shadowfox-intermediate

August 30, 2025

[50]:
```
#Import Libraries & Load Dataset
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Load dataset
df = pd.read_csv(r"C:\Users\varsh\Downloads\delhiaqi.csv")
# Preview data
print(df.head())
print(df.info())
```

```
                 date       co     no    no2    o3    so2   pm2_5    pm10  \
0  2023-01-01 00:00:00  1655.58   1.66  39.41  5.90  17.88  169.29  194.64
1  2023-01-01 01:00:00  1869.20   6.82  42.16  1.99  22.17  182.84  211.08
2  2023-01-01 02:00:00  2510.07  27.72  43.87  0.02  30.04  220.25  260.68
3  2023-01-01 03:00:00  3150.94  55.43  44.55  0.85  35.76  252.90  304.12
4  2023-01-01 04:00:00  3471.37  68.84  45.24  5.45  39.10  266.36  322.80

     nh3
0   5.83
1   7.66
2  11.40
3  13.55
4  14.19
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 561 entries, 0 to 560
Data columns (total 9 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   date    561 non-null    object
 1   co      561 non-null    float64
 2   no      561 non-null    float64
 3   no2     561 non-null    float64
 4   o3      561 non-null    float64
 5   so2     561 non-null    float64
 6   pm2_5   561 non-null    float64
 7   pm10    561 non-null    float64
```

```
 8   nh3     561 non-null    float64
dtypes: float64(8), object(1)
memory usage: 39.6+ KB
None
```

[40]:
```
#Preprocess Dataset
df['date'] = pd.to_datetime(df['date'], errors='coerce')
df['year'] = df['date'].dt.year
df['month'] = df['date'].dt.month
def assign_season(month):
    if month in [12, 1, 2]:
        return 'Winter'
    elif month in [3, 4, 5]:
        return 'Summer'
    elif month in [6, 7, 8, 9]:
        return 'Monsoon'
    else:
        return 'Post-Monsoon'
df['season'] = df['month'].apply(assign_season)
print(df.head())
```

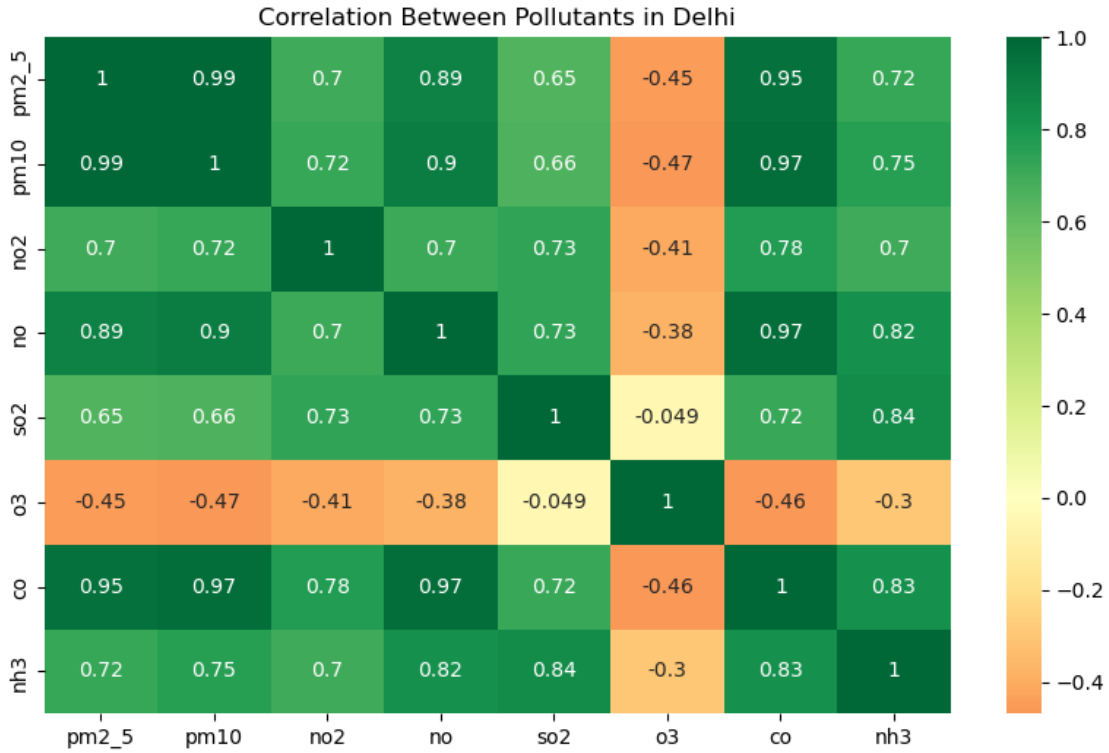
```
                 date       co     no    no2    o3    so2   pm2_5    pm10  \
0 2023-01-01 00:00:00  1655.58   1.66  39.41  5.90  17.88  169.29  194.64
1 2023-01-01 01:00:00  1869.20   6.82  42.16  1.99  22.17  182.84  211.08
2 2023-01-01 02:00:00  2510.07  27.72  43.87  0.02  30.04  220.25  260.68
3 2023-01-01 03:00:00  3150.94  55.43  44.55  0.85  35.76  252.90  304.12
4 2023-01-01 04:00:00  3471.37  68.84  45.24  5.45  39.10  266.36  322.80

     nh3  year  month  season
0   5.83  2023      1  Winter
1   7.66  2023      1  Winter
2  11.40  2023      1  Winter
3  13.55  2023      1  Winter
4  14.19  2023      1  Winter
```

