In [55]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn import tree
from sklearn.metrics import accuracy_score
```

In [56]:
```
data = pd.read_csv("seattle-weather.csv")
data.head()
```

Out[56]:

	date	precipitation	temp_max	temp_min	wind	weather
0	2012-01-01	0.0	12.8	5.0	4.7	drizzle
1	2012-01-02	10.9	10.6	2.8	4.5	rain
2	2012-01-03	0.8	11.7	7.2	2.3	rain
3	2012-01-04	20.3	12.2	5.6	4.7	rain
4	2012-01-05	1.3	8.9	2.8	6.1	rain

In [40]:
```
data.tail()
```

Out[40]:

	date	precipitation	temp_max	temp_min	wind	weather
1456	2015-12-27	8.6	4.4	1.7	2.9	rain
1457	2015-12-28	1.5	5.0	1.7	1.3	rain
1458	2015-12-29	0.0	7.2	0.6	2.6	fog
1459	2015-12-30	0.0	5.6	-1.0	3.4	sun
1460	2015-12-31	0.0	5.6	-2.1	3.5	sun

In [41]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1461 entries, 0 to 1460
Data columns (total 6 columns):
date                 1461 non-null object
precipitation        1461 non-null float64
temp_max             1461 non-null float64
temp_min             1461 non-null float64
wind                 1461 non-null float64
weather              1461 non-null object
dtypes: float64(4), object(2)
memory usage: 68.6+ KB
```

In [42]:
```
data.isnull()
```

Out[42]:

	date	precipitation	temp_max	temp_min	wind	weather
0	False	False	False	False	False	False
1	False	False	False	False	False	False
2	False	False	False	False	False	False
3	False	False	False	False	False	False
4	False	False	False	False	False	False
...	...	...	...	...	...	...
1456	False	False	False	False	False	False
1457	False	False	False	False	False	False
1458	False	False	False	False	False	False
1459	False	False	False	False	False	False
1460	False	False	False	False	False	False

1461 rows × 6 columns

In [43]:
```
data.isnull().sum()
```

Out[43]:
```
date             0
precipitation    0
temp_max         0
temp_min         0
wind             0
weather          0
dtype: int64
```

In [44]:
```
data.columns
```

Out[44]:
```
Index(['date', 'precipitation', 'temp_max', 'temp_min', 'wind', 'weather'], dty
pe='object')
```

In [45]:
```
data.describe()
```

Out[45]:

	precipitation	temp_max	temp_min	wind
count	1461.000000	1461.000000	1461.000000	1461.000000
mean	3.029432	16.439083	8.234771	3.241136
std	6.680194	7.349758	5.023004	1.437825
min	0.000000	-1.600000	-7.100000	0.400000
25%	0.000000	10.600000	4.400000	2.200000
50%	0.000000	15.600000	8.300000	3.000000
75%	2.800000	22.200000	12.200000	4.000000
max	55.900000	35.600000	18.300000	9.500000

In [86]:
```
data["temp_max"].count()
```

Out[86]:
```
1461
```

In [98]:
```
data["temp_max"].max()
```

Out[98]:
```
35.6
```

In [99]:
```
data["wind"].max()
```

Out[99]:
```
9.5
```

In [103]:
```
x=data["temp_max"]
y=data["weather"]=='rain'
z=x[y]
z
```

Out[103]:
```
1       10.6
2       11.7
3       12.2
4        8.9
5        4.4
        ...
1452     5.0
1453     5.6
1454     5.0
1456     4.4
1457     5.0
Name: temp_max, Length: 641, dtype: float64
```

In [106]:
```
x=data["wind"]
y=data["weather"]=='rain'
z=x[y]
z
```

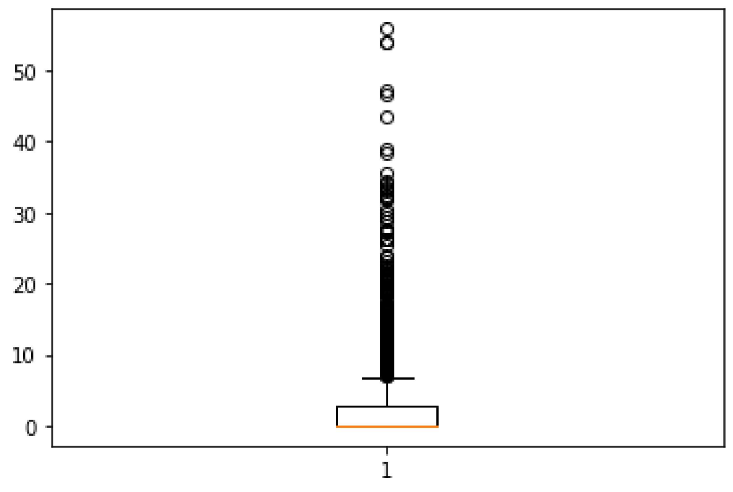
Out[106]:
```
1       4.5
2       2.3
3       4.7
4       6.1
5       2.2
       ...
1452    7.6
1453    4.3
1454    1.5
1456    2.9
1457    1.3
Name: wind, Length: 641, dtype: float64
```

In [46]:
```
data.duplicated().sum()
```

Out[46]:
```
0
```

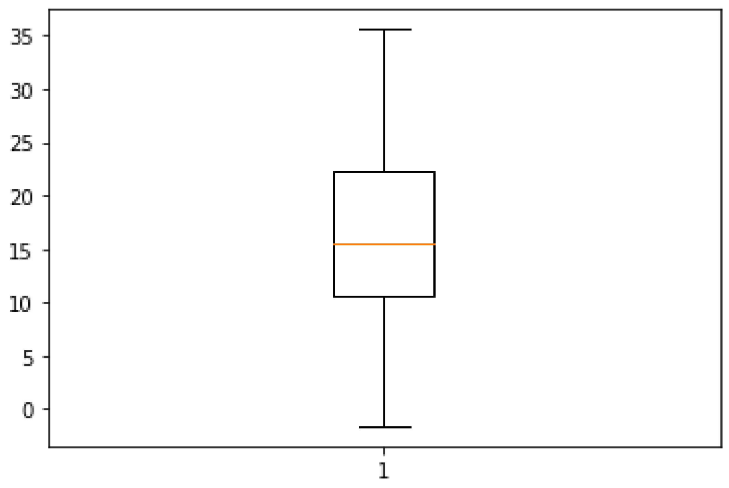
In [47]:
```
plt.boxplot(data["precipitation"])
```

Out[47]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x1d121a80fc8>,
  <matplotlib.lines.Line2D at 0x1d121a80bc8>],
 'caps': [<matplotlib.lines.Line2D at 0x1d121a85c88>,
  <matplotlib.lines.Line2D at 0x1d121a85d88>],
 'boxes': [<matplotlib.lines.Line2D at 0x1d121a80948>],
 'medians': [<matplotlib.lines.Line2D at 0x1d121a85e08>],
 'fliers': [<matplotlib.lines.Line2D at 0x1d121a8ccc8>],
 'means': []}
```



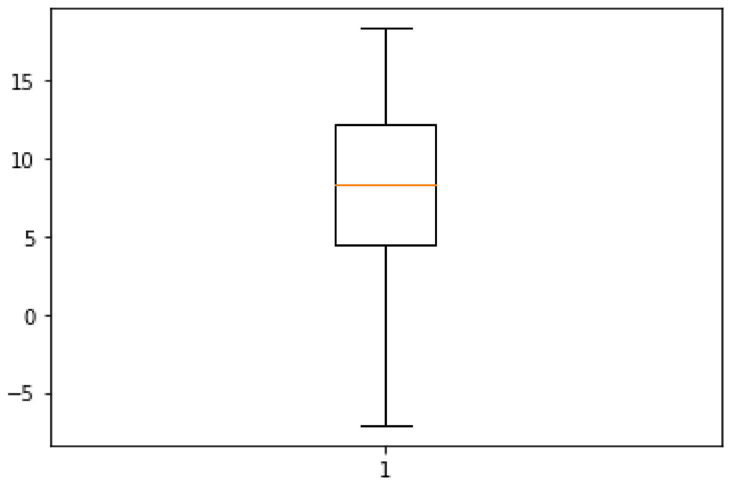
In [48]:
```
plt.boxplot(data['temp_max'])
```

Out[48]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x1d121ae7848>,
  <matplotlib.lines.Line2D at 0x1d121ae7fc8>],
 'caps': [<matplotlib.lines.Line2D at 0x1d121ae7f88>,
  <matplotlib.lines.Line2D at 0x1d121aedf88>],
 'boxes': [<matplotlib.lines.Line2D at 0x1d121ae2d48>],
 'medians': [<matplotlib.lines.Line2D at 0x1d121aedf08>],
 'fliers': [<matplotlib.lines.Line2D at 0x1d121af0f08>],
 'means': []}
```



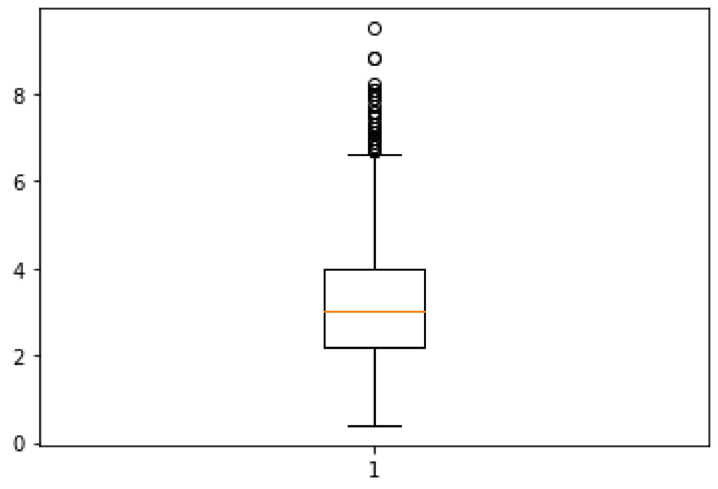
In [49]:
```
plt.boxplot(data["temp_min"])
```

Out[49]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x1d121b53fc8>,
  <matplotlib.lines.Line2D at 0x1d121b57a48>],
 'caps': [<matplotlib.lines.Line2D at 0x1d121b57b48>,
  <matplotlib.lines.Line2D at 0x1d121b5c9c8>],
 'boxes': [<matplotlib.lines.Line2D at 0x1d121b53788>],
 'medians': [<matplotlib.lines.Line2D at 0x1d121b5cac8>],
 'fliers': [<matplotlib.lines.Line2D at 0x1d121b61948>],
 'means': []}
```



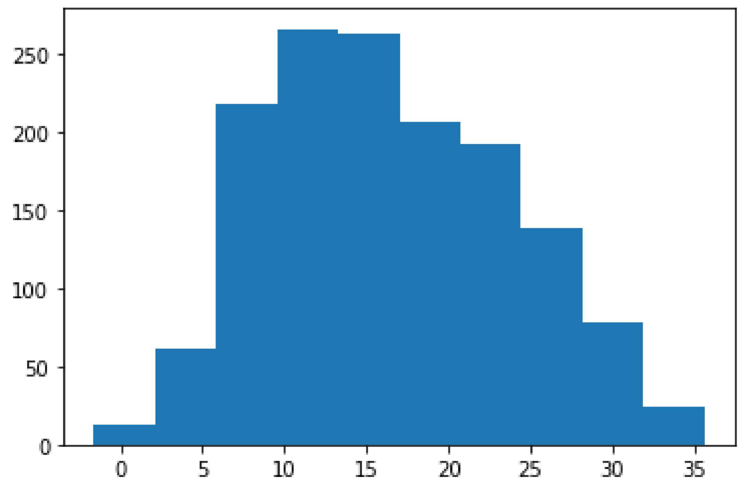
In [50]:
```
plt.boxplot(data["wind"])
```

Out[50]:
```
{'whiskers': [<matplotlib.lines.Line2D at 0x1d121bbcfc8>,
  <matplotlib.lines.Line2D at 0x1d121bbcf48>],
 'caps': [<matplotlib.lines.Line2D at 0x1d121bbef88>,
  <matplotlib.lines.Line2D at 0x1d121bbef08>],
 'boxes': [<matplotlib.lines.Line2D at 0x1d121bbc508>],
 'medians': [<matplotlib.lines.Line2D at 0x1d121bc2f08>],
 'fliers': [<matplotlib.lines.Line2D at 0x1d121bc2e88>],
 'means': []}
```



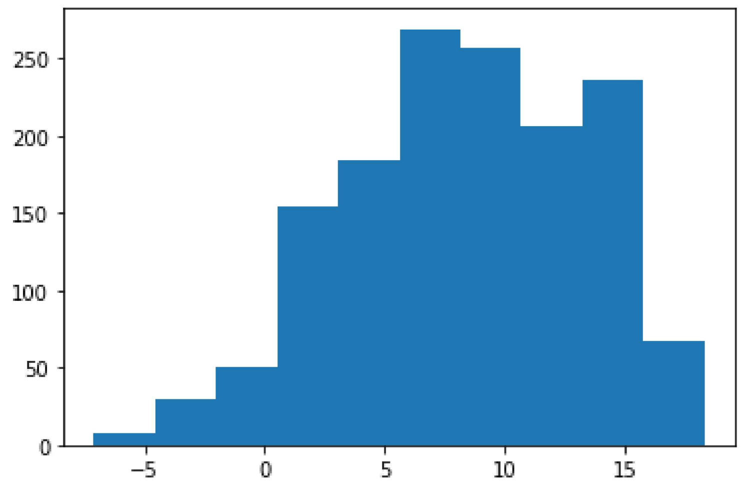
In [51]:
```
plt.hist(data['temp_max'])
```

Out[51]:
```
(array([ 12.,  61., 218., 266., 263., 207., 193., 139.,  78.,  24.]),
 array([-1.6 ,  2.12,  5.84,  9.56, 13.28, 17.  , 20.72, 24.44, 28.16,
        31.88, 35.6 ]),
 <a list of 10 Patch objects>)
```



In [52]:
```
plt.hist(data["temp_min"])
```

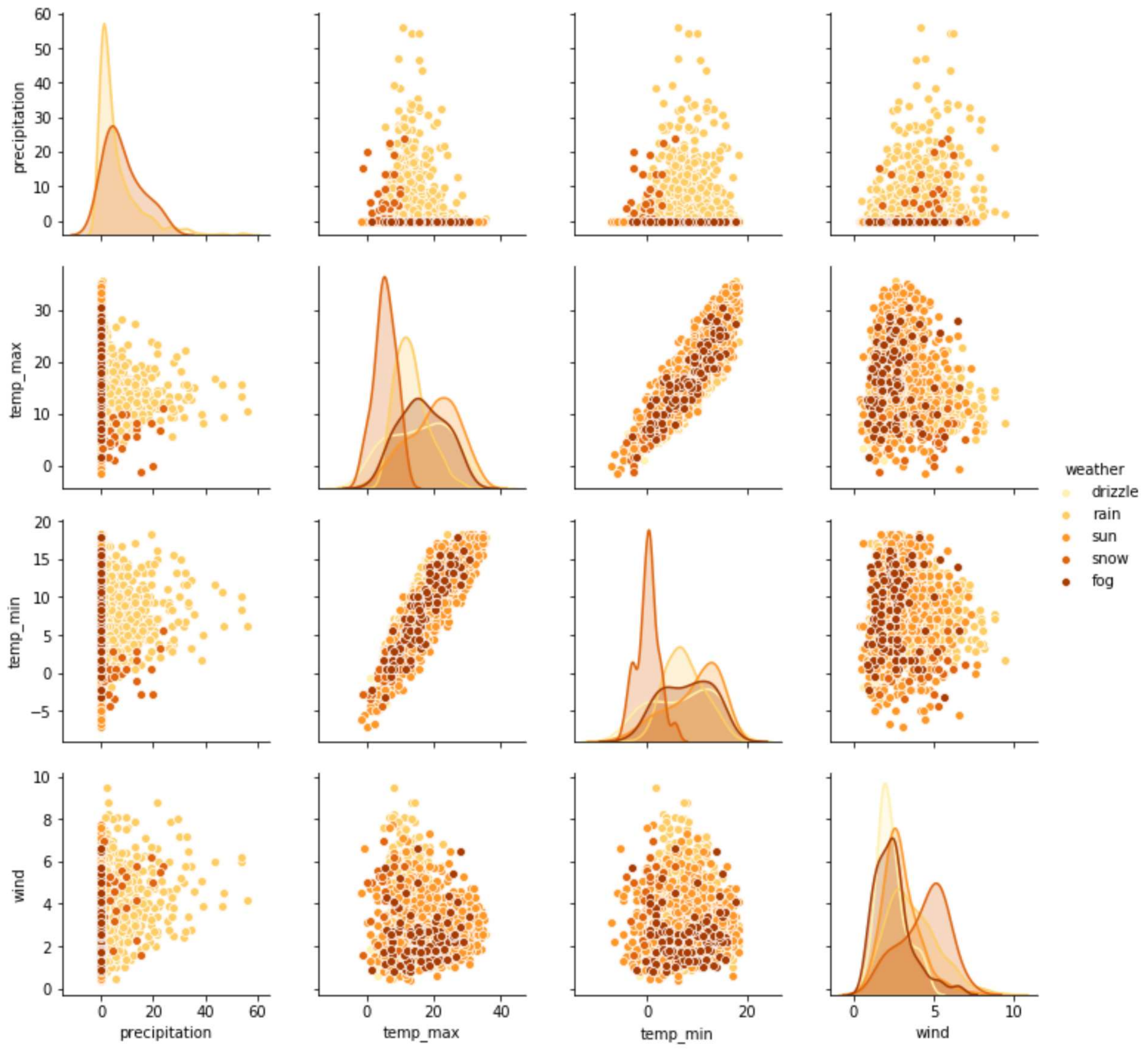
Out[52]:
```
(array([  8.,  30.,  50., 154., 184., 269., 257., 206., 236.,  67.]),
 array([-7.1 , -4.56, -2.02,  0.52,  3.06,  5.6 ,  8.14, 10.68, 13.22,
        15.76, 18.3 ]),
 <a list of 10 Patch objects>)
```



In [53]:
```
plt.figure(figsize=(10,5))
sns.pairplot(data.drop('date',axis=1),hue='weather',palette="YlOrBr")
plt.show()
```

```
C:\Users\Rgukt iiit\Anaconda3\anaconda\lib\site-packages\statsmodels\nonparamet
ric\kde.py:487: RuntimeWarning: invalid value encountered in true_divide
  binned = fast_linbin(X, a, b, gridsize) / (delta * nobs)
C:\Users\Rgukt iiit\Anaconda3\anaconda\lib\site-packages\statsmodels\nonparamet
ric\kdetools.py:34: RuntimeWarning: invalid value encountered in double_scalars
  FAC1 = 2*(np.pi*bw/RANGE)**2

<Figure size 720x360 with 0 Axes>
```

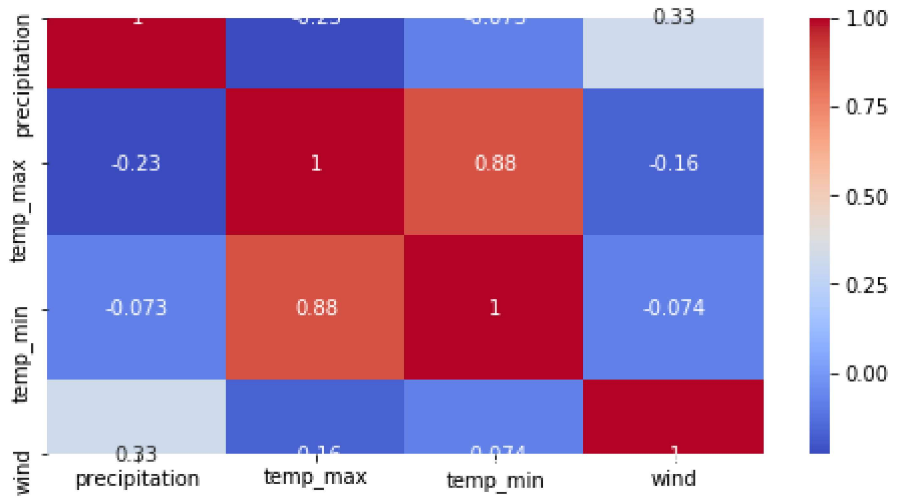


In [54]:
```
data.corr()
```

Out[54]:

	precipitation	temp_max	temp_min	wind
precipitation	1.000000	-0.228555	-0.072684	0.328045
temp_max	-0.228555	1.000000	0.875687	-0.164857
temp_min	-0.072684	0.875687	1.000000	-0.074185
wind	0.328045	-0.164857	-0.074185	1.000000

In [19]:
```
cor=data.corr()
plt.figure(figsize=(8,4))
sns.heatmap(cor,annot=True,cmap="coolwarm")
plt.show()
```



In [20]:
```
x=data[["precipitation","temp_max","temp_min","wind"]]
y=data["weather"]
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
```

In [21]:
```
x_test
```

Out[21]:

	precipitation	temp_max	temp_min	wind
530	0.0	20.0	12.2	3.7
657	0.0	10.6	7.8	1.4
459	8.4	14.4	10.0	3.0
279	0.0	23.9	7.8	5.1
656	0.0	12.8	7.2	1.2
...	...	...	...	...
440	4.3	10.6	4.4	6.4
634	0.0	17.2	7.2	2.2
494	0.0	22.8	10.0	1.3
61	2.0	6.7	3.9	5.1
517	0.0	22.8	12.2	2.5

293 rows × 4 columns

In [22]:
```
x_train
```

Out[22]:

	precipitation	temp_max	temp_min	wind
646	6.9	13.9	7.8	3.0
92	0.0	16.7	4.4	3.1
818	14.0	11.7	7.2	5.1
302	10.9	15.6	10.0	4.9
1259	0.0	23.9	9.4	2.6
...	...	...	...	...
763	0.0	8.9	1.1	2.5
835	0.5	14.4	7.8	4.0
1216	0.0	18.3	8.9	3.7
559	0.0	26.1	11.1	3.1
684	3.0	10.6	7.2	6.0

1168 rows × 4 columns

In [23]:
```
y_test
```

Out[23]:
```
530     sun
657     sun
459    rain
279     sun
656     sun
       ... 
440    rain
634     sun
494     sun
61     rain
517     sun
Name: weather, Length: 293, dtype: object
```

In [24]:
```
y_train
```

Out[24]:
```
646     rain
92       sun
818     rain
302     rain
1259     sun
        ... 
763      sun
835     rain
1216     sun
559      sun
684     rain
Name: weather, Length: 1168, dtype: object
```

In [25]:
```
print(x_train.shape)
print(y_train.shape)
print(x_test.shape)
print(y_test.shape)
```

```
(1168, 4)
(1168,)
(293, 4)
(293,)
```

In [26]:
```
model=tree.DecisionTreeClassifier()
my_model=model.fit(x_train,y_train)
y_pred=my_model.predict(x_test)
y_pred
```

Out[26]:
```
array(['sun', 'drizzle', 'rain', 'sun', 'drizzle', 'rain', 'rain', 'rain',
       'sun', 'rain', 'drizzle', 'rain', 'sun', 'sun', 'drizzle', 'sun',
       'drizzle', 'rain', 'rain', 'rain', 'rain', 'rain', 'rain', 'sun',
       'drizzle', 'sun', 'sun', 'rain', 'sun', 'sun', 'rain', 'rain',
       'rain', 'rain', 'sun', 'rain', 'sun', 'rain', 'sun', 'rain', 'sun',
       'rain', 'rain', 'rain', 'drizzle', 'drizzle', 'drizzle', 'sun',
       'sun', 'sun', 'sun', 'sun', 'sun', 'rain', 'rain', 'drizzle',
       'sun', 'rain', 'sun', 'rain', 'sun', 'sun', 'sun', 'sun', 'rain',
       'rain', 'sun', 'sun', 'sun', 'rain', 'rain', 'snow', 'rain',
       'drizzle', 'fog', 'sun', 'sun', 'drizzle', 'rain', 'rain', 'sun',
       'sun', 'rain', 'rain', 'rain', 'drizzle', 'sun', 'rain', 'sun',
       'rain', 'sun', 'sun', 'rain', 'sun', 'rain', 'snow', 'sun',
       'drizzle', 'sun', 'sun', 'rain', 'rain', 'sun', 'rain', 'rain',
       'sun', 'rain', 'rain', 'fog', 'sun', 'rain', 'sun', 'rain', 'fog',
       'sun', 'sun', 'sun', 'rain', 'rain', 'rain', 'sun', 'rain', 'rain',
       'sun', 'rain', 'sun', 'sun', 'sun', 'sun', 'rain', 'sun', 'rain',
       'sun', 'rain', 'rain', 'rain', 'drizzle', 'sun', 'sun', 'sun',
       'rain', 'rain', 'sun', 'sun', 'drizzle', 'rain', 'rain', 'sun',
       'fog', 'sun', 'sun', 'sun', 'sun', 'rain', 'fog', 'rain', 'sun',
       'sun', 'rain', 'sun', 'sun', 'drizzle', 'rain', 'fog', 'rain',
       'rain', 'fog', 'rain', 'rain', 'rain', 'fog', 'sun', 'rain', 'sun',
       'rain', 'rain', 'sun', 'sun', 'sun', 'sun', 'sun', 'sun', 'rain',
       'fog', 'rain', 'drizzle', 'sun', 'rain', 'rain', 'rain', 'sun',
       'sun', 'fog', 'sun', 'rain', 'rain', 'sun', 'drizzle', 'rain',
       'sun', 'rain', 'rain', 'fog', 'sun', 'rain', 'rain', 'rain',
       'drizzle', 'fog', 'drizzle', 'snow', 'rain', 'rain', 'rain', 'fog',
       'sun', 'fog', 'rain', 'rain', 'rain', 'fog', 'sun', 'sun', 'rain',
       'sun', 'sun', 'sun', 'sun', 'rain', 'drizzle', 'drizzle', 'rain',
       'rain', 'rain', 'rain', 'sun', 'rain', 'fog', 'rain', 'rain',
       'sun', 'rain', 'sun', 'rain', 'rain', 'drizzle', 'rain', 'sun',
       'fog', 'sun', 'sun', 'sun', 'rain', 'snow', 'sun', 'sun', 'sun',
       'sun', 'sun', 'sun', 'rain', 'fog', 'sun', 'sun', 'fog', 'rain',
       'sun', 'rain', 'drizzle', 'fog', 'sun', 'sun', 'rain', 'rain',
       'drizzle', 'sun', 'rain', 'sun', 'rain', 'fog', 'rain', 'rain',
       'rain', 'sun', 'sun', 'sun', 'rain', 'sun', 'rain', 'fog', 'fog',
       'rain', 'sun'], dtype=object)
```

In [27]:
```
accuracy_score(y_pred,y_test)
```

Out[27]:
```
0.6996587030716723
```

In [28]:
```
model_2=RandomForestClassifier()
rmodel=model_2.fit(x_train,y_train)
r_ypred=rmodel.predict(x_test)
r_ypred
```

```
C:\Users\Rgukt iiit\Anaconda3\anaconda\lib\site-packages\sklearn\ensemble\fores
t.py:245: FutureWarning: The default value of n_estimators will change from 10
in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
```

Out[28]:
```
array(['sun', 'drizzle', 'rain', 'sun', 'fog', 'rain', 'rain', 'rain',
       'sun', 'rain', 'sun', 'rain', 'sun', 'sun', 'drizzle', 'sun',
       'fog', 'rain', 'rain', 'rain', 'rain', 'rain', 'rain', 'sun',
       'sun', 'sun', 'sun', 'rain', 'sun', 'sun', 'rain', 'sun', 'rain',
       'rain', 'sun', 'rain', 'sun', 'rain', 'sun', 'rain', 'sun', 'rain',
       'rain', 'rain', 'sun', 'fog', 'drizzle', 'sun', 'sun', 'sun',
       'sun', 'sun', 'sun', 'rain', 'rain', 'sun', 'sun', 'rain', 'sun',
       'rain', 'sun', 'sun', 'sun', 'sun', 'fog', 'rain', 'sun', 'sun',
       'sun', 'rain', 'rain', 'rain', 'rain', 'fog', 'sun', 'sun', 'sun',
       'sun', 'rain', 'rain', 'sun', 'sun', 'rain', 'rain', 'rain', 'sun',
       'sun', 'rain', 'sun', 'rain', 'sun', 'sun', 'rain', 'sun', 'rain',
       'rain', 'sun', 'sun', 'sun', 'sun', 'rain', 'sun', 'sun', 'rain',
       'rain', 'sun', 'rain', 'rain', 'sun', 'sun', 'rain', 'sun', 'rain',
       'sun', 'sun', 'sun', 'sun', 'fog', 'rain', 'rain', 'sun', 'rain',
       'rain', 'fog', 'rain', 'sun', 'sun', 'sun', 'sun', 'rain', 'rain',
       'rain', 'sun', 'rain', 'rain', 'rain', 'sun', 'sun', 'sun', 'sun',
       'rain', 'rain', 'sun', 'sun', 'sun', 'rain', 'rain', 'sun', 'sun',
       'sun', 'sun', 'sun', 'sun', 'rain', 'fog', 'rain', 'sun', 'sun',
       'rain', 'sun', 'sun', 'drizzle', 'rain', 'sun', 'rain', 'rain',
       'fog', 'rain', 'rain', 'rain', 'sun', 'sun', 'sun', 'sun', 'rain',
       'rain', 'sun', 'sun', 'sun', 'sun', 'sun', 'sun', 'rain', 'sun',
       'rain', 'sun', 'sun', 'sun', 'rain', 'sun', 'sun', 'sun', 'sun',
       'sun', 'rain', 'rain', 'sun', 'sun', 'sun', 'sun', 'rain', 'rain',
       'fog', 'sun', 'rain', 'rain', 'rain', 'fog', 'sun', 'fog', 'rain',
       'rain', 'rain', 'rain', 'fog', 'sun', 'sun', 'rain', 'rain',
       'rain', 'fog', 'sun', 'sun', 'rain', 'sun', 'sun', 'sun', 'sun',
       'rain', 'sun', 'sun', 'rain', 'rain', 'rain', 'rain', 'sun',
       'rain', 'fog', 'rain', 'rain', 'sun', 'rain', 'sun', 'rain',
       'rain', 'sun', 'rain', 'sun', 'sun', 'sun', 'sun', 'sun', 'rain',
       'rain', 'sun', 'sun', 'sun', 'sun', 'sun', 'sun', 'rain', 'fog',
       'sun', 'sun', 'sun', 'rain', 'sun', 'rain', 'sun', 'fog', 'fog',
       'sun', 'rain', 'rain', 'drizzle', 'sun', 'rain', 'sun', 'rain',
       'sun', 'rain', 'rain', 'rain', 'sun', 'sun', 'sun', 'rain', 'sun',
       'rain', 'sun', 'sun', 'rain', 'sun'], dtype=object)
```

In [29]:
```
accuracy_score(r_ypred,y_test)
```

Out[29]:
```
0.7610921501706485
```

In [30]:
```
model_2=RandomForestClassifier(criterion='entropy',n_estimators=100)
rmodel=model_2.fit(x_train,y_train)
r_ypred=rmodel.predict(x_test)
r_ypred
```

Out[30]:
```
array(['sun', 'drizzle', 'rain', 'sun', 'fog', 'rain', 'rain', 'rain',
       'sun', 'rain', 'sun', 'rain', 'sun', 'sun', 'drizzle', 'sun',
       'fog', 'rain', 'rain', 'rain', 'rain', 'rain', 'rain', 'sun',
       'sun', 'sun', 'sun', 'rain', 'sun', 'sun', 'rain', 'sun', 'rain',
       'rain', 'sun', 'rain', 'sun', 'rain', 'sun', 'rain', 'sun', 'rain',
       'rain', 'rain', 'sun', 'sun', 'drizzle', 'sun', 'sun', 'sun',
       'sun', 'sun', 'sun', 'rain', 'rain', 'sun', 'sun', 'rain', 'sun',
       'rain', 'sun', 'sun', 'sun', 'sun', 'fog', 'rain', 'sun', 'sun',
       'sun', 'rain', 'rain', 'rain', 'rain', 'sun', 'sun', 'sun', 'sun',
       'sun', 'rain', 'rain', 'sun', 'sun', 'rain', 'rain', 'rain',
       'drizzle', 'sun', 'rain', 'sun', 'rain', 'sun', 'sun', 'rain',
       'sun', 'rain', 'rain', 'sun', 'sun', 'sun', 'sun', 'rain', 'rain',
       'sun', 'rain', 'rain', 'sun', 'rain', 'rain', 'sun', 'sun', 'rain',
       'sun', 'rain', 'sun', 'sun', 'sun', 'sun', 'sun', 'rain', 'rain',
       'sun', 'rain', 'rain', 'sun', 'rain', 'sun', 'sun', 'sun', 'sun',
       'rain', 'sun', 'rain', 'sun', 'rain', 'rain', 'rain', 'sun', 'sun',
       'sun', 'sun', 'rain', 'rain', 'sun', 'sun', 'sun', 'rain', 'rain',
       'sun', 'sun', 'sun', 'sun', 'sun', 'sun', 'rain', 'fog', 'rain',
       'sun', 'sun', 'rain', 'sun', 'sun', 'drizzle', 'rain', 'sun',
       'rain', 'rain', 'fog', 'rain', 'rain', 'rain', 'sun', 'sun', 'sun',
       'sun', 'rain', 'rain', 'sun', 'sun', 'sun', 'sun', 'sun', 'sun',
       'rain', 'sun', 'rain', 'fog', 'sun', 'rain', 'rain', 'sun', 'sun',
       'sun', 'sun', 'sun', 'rain', 'rain', 'sun', 'sun', 'sun', 'sun',
       'rain', 'sun', 'sun', 'sun', 'rain', 'rain', 'rain', 'sun', 'sun',
       'sun', 'snow', 'rain', 'rain', 'rain', 'fog', 'sun', 'sun', 'rain',
       'rain', 'rain', 'sun', 'sun', 'sun', 'rain', 'sun', 'sun', 'sun',
       'sun', 'rain', 'sun', 'fog', 'rain', 'rain', 'rain', 'rain', 'sun',
       'rain', 'sun', 'rain', 'rain', 'sun', 'rain', 'sun', 'rain',
       'rain', 'sun', 'rain', 'sun', 'sun', 'sun', 'sun', 'sun', 'rain',
       'rain', 'sun', 'sun', 'sun', 'sun', 'sun', 'sun', 'rain', 'fog',
       'sun', 'sun', 'sun', 'rain', 'sun', 'rain', 'sun', 'fog', 'sun',
       'sun', 'rain', 'rain', 'drizzle', 'sun', 'rain', 'sun', 'rain',
       'sun', 'rain', 'rain', 'rain', 'sun', 'drizzle', 'sun', 'rain',
       'sun', 'rain', 'sun', 'sun', 'rain', 'sun'], dtype=object)
```

In [31]:
```
accuracy_score(r_ypred,y_test)
```

Out[31]:
```
0.764505119453925
```

In [32]:
```
r_ypred=rmodel.predict([[0.5,29.0,26.0,14.0]])
r_ypred
```

Out[32]:
```
array(['rain'], dtype=object)
```

In [33]:
```
r_ypred=model.predict([[0,31.0,29.0,21.0]])
r_ypred
```

Out[33]:
```
array(['fog'], dtype=object)
```

In [62]:
```
r_ypred=model.predict([[5.4,31.5,28.0,28.0]])
r_ypred
```

Out[62]:
```
array(['rain'], dtype=object)
```

In [36]:
```
r_ypred=model.predict([[0,8.3,2.8,4.1]])
r_ypred
```

Out[36]:
```
array(['fog'], dtype=object)
```

In [69]:
```
new_data=[[0.0,0.0,0.0,0.0]]
precipitation=float(input("enter precipitation"))
temp_max=float(input("enter max temperature"))
temp_min=float(input("enter min temperature"))
wind=float(input("enter wind"))
new_data=[[precipitation,temp_max,temp_min,wind]]
if(temp_max==0 or temp_min==0 or wind==0):
    print('invalid')
else:
    r_ypred=model.predict(new_data)
    print(r_ypred)
```

```
enter precipitation0
enter max temperature0
enter min temperature0
enter wind0
invalid
```

In [ ]: