INFOSYS SPRINGBOARD VIRTUAL INTERNSHIP

FWI PREDICTOR

By

Srinanda C S

MILESTONE 1

Dataset Source : Kaggle

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder
```

These libraries are used for data analysis, preprocessing, visualization, and encoding.

```
df = pd.read_csv("FWI Dataset.csv")
if 'Region' in df.columns:
    df['Region'] = df['Region'].astype('category').cat.codes
```

Dataset is loaded using pandas.

The following information is displayed:

- Entire dataset
- Dataset structure (df.info())
- Statistical summary (df.describe())
- First & last 5 rows

Converts the *Region* column from strings to numerical category codes. Useful for machine learning algorithms that accept numeric inputs.

```
numeric_df = df.select_dtypes(include=['int64', 'float64'])
```

Extracts only numerical features for later analysis.

```
print(df.shape)

print(df.columns)
```

To ensure the dataset structure is correct.

```
df.isnull().sum()

df[df.isnull().any(axis=1)]

df.columns = df.columns.str.strip()
```

Identifies missing values in each column.

Displays rows containing incomplete data.

Removes extra spaces in column names (common in raw CSV files).

```
for col in df.columns:
    if df[col].dtype == 'object':
        df[col] = df[col].astype(str).str.strip()
```

Removes unnecessary spaces in string values.

Ensures uniform data format.

```
for col in df.columns:
    if df[col].dtype == 'object':
        df[col] = df[col].str.replace("  ", " ")
        if df[col].str.contains(" ").any():
            df[col] = df[col].str.split(" ").str[0]
```

This ensures numeric columns convert cleanly.

```
numeric_cols =
['Temperature','RH','Ws','Rain','FFMC','DMC','DC','ISI','BUI',
'FWI']

for col in numeric_cols:

    df[col] = pd.to_numeric(df[col], errors='coerce')
```

Converts corrupted strings into numeric format.

Non-convertible values become Nan.

```
df['Region'] = df['Region'].fillna(df['Region'].mode()[0])

df['Classes'] = df['Classes'].fillna(df['Classes'].mode()[0])
```

Uses mode (most frequent value) for categorical features.

Prevents ML models from failing due to null values.

Label Encoding Categorical Columns

```
le_region = LabelEncoder()

df['Region_encoded'] = le_region.fit_transform(df['Region'])


le_class = LabelEncoder()

df['Classes_encoded'] = le_class.fit_transform(df['Classes'])
```

Convert string labels to numeric classes for ML model training.

Encoding All Remaining Categorical Columns

```
df_encoded = df.copy()

label_encoders = {}
```

```
for col in df_encoded.columns:

    if df_encoded[col].dtype == 'object':

        le = LabelEncoder()

        df_encoded[col] =
le.fit_transform(df_encoded[col].astype(str))

        label_encoders[col] = le
```

This ensures all non-numeric features are usable in correlation analysis.

Correlation Heatmap

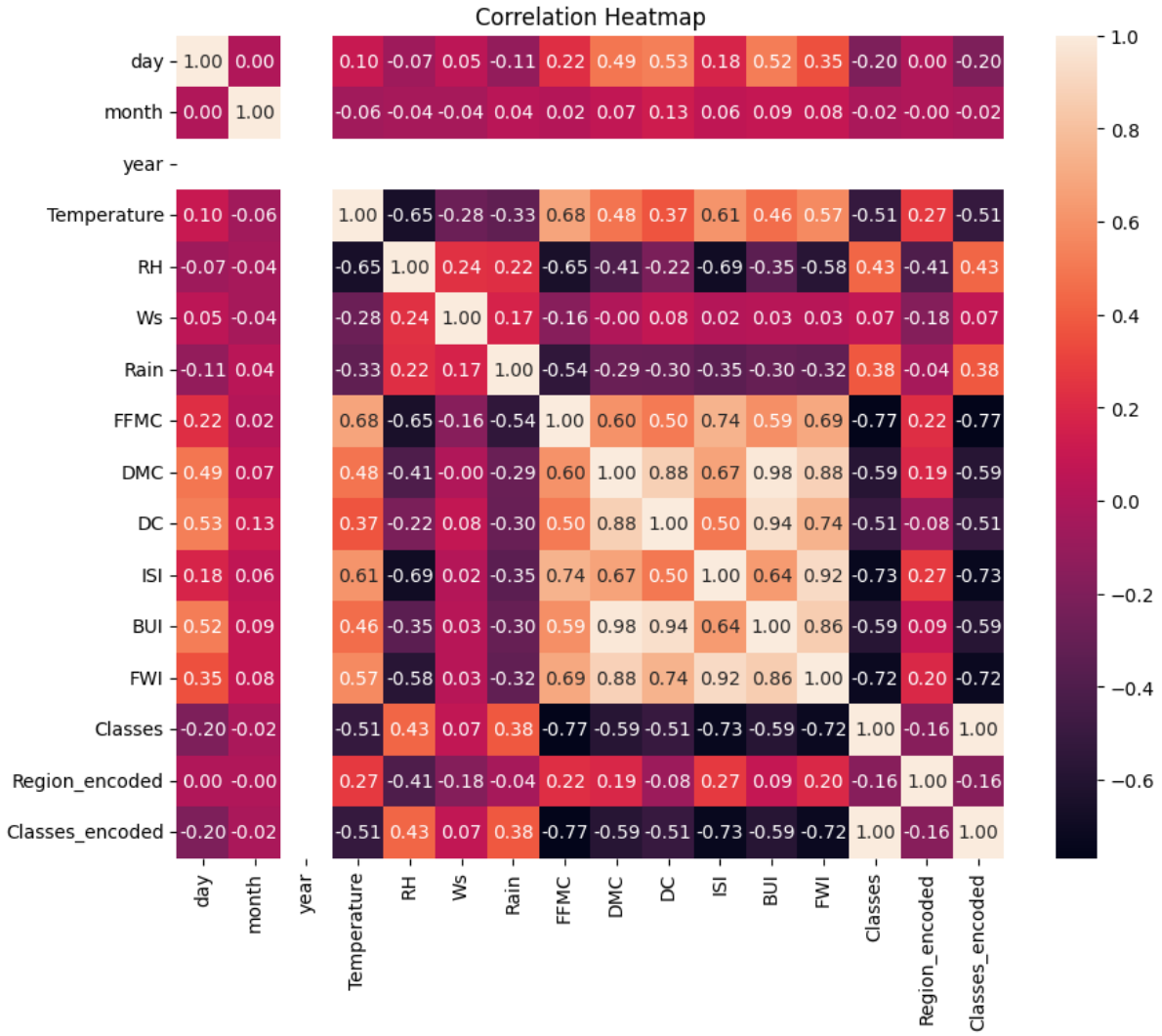
```
plt.figure(figsize=(10,8))

sns.heatmap(numeric_df.corr(), annot=True)

plt.show()
```

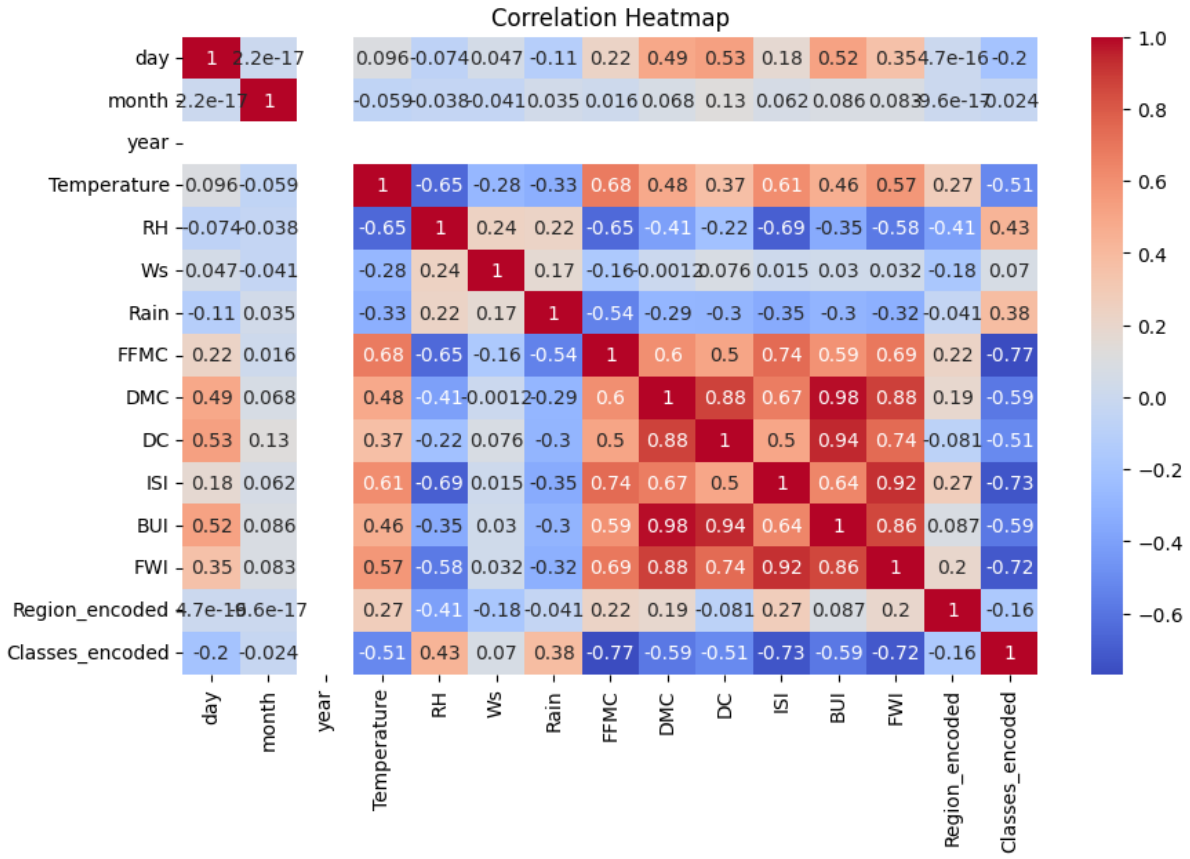
Used to understand:

Feature relationships

Which variables strongly influence FWI

Multicollinearity issues


Correlation Heatmap
day 1.00 0.00 0.10 -0.07 0.05 -0.11 0.22 0.49 0.53 0.18 0.52 0.35 -0.20 0.00 -0.20
month 0.00 1.00 -0.06 -0.04 -0.04 0.04 0.02 0.07 0.13 0.06 0.09 0.08 -0.02 -0.00 -0.02
year
Temperature 0.10 -0.06 1.00 -0.65 -0.28 -0.33 0.68 0.48 0.37 0.61 0.46 0.57 -0.51 0.27 -0.51
RH -0.07 -0.04 -0.65 1.00 0.24 0.22 -0.65 -0.41 -0.22 -0.69 -0.35 -0.58 0.43 -0.41 0.43
Ws 0.05 -0.04 -0.28 0.24 1.00 0.17 -0.16 -0.00 0.08 0.02 0.03 0.03 0.07 -0.18 0.07
Rain -0.11 0.04 -0.33 0.22 0.17 1.00 -0.54 -0.29 -0.30 -0.35 -0.30 -0.32 0.38 -0.04 0.38
FFMC 0.22 0.02 0.68 -0.65 -0.16 -0.54 1.00 0.60 0.50 0.74 0.59 0.69 -0.77 0.22 -0.77
DMC 0.49 0.07 0.48 -0.41 -0.00 -0.29 0.60 1.00 0.88 0.67 0.98 0.88 -0.59 0.19 -0.59
DC 0.53 0.13 0.37 -0.22 0.08 -0.30 0.50 0.88 1.00 0.50 0.94 0.74 -0.51 -0.08 -0.51
ISI 0.18 0.06 0.61 -0.69 0.02 -0.35 0.74 0.67 0.50 1.00 0.64 0.92 -0.73 0.27 -0.73
BUI 0.52 0.09 0.46 -0.35 0.03 -0.30 0.59 0.98 0.94 0.64 1.00 0.86 -0.59 0.09 -0.59
FWI 0.35 0.08 0.57 -0.58 0.03 -0.32 0.69 0.88 0.74 0.92 0.86 1.00 -0.72 0.20 -0.72
Classes -0.20 -0.02 -0.51 0.43 0.07 0.38 -0.77 -0.59 -0.51 -0.73 -0.59 -0.72 1.00 -0.16 1.00
Region_encoded 0.00 -0.00 0.27 -0.41 -0.18 -0.04 0.22 0.19 -0.08 0.27 0.09 0.20 -0.16 1.00 -0.16
Classes_encoded -0.20 -0.02 -0.51 0.43 0.07 0.38 -0.77 -0.59 -0.51 -0.73 -0.59 -0.72 1.00 -0.16 1.00
day
month
year
Temperature
RH
Ws
Rain
FFMC
DMC
DC
ISI
BUI
FWI
Classes
Region_encoded
Classes_encoded
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6

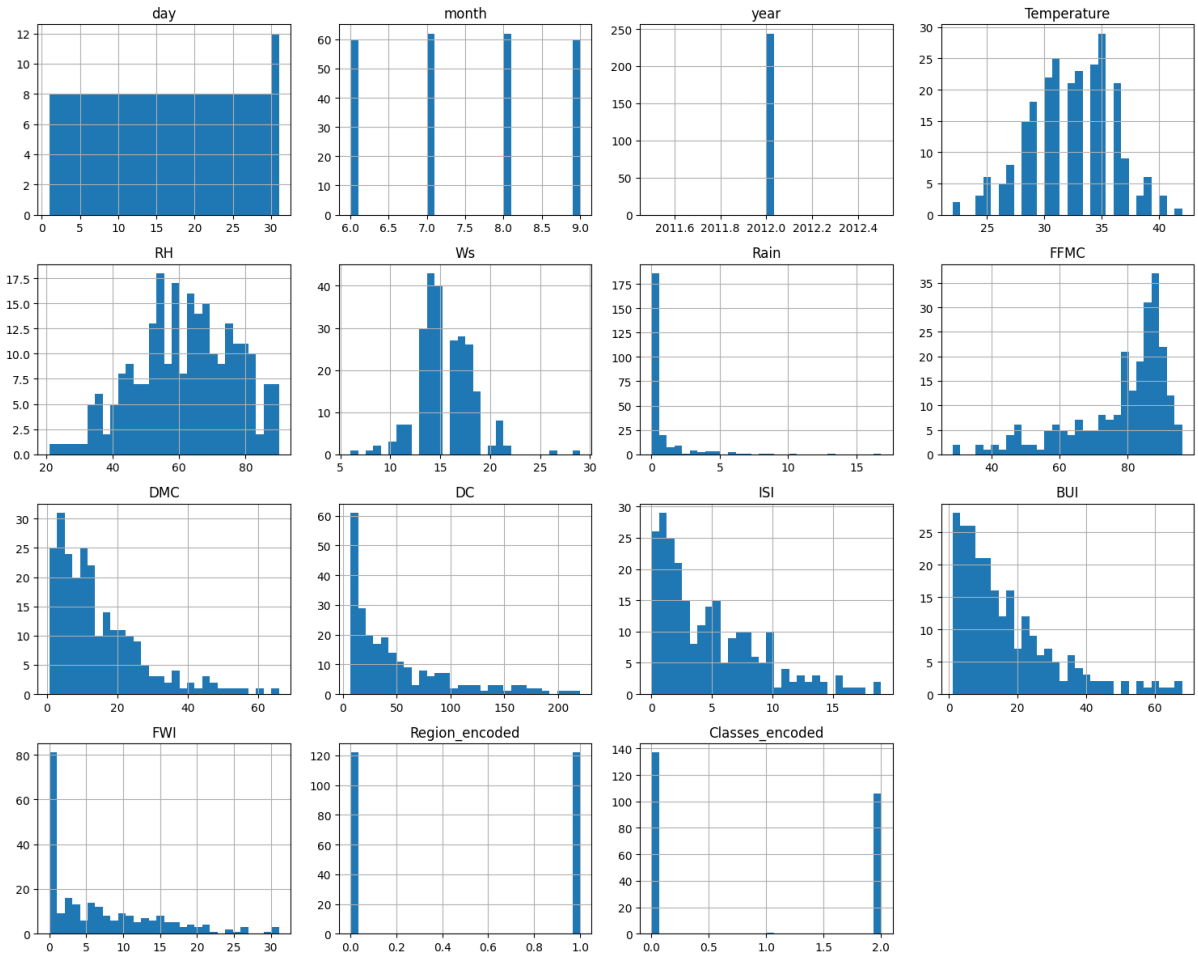


Histogram Plots

