CSV Data Exploration & Visualization

```
# Importing required libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

```
# Loading csv file
df = pd.read_csv('electricity_demand.csv')
```

```
# Displaying first 10 rows
df.head(10)
```

	Timestamp	hour	dayofweek	month	year	dayofyear	Temperature	Humidity	Demand
0	01-Jan-20	0.0	2.0	1.0	2020.0	1.0	3.000000	61.288951	2457.119872
1	01-Jan-20	1.0	2.0	1.0	2020.0	1.0	3.000000	52.873702	2269.904712
2	01-Jan-20	2.0	2.0	1.0	2020.0	1.0	4.244482	36.341783	2215.640403
3	01-Jan-20	3.0	2.0	1.0	2020.0	1.0	3.000000	72.629378	2174.232413
4	01-Jan-20	4.0	2.0	1.0	2020.0	1.0	3.881208	90.582444	2472.453006
5	01-Jan-20	5.0	2.0	1.0	2020.0	1.0	10.822571	67.753433	3104.845505
6	01-Jan-20	6.0	2.0	1.0	2020.0	1.0	6.306673	94.912591	3759.476912
7	01-Jan-20	7.0	2.0	1.0	2020.0	1.0	7.464640	74.456860	4114.486001
8	01-Jan-20	8.0	2.0	1.0	2020.0	1.0	14.746876	66.725005	4575.159503
9	01-Jan-20	9.0	2.0	1.0	2020.0	1.0	19.738254	55.036160	4512.169696

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
# Displaying Shape of the dataset
df.shape
```

```
(17105, 9)
```

```
# Displaying column names
df.columns
```

```
Index(['Timestamp', 'hour', 'dayofweek', 'month', 'year', 'dayofyear',
       'Temperature', 'Humidity', 'Demand'],
      dtype='object')
```

```
# Displaying data types of each column
df.dtypes
```

	0
Timestamp	object
hour	float64
dayofweek	float64
month	float64
year	float64
dayofyear	float64
Temperature	float64
Humidity	float64
Demand	float64

dtype: object

```
# Getting only numeric columns
numeric_cols = df.select_dtypes(include=['number']).columns
```

```
# Displaying numeric columns
numeric_cols
```

```
Index(['hour', 'dayofweek', 'month', 'year', 'dayofyear', 'Temperature',
       'Humidity', 'Demand'],
      dtype='object')
```

```
# Doing .describe() for numeric columns.
df[numeric_cols].describe()
```

	hour	dayofweek	month	year	dayofyear	Temperature	Humidity	Demand
count	17098.000000	17101.000000	17101.000000	17101.000000	17101.00000	17101.000000	17100.000000	17100.000000
mean	11.494093	3.003567	6.380212	2020.486346	178.83767	25.414858	59.841600	4952.272896
std	6.920812	1.999544	3.378327	0.499828	103.09274	12.705231	18.474323	1407.556819
min	0.000000	0.000000	1.000000	2020.000000	1.00000	3.000000	20.000000	1701.103609
25%	5.000000	1.000000	3.000000	2020.000000	90.00000	15.698773	46.105224	3964.182972
50%	11.000000	3.000000	6.000000	2020.000000	179.00000	25.454371	59.870157	4980.331904
75%	17.000000	5.000000	9.000000	2021.000000	268.00000	34.929106	73.800433	5950.530153
max	23.000000	6.000000	12.000000	2021.000000	366.00000	50.000000	95.000000	8639.979887

```
# Checking null values in each column
df.isna().sum()
```

	0
Timestamp	0
hour	7
dayofweek	4
month	4
year	4
dayofyear	4
Temperature	4
Humidity	5
Demand	5

dtype: int64

```
# Filling mean values
df.fillna(df.mean(numeric_only=True), inplace=True)
```

```
# Checking for any null values after filling mean values
df.isna().sum()
```

	0
Timestamp	0
hour	0
dayofweek	0
month	0
year	0
dayofyear	0
Temperature	0
Humidity	0
Demand	0

dtype: int64

```
df.head()
```

	Timestamp	hour	dayofweek	month	year	dayofyear	Temperature	Humidity	Demand
0	01-Jan-20	0.0	2.0	1.0	2020.0	1.0	3.000000	61.288951	2457.119872
1	01-Jan-20	1.0	2.0	1.0	2020.0	1.0	3.000000	52.873702	2269.904712
2	01-Jan-20	2.0	2.0	1.0	2020.0	1.0	4.244482	36.341783	2215.640403
3	01-Jan-20	3.0	2.0	1.0	2020.0	1.0	3.000000	72.629378	2174.232413
4	01-Jan-20	4.0	2.0	1.0	2020.0	1.0	3.881208	90.582444	2472.453006

Next steps: Generate code with df View recommended plots New interactive sheet

```
# Filtering rows (1)
df.loc[df['Humidity'] > 50]
```

```
       Timestamp  hour  dayofweek  month    year  dayofyear  Temperature   Humidity       Demand
0      01-Jan-20   0.0        2.0    1.0  2020.0        1.0     3.000000  61.288951  2457.119872
1      01-Jan-20   1.0        2.0    1.0  2020.0        1.0     3.000000  52.873702  2269.904712
3      01-Jan-20   3.0        2.0    1.0  2020.0        1.0     3.000000  72.629378  2174.232413
4      01-Jan-20   4.0        2.0    1.0  2020.0        1.0     3.881208  90.582444  2472.453006
5      01-Jan-20   5.0        2.0    1.0  2020.0        1.0    10.822571  67.753433  3104.845505
...          ...   ...        ...    ...     ...        ...          ...        ...          ...
17097  13-Dec-21   9.0        0.0   12.0  2021.0      347.0    15.325196  63.197049  4381.335825
17099  13-Dec-21  11.0        0.0   12.0  2021.0      347.0    19.138703  60.166684  4251.877221
17100  13-Dec-21  12.0        0.0   12.0  2021.0      347.0    17.448002  75.748361  4469.636222
17101  13-Dec-21  13.0        0.0   12.0  2021.0      347.0    19.548059  63.738039  4659.771561
17104  13-Dec-21  16.0        0.0   12.0  2021.0      347.0    13.247942  59.841600  4952.272896

11657 rows × 9 columns
```

```
# Filtering rows (2)
df.loc[df['year']== 2020]
```

```
      Timestamp  hour  dayofweek  month    year  dayofyear  Temperature   Humidity       Demand
0     01-Jan-20   0.0        2.0    1.0  2020.0        1.0     3.000000  61.288951  2457.119872
1     01-Jan-20   1.0        2.0    1.0  2020.0        1.0     3.000000  52.873702  2269.904712
2     01-Jan-20   2.0        2.0    1.0  2020.0        1.0     4.244482  36.341783  2215.640403
3     01-Jan-20   3.0        2.0    1.0  2020.0        1.0     3.000000  72.629378  2174.232413
4     01-Jan-20   4.0        2.0    1.0  2020.0        1.0     3.881208  90.582444  2472.453006
...         ...   ...        ...    ...     ...        ...          ...        ...          ...
8779  31-Dec-20  19.0        3.0   12.0  2020.0      366.0    11.816995  46.331397  4745.575444
8780  31-Dec-20  20.0        3.0   12.0  2020.0      366.0    11.594246  43.219001  4561.973620
8781  31-Dec-20  21.0        3.0   12.0  2020.0      366.0     3.000000  52.711842  3766.177523
8782  31-Dec-20  22.0        3.0   12.0  2020.0      366.0     3.000000  64.753917  3120.111392
8783  31-Dec-20  23.0        3.0   12.0  2020.0      366.0     3.000000  49.047211  2691.443838

8784 rows × 9 columns
```

```
# Sorting the dataset by column 'Demand' in descending order.
df.sort_values(by='Demand', ascending=False)
```

```
       Timestamp  hour  dayofweek  month    year  dayofyear  Temperature   Humidity       Demand
10962  01-Apr-21  18.0        3.0    4.0  2021.0       91.0    19.396609  33.345002  8639.979887
4578   09-Jul-20  18.0        3.0    7.0  2020.0      191.0    43.191233  62.467953  8339.168142
13073  28-Jun-21  17.0        0.0    6.0  2021.0      179.0    47.600283  61.404128  8331.474310
4745   16-Jul-20  17.0        3.0    7.0  2020.0      198.0    48.619821  74.674535  8323.272995
13578  19-Jul-21  18.0        0.0    7.0  2021.0      200.0    41.180157  64.169168  8289.606440
...          ...   ...        ...    ...     ...        ...          ...        ...          ...
435    19-Jan-20   3.0        6.0    1.0  2020.0       19.0     3.000000  73.050075  1831.830411
9170   17-Jan-21   2.0        6.0    1.0  2021.0       17.0     4.377993  59.411704  1778.824719
411    18-Jan-20   3.0        5.0    1.0  2020.0       18.0     3.000000  60.237501  1769.569040
9338   24-Jan-21   2.0        6.0    1.0  2021.0       24.0     3.000000  52.168535  1743.392220
602    26-Jan-20   2.0        6.0    1.0  2020.0       26.0     3.000000  35.166937  1701.103609

17105 rows × 9 columns
```