[26]:
```
# Check missing values
print("Missing values before filling:")
print(df.isnull().sum())

# Fill only numeric columns with their median
numeric_cols = df.select_dtypes(include=['number']).columns
df[numeric_cols] = df[numeric_cols].fillna(df[numeric_cols].median())

print("\nMissing values after filling:")
print(df.isnull().sum())
```

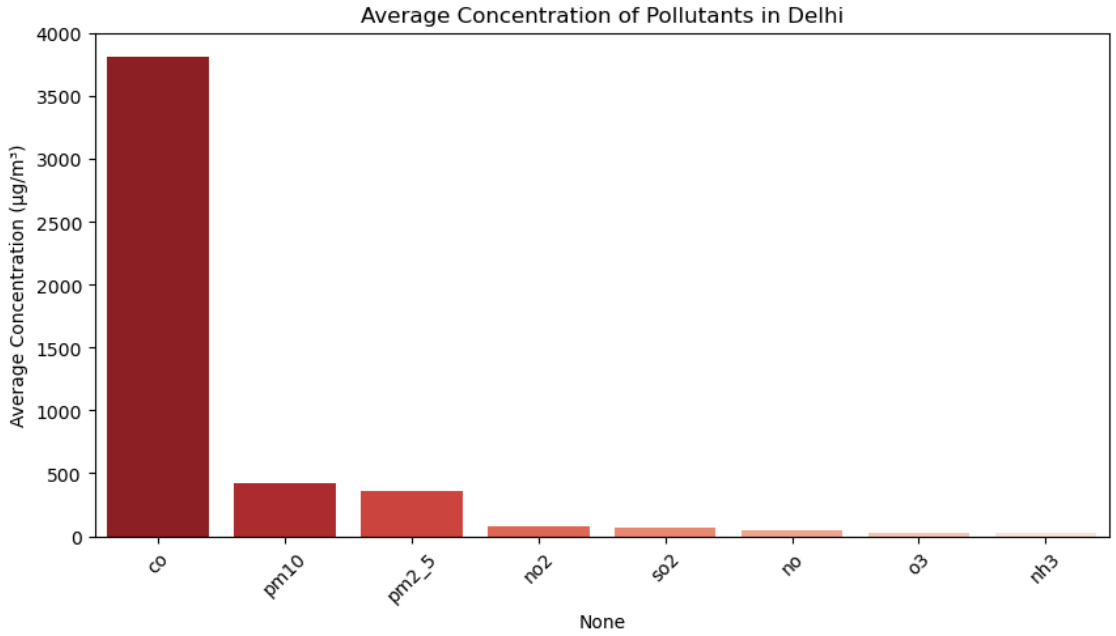
```
Missing values before filling:
date      0
```

```
co        0
no        0
no2       0
o3        0
so2       0
pm2_5     0
pm10      0
nh3       0
year      0
month     0
season    0
dtype: int64

Missing values after filling:
date      0
co        0
no        0
no2       0
o3        0
so2       0
pm2_5     0
pm10      0
nh3       0
year      0
month     0
season    0
dtype: int64
```

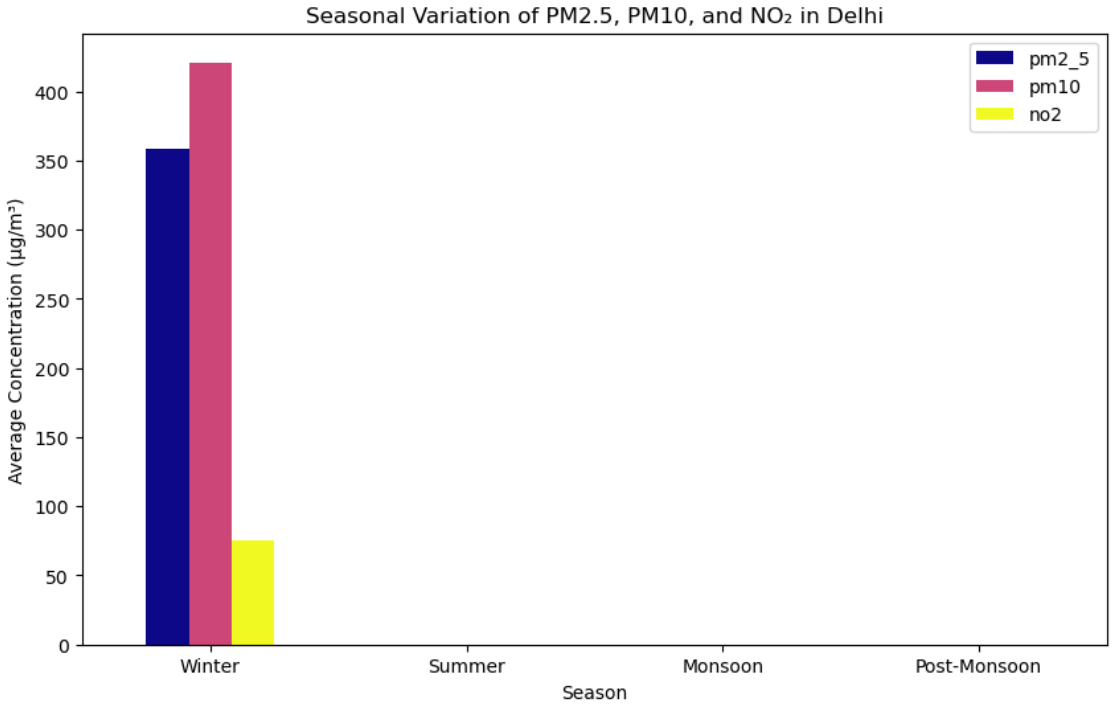
[42]:
```python
# Correlation Analysis of Pollutants
pollutants = ['pm2_5', 'pm10', 'no2', 'no', 'so2', 'o3', 'co', 'nh3']
plt.figure(figsize=(10,6))
sns.heatmap(df[pollutants].corr(), annot=True, cmap="RdYlGn", center=0)
plt.title("Correlation Between Pollutants in Delhi")
plt.show()
```



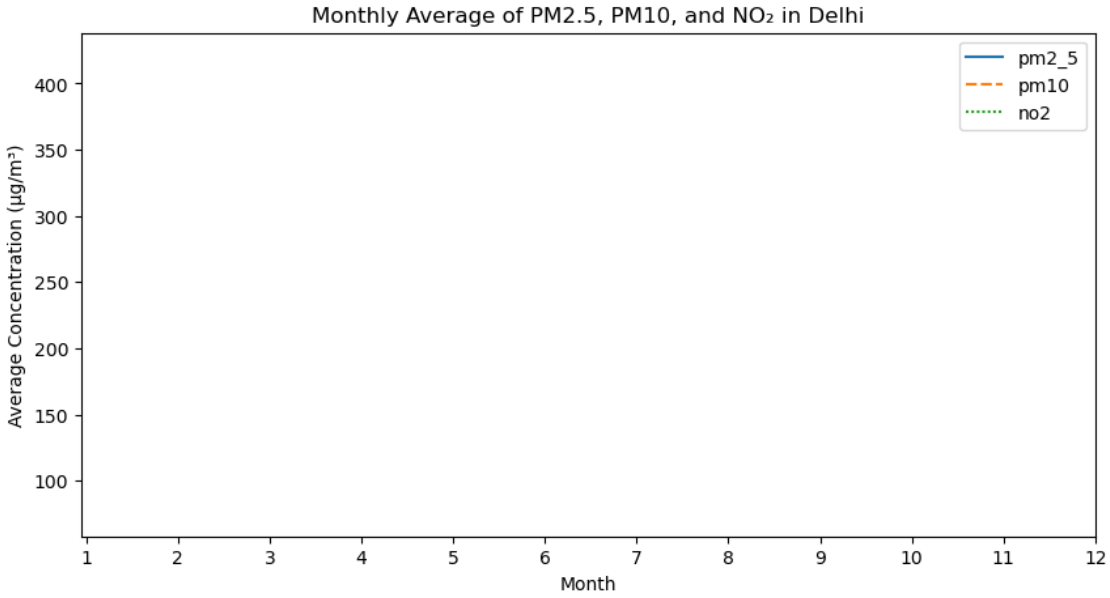
[46]:
```
#Average Concentration of Pollutants
avg_pollutants = df[pollutants].mean().sort_values(ascending=False)
plt.figure(figsize=(10,5))
sns.barplot(
    x=avg_pollutants.index,
    y=avg_pollutants.values,
    hue=avg_pollutants.index,
    palette="Reds_r",
    legend=False
)
plt.ylabel("Average Concentration (µg/m³)")
plt.title("Average Concentration of Pollutants in Delhi")
plt.xticks(rotation=45)
plt.show()
```



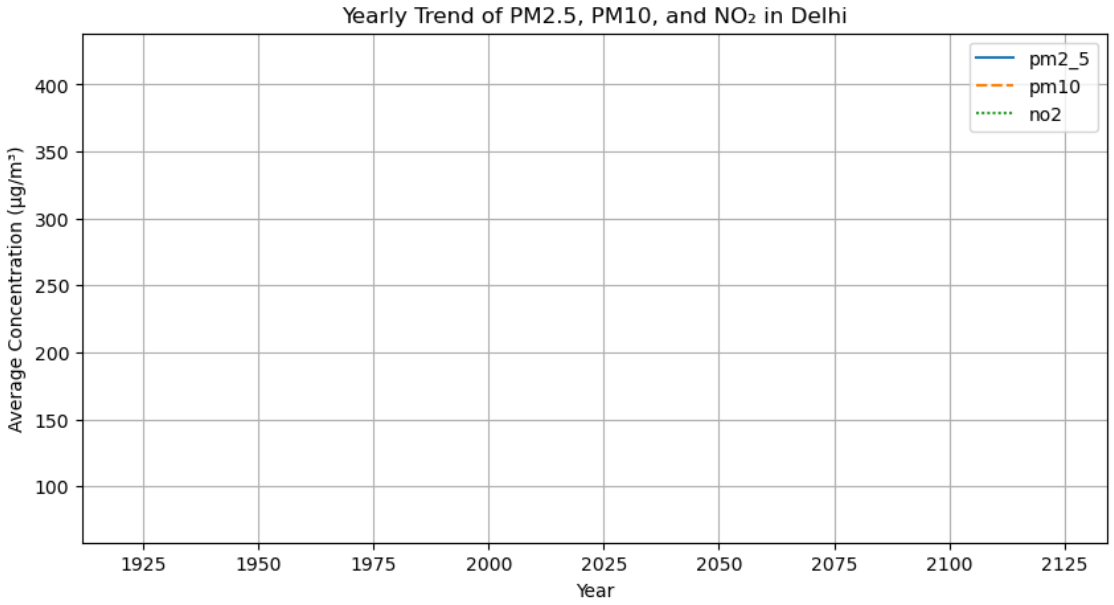
[44]:
```
# Seasonal Variation of Key Pollutants
seasonal_avg = df.groupby("season")[["pm2_5", "pm10", "no2"]].mean()
seasonal_avg = seasonal_avg.reindex(["Winter", "Summer", "Monsoon",␣
 ↪"Post-Monsoon"])

# Bar plot
seasonal_avg.plot(kind="bar", figsize=(10,6), colormap="plasma")
plt.title("Seasonal Variation of PM2.5, PM10, and NO  in Delhi")
plt.ylabel("Average Concentration (µg/m³)")
plt.xlabel("Season")
plt.xticks(rotation=0)
plt.show()
```



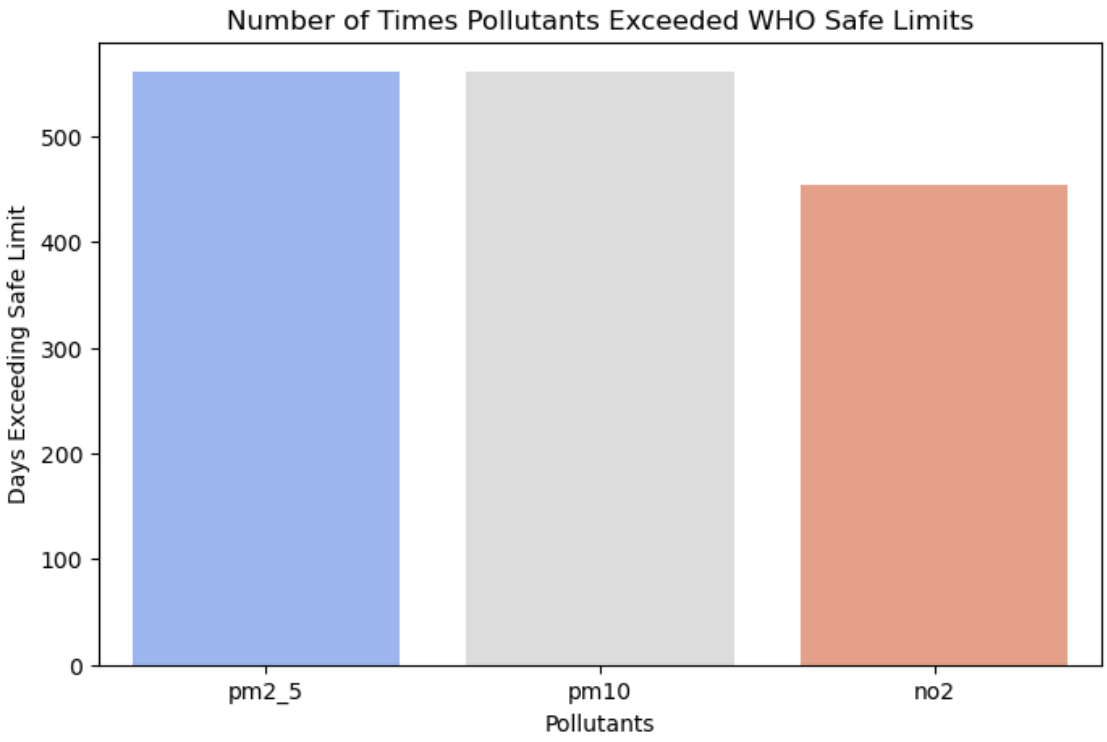
[48]:
```
# Monthly AQI Trend
monthly_avg = df.groupby("month")[["pm2_5", "pm10", "no2"]].mean()

plt.figure(figsize=(10,5))
sns.lineplot(data=monthly_avg)
plt.title("Monthly Average of PM2.5, PM10, and NO  in Delhi")
plt.ylabel("Average Concentration (µg/m³)")
plt.xlabel("Month")
plt.xticks(range(1,13))
plt.show()
```



[68]:
```
# Yearly AQI Trend
df['date'] = pd.to_datetime(df['date'], errors='coerce')
df['year'] = df['date'].dt.year
yearly_avg = df.groupby("year")[["pm2_5", "pm10", "no2"]].mean()
plt.figure(figsize=(10, 5))
sns.lineplot(data=yearly_avg)
plt.title("Yearly Trend of PM2.5, PM10, and NO  in Delhi")
plt.ylabel("Average Concentration (µg/m³)")
plt.xlabel("Year")
plt.grid(True)
plt.show()
```



[64]:
```python
# Pollution Episodes (Exceeding Safe Limits)
safe_limits = {'pm2_5': 25, 'pm10': 50, 'no2': 40}
exceedances = {
    pollutant: (df[pollutant] > safe_limits[pollutant]).sum()
    for pollutant in safe_limits
}

plt.figure(figsize=(8, 5))
sns.barplot(
    x=list(exceedances.keys()),
    y=list(exceedances.values()),
    hue=list(exceedances.keys()),
    palette="coolwarm",
    legend=False
)
plt.title("Number of Times Pollutants Exceeded WHO Safe Limits")
plt.ylabel("Days Exceeding Safe Limit")
plt.xlabel("Pollutants")
plt.show()
```


Number of Times Pollutants Exceeded WHO Safe Limits
Days Exceeding Safe Limit
0
100
200
300
400
500
pm2_5
pm10
no2
Pollutants

[ ]: