weather-prediction

May 31, 2023

[633]:
```
import numpy as np
import pandas as pd
```

[634]:
```
df=pd.read_csv('daily_weather.csv')
```

[635]:
```
df.head(3)
```

[635]:
```
   number  air_pressure_9am  air_temp_9am  avg_wind_direction_9am  \
0       0        918.060000     74.822000              271.100000
1       1        917.347688     71.403843              101.935179
2       2        923.040000     60.638000               51.000000

   avg_wind_speed_9am  max_wind_direction_9am  max_wind_speed_9am  \
0            2.080354              295.400000            2.863283
1            2.443009              140.471548            3.533324
2           17.067852               63.700000           22.100967

   rain_accumulation_9am  rain_duration_9am  relative_humidity_9am  \
0                    0.0                0.0              42.420000
1                    0.0                0.0              24.328697
2                    0.0               20.0               8.900000

   relative_humidity_3pm
0              36.160000
1              19.426597
2              14.460000
```

[636]:
```
df.tail()
```

[636]:
```
      number  air_pressure_9am  air_temp_9am  avg_wind_direction_9am  \
1090    1090            918.90        63.104                   192.9
1091    1091            918.71        49.568                   241.6
1092    1092            916.60        71.096                   189.3
1093    1093            912.60        58.406                   172.7
1094    1094            921.53        77.702                    97.1

      avg_wind_speed_9am  max_wind_direction_9am  max_wind_speed_9am  \
```

```
1090            3.869906                   207.3            5.212070
1091            1.811921                   227.4            2.371156
1092            3.064608                   200.8            3.892276
1093            3.825167                   189.1            4.764682
1094            3.265932                   125.9            4.451511

      rain_accumulation_9am  rain_duration_9am  relative_humidity_9am  \
1090                    0.0                0.0                  26.02
1091                    0.0                0.0                  90.35
1092                    0.0                0.0                  45.59
1093                    0.0                0.0                  64.84
1094                    0.0                0.0                  14.56

      relative_humidity_3pm
1090                  38.18
1091                  73.34
1092                  52.31
1093                  58.28
1094                  15.10
```

[637]:
```
df.shape
```

[637]:
```
(1095, 11)
```

[638]:
```
df.isnull().sum()
```

[638]:
```
number                   0
air_pressure_9am         3
air_temp_9am             5
avg_wind_direction_9am   4
avg_wind_speed_9am       3
max_wind_direction_9am   3
max_wind_speed_9am       4
rain_accumulation_9am    6
rain_duration_9am        3
relative_humidity_9am    0
relative_humidity_3pm    0
dtype: int64
```

[639]:
```
df.dropna(inplace=True)
```

[640]:
```
df.isnull().sum()
```

[640]:
```
number                   0
air_pressure_9am         0
air_temp_9am             0
avg_wind_direction_9am   0
```

```
avg_wind_speed_9am        0
max_wind_direction_9am    0
max_wind_speed_9am        0
rain_accumulation_9am     0
rain_duration_9am         0
relative_humidity_9am     0
relative_humidity_3pm     0
dtype: int64
```

[641]:
```
df.head(2)
```

[641]:
```
   number  air_pressure_9am  air_temp_9am  avg_wind_direction_9am  \
0       0        918.060000     74.822000              271.100000
1       1        917.347688     71.403843              101.935179

   avg_wind_speed_9am  max_wind_direction_9am  max_wind_speed_9am  \
0            2.080354              295.400000            2.863283
1            2.443009              140.471548            3.533324

   rain_accumulation_9am  rain_duration_9am  relative_humidity_9am  \
0                    0.0                0.0              42.420000
1                    0.0                0.0              24.328697

   relative_humidity_3pm
0              36.160000
1              19.426597
```

[642]:
```
df.drop(columns='number',inplace=True)
```

[643]:
```
df
```

[643]:
```
      air_pressure_9am  air_temp_9am  avg_wind_direction_9am  \
0           918.060000     74.822000              271.100000
1           917.347688     71.403843              101.935179
2           923.040000     60.638000               51.000000
3           920.502751     70.138895              198.832133
4           921.160000     44.294000              277.800000
…                  …              …                       …
1090        918.900000     63.104000              192.900000
1091        918.710000     49.568000              241.600000
1092        916.600000     71.096000              189.300000
1093        912.600000     58.406000              172.700000
1094        921.530000     77.702000               97.100000

      avg_wind_speed_9am  max_wind_direction_9am  max_wind_speed_9am  \
0               2.080354              295.400000            2.863283
1               2.443009              140.471548            3.533324
```

```
2              17.067852              63.700000          22.100967
3               4.337363             211.203341           5.190045
4               1.856660             136.500000           2.863283
…                    …                     …                  …
1090            3.869906             207.300000           5.212070
1091            1.811921             227.400000           2.371156
1092            3.064608             200.800000           3.892276
1093            3.825167             189.100000           4.764682
1094            3.265932             125.900000           4.451511

      rain_accumulation_9am  rain_duration_9am  relative_humidity_9am  \
0                       0.0                0.0              42.420000
1                       0.0                0.0              24.328697
2                       0.0               20.0               8.900000
3                       0.0                0.0              12.189102
4                       8.9            14730.0              92.410000
…                       …                  …                  …
1090                    0.0                0.0              26.020000
1091                    0.0                0.0              90.350000
1092                    0.0                0.0              45.590000
1093                    0.0                0.0              64.840000
1094                    0.0                0.0              14.560000

      relative_humidity_3pm
0                 36.160000
1                 19.426597
2                 14.460000
3                 12.742547
4                 76.740000
…                       …
1090              38.180000
1091              73.340000
1092              52.310000
1093              58.280000
1094              15.100000

[1064 rows x 10 columns]
```

[644]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1064 entries, 0 to 1094
Data columns (total 10 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   air_pressure_9am       1064 non-null   float64
 1   air_temp_9am           1064 non-null   float64
```

```
 2   avg_wind_direction_9am  1064 non-null   float64
 3   avg_wind_speed_9am      1064 non-null   float64
 4   max_wind_direction_9am  1064 non-null   float64
 5   max_wind_speed_9am      1064 non-null   float64
 6   rain_accumulation_9am   1064 non-null   float64
 7   rain_duration_9am       1064 non-null   float64
 8   relative_humidity_9am   1064 non-null   float64
 9   relative_humidity_3pm   1064 non-null   float64
dtypes: float64(10)
memory usage: 91.4 KB
```

[645]:
```
df.describe()
```

[645]:

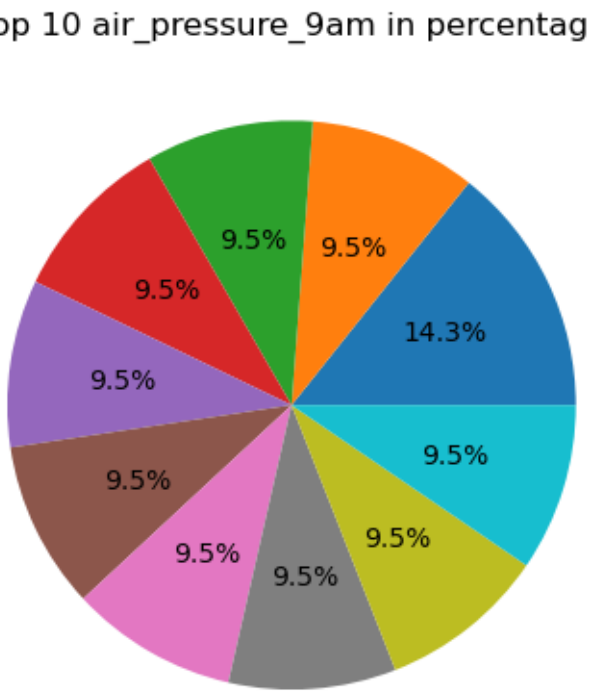
	air_pressure_9am	air_temp_9am	avg_wind_direction_9am	avg_wind_speed_9am	max_wind_direction_9am	max_wind_speed_9am	rain_accumulation_9am	rain_duration_9am	relative_humidity_9am	relative_humidity_3pm
count	1064.000000	1064.000000	1064.000000	1064.000000	1064.000000	1064.000000	1064.000000	1064.000000	1064.000000	1064.000000
mean	918.903180	65.022609	142.306756	5.485793	148.480424	6.999714	0.182023	266.393697	34.077440	35.148381
std	3.179040	11.168033	69.149472	4.534427	67.154911	5.590790	1.534493	1503.092216	25.356668	22.365475
min	907.990000	36.752000	15.500000	0.693451	28.900000	1.185578	0.000000	0.000000	6.090000	
25%	916.595376	57.398000	65.979244	2.245529	76.335351	3.064608	0.000000	0.000000	15.093365	
50%	918.942281	65.778479	165.937461	3.869906	176.350000	4.943637	0.000000	0.000000	23.135000	
75%	921.169054	73.530872	191.100000	7.264463	201.125000	8.747888	0.000000	0.000000	44.660000	
max	929.320000	98.906000	343.400000	23.554978	312.200000	29.840780	24.020000	17704.000000	92.620000	

```
min              5.300000
25%             17.360468
50%             24.371286
75%             51.922500
max             92.250000
```

[646]:
```
pie=df['air_pressure_9am'].value_counts().head(10)
```

[647]:
```
import matplotlib.pyplot as plt
plt.pie(pie,autopct='%1.1f%%')
plt.title('Top 10 air_pressure_9am in percentage')
```

[647]:
```
Text(0.5, 1.0, 'Top 10 air_pressure_9am in percentage')
```



[648]:
```
pie
```

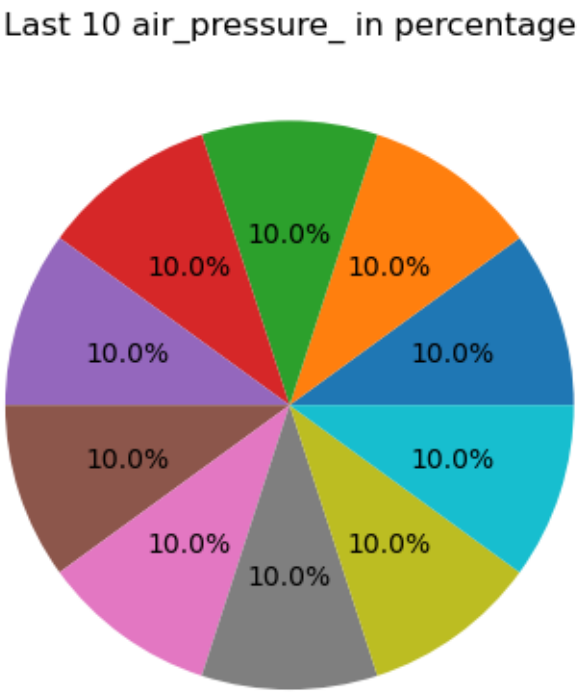
[648]:
```
918.60    3
915.20    2
917.60    2
917.01    2
919.30    2
925.30    2
```

```
921.34    2
923.60    2
918.71    2
916.10    2
Name: air_pressure_9am, dtype: int64
```

[649]:
```
pie2=df['air_pressure_9am'].value_counts().tail(10)
```

[650]:
```
plt.pie(pie2,autopct='%1.1f%%')
plt.title('Last 10 air_pressure_ in percentage')
```

[650]: Text(0.5, 1.0, 'Last 10 air_pressure_ in percentage')



[651]:
```
df.head()
```

[651]:

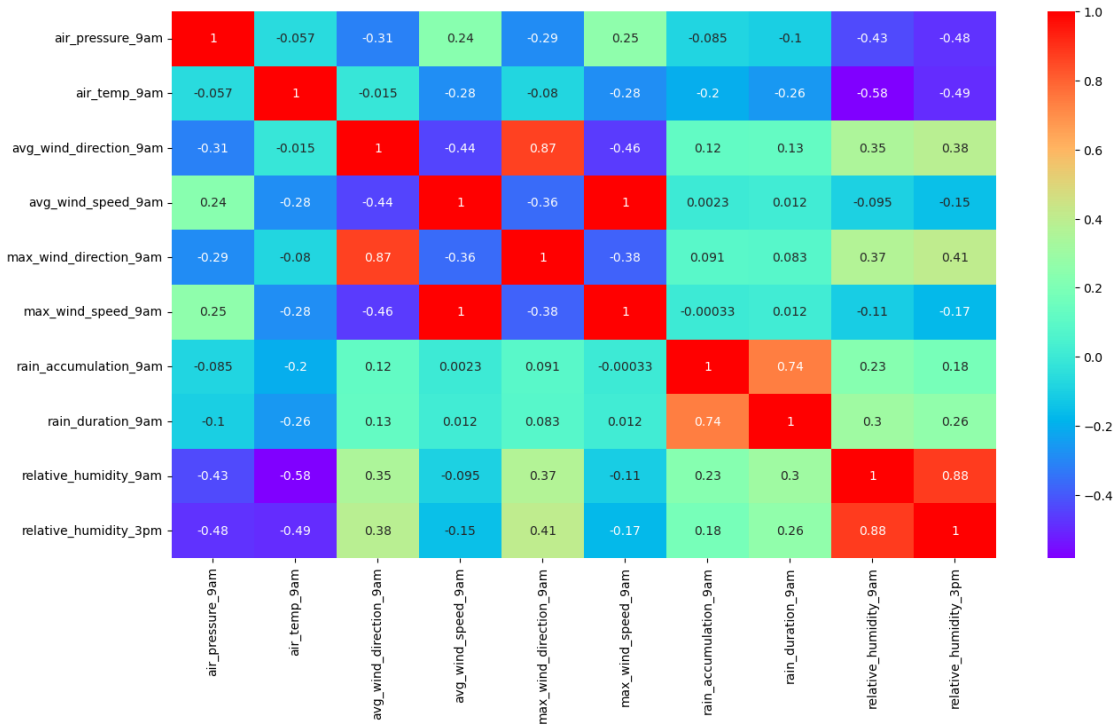
	air_pressure_9am	air_temp_9am	avg_wind_direction_9am	avg_wind_speed_9am	\
0	918.060000	74.822000	271.100000	2.080354	
1	917.347688	71.403843	101.935179	2.443009	
2	923.040000	60.638000	51.000000	17.067852	
3	920.502751	70.138895	198.832133	4.337363	
4	921.160000	44.294000	277.800000	1.856660	

	max_wind_direction_9am	max_wind_speed_9am	rain_accumulation_9am	\
0	295.400000	2.863283	0.0	
1	140.471548	3.533324	0.0	
2	63.700000	22.100967	0.0	
3	211.203341	5.190045	0.0	
4	136.500000	2.863283	8.9	

	rain_duration_9am	relative_humidity_9am	relative_humidity_3pm
0	0.0	42.420000	36.160000
1	0.0	24.328697	19.426597
2	20.0	8.900000	14.460000
3	0.0	12.189102	12.742547
4	14730.0	92.410000	76.740000

[652]:
```
import seaborn as sns
plt.figure(figsize=(15,8))
sns.heatmap(df.corr(),annot=True,cmap='rainbow')
```

[652]: <Axes: >



[ ]:
```
df.info()
```

[653]:
```
x=df.drop(columns=['relative_humidity_3pm'],axis=1)
```

[654]:
```
y=df['relative_humidity_3pm']
```

[655]:
```
from sklearn.model_selection import train_test_split
```

[656]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
```