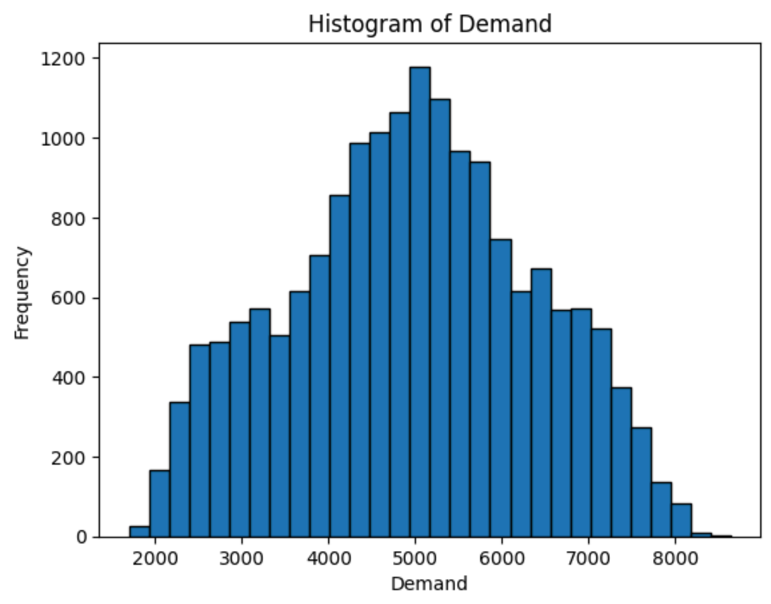
```
# Grouped by column 'Timestamp'(Not a numeric) and calculated mean of column 'Demand'(Numeric)
df_grouped = df.groupby('Timestamp')['Demand'].mean().reset_index()

# Displaying grouped and aggregated values
df_grouped
```

	Timestamp	Demand
0	01-Apr-20	4712.549076
1	01-Apr-21	4883.449609
2	01-Aug-20	5440.966254
709	31-May-20	5413.860969

Next steps: Generate code with df_grouped View recommended plots New interactive sheet

```
# Histogram for numeric column
plt.hist(df['Demand'], bins=30, edgecolor='black')
plt.title('Histogram of Demand')
plt.xlabel('Demand')
plt.ylabel('Frequency')
plt.show()
```



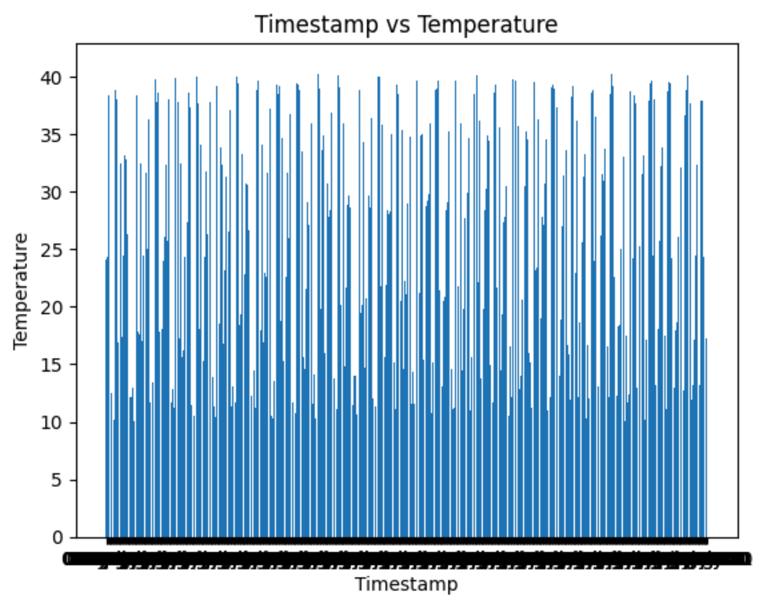
```
# Bar chart
df.head()
```

	Timestamp	hour	dayofweek	month	year	dayofyear	Temperature	Humidity	Demand
0	01-Jan-20	0.0	2.0	1.0	2020.0	1.0	3.000000	61.288951	2457.119872
1	01-Jan-20	1.0	2.0	1.0	2020.0	1.0	3.000000	52.873702	2269.904712
2	01-Jan-20	2.0	2.0	1.0	2020.0	1.0	4.244482	36.341783	2215.640403
3	01-Jan-20	3.0	2.0	1.0	2020.0	1.0	3.000000	72.629378	2174.232413
4	01-Jan-20	4.0	2.0	1.0	2020.0	1.0	3.881208	90.582444	2472.453006

Next steps: Generate code with df View recommended plots New interactive sheet

```
grouped_1 = df.groupby('Timestamp')['Temperature'].mean()
```

```
plt.bar(grouped_1.index, grouped_1.values)
plt.title('Timestamp vs Temperature')
plt.xlabel('Timestamp')
plt.ylabel('Temperature')
plt.show()
```



Could not connect to the reCAPTCHA service. Please check your internet connection and reload to get a reCAPTCHA challenge.