TUGAS 1 - MACHINE LEARNING
Data Preparation

KELOMPOK 9

- Fazhira Rizky Harmayani (2208107010012)
- Cut Dahliana (2208107010027)
- Naufal Aqil (2208107010043)
- Hidayat Nur Hakim (2208107010063)
- Riska Haqika Situmorang (2208107010086)

Data Description

Nama Dataset :

PRSA_Data_Aotizhongxin 2013-2017

Sumber Kaggle :

Deskripsi Dataset :

Dataset ini berisi data polusi udara dari kota Aotizhongxin, mencakup berbagai parameter kualitas udara serta faktor lingkungan yang dapat mempengaruhi tingkat polusi. Data ini dapat digunakan untuk menganalisis tren polusi, mengidentifikasi korelasi dengan kondisi kesehatan masyarakat, dan mengevaluasi dampak kebijakan lingkungan terhadap kualitas udara.

Jumlah Data :

Dataset polusi udara Aotizhongxin berisi 35.064 baris dan 18 kolom, mencakup waktu pencatatan (year, month, day, hour), parameter kualitas udara (PM2.5, PM10, SO2, NO2, CO, O3), serta faktor lingkungan (TEMP, PRES, DEWP, RAIN, WSPM, wd). Informasi tambahan mencakup nomor data (No) dan stasiun pemantauan (station). Dataset ini berguna untuk menganalisis tren polusi, dampaknya pada kesehatan, dan efektivitas kebijakan lingkungan.

Format Data : CSV

Data Loading :

- **Library yang digunakan**

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

Kode ini menggunakan Pandas untuk manipulasi data, NumPy untuk perhitungan numerik, Matplotlib dan Seaborn untuk visualisasi, serta MinMaxScaler dari Scikit-learn untuk normalisasi data.

- **Memuat Dataset**

```
# Memuat dataset
df = pd.read_csv("PRSA_Data_Aotizhongxin.csv")
```

Fungsi read_csv() dari Pandas membaca file CSV dan mengonversinya menjadi DataFrame untuk memudahkan manipulasi dan analisis data.

- **Tantangan dalam memuat data ?**

Ukuran datanya tidak terlalu besar sehingga tidak menjadi tantangan dalam memuat datasetnya

Data Understanding

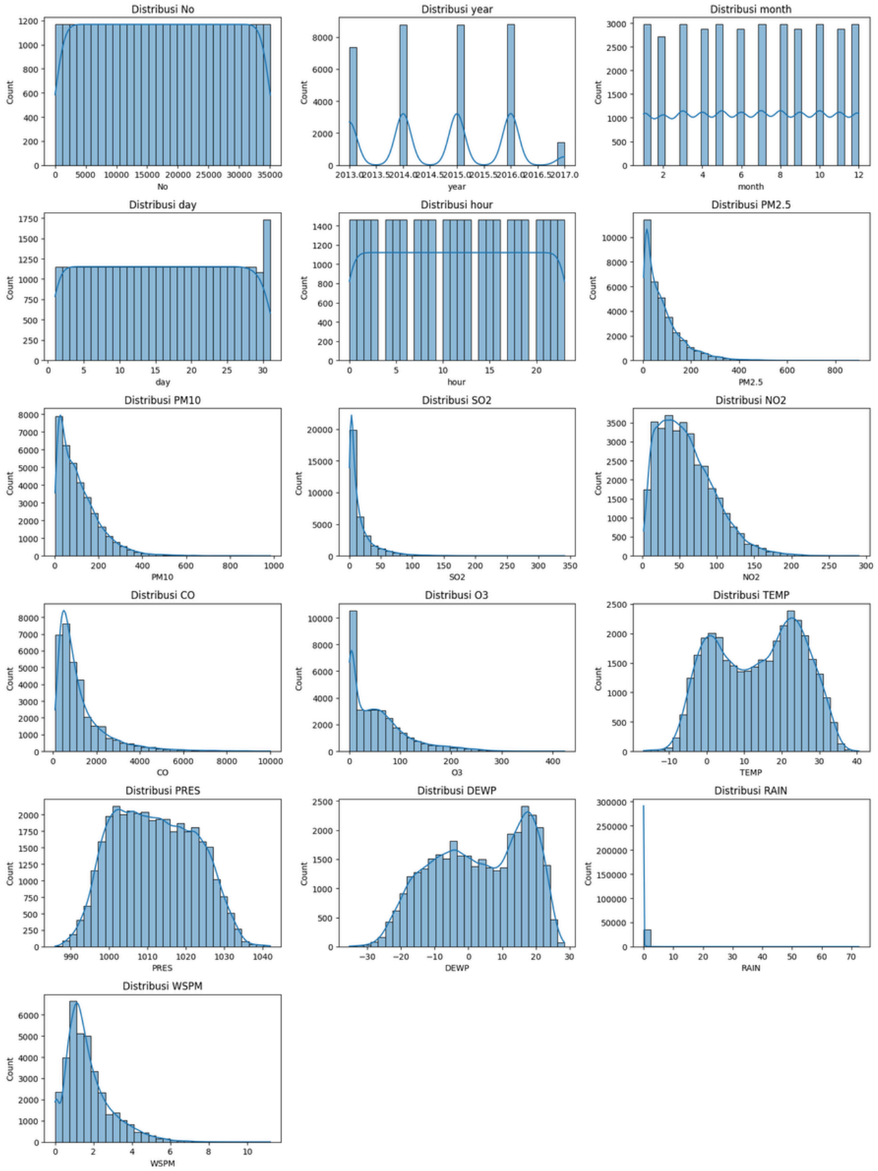
Struktur Data :

```
Struktur Data:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 35064 entries, 0 to 35063
Data columns (total 18 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   No       35064 non-null  int64
 1   year     35064 non-null  int64
 2   month    35064 non-null  int64
 3   day      35064 non-null  int64
 4   hour     35064 non-null  int64
 5   PM2.5    34139 non-null  float64
 6   PM10     34346 non-null  float64
 7   SO2      34129 non-null  float64
 8   NO2      34041 non-null  float64
 9   CO       33288 non-null  float64
 10  O3       33345 non-null  float64
 11  TEMP     35044 non-null  float64
 12  PRES     35044 non-null  float64
 13  DEWP     35044 non-null  float64
 14  RAIN     35044 non-null  float64
 15  wd       34983 non-null  object
 16  WSPM     35050 non-null  float64
 17  station  35064 non-null  object
dtypes: float64(11), int64(5), object(2)
memory usage: 4.8+ MB
None
```

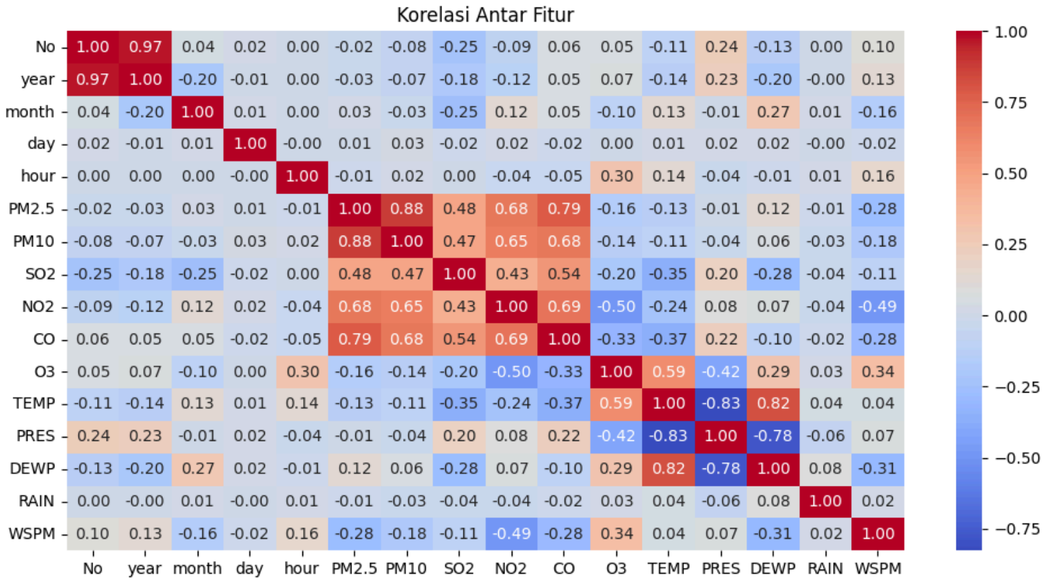
Missing Value :

```
Missing Values:
No            0
year          0
month         0
day           0
hour          0
PM2.5       925
PM10        718
SO2         935
NO2        1023
CO         1776
O3         1719
TEMP         20
PRES         20
DEWP         20
RAIN         20
wd           81
WSPM         14
station       0
dtype: int64
```

Distribusi Data :



Kolerasi Data :



Penjelasan :

Struktur data dapat dilihat tipe data dari kolom-kolom feature. Untuk Missing Value terdapat pada beberapa kolom, Viualisasi distribusi data ini menunjukkan nilai dari setiap variable numerik dalam dataset ada 5 kolom yang memiliki distribusi cenderung skewed ke kanan ada 3 kolom lebih mendektai distribusi normal sedangkan 1 kolom memiliki ditribusi yang tidak merata. Heatmap korelasi menunjjukan hubungan antara berbagai fitur numerik dalam dataset.