[657]:
```
df.shape
```

[657]: (1064, 10)

[658]:
```
x_train.shape,x_test.shape,y_train.shape,y_test.shape
```

[658]: ((851, 9), (213, 9), (851,), (213,))

[659]:
```
from sklearn.tree import DecisionTreeRegressor
```

[660]:
```
model= DecisionTreeRegressor()
```

[661]:
```
model.fit(x_train,y_train)
```

[661]: DecisionTreeRegressor()

[662]:
```
model.predict(x_test)
```

[662]:
```
array([21.73      , 39.52      ,  7.30218051, 71.65      , 20.3686575 ,
       44.8       , 71.9       , 32.97      , 25.14      , 44.82      ,
       19.60693827, 13.77      , 77.87      , 21.86553243, 44.27      ,
       19.52606277, 91.16      , 23.8       , 17.33744917, 14.41069359,
       17.75      , 53.02      , 13.48750448, 14.05      , 68.05      ,
       54.77      , 15.25344497, 12.69854598, 18.65026305, 44.85      ,
       21.28846893, 24.37743921, 10.376058  , 14.05374792, 69.67      ,
       44.82      , 54.77      , 56.25      , 48.38      , 14.27588983,
       88.16      , 22.60694278,  8.03      , 13.21      , 53.17      ,
       13.55961808, 13.62      , 12.52      , 75.97      , 71.9       ,
        8.62      , 14.15673274, 14.05      , 13.21      , 20.52      ,
       13.48      , 66.57      , 21.77      , 22.18990678, 84.39      ,
       16.87177039, 23.03597591, 88.67      , 30.57      , 10.376058  ,
       19.59      , 15.52      , 43.96      , 46.6       , 13.77      ,
       42.95      , 62.81      , 84.39      , 12.36298408, 64.52      ,
       14.64990936, 74.1       , 20.9       , 18.27451897, 13.21      ,
       14.15673274, 19.92657856, 51.1       , 53.23      , 28.82      ,
       55.52      , 71.9       , 17.45      , 49.39      ,  7.52      ,
       68.39      , 44.59      ,  7.94640849, 54.77      , 19.2       ,
       17.71      , 53.84      , 13.52559044, 33.19      , 20.04530549,
       45.19      , 21.03831789, 15.25344497, 22.72      , 19.23      ,
       30.        , 20.21      , 79.38      , 19.52606277, 15.64703999,
       17.3342963 , 18.9414337 , 16.68      , 19.04826571, 79.09      ,
```

```
       11.43602921, 38.85      , 17.76      , 18.25795332, 27.45      ,
       28.82      , 21.15      , 90.06      , 15.46261112, 76.94      ,
       73.65      , 40.91      , 47.58      ,  8.76931379, 23.8       ,
       44.27      , 19.04826571, 23.8       , 52.12      , 51.52      ,
       36.55      , 12.15540564, 21.28846893, 37.8       , 14.22808216,
       68.27      , 11.10544045, 91.22      , 15.87360525, 44.82      ,
       22.77      , 21.28846893, 74.1       , 63.44      , 21.73      ,
       11.24      ,  9.98407134, 42.95      , 48.81      , 69.19      ,
       20.9       , 19.68      , 19.23      , 28.17      , 48.98      ,
       42.22      , 16.81584367, 28.12      , 53.02      , 21.16905026,
       13.17      , 19.30666431, 69.74      , 27.45      , 28.43      ,
       78.51      , 43.96      , 13.35131297, 28.82      , 22.16642603,
       23.11818221, 14.46      , 27.45      , 22.72      , 68.39      ,
       56.93      , 68.05      , 17.63      , 51.1       , 14.05      ,
       22.4       , 42.22      , 22.78455864, 17.28      , 76.94      ,
       24.37743921, 20.71      , 17.76      , 91.22      , 19.2       ,
       39.21      , 12.01883617, 53.02      ,  7.94640849, 85.96      ,
       22.60694278, 26.94      , 48.38      , 27.88      , 59.69      ,
       12.52      , 17.02234251, 25.11      , 16.66230871, 80.61      ,
       13.09702532, 52.97      , 76.7       ])
```

[663]:
```
model.score(x_test,y_test)
```

[663]:
```
0.7416526810297336
```

[664]:
```
df.head()
```

[664]:

	air_pressure_9am	air_temp_9am	avg_wind_direction_9am	avg_wind_speed_9am	max_wind_direction_9am	max_wind_speed_9am	rain_accumulation_9am	rain_duration_9am	relative_humidity_9am	relative_humidity_3pm
0	918.060000	74.822000	271.100000	2.080354	295.400000	2.863283	0.0	0.0	42.420000	36.160000
1	917.347688	71.403843	101.935179	2.443009	140.471548	3.533324	0.0	0.0	24.328697	19.426597
2	923.040000	60.638000	51.000000	17.067852	63.700000	22.100967	0.0	20.0	8.900000	14.460000
3	920.502751	70.138895	198.832133	4.337363	211.203341	5.190045	0.0	0.0	12.189102	12.742547
4	921.160000	44.294000	277.800000	1.856660	136.500000	2.863283	8.9	14730.0	92.410000	76.740000

[665]:
```
x=df.
  ↪drop(columns=['relative_humidity_3pm','max_wind_speed_9am','rain_accumulation_9am','avg_win
y=df['relative_humidity_3pm']
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
```

[699]:
```
mode_r1= DecisionTreeRegressor(max_leaf_nodes=10)
```

[700]:
```
mode_r1.fit(x_train,y_train)
```

[700]:
```
DecisionTreeRegressor(max_leaf_nodes=10)
```

[701]:
```
mode_r1.score(x_test,y_test)
```

[701]:
```
0.8142315832148777
```

[702]:
```
from sklearn.preprocessing import StandardScaler
```

[703]:
```
scale=StandardScaler()
```

[704]:
```
scale.fit(x_train,y_train)
```

[704]:
```
StandardScaler()
```

[705]:
```
x_test_scale=scale.fit_transform(x_test)
x_train_scale=scale.fit_transform(x_train)
```

[706]:
```
mode_r1.fit(x_train_scale,y_train)
```

[706]:
```
DecisionTreeRegressor(max_leaf_nodes=10)
```

[707]:
```
mode_r1.score(x_test_scale,y_test)
```

[707]:
```
0.7961597898639061
```

[708]:
```
from sklearn.ensemble import RandomForestRegressor
```

[709]:
```
model2=RandomForestRegressor()
```

[710]:
```
model2.fit(x_train,y_train)
```

[710]:
```
RandomForestRegressor()
```

[712]:
```
model2.score(x_test_scale,y_test)
```

[712]:
```
0.821154255074222
```

[711]:
```
model2.fit(x_train_scale,y_train)
```

[711]:
```
RandomForestRegressor()
```

[713]:
```
model2.score(x_test_scale,y_test)
```

[713]:
```
0.821154255074222
```

[715]:
```
from sklearn.linear_model import LinearRegression
```

[716]:
```
model3=LinearRegression()
```

[717]:
```
model3.fit(x_train,y_train)
```

[717]:
```
LinearRegression()
```

[718]:
```
model3.score(x_test,y_test)
```

[718]:
```
0.8117287686999615
```

[719]:
```
model3.fit(x_train_scale,y_train)
```

[719]:
```
LinearRegression()
```

[720]:
```
model3.score(x_test_scale,y_test)
```

[720]:
```
0.8021842202929085
```

[721]:
```
from sklearn.preprocessing import PolynomialFeatures
```

[722]:
```
from sklearn.pipeline import make_pipeline
```

[723]:
```
model_0 = make_pipeline(PolynomialFeatures(degree=1), LinearRegression())
```