```
numeric_df.hist(figsize=(15, 12), bins=30)

plt.show()
```

Shows distribution of each numeric feature:

Normal, Skewed, Outliers



Density (KDE) Plots

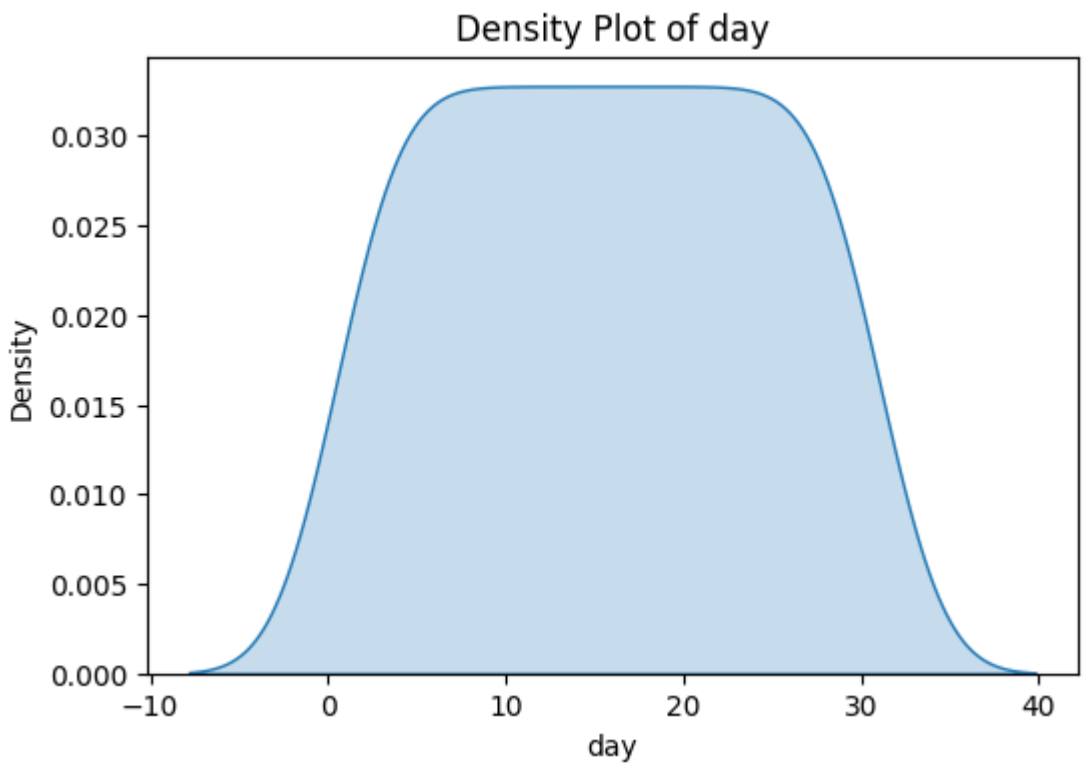
```
sns.kdeplot(numeric_df[col], fill=True)
```

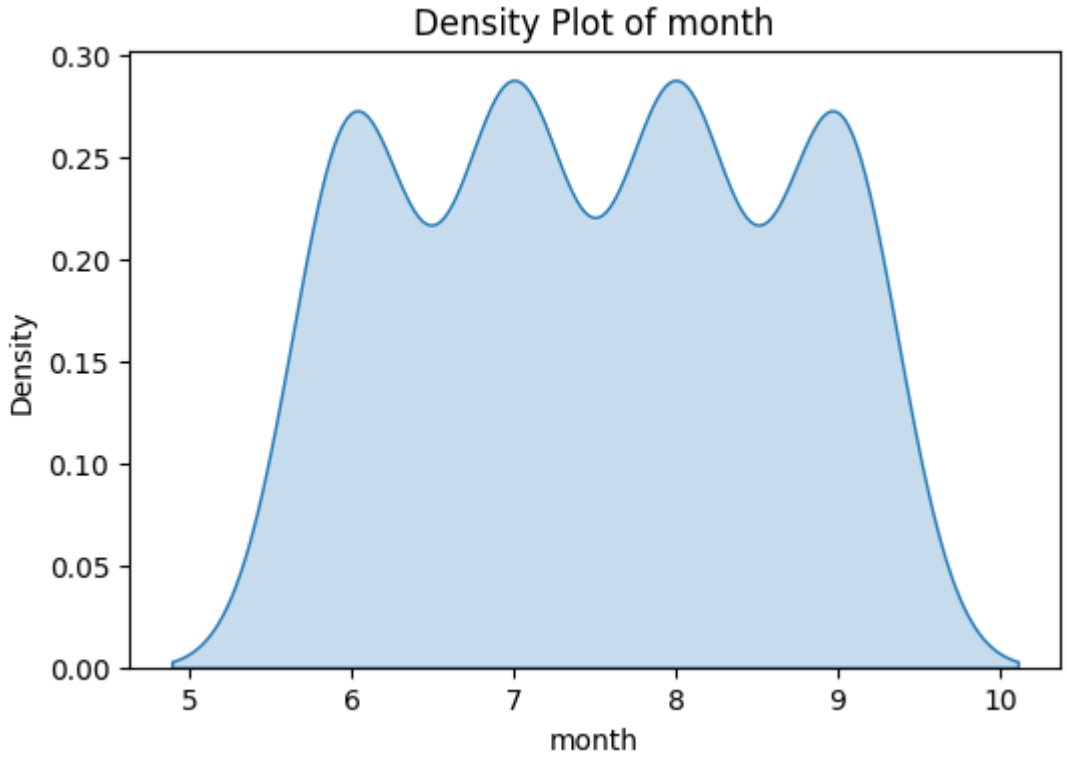
These help understand:

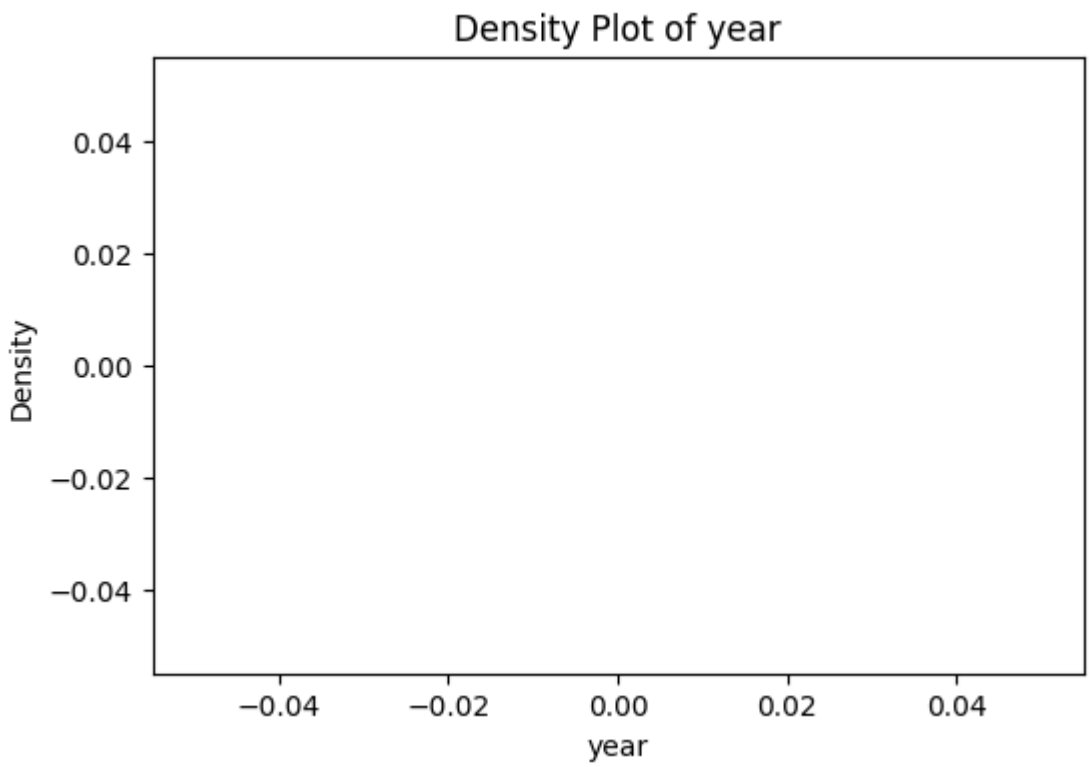
Probability distribution

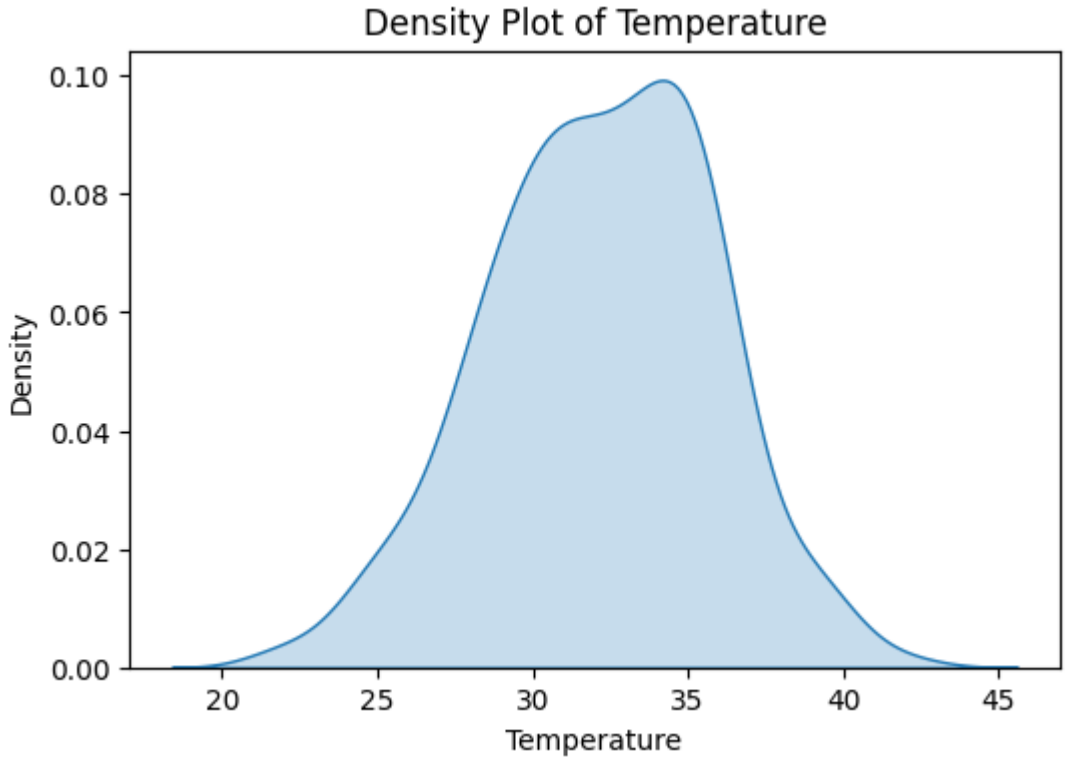
Spread of data

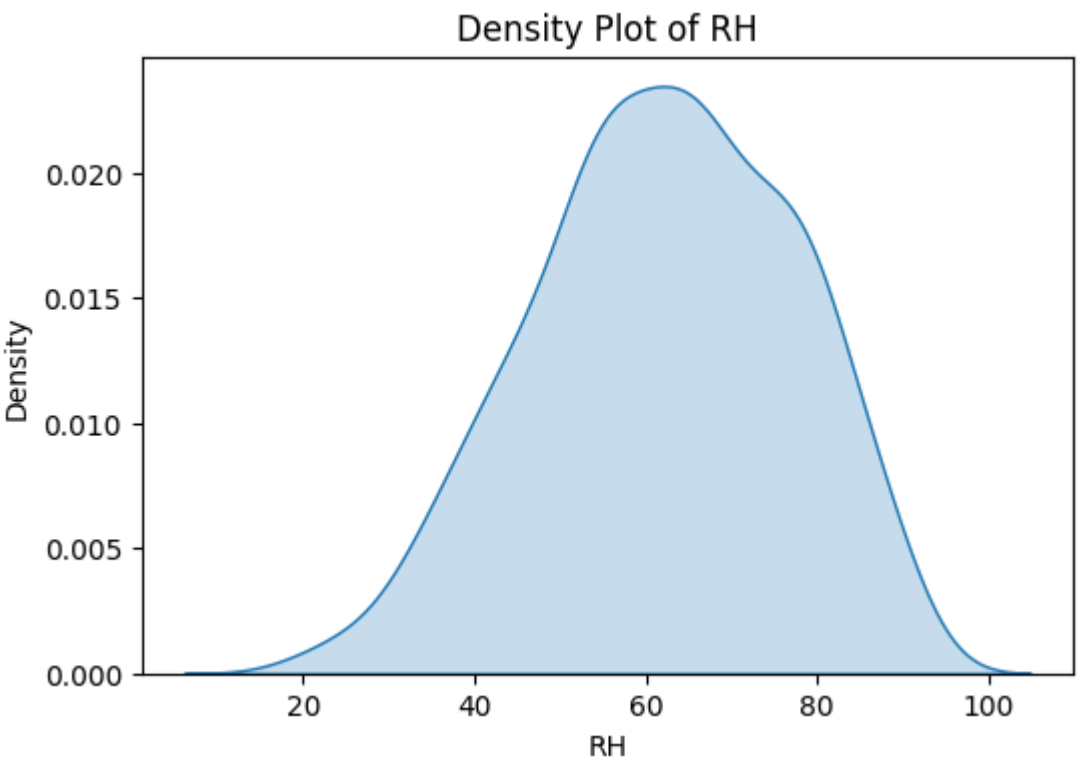
Detecting skewness

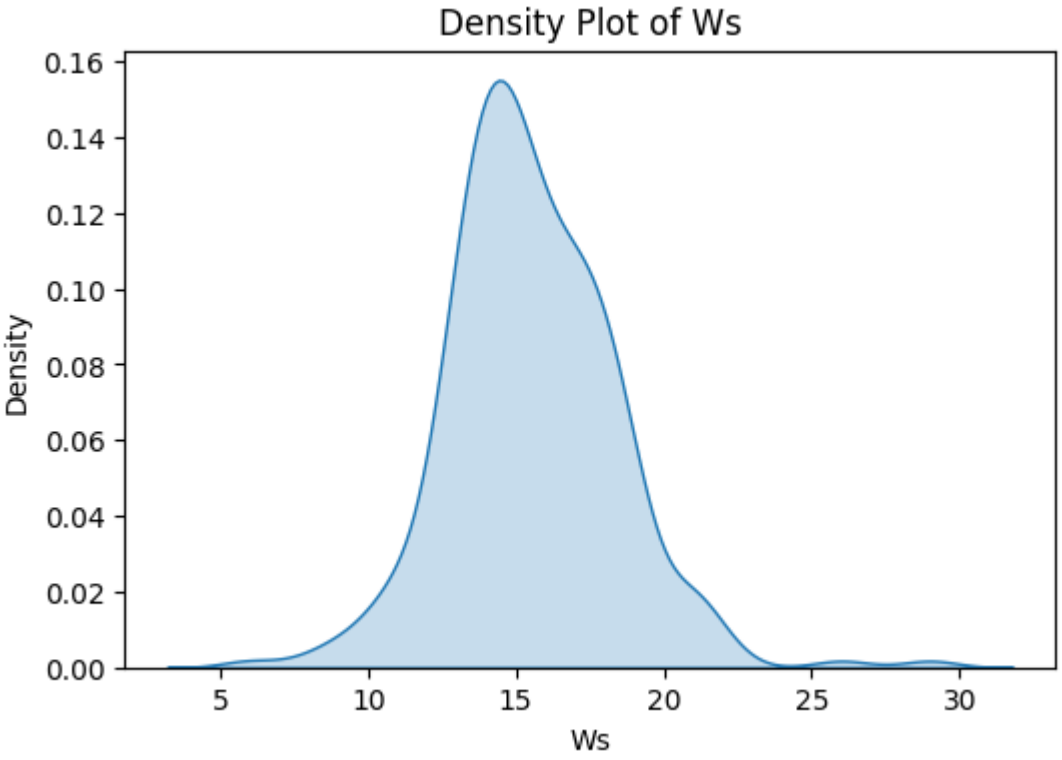

Density Plot of day
Density
0.030
0.025
0.020
0.015
0.010
0.005
0.000
−10
0
10
20
