphase	\
0	3.1	4.753946	4.774935	272.9	0.0	3.1	0.60	
4	8.1	5.371220	3.674235	100.4	38.4	3.1	0.81	
5	10.2	5.523586	3.847077	102.0	30.7	2.4	0.86	
6	9.7	5.629387	4.289522	220.8	24.1	2.7	0.91	

```
7   4.4  5.144900   3.987480    259.7         0.0         3.2       0.94

          conditions
0              Clear
4  Partially cloudy
5  Partially cloudy
6  Partially cloudy
7              Clear
```

[56]:
```
df1=df.drop(['City'],axis=1)
```

[57]:
```
df1.head()
```

[57]:
```
   tempmax  tempmin  temp  feelslikemax  feelslikemin  feelslike   dew  \
0     34.0     19.0  27.1          31.6          19.0       26.1   3.1
4     38.8     21.0  29.9          37.1          21.0       28.9   8.1
5     38.0     22.6  30.4          37.2          22.6       29.5  10.2
6     36.0     23.4  29.6          34.6          23.4       28.7   9.7
7     34.9     20.9  27.6          32.6          20.9       26.7   4.4

   humidity  windspeed  winddir  cloudcover  visibility  moonphase  \
0  4.753946   4.774935    272.9         0.0         3.1       0.60
4  5.371220   3.674235    100.4        38.4         3.1       0.81
5  5.523586   3.847077    102.0        30.7         2.4       0.86
6  5.629387   4.289522    220.8        24.1         2.7       0.91
7  5.144900   3.987480    259.7         0.0         3.2       0.94

          conditions
0              Clear
4  Partially cloudy
5  Partially cloudy
6  Partially cloudy
7              Clear
```

[61]:
```
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.metrics import␣
 ↪accuracy_score,confusion_matrix,classification_report
```

[62]:
```
lc=LabelEncoder()
df1['conditions']=lc.fit_transform(df1["conditions"])
```

[64]:
```
df1.head()
```

[64]:

	tempmax	tempmin	temp	feelslikemax	feelslikemin	feelslike	dew	\
0	34.0	19.0	27.1	31.6	19.0	26.1	3.1	
4	38.8	21.0	29.9	37.1	21.0	28.9	8.1	
5	38.0	22.6	30.4	37.2	22.6	29.5	10.2	
6	36.0	23.4	29.6	34.6	23.4	28.7	9.7	
7	34.9	20.9	27.6	32.6	20.9	26.7	4.4	

	humidity	windspeed	winddir	cloudcover	visibility	moonphase	conditions
0	4.753946	4.774935	272.9	0.0	3.1	0.60	0
4	5.371220	3.674235	100.4	38.4	3.1	0.81	2
5	5.523586	3.847077	102.0	30.7	2.4	0.86	2
6	5.629387	4.289522	220.8	24.1	2.7	0.91	2
7	5.144900	3.987480	259.7	0.0	3.2	0.94	0

[65]:
```
x=((df1.loc[:,df1.columns!="conditions"]).astype(int)).values[:,0:]
y=df1["conditions"].values
```

[66]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=1)
```

[67]:
```
knn=KNeighborsClassifier()
knn.fit(x_train,y_train)
print("KNN Accuracy:{:2f}%".format(knn.score(x_test,y_test)*100))
```

```
KNN Accuracy:86.233270%
```

[69]:
```
svm=SVC()
svm.fit(x_train,y_train)
print("SVM Accuracy:{:2f}%".format(svm.score(x_test,y_test)*100))
```

```
SVM Accuracy:86.462715%
```

[72]:
```
gbc=GradientBoostingClassifier(subsample=0.
 ↪5,n_estimators=450,max_depth=5,max_leaf_nodes=2)
gbc.fit(x_train,y_train)
print('Gradient Boosting Accuracy:{:2f}%'.format(gbc.score(x_test,y_test)*100))
```

```
Gradient Boosting Accuracy:89.139579%
```