Data Preparation

Menangani Missing Value :

```
# Kolom dengan missing values kecil (<100) diimputasi (mengisi)
# Mengisi kolom numerik dengan median tanpa inplace
df_numeric['WSPM'] = df_numeric['WSPM'].fillna(df_numeric['WSPM'].median())


# Disarankan untuk diimputasi dengan median atau interpolasi, karena data polutan udara biasanya memiliki outlier.
# Mengisi kolom dengan missing values sedang menggunakan median
cols_with_median = ['PM2.5', 'PM10', 'SO2', 'NO2', 'CO', 'O3', 'TEMP', 'PRES', 'DEWP', 'RAIN']
df_numeric[cols_with_median] = df_numeric[cols_with_median].apply(lambda x: x.fillna(x.median()))

# Mengecek kembali jumlah missing values setelah imputasi
print("Missing values setelah penanganan:")
print(df_numeric.isnull().sum())
```

```
Missing values setelah penanganan:
No       0
year     0
month    0
day      0
hour     0
PM2.5    0
PM10     0
SO2      0
NO2      0
CO       0
O3       0
TEMP     0
PRES     0
DEWP     0
RAIN     0
WSPM     0
dtype: int64
```

Missing values diisi dengan mode untuk data kategorikal (wd) dan median untuk data numerik (WSPM, PM2.5, CO, dll.) agar lebih stabil terhadap outlier. Kolom CO tidak dihapus karena masih memiliki 82% data, sehingga lebih baik diimputasi menggunakan median.

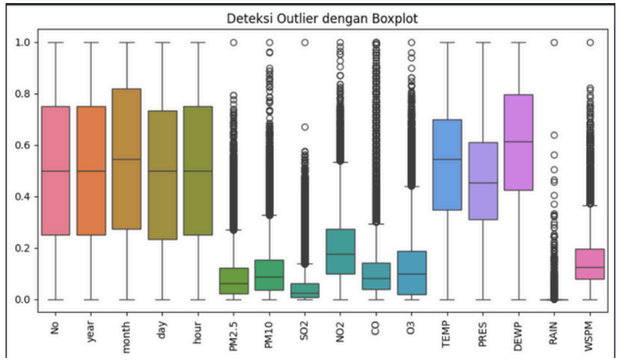
Normalisasi Data :

```
# Inisialisasi scaler untuk masing-masing dataset
scaler = MinMaxScaler()

# Daftar kolom numerik yang akan dinormalisasi
num_cols = df_numeric.select_dtypes(include=['float64', 'int64']).columns.tolist()

# Normalisasi dataset train (fit dan transform di train)
df_numeric[num_cols] = scaler.fit_transform(df_numeric[num_cols])
```

Normalisasi diperlukan karena dataset memiliki variabel dengan skala berbeda yang dapat menyebabkan algoritma ML lebih condong ke fitur yang besar

Mendektesi Outlier :

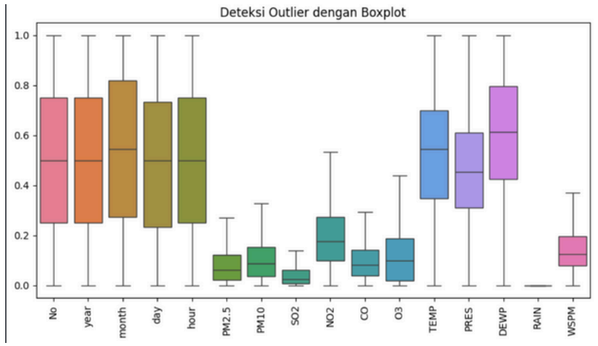
```
# Visualisasi boxplot untuk semua kolom numerik
plt.figure(figsize=(10,5))
sns.boxplot(data=df_numeric)
plt.xticks(rotation=90)  # Rotasi label agar lebih mudah dibaca
plt.title("Deteksi Outlier dengan Boxplot")
plt.show()
```



Boxplot menunjukkan bahwa terdapat banyak outlier pada kolom PM2.5, PM10, SO2, NO2, CO, O3, RAIN, dan WSPM, yang ditandai dengan titik-titik di luar batas whisker. Outlier ini mengindikasikan adanya variasi ekstrem dalam data polusi udara dan cuaca, yang perlu dianalisis lebih lanjut.

Penangganan Outlier :

```
def cap_outliers(df_numeric, columns):
    for column in columns:
        Q1 = df_numeric[column].quantile(0.25)
        Q3 = df_numeric[column].quantile(0.75)
        IQR = Q3 - Q1

        lower_bound = Q1 - 1.5 * IQR
        upper_bound = Q3 + 1.5 * IQR

        # Ganti nilai outlier dengan batas bawah dan atas
        df_numeric[column] = df_numeric[column].clip(lower=lower_bound, upper=upper_bound)

    return df_numeric

# Kolom numerik yang mengandung outlier
feature_num = ['PM2.5', 'PM10', 'SO2', 'NO2', 'CO', 'O3', 'RAIN', 'WSPM']

# Menangani outlier dengan metode capping
df_numeric = cap_outliers(df_numeric, feature_num)
```



Setelah dilakukan capping, data outlier pada kolom PM2.5, PM10, SO2, NO2, CO, O3, RAIN, dan WSPM berhasil dikurangi, sehingga distribusi data menjadi lebih stabil. Hal ini membantu dalam mengurangi pengaruh nilai ekstrem tanpa menghilangkan informasi penting dalam dataset.