30
40
day

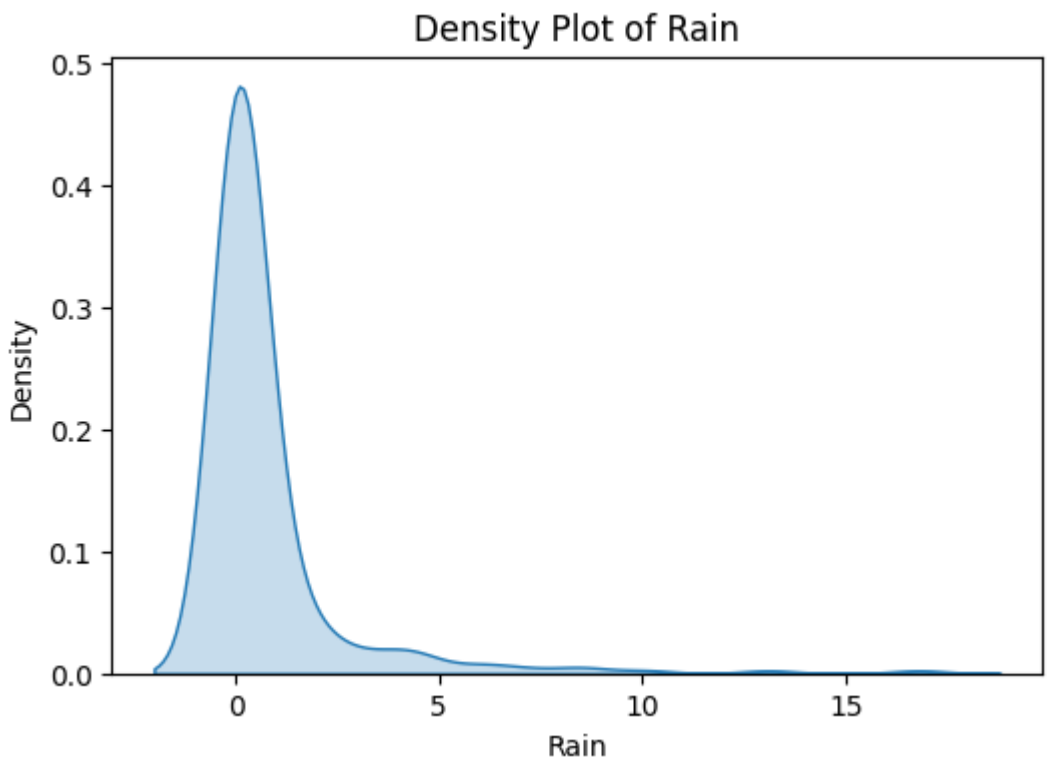

Density Plot of month
Density
0.30
0.25
0.20
0.15
0.10
0.05
0.00
5
6
7
8
9
10
month

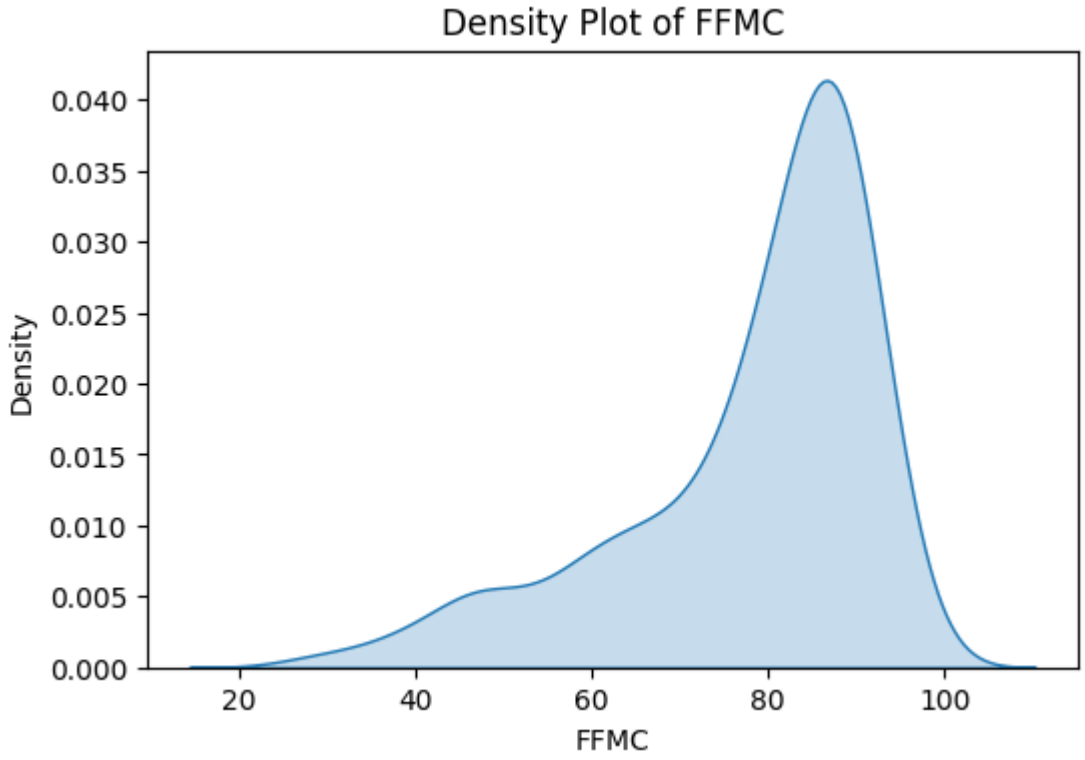

Density Plot of year
0.04
0.02
0.00
−0.02
−0.04
Density
−0.04
−0.02
0.00
0.02
0.04
year

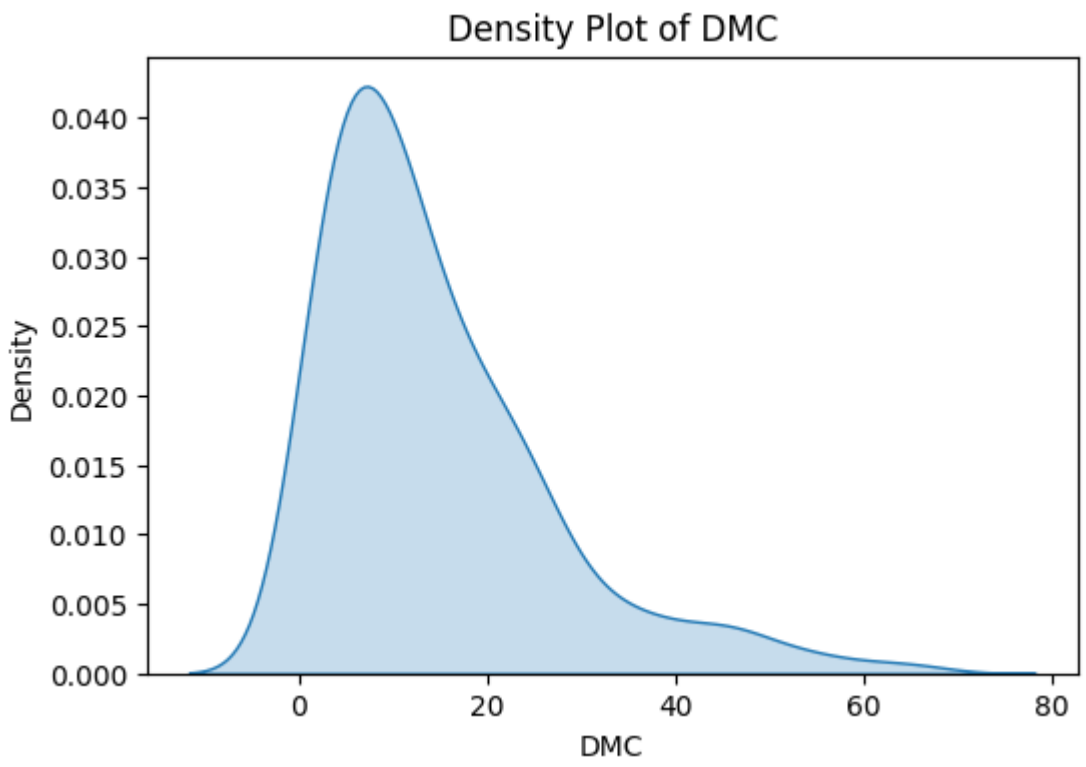

Density Plot of Temperature
0.10
0.08
0.06
0.04
0.02
0.00
Density
20
25
30
35
40
45
Temperature

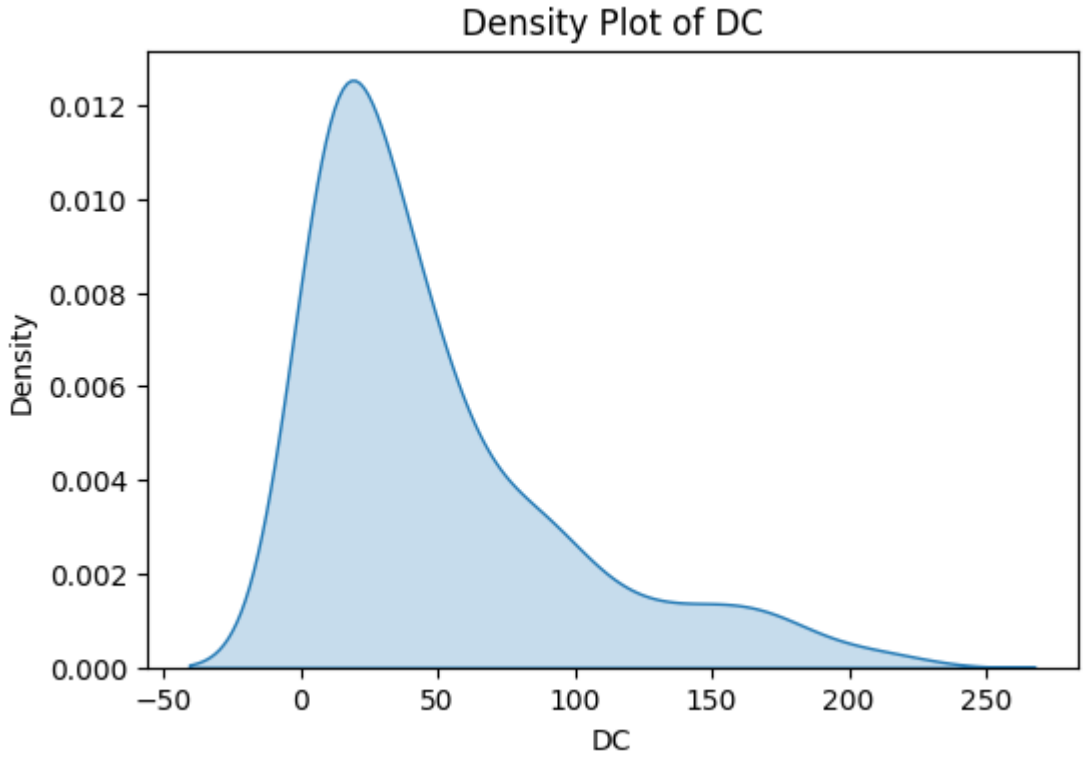

Density Plot of RH
Density
0.020
0.015
0.010
0.005
0.000
20
40
60
80
100
RH

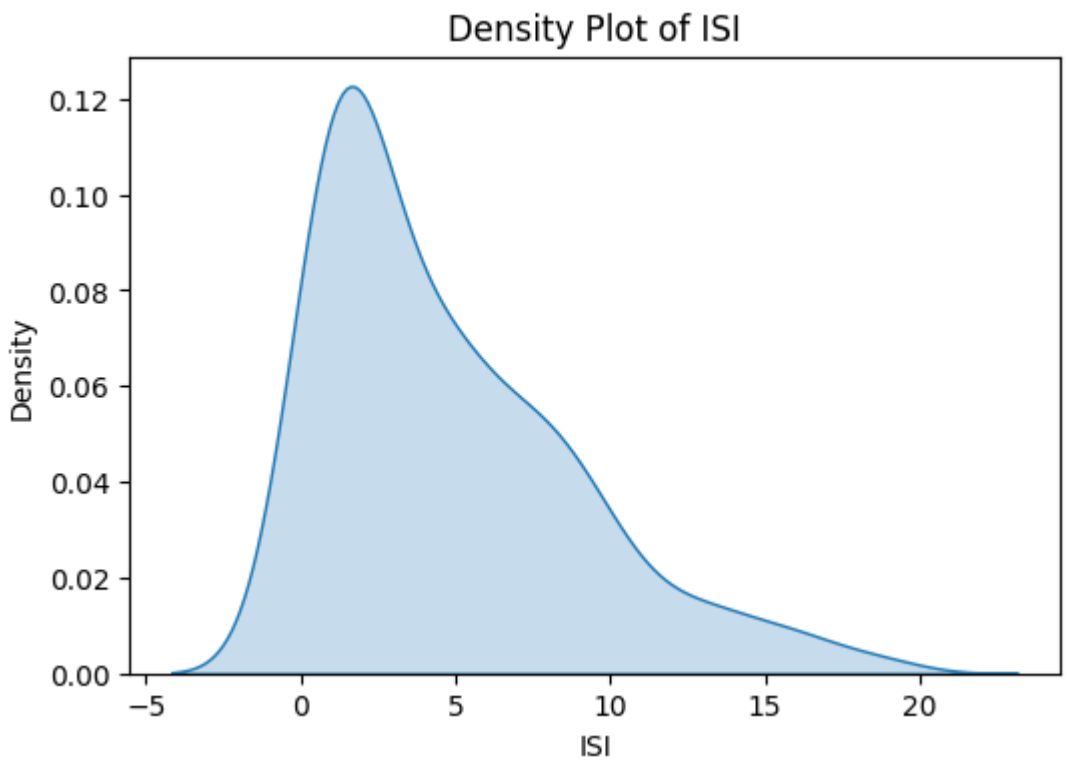

Density Plot of Ws
Density
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
5
10
15
20
25
30
Ws

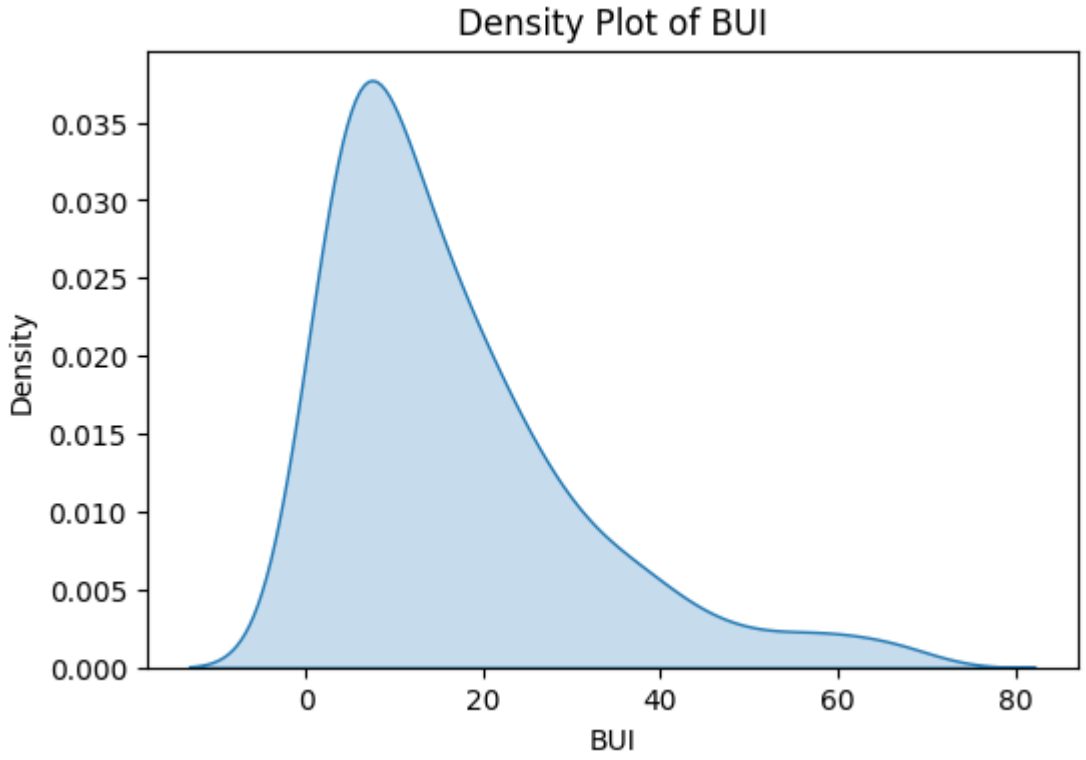

Density Plot of Rain
Density
Rain
0.0
0.1
0.2
0.3
0.4
0.5
0
5
10
15

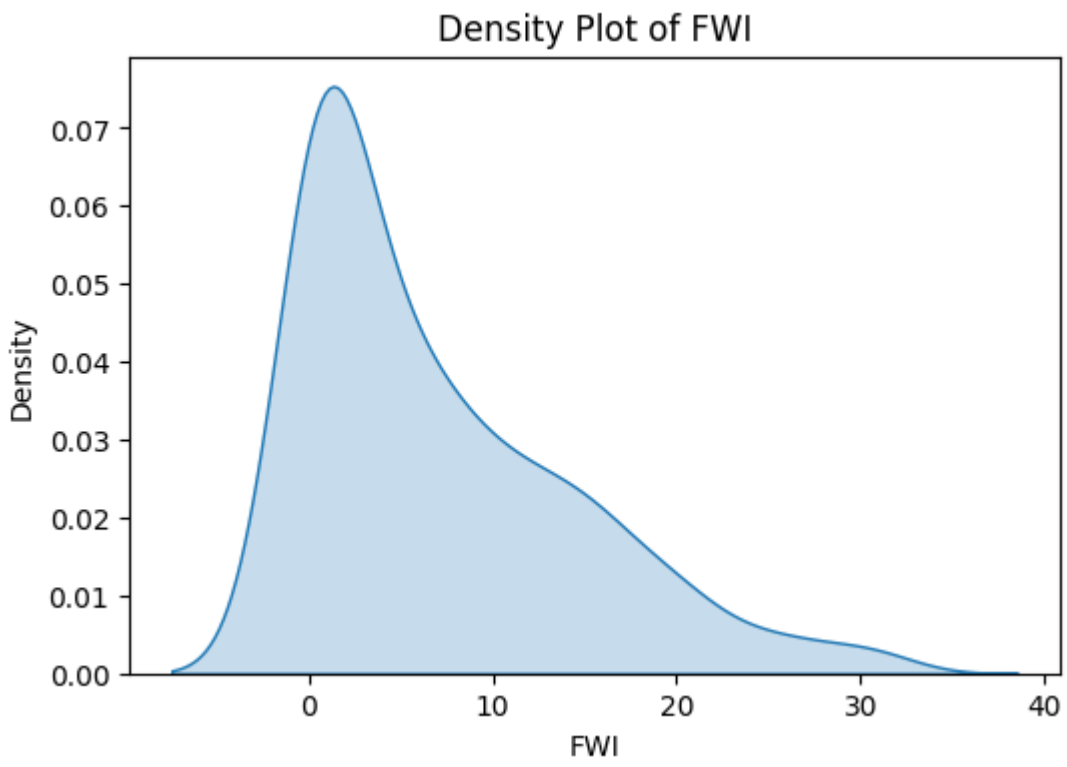

Density Plot of FFMC
Density
FFMC
0.000
0.005
0.010
0.015
0.020
0.025
0.030
0.035
0.040
20
40
60
80
100

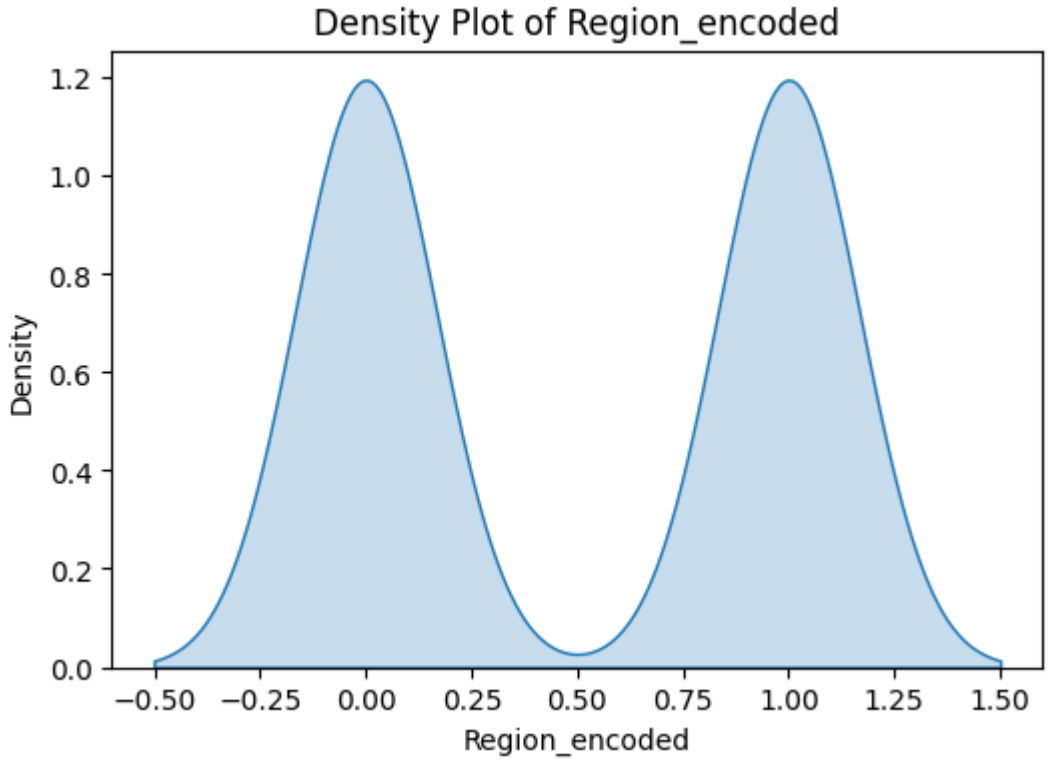

Density Plot of DMC
Density
0.040
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
0
20
40
60
80
DMC

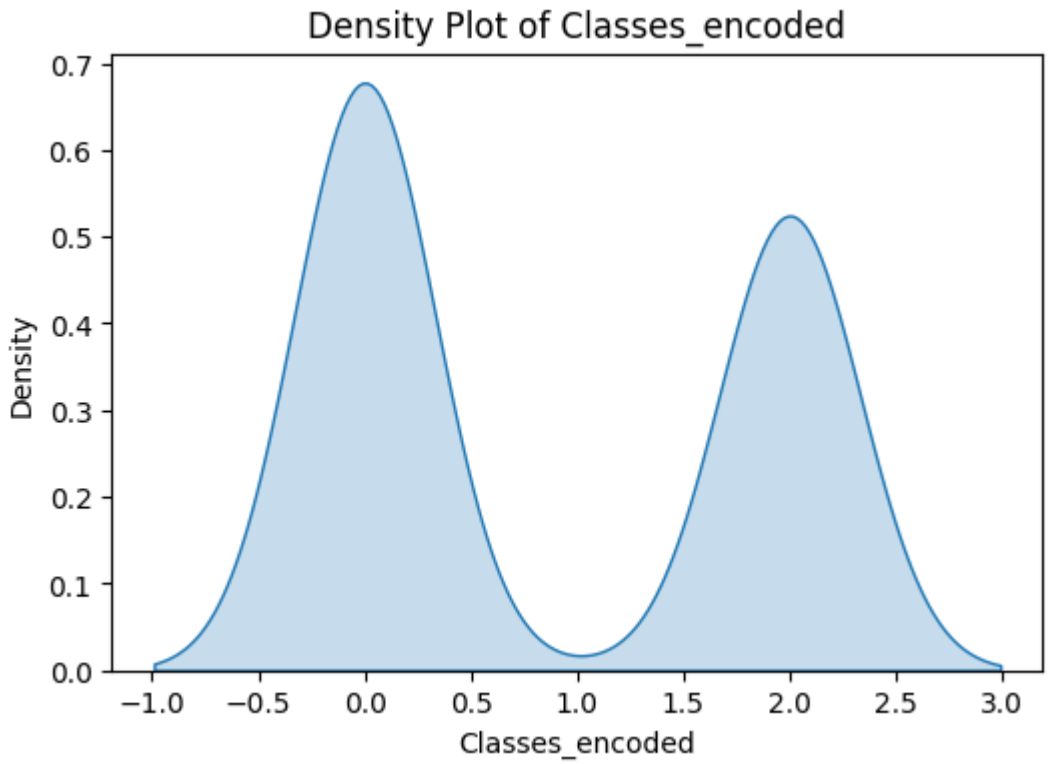

Density Plot of DC
Density
0.012
0.010
0.008
0.006
0.004
0.002
0.000
−50
0
50
100
150
200
250
DC


Density Plot of ISI
Density
0.12
0.10
0.08
0.06
0.04
0.02
0.00
−5
0
5
10
15
20
ISI


Density Plot of BUI
Density
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
0
20
40
60
80
BUI


Density Plot of FWI
Density
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0
10
20
30
40
FWI


Density Plot of Region_encoded
Density
0.0
0.2
0.4
0.6
0.8
1.0
1.2
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
1.25
1.50
Region_encoded



Boxplots for Outlier Detection

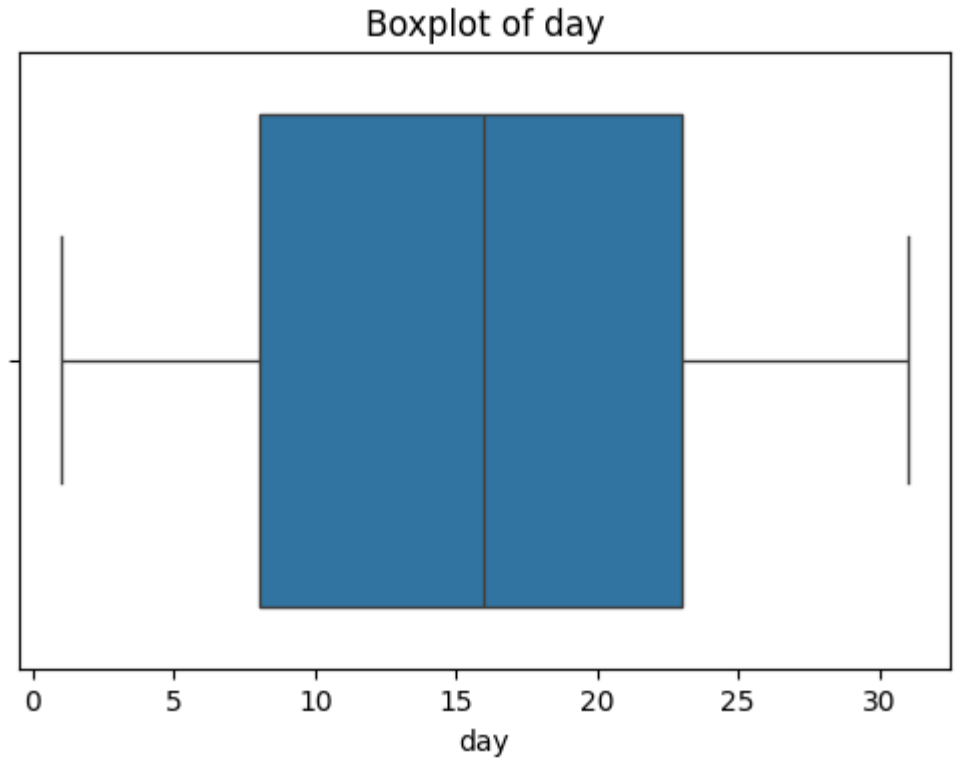
```
sns.boxplot(x=df[col])
```

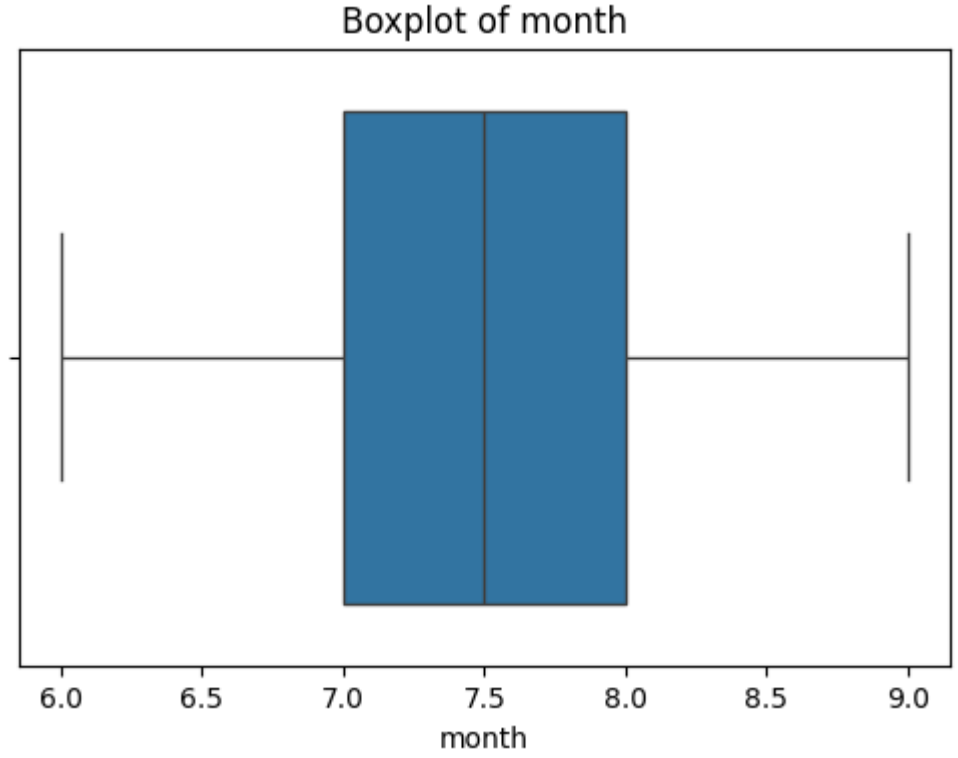
Used to visually identify:

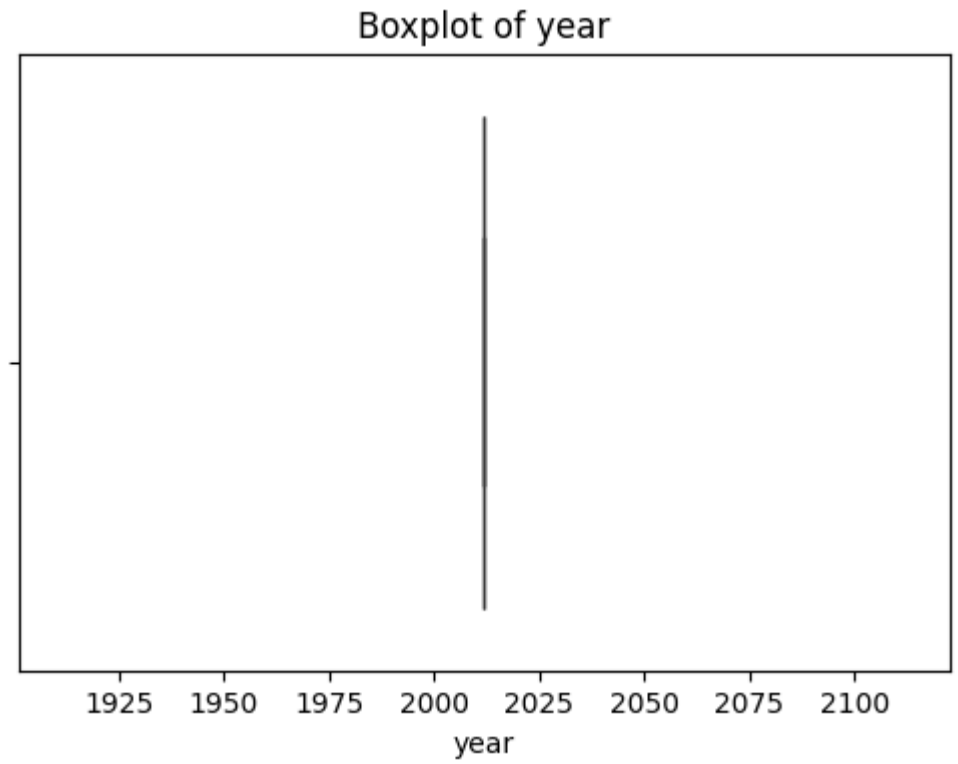
Outliers

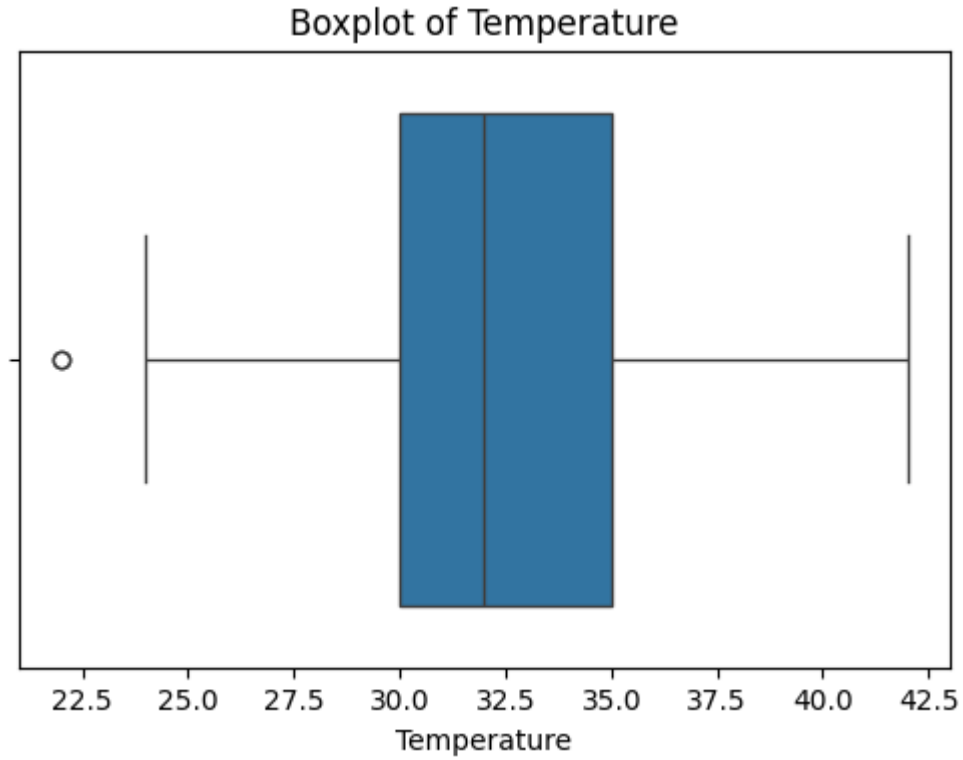
Data spread

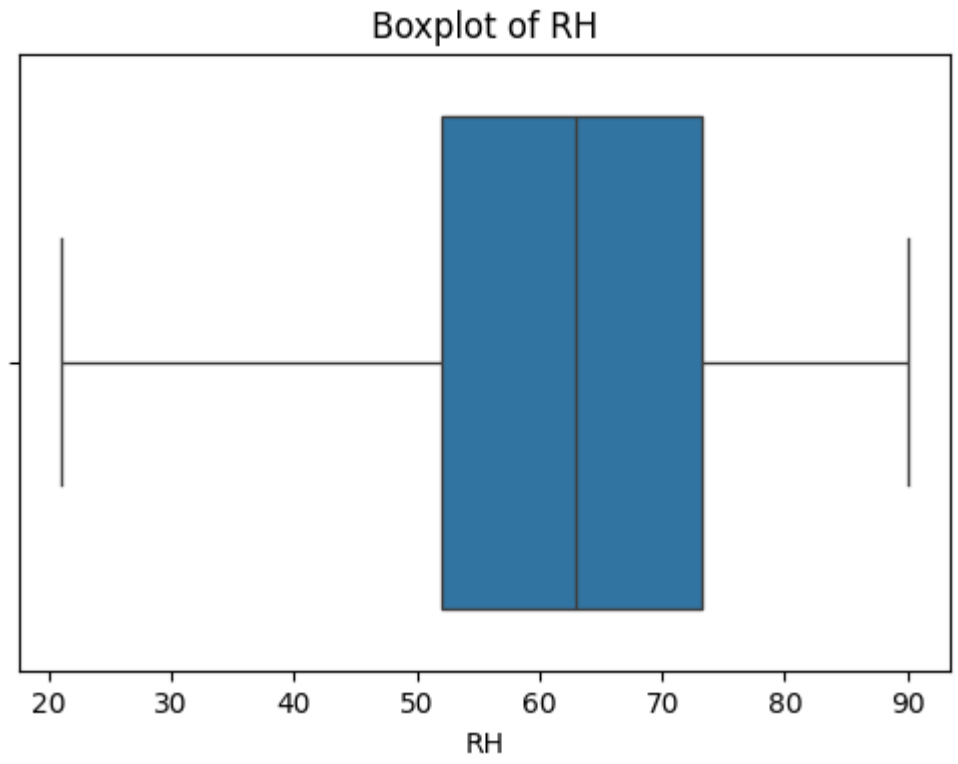
Extreme values

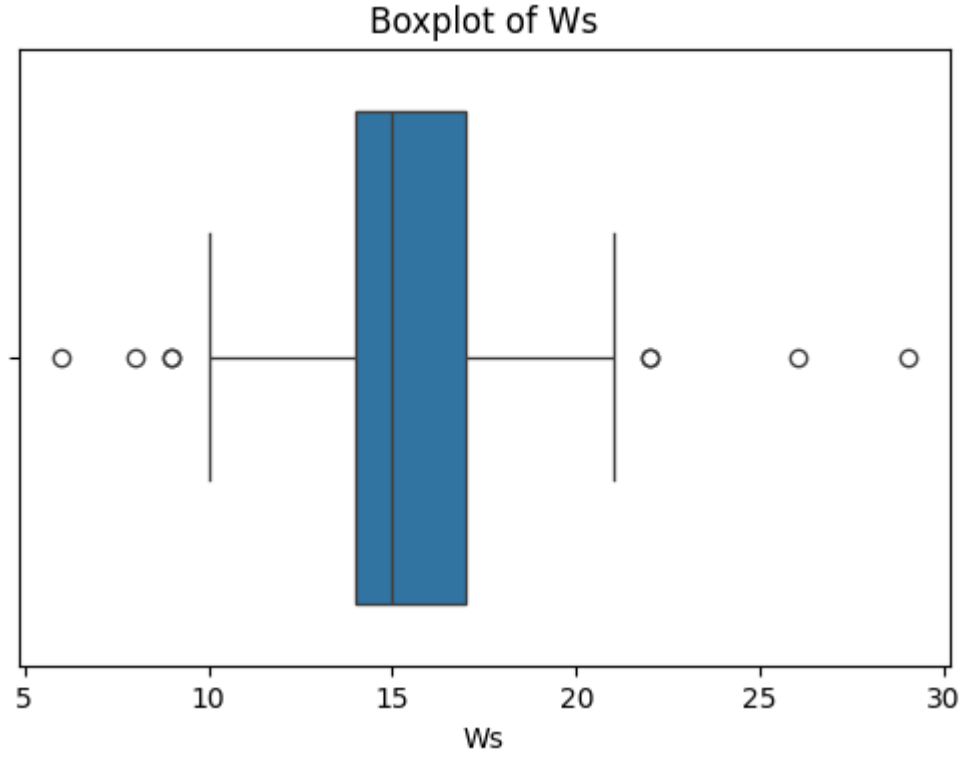

Boxplot of day
0
5
10
15
20
25
30
day

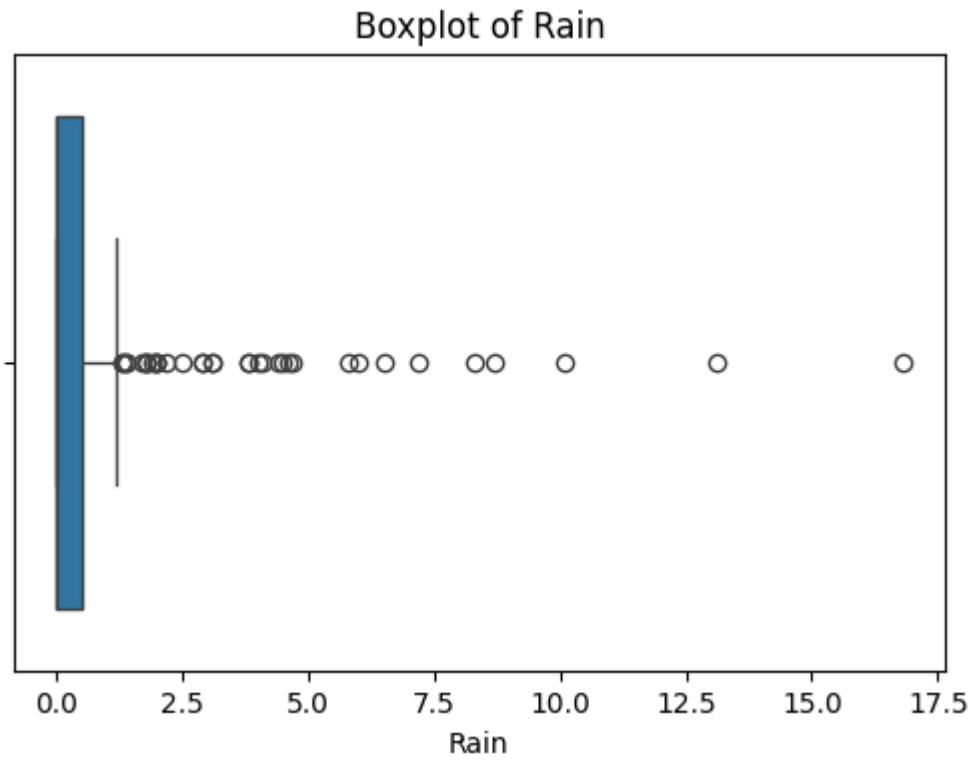

Boxplot of month
6.0
6.5
7.0
7.5
8.0
8.5
9.0
month

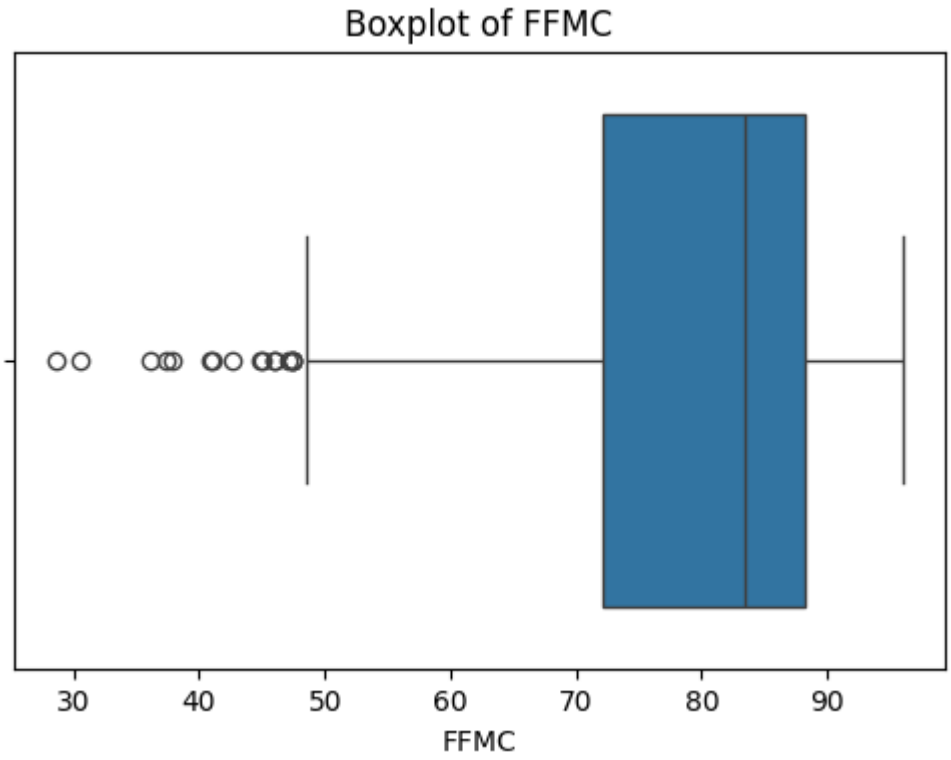

Boxplot of year
1925
1950
1975
2000
2025
2050
2075
2100
year

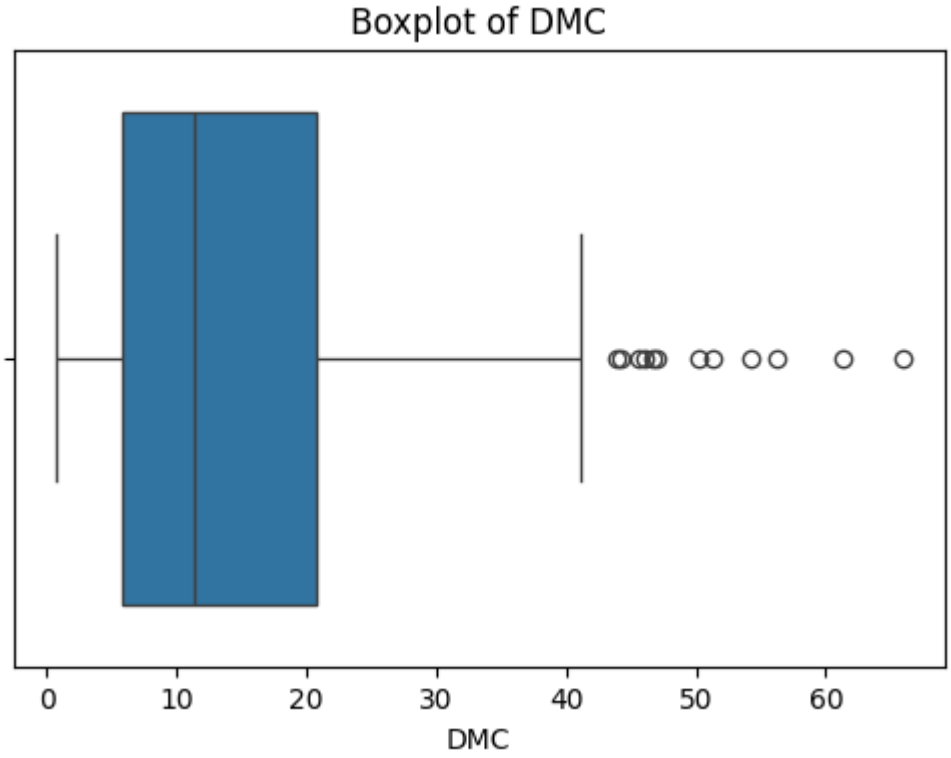

Boxplot of Temperature
22.5
25.0
27.5
30.0
32.5
35.0
37.5
40.0
42.5
Temperature

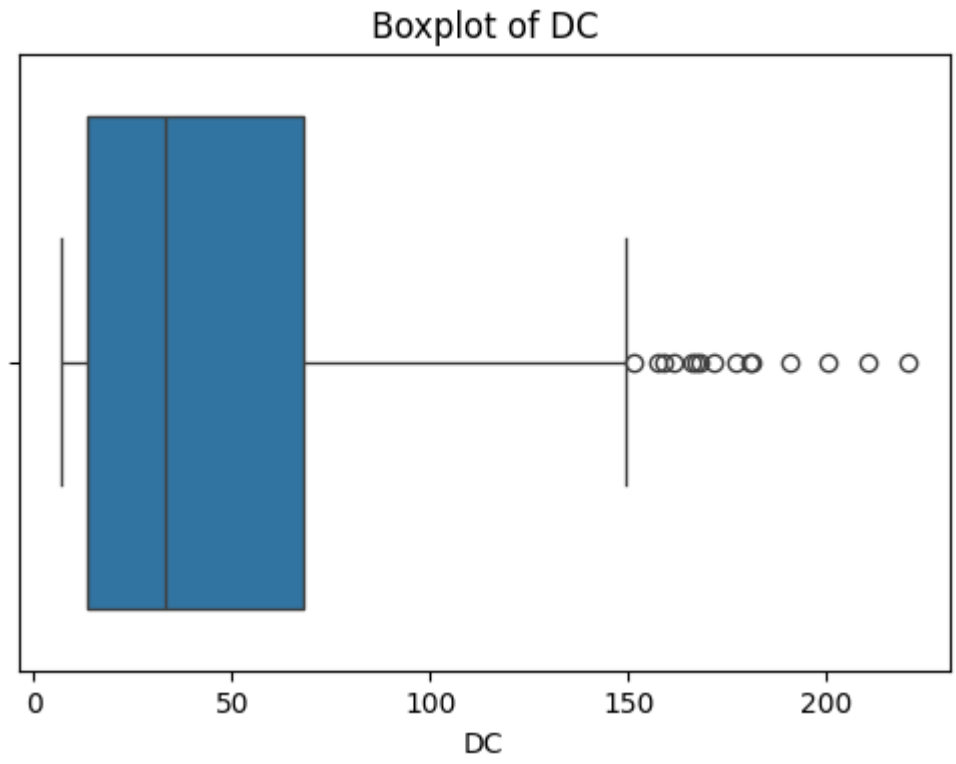

Boxplot of RH
20
30
40
50
60
70
80
90
RH

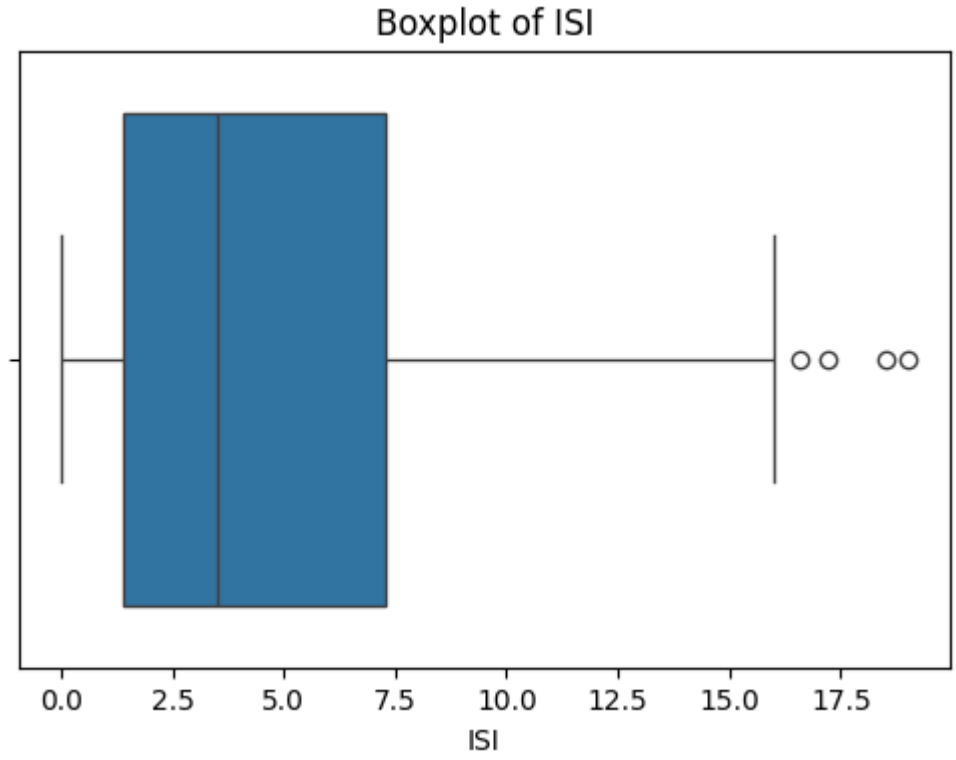

Boxplot of Ws
5
10
15
20
25
30
Ws

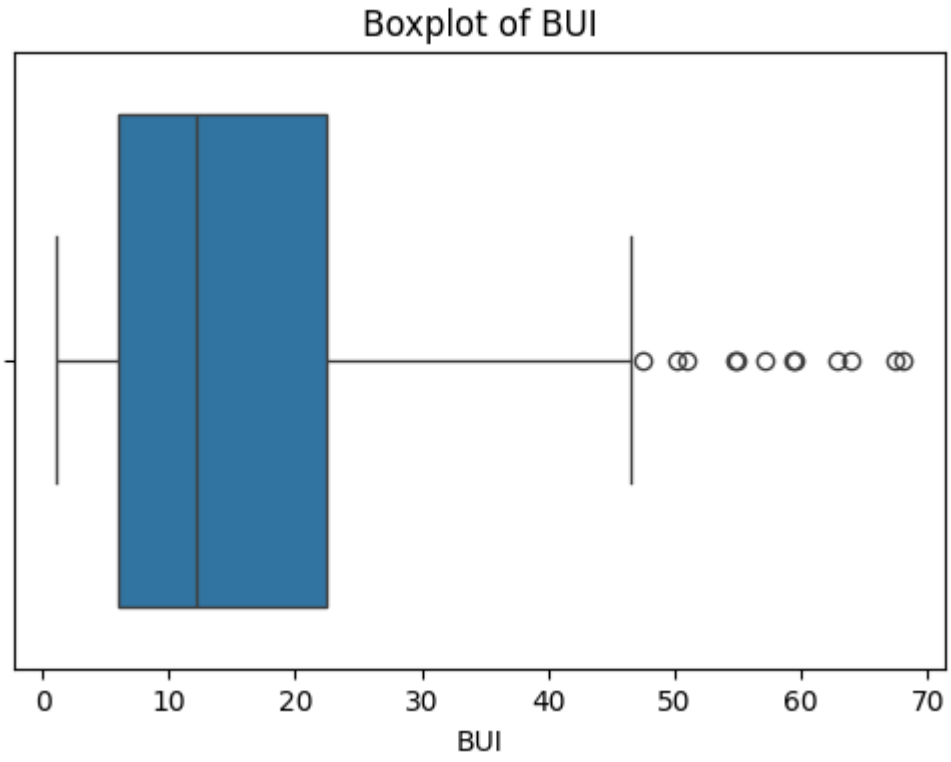

Boxplot of Rain
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
Rain

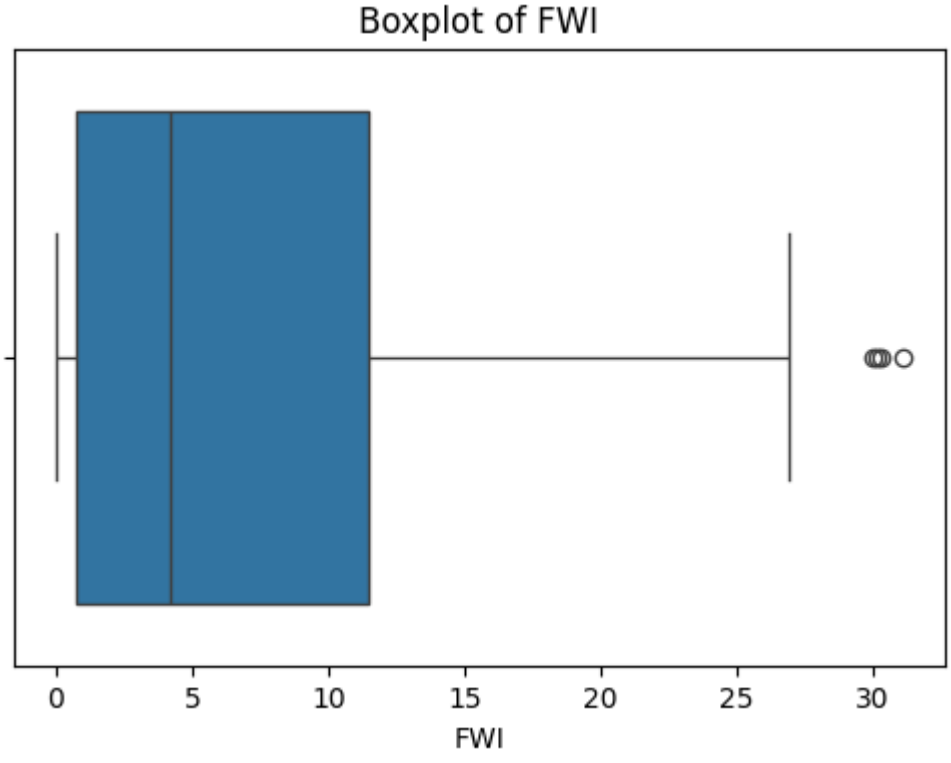

Boxplot of FFMC
30
40
50
60
70
80
90
FFMC

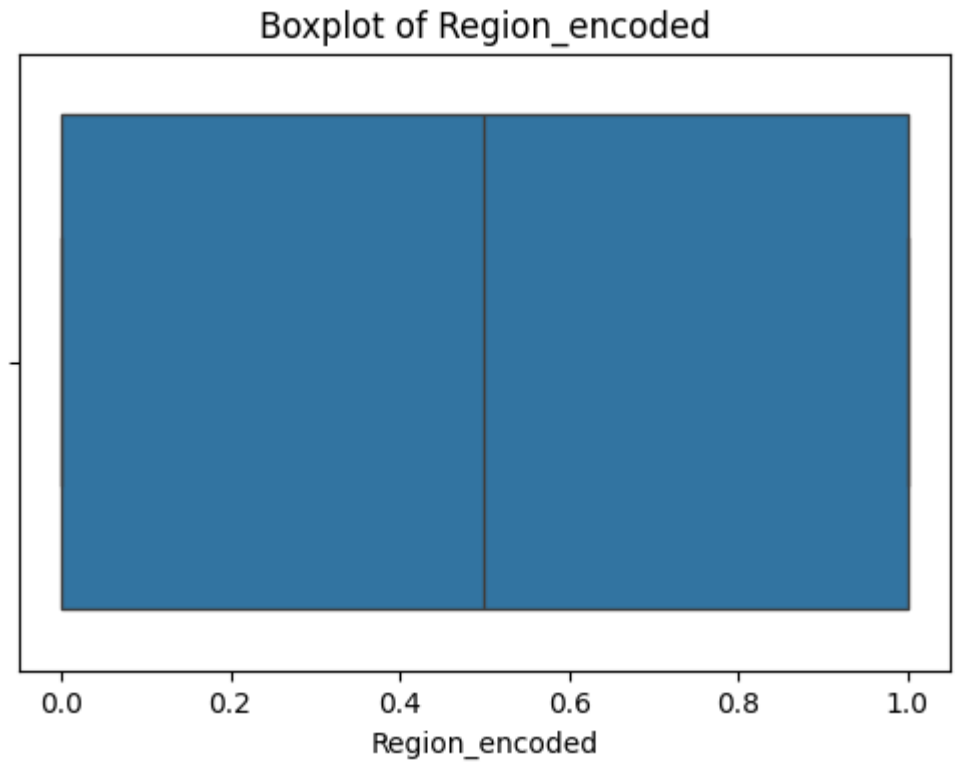

Boxplot of DMC
0
10
20
30
40
50
60
DMC

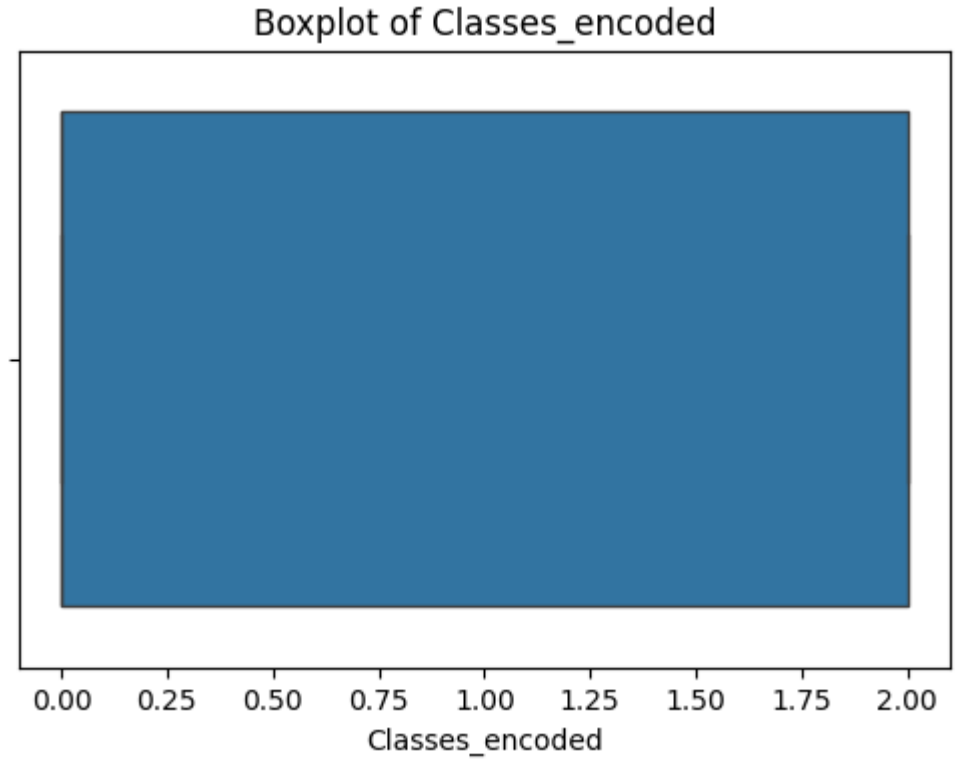

Boxplot of DC
0
50
100
150
200
DC


Boxplot of ISI
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
ISI


Boxplot of BUI
0
10
20
30
40
50
60
70
BUI


Boxplot of FWI
0
5
10
15
20
25
30
FWI





Outlier Treatment using IQR

```
Q1 = numeric_df[col].quantile(0.25)
```

```
Q3 = numeric_df[col].quantile(0.75)

IQR = Q3 - Q1

df[col] = df[col].clip(lower, upper)
```

Purpose:

Removes extreme values

Prevents model distortion

Makes distributions more stable

Scatter Plots

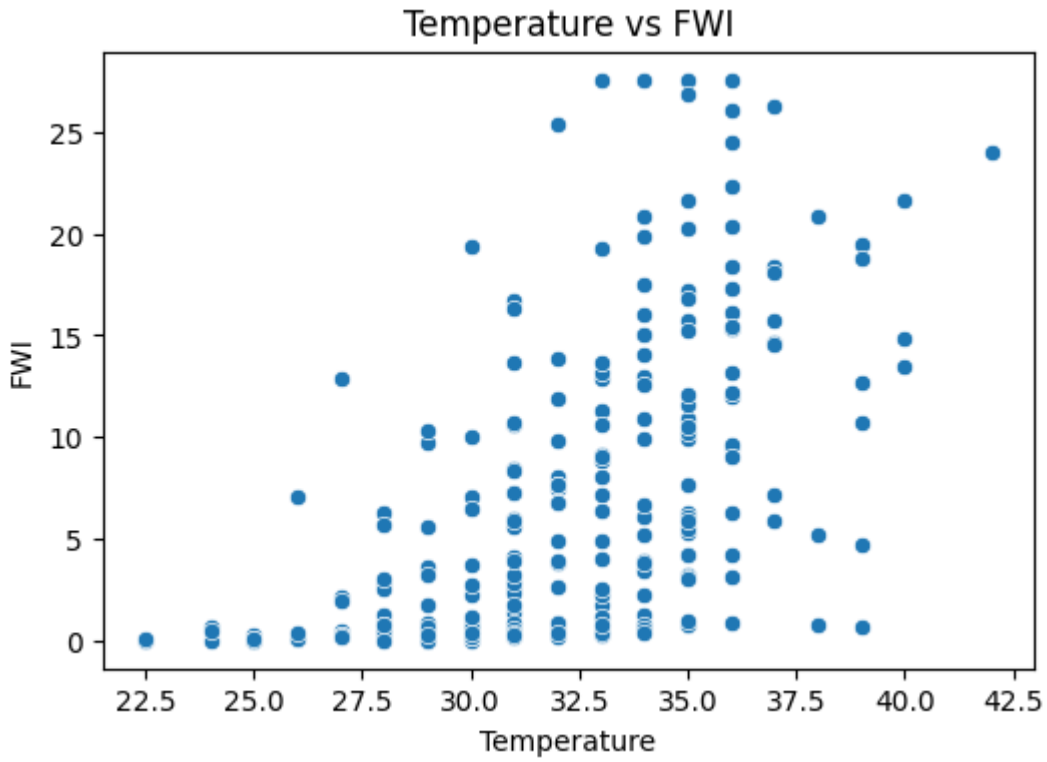
```
sns.scatterplot(x=df['Temperature'], y=df['FWI'])

sns.scatterplot(x=df['Ws'], y=df['FWI'])

sns.scatterplot(x=df['RH'], y=df['FWI'])
```

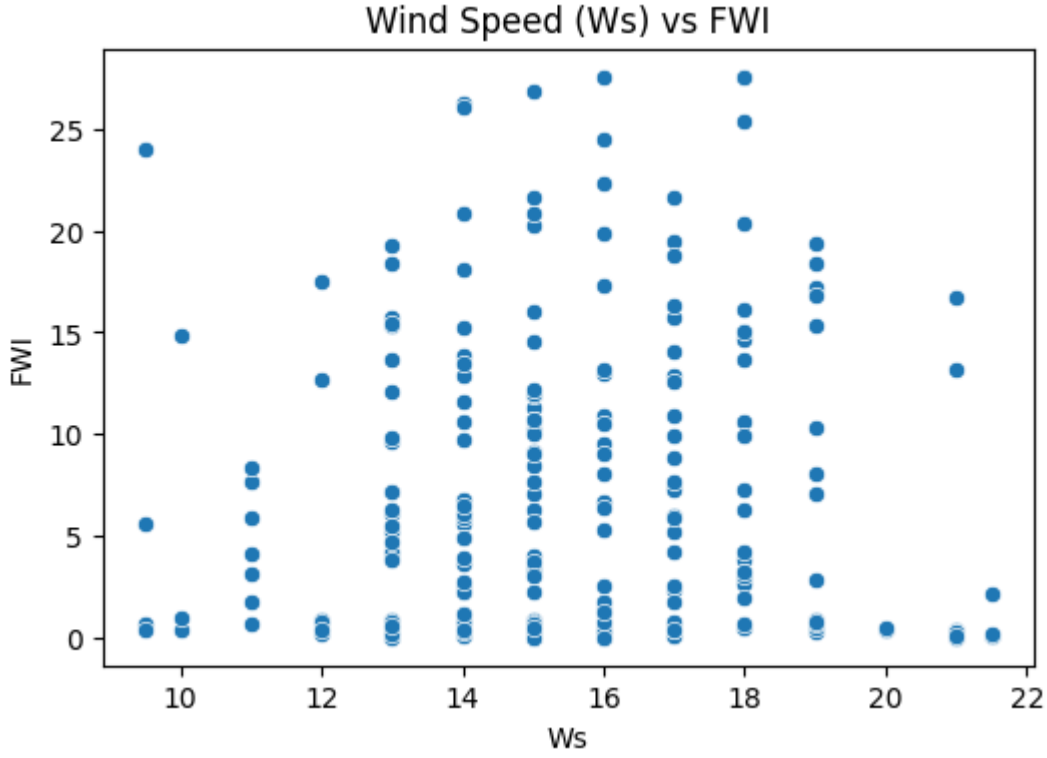
Shows:

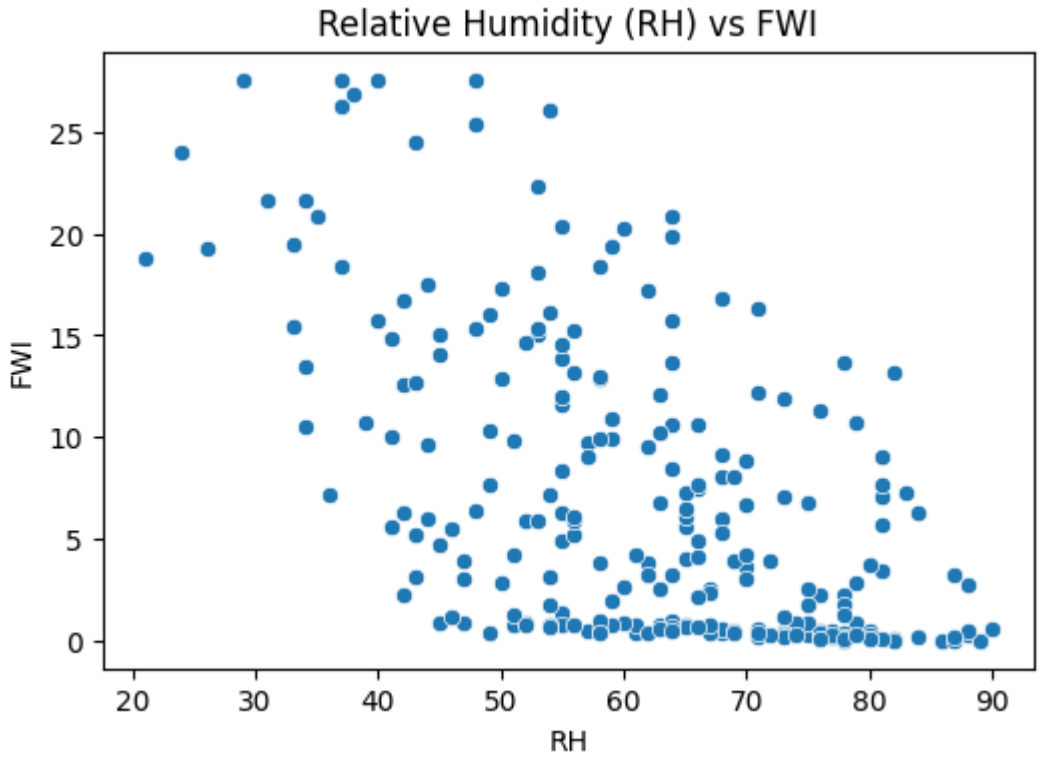
How individual features impact FWI

Linear / non-linear trends

Data clusters


Temperature vs FWI
FWI
Temperature
0
5
10
15
20
25
22.5
25.0
27.5
30.0
32.5
35.0
37.5
40.0
42.5


Wind Speed (Ws) vs FWI
FWI
Ws
0
5
10
15
20
25
10
12
14
16
18
20
22



To save the cleaned dataset.

```
df.to_csv("FWI Cleaned.csv", index=False)
```