[724]:
```
model_0.fit(x_train, y_train)
```

[724]:
```
Pipeline(steps=[('polynomialfeatures', PolynomialFeatures(degree=1)),
                ('linearregression', LinearRegression())])
```

[725]:
```
model_0.score(x_test, y_test)
```

[725]:
```
0.8117287686999659
```

[726]:
```
model_01 = make_pipeline(PolynomialFeatures(degree=2), DecisionTreeRegressor())
```

[727]:
```
model_01.fit(x_train, y_train)
```

[727]:
```
Pipeline(steps=[('polynomialfeatures', PolynomialFeatures()),
                ('decisiontreeregressor', DecisionTreeRegressor())])
```

[728]:
```
model_01.score(x_test, y_test)
```

[728]:
```
0.7226019380451649
```

[760]:
```
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.feature_selection import SelectKBest
from sklearn.ensemble import RandomForestClassifier

pipe = Pipeline([
    ('scaler', StandardScaler()),
    ('kbest', SelectKBest(k=6)),
    ('rf', RandomForestRegressor(n_estimators=10))
])
```

[761]:
```
pipe.fit(x_train, y_train)
```

```
C:\Users\User\AppData\Roaming\Python\Python310\site-
packages\sklearn\feature_selection\_univariate_selection.py:113: RuntimeWarning:
divide by zero encountered in divide
  f = msb / msw
```

[761]:
```
Pipeline(steps=[('scaler', StandardScaler()), ('kbest', SelectKBest(k=6)),
                ('rf', RandomForestRegressor(n_estimators=10))])
```

[762]:
```
pipe.score(x_test, y_test)
```

[762]:
```
0.8387840758392073
```

[763]:
```
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.feature_selection import SelectKBest
from sklearn.ensemble import RandomForestClassifier

pipe = Pipeline([
    ('scaler', StandardScaler()),
    ('kbest', SelectKBest(k=6)),
    ('rf',LinearRegression())
])
```

[764]:
```
pipe.fit(x_train, y_train)
```

```
C:\Users\User\AppData\Roaming\Python\Python310\site-
packages\sklearn\feature_selection\_univariate_selection.py:113: RuntimeWarning:
divide by zero encountered in divide
  f = msb / msw
```

[764]:
```
Pipeline(steps=[('scaler', StandardScaler()), ('kbest', SelectKBest(k=6)),
                ('rf', LinearRegression())])
```

[765]:
```
pipe.score(x_test, y_test)
```

[765]:
```
0.8117287686999617
```

[873]:
```
from sklearn.pipeline import Pipeline
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import RandomizedSearchCV
from sklearn.preprocessing import StandardScaler

# Define the pipeline
pipeline = Pipeline([
    ('scaler', StandardScaler()),
    ('classifier', RandomForestRegressor()),
])

# Define the parameter grid to search over
param_grid = {
    'classifier__n_estimators': [100, 200, 300],
    'classifier__max_depth': [5, 10, 15, 20],
    'classifier__min_samples_split': [2, 5, 10],
    'classifier__min_samples_leaf': [1, 2, 4],
}

# Define the RandomizedSearchCV object
search = RandomizedSearchCV(
    pipeline,
    param_grid,
    n_iter=2,
    cv=2,
    verbose=5,
    n_jobs=-1
)
```

[874]:
```
search.fit(x_train, y_train)
```

```
Fitting 2 folds for each of 2 candidates, totalling 4 fits
```

[874]:
```
RandomizedSearchCV(cv=2,
                   estimator=Pipeline(steps=[('scaler', StandardScaler()),
                                             ('classifier',
                                              RandomForestRegressor())]),
                   n_iter=2, n_jobs=-1,
                   param_distributions={'classifier__max_depth': [5, 10, 15,
                                                                  20],
```

```
                                                'classifier__min_samples_leaf': [1, 2,
                                                                                 4],
                                                'classifier__min_samples_split': [2, 5,
                                                                                  10],
                                                'classifier__n_estimators': [100, 200,
                                                                             300]},
                             verbose=5)
```

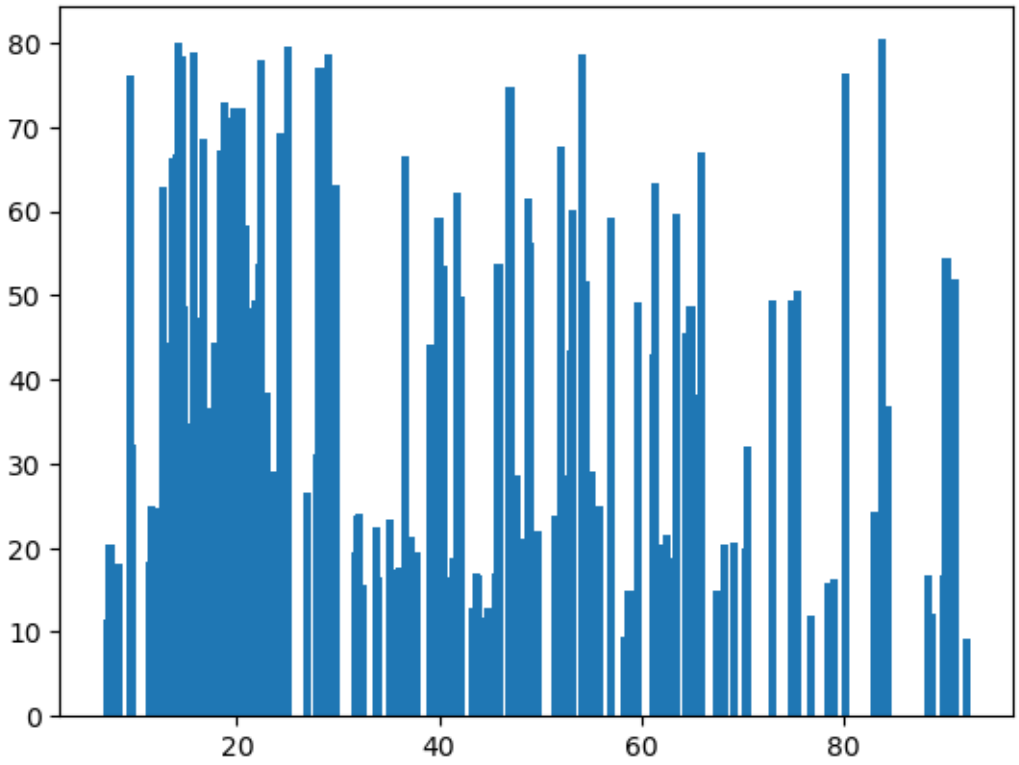
[881]:
```
search.score(x_test, y_test)
```

[881]:
```
0.8409377024435153
```

[900]:
```
humidity_3pm = df.relative_humidity_3pm[:213]
```

[901]:
```
plt.bar(humidity_3pm, bar)
```

[901]:
```
<BarContainer object of 213 artists>
```



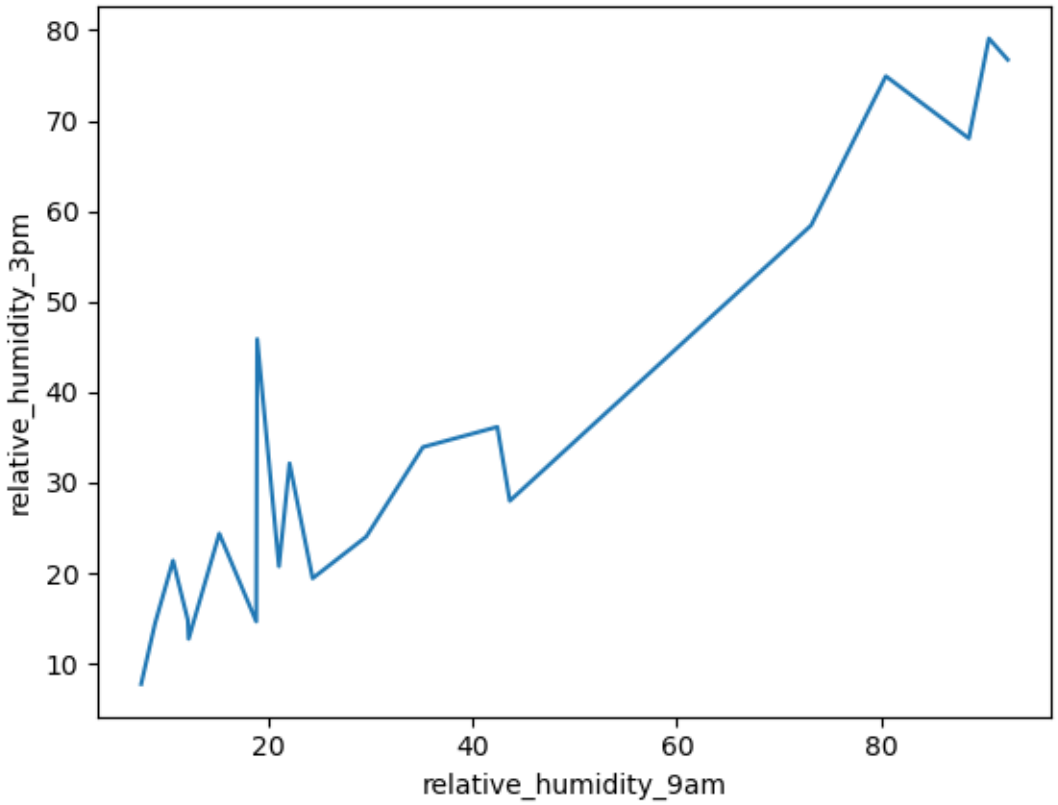
[892]:
```
df.head(2)
```

[892]:
```
    air_pressure_9am  air_temp_9am  avg_wind_direction_9am  avg_wind_speed_9am  \
0         918.060000     74.822000              271.100000            2.080354
```

```
1        917.347688     71.403843              101.935179             2.443009

   max_wind_direction_9am  max_wind_speed_9am  rain_accumulation_9am  \
0              295.400000            2.863283                    0.0
1              140.471548            3.533324                    0.0

   rain_duration_9am  relative_humidity_9am  relative_humidity_3pm
0                0.0              42.420000              36.160000
1                0.0              24.328697              19.426597
```

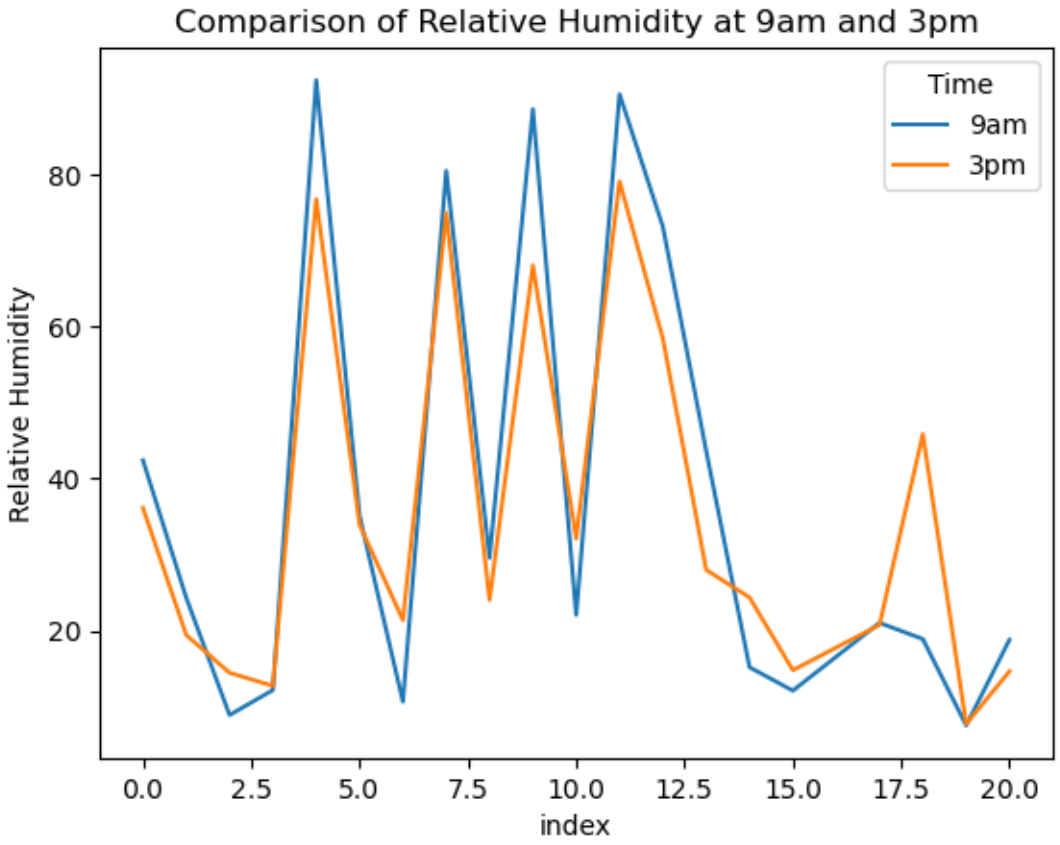
[910]:
```
first=df.relative_humidity_9am.head(20)
second=df.relative_humidity_3pm.head(20)
sns.lineplot(x=first,y=second)
```

[910]: <Axes: xlabel='relative_humidity_9am', ylabel='relative_humidity_3pm'>



[933]:
```
import seaborn as sns
import matplotlib.pyplot as plt

first = df.relative_humidity_9am.head(20)
second = df.relative_humidity_3pm.head(20)
```

```
data = pd.DataFrame({'9am': first, '3pm': second})


data = data.reset_index()


data = data.melt('index', var_name='Time', value_name='Relative Humidity')


sns.lineplot(data=data, x='index', y='Relative Humidity', hue='Time')



plt.ylabel('Relative Humidity')
plt.title('Comparison of Relative Humidity at 9am and 3pm')
plt.show()
```



[927]:
```
columns=['relative_humidity_3pm','relative_humidity_9am']
```

[928]:
```
df[columns]
```

[928]:
```
      relative_humidity_3pm  relative_humidity_9am
0                 36.160000              42.420000
1                 19.426597              24.328697
2                 14.460000               8.900000
3                 12.742547              12.189102
4                 76.740000              92.410000
…                       …                      …
1090              38.180000              26.020000
1091              73.340000              90.350000
1092              52.310000              45.590000
1093              58.280000              64.840000
1094              15.100000              14.560000

[1064 rows x 2 columns]
```

[915]:
```
df.head(2)
```

[915]:
```
   air_pressure_9am  air_temp_9am  avg_wind_direction_9am  avg_wind_speed_9am  \
0        918.060000     74.822000              271.100000            2.080354
1        917.347688     71.403843              101.935179            2.443009

   max_wind_direction_9am  max_wind_speed_9am  rain_accumulation_9am  \
0              295.400000            2.863283                    0.0
1              140.471548            3.533324                    0.0

   rain_duration_9am  relative_humidity_9am  relative_humidity_3pm
0                0.0              42.420000              36.160000
1                0.0              24.328697              19.426597
```

[924]:
```
df.columns
```

[924]:
```
Index(['air_pressure_9am', 'air_temp_9am', 'avg_wind_direction_9am',
       'avg_wind_speed_9am', 'max_wind_direction_9am', 'max_wind_speed_9am',
       'rain_accumulation_9am', 'rain_duration_9am', 'relative_humidity_9am',
       'relative_humidity_3pm'],
      dtype='object')
```

[935]:
```
pred=search.predict(x_test)
```

[936]:
```
pred
```

[936]:
```
array([17.57243845, 23.77754529, 16.95008649, 16.23809775, 11.91841954,
       16.4803811 , 48.49464826, 26.46887882, 14.47584603,  9.40653724,
       24.0250184 , 16.19651861,  9.519366  , 31.14660244, 69.26240659,
       14.94165528, 58.35702346, 53.68780015, 11.76686343, 14.65498221,
```

```
       12.97874143, 21.34095256, 66.97581498, 17.46205321, 74.70343472,
       52.00939102, 16.82342456, 36.76437211, 21.33642087, 16.60116126,
       15.23191178, 23.87241921, 15.22239994, 16.57384305, 39.86091483,
       16.65948581, 59.2497227 , 16.80865784, 18.5235033 , 63.3059451 ,
       10.76640843, 15.02593408, 18.54318033, 66.38477922, 61.49880682,
       20.5437302 , 48.68767859, 72.14201337, 44.49042717, 10.70604985,
       78.60939765, 62.93346575, 11.48115603, 72.91516903, 62.08392586,
       76.45409682, 44.29057488, 20.62282615, 20.42891192, 64.02294837,
       16.70774834, 15.17955118, 11.71364036, 17.64697472, 45.83503716,
       27.29131362, 18.4208507 , 19.48561187, 44.1288649 , 52.78051355,
       29.30849223, 18.11050788, 51.59674448, 44.21592687, 48.7511616 ,
       79.9215127 , 47.32151614, 53.4624155 , 59.25726656, 26.3796573 ,
       18.75162672, 67.76444309, 17.8541254 , 26.61438934, 59.79107825,
       19.18905097, 23.18612873, 57.61944194, 29.04241619, 23.67777368,
       49.81289783, 16.66850841, 49.45989075, 14.50549918, 66.86044206,
       26.67416756, 67.17034298, 15.8326133 , 23.36276116, 26.57944828,
       70.94165738, 17.45527987, 80.40403665, 32.24090993, 17.9286429 ,
       38.19385379, 24.75122554, 15.26457471, 66.25852718, 25.6905802 ,
       12.76291363, 21.13518022, 24.16921558, 12.96065411, 53.60239179,
       36.54354668, 63.18385587, 45.50531681, 77.13383103, 12.66684859,
       22.37806395, 40.47163421, 66.57457705, 61.24944428, 19.27881255,
       42.02096509, 17.36823668, 19.05929126, 21.91498768, 31.99101065,
       19.39270086, 23.93809152, 30.13030535, 28.64643014,  9.96692614,
       49.23248637, 49.35538668, 30.40631521, 16.87972701, 23.68795133,
       30.34141264, 17.06456603, 18.88063582, 78.558013  , 47.44369879,
       17.03475024, 13.28221761, 20.42708138, 17.26652423, 15.02218844,
       11.73889446, 29.1763011 , 26.18290365, 21.51521139, 24.77429214,
       50.99727859, 38.44044377, 49.36757333, 17.74998581, 50.54576354,
       16.23919355, 33.43181118, 24.81092724, 20.45618373, 25.03577347,
       28.68566477, 15.02218844, 12.2432319 , 71.08293024, 21.25306833,
       16.17957236, 18.61009884, 16.6559143 , 19.58078439, 77.90475452,
       79.53238779, 25.02273207, 43.11147672, 56.14268016, 16.56988981,
       34.77261083, 17.44461617, 76.19047319, 43.50783339, 53.83071621,
       72.28141966, 54.43222674, 19.19783723, 44.34388574, 18.97128044,
       28.72472229, 19.85879655, 21.0861607 , 15.21248029, 78.42215647,
       16.62072307, 23.74517324, 24.97602047, 19.70828239, 18.3825252 ,
       31.73416282,  9.21498814, 24.20264126, 15.48349726, 60.06214547,
       14.91209476, 21.13816854, 71.12341299, 78.77885676, 38.12668596,
       20.31708064, 68.62622195, 23.93336637])
```

[938]:
```
pred.shape
```

[938]: (213,)

[940]:
```
y_test.shape
```

[940]: (213,)

[949]:
```
df.describe()
```

[949]:

	air_pressure_9am	air_temp_9am	avg_wind_direction_9am
count	1064.000000	1064.000000	1064.000000
mean	918.903180	65.022609	142.306756
std	3.179040	11.168033	69.149472
min	907.990000	36.752000	15.500000
25%	916.595376	57.398000	65.979244
50%	918.942281	65.778479	165.937461
75%	921.169054	73.530872	191.100000
max	929.320000	98.906000	343.400000

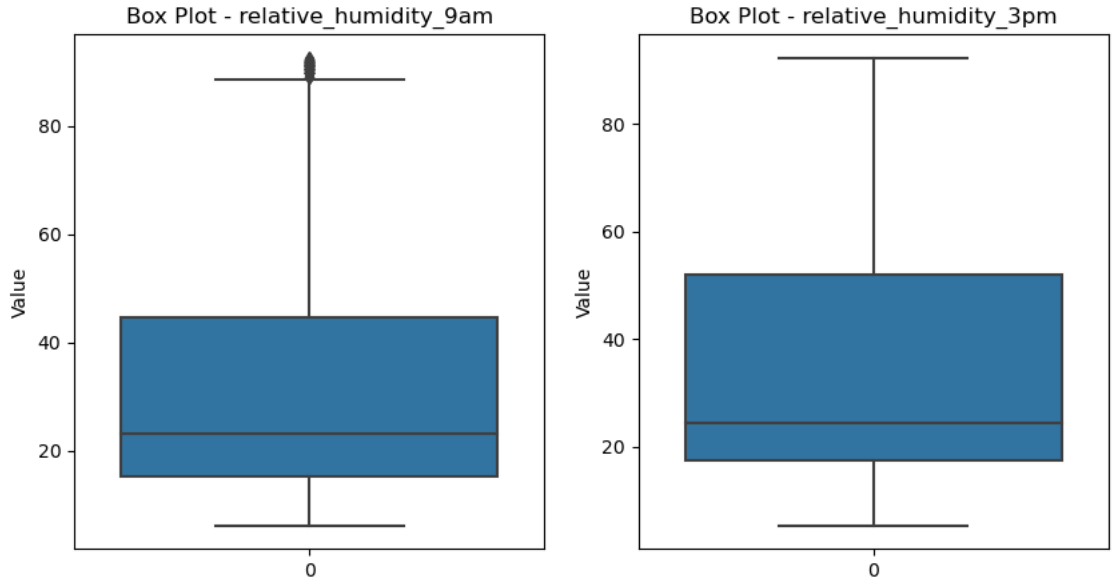
	avg_wind_speed_9am	max_wind_direction_9am	max_wind_speed_9am
count	1064.000000	1064.000000	1064.000000
mean	5.485793	148.480424	6.999714
std	4.534427	67.154911	5.590790
min	0.693451	28.900000	1.185578
25%	2.245529	76.335351	3.064608
50%	3.869906	176.350000	4.943637
75%	7.264463	201.125000	8.747888
max	23.554978	312.200000	29.840780

	rain_accumulation_9am	rain_duration_9am	relative_humidity_9am
count	1064.000000	1064.000000	1064.000000
mean	0.182023	266.393697	34.077440
std	1.534493	1503.092216	25.356668
min	0.000000	0.000000	6.090000
25%	0.000000	0.000000	15.093365
50%	0.000000	0.000000	23.135000
75%	0.000000	0.000000	44.660000
max	24.020000	17704.000000	92.620000

	relative_humidity_3pm
count	1064.000000
mean	35.148381
std	22.365475
min	5.300000
25%	17.360468
50%	24.371286
75%	51.922500
max	92.250000

[962]:
```
import seaborn as sns
import matplotlib.pyplot as plt

columns = ['relative_humidity_9am', 'relative_humidity_3pm']
```

```
fig, axes = plt.subplots(nrows=1, ncols=len(columns), figsize=(10, 5))


for i, column in enumerate(columns):
    sns.boxplot(data=df[column], ax=axes[i])
    axes[i].set_ylabel('Value')
    axes[i].set_title(f'Box Plot - {column}')
```



[964]:
```
df.head(2)
```

[964]:

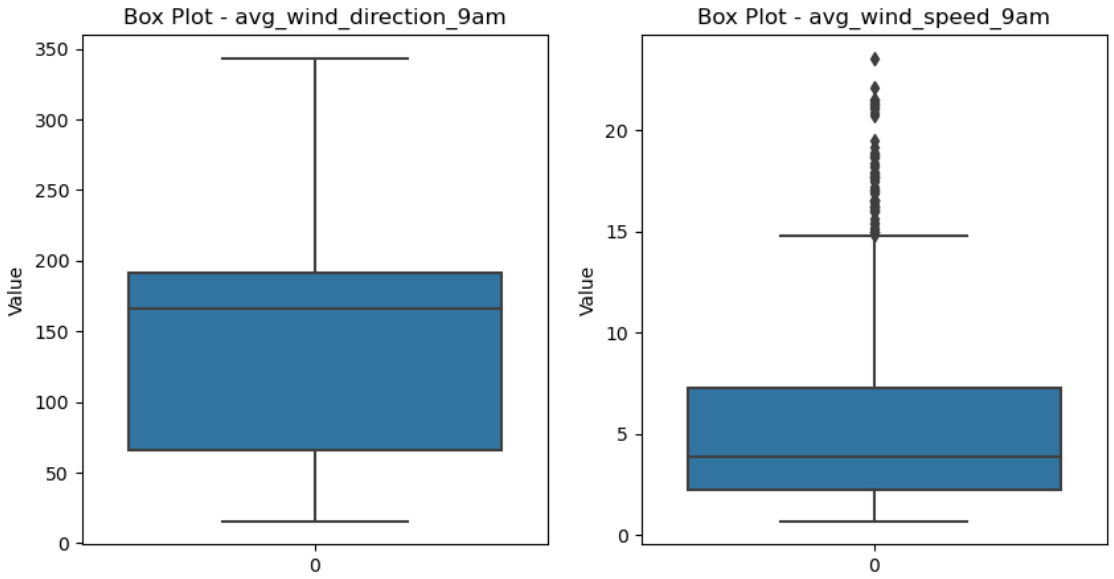
	air_pressure_9am	air_temp_9am	avg_wind_direction_9am	avg_wind_speed_9am	\
0	918.060000	74.822000	271.100000	2.080354	
1	917.347688	71.403843	101.935179	2.443009	

	max_wind_direction_9am	max_wind_speed_9am	rain_accumulation_9am	\
0	295.400000	2.863283	0.0	
1	140.471548	3.533324	0.0	

	rain_duration_9am	relative_humidity_9am	relative_humidity_3pm
0	0.0	42.420000	36.160000
1	0.0	24.328697	19.426597

[965]:
```
import seaborn as sns
import matplotlib.pyplot as plt

columns = ['avg_wind_direction_9am', 'avg_wind_speed_9am']
```

```
fig, axes = plt.subplots(nrows=1, ncols=len(columns), figsize=(10, 5))


for i, column in enumerate(columns):
    sns.boxplot(data=df[column], ax=axes[i])
    axes[i].set_ylabel('Value')
    axes[i].set_title(f'Box Plot - {column}')
```



[966]:
```
import pandas as pd
import numpy as np

# Assuming you have a DataFrame called 'df' with multiple columns

# Define a function to detect outliers using Tukey's fences method
def detect_outliers(data):
    Q1 = np.percentile(data, 25)
    Q3 = np.percentile(data, 75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    outliers = data[(data < lower_bound) | (data > upper_bound)]
    return outliers

# Iterate over each column and find outliers
for column in df.columns:
```

```
    outliers = detect_outliers(df[column])

    if outliers.empty:
        print(f"No outliers found in column: {column}")
    else:
        print(f"Outliers found in column: {column}")
        print(outliers)
    print()
```

```
Outliers found in column: air_pressure_9am
204    907.99
281    929.32
342    908.42
474    908.97
708    909.19
Name: air_pressure_9am, dtype: float64

Outliers found in column: air_temp_9am
720    98.906
Name: air_temp_9am, dtype: float64

No outliers found in column: avg_wind_direction_9am

Outliers found in column: avg_wind_speed_9am
2       17.067852
61      16.486248
67      17.527932
97      23.554978
104     16.126429
          …
1012    18.698476
1017    14.987498
1037    18.653986
1044    17.188835
1078    15.162800
Name: avg_wind_speed_9am, Length: 67, dtype: float64

No outliers found in column: max_wind_direction_9am

Outliers found in column: max_wind_speed_9am
2       22.100967
61      18.029736
67      20.971183
71      19.214314
97      29.840780
          …
1012    23.377962
```

```
1017    19.617964
1037    23.268404
1044    22.195713
1078    19.803578
Name: max_wind_speed_9am, Length: 86, dtype: float64

Outliers found in column: rain_accumulation_9am
4       8.900
5       0.020
11      0.550
42      1.530
46      0.021
        …
1009    0.010
1011    0.010
1018    0.010
1027    5.272
1059    0.080
Name: rain_accumulation_9am, Length: 121, dtype: float64

Outliers found in column: rain_duration_9am
2          20.000000
4       14730.000000
5         170.000000
11       1770.000000
27        220.000000
            …
1037       11.024881
1039        6.029673
1044       19.348384
1059      520.000000
1076       10.000000
Name: rain_duration_9am, Length: 176, dtype: float64

Outliers found in column: relative_humidity_9am
4       92.41
11      90.56
26      92.10
36      89.77
38      91.22
        …
1082    89.80
1083    91.03
1085    92.30
1086    91.11
1091    90.35
Name: relative_humidity_9am, Length: 74, dtype: float64
```

```
No outliers found in column: relative_humidity_3pm
```

[967]:
```
df.head(2)
```

[967]:
```
   air_pressure_9am  air_temp_9am  avg_wind_direction_9am  avg_wind_speed_9am  \
0        918.060000     74.822000              271.100000            2.080354
1        917.347688     71.403843              101.935179            2.443009

   max_wind_direction_9am  max_wind_speed_9am  rain_accumulation_9am  \
0              295.400000            2.863283                    0.0
1              140.471548            3.533324                    0.0

   rain_duration_9am  relative_humidity_9am  relative_humidity_3pm
0                0.0              42.420000              36.160000
1                0.0              24.328697              19.426597
```

[971]:
```
df.columns
```

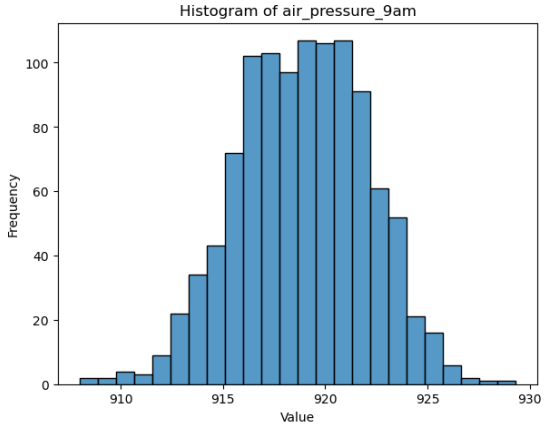
[971]:
```
Index(['air_pressure_9am', 'air_temp_9am', 'avg_wind_direction_9am',
       'avg_wind_speed_9am', 'max_wind_direction_9am', 'max_wind_speed_9am',
       'rain_accumulation_9am', 'rain_duration_9am', 'relative_humidity_9am',
       'relative_humidity_3pm'],
      dtype='object')
```

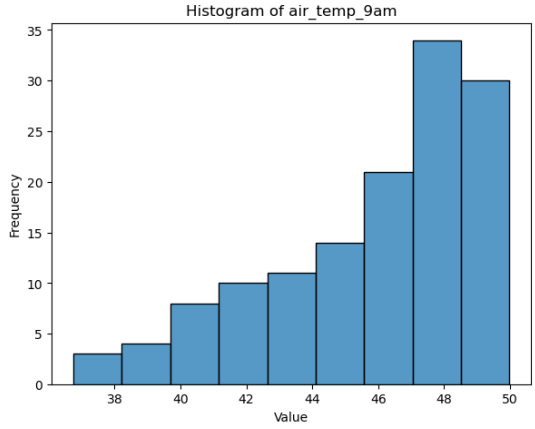
[979]:
```
df.describe()
```

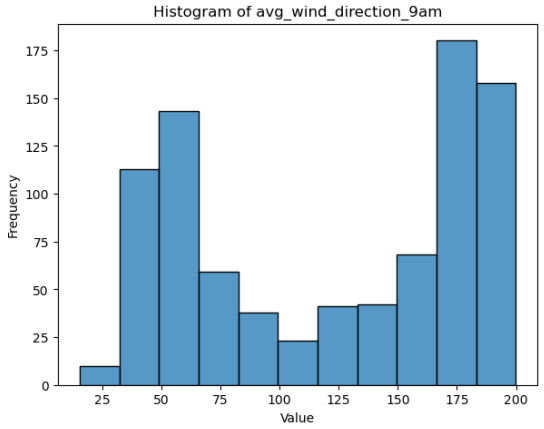
[979]:
```
       air_pressure_9am  air_temp_9am  avg_wind_direction_9am  \
count       1064.000000   1064.000000             1064.000000
mean         918.903180     65.022609              142.306756
std            3.179040     11.168033               69.149472
min          907.990000     36.752000               15.500000
25%          916.595376     57.398000               65.979244
50%          918.942281     65.778479              165.937461
75%          921.169054     73.530872              191.100000
max          929.320000     98.906000              343.400000

       avg_wind_speed_9am  max_wind_direction_9am  max_wind_speed_9am  \
count         1064.000000             1064.000000         1064.000000
mean             5.485793              148.480424            6.999714
std              4.534427               67.154911            5.590790
min              0.693451               28.900000            1.185578
25%              2.245529               76.335351            3.064608
50%              3.869906              176.350000            4.943637
75%              7.264463              201.125000            8.747888
max             23.554978              312.200000           29.840780
```

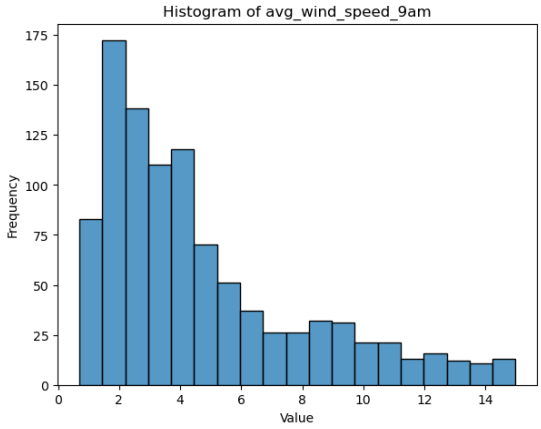
```
       rain_accumulation_9am  rain_duration_9am  relative_humidity_9am  \
count            1064.000000        1064.000000            1064.000000
mean                0.182023         266.393697              34.077440
std                 1.534493        1503.092216              25.356668
min                 0.000000           0.000000               6.090000
25%                 0.000000           0.000000              15.093365
50%                 0.000000           0.000000              23.135000
75%                 0.000000           0.000000              44.660000
max                24.020000       17704.000000              92.620000

       relative_humidity_3pm
count            1064.000000
mean               35.148381
std                22.365475
min                 5.300000
25%                17.360468
50%                24.371286
75%                51.922500
max                92.250000
```

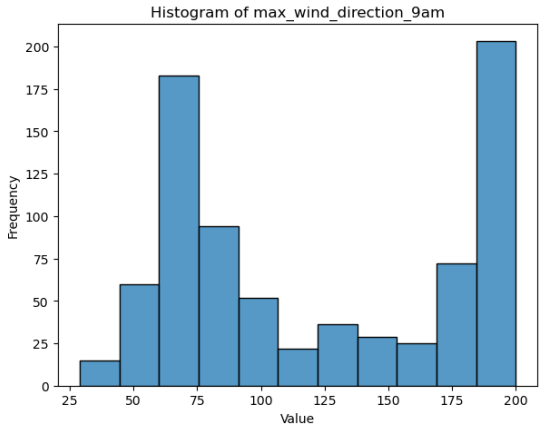
[1000]:
```
import seaborn as sns
import matplotlib.pyplot as plt

air_pressure_9am = df[df['air_pressure_9am'] > 907]['air_pressure_9am']
air_temp_9am = df[df['air_temp_9am'] < 50]['air_temp_9am']
avg_wind_direction_9am = df[df['avg_wind_direction_9am'] <␣
 ↪200]['avg_wind_direction_9am']
avg_wind_speed_9am = df[df['avg_wind_speed_9am'] < 15]['avg_wind_speed_9am']
max_wind_direction_9am = df[df['max_wind_direction_9am'] <␣
 ↪200]['max_wind_direction_9am']

show = [air_pressure_9am, air_temp_9am, avg_wind_direction_9am,␣
 ↪avg_wind_speed_9am, max_wind_direction_9am]
variables = ['air_pressure_9am', 'air_temp_9am', 'avg_wind_direction_9am',␣
 ↪'avg_wind_speed_9am', 'max_wind_direction_9am']

plt.figure(figsize=(15, 30))
for i, col in enumerate(show):
    plt.subplot(5, 2, i + 1)
    sns.histplot(x=col)
    plt.xlabel('Value')
    plt.ylabel('Frequency')
    plt.title(f'Histogram of {variables[i]}')
```


Histogram of air_pressure_9am
Frequency
Value
910
915
920
925
930


Histogram of air_temp_9am
Frequency
Value
38
40
42
44
46
48
50


Histogram of avg_wind_direction_9am
Frequency
Value
25
50
75
100
125
150
175
200


Histogram of avg_wind_speed_9am
Frequency
Value
0
2
4
6
8
10
12
14


Histogram of max_wind_direction_9am
Frequency
Value
25
50
75
100
125
150
175
200

[ ]:

[ ]: