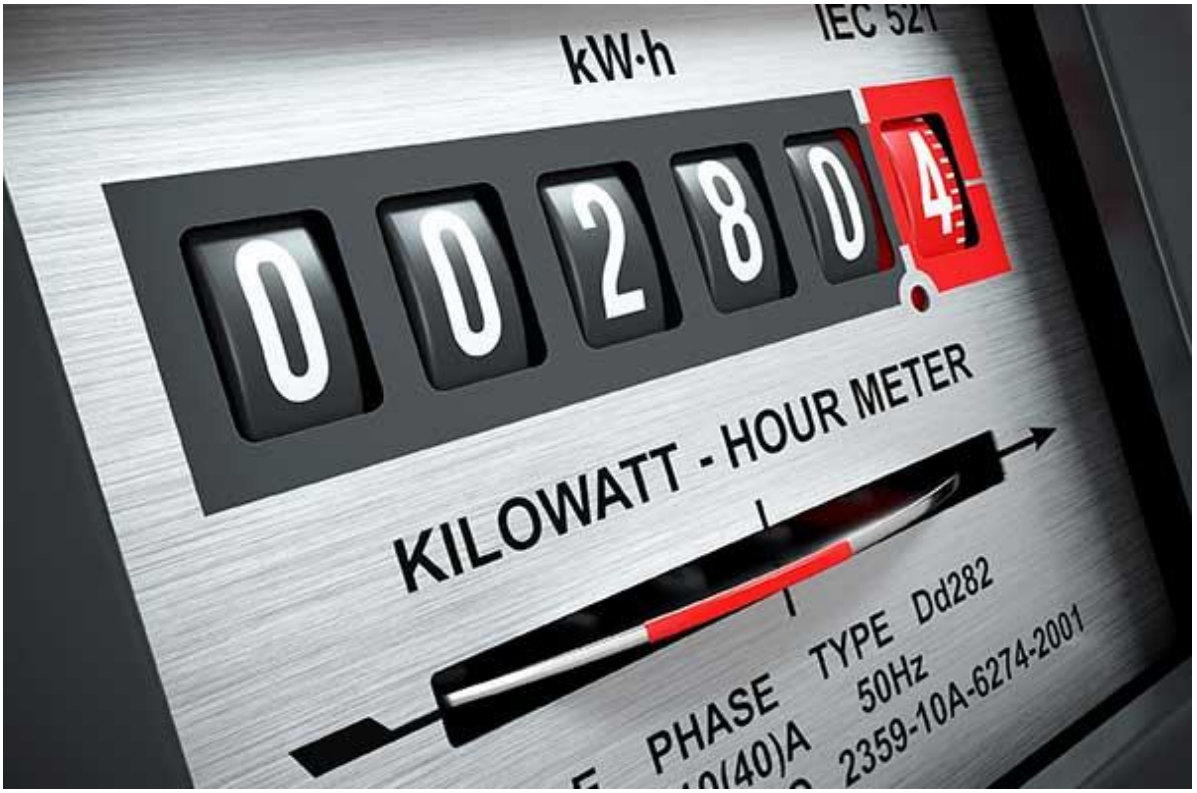
PHASE 3

DEVELOPMENT PART 1

MEASURE ENERGY CONSUMPTION



The Measure Energy Consumption Dataset (also known as the Household Electric Power Consumption Dataset) is a publicly available dataset on energy consumption in a single household over a period of almost 4 years. The dataset was collected at a one-minute sampling rate and contains measurements of global active power, global reactive power, voltage, current intensity, and sub-metering values for three electrical appliances.

The dataset is a valuable resource for researchers and practitioners working on energy forecasting, energy efficiency, and smart grid technologies. It can be used to develop and train machine learning models to predict energy consumption, identify patterns in energy usage, and develop energy-saving strategies.

The dataset contains a variety of features, including:

- *Energy consumption data (in kWh or other units)*
- *Time data (date and time of measurement)*
- *Weather data (temperature, humidity, etc.)*
- *Building metadata (square footage, number of occupants, etc.)*
- *Household metadata (appliance ownership, etc.)*

The dataset can be used for a variety of purposes, including:

- *Analyzing energy consumption patterns*
- *Forecasting future energy demand*
- *Developing energy efficiency strategies*
- *Evaluating the impact of energy policies*

***Example Use Cases**:*

Here are a few examples of how the Measure Energy Consumption Dataset can be used:

- *A researcher could use the dataset to study the relationship between energy consumption and weather conditions.*
- *An energy utility could use the dataset to forecast future energy demand and plan for capacity needs.*
- *A government agency could use the dataset to evaluate the impact of energy efficiency programs.*
- *A homeowner could use the dataset to identify areas where they can reduce their energy consumption.*

There are many different datasets available for measuring energy consumption, depending on the specific needs of the user. Here is a list of datasets for measuring energy consumption.Some examples include:

- ***Electricity Consumption Dataset:***

  This dataset contains electricity consumption data for a single household over a period of 4 years. The data is sampled at a rate of one minute, and includes measurements of global active power, global reactive power, voltage, and current intensity.

- ***Buildings Energy Consumption Dataset:***

  This dataset contains energy consumption data for a variety of commercial and residential buildings. The data is sampled at a rate of one hour, and includes measurements of total electricity consumption, heating consumption, cooling consumption, and weather data.

  This dataset contains hourly energy consumption data for 28 buildings over a period of two years. The data includes information on building type, size, location, and weather conditions.

- ***Energy Consumption Time Series Dataset:***

This dataset contains energy consumption data for a variety of electrical devices, including refrigerators, air conditioners, and washing machines. The data is sampled at a rate of 15 minutes, and includes measurements of power consumption.

- ***Household Electric Power Consumption Dataset*** *:*

  This dataset contains minute-level electricity consumption data for a single household over a period of four years. The data includes information on the total energy consumption, as well as the energy consumption of individual appliances.

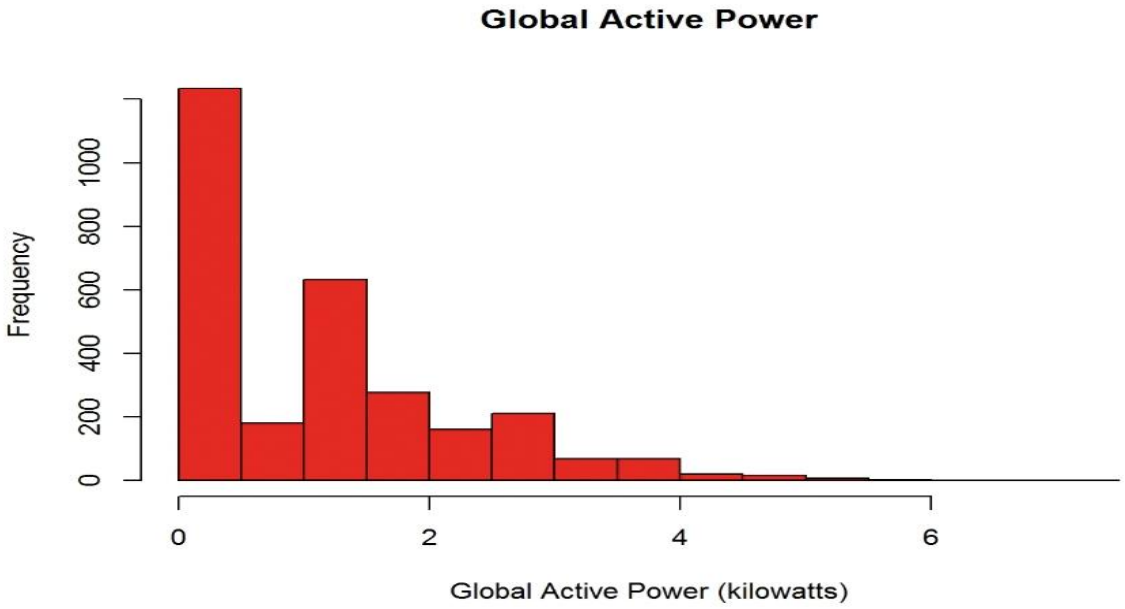
- ***Smart Meter Dataset*** *:*

  This dataset contains hourly electricity consumption data for 100 households over a period of two years. The data includes information on the total energy consumption, as well as the energy consumption of individual appliances.

- ***Buildings Energy Dataset****:*

  The CEED database contains energy consumption data for over 100,000 commercial buildings in the United States. The data includes information on building type, size, location, and weather conditions.

```
fh <- file("household_power_consumption.txt")
ba <- read.table(text = grep("^[1,2]/2/2007", readLines(fh), value = TRUE), col.names = c("Date", "Tim
e", "Global_active_power", "Global_reactive_power", "Voltage", "Global_intensity", "Sub_metering_1",
"Sub_metering_2", "Sub_metering_3"), sep = ";", header = TRUE)

# Generating Plot 1
hist(ba$Global_active_power, col = "red", main = paste("Global Active Power"), xlab = "Global Active P
ower (kilowatts)")
```



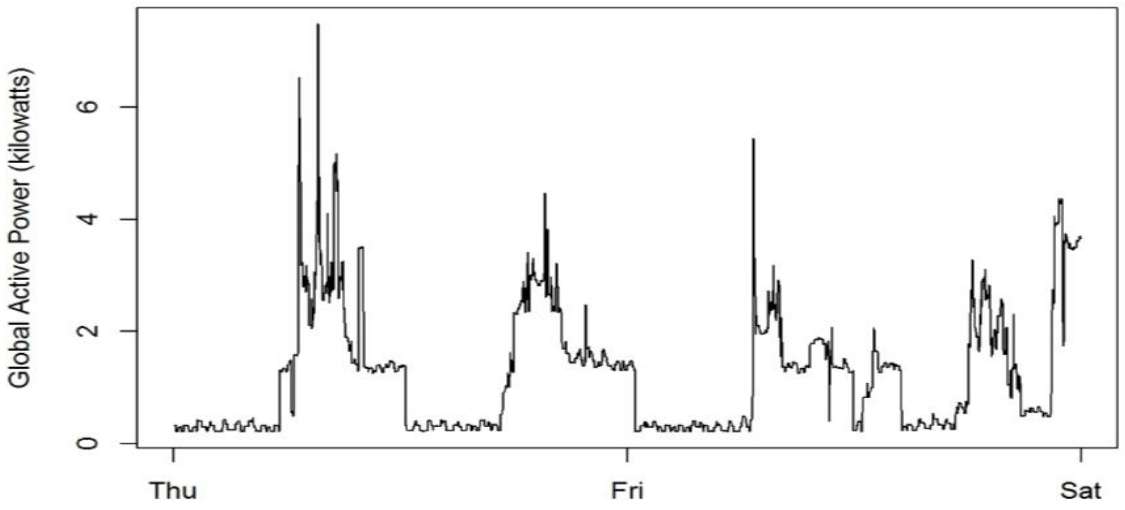
These datasets can be used for a variety of purposes, such as:

- ***Developing Energy Forecasting Models:*** *Energy forecasting models can be used to predict future energy consumption, which can help businesses and governments to make better decisions about energy planning and resource allocation.*

- ***Identifying Energy Efficiency Opportunities:*** *Energy consumption data can be used to identify areas where energy can be saved. For example, a business might use energy consumption data to identify which devices are using the most energy, and then take steps to reduce their consumption.*

- ***Understanding Energy Consumption Patterns:*** *Energy consumption data can be used to understand how energy is consumed in different sectors, such as households, businesses, and industry. This information can be used to develop policies and programs to promote energy efficiency and reduce greenhouse gas emissions.*

```r
## Getting full dataset
data_full <- read.csv("household_power_consumption.txt", header = T, sep = ';',
                      na.strings = "?", nrows = 2075259, check.names = F,
                      stringsAsFactors = F, comment.char = "", quote = '\"')
data_full$Date <- as.Date(data_full$Date, format = "%d/%m/%Y")

## Subsetting the data
data <- subset(data_full, subset = (Date >= "2007-02-01" & Date <= "2007-02-02"))
rm(data_full)

## Converting dates
datetime <- paste(as.Date(data$Date), data$Time)
data$Datetime <- as.POSIXct(datetime)

## Generating Plot 2
plot(data$Global_active_power ~ data$Datetime, type = "l",
     ylab = "Global Active Power (kilowatts)", xlab = "")
```



When choosing a dataset for measuring energy consumption, it is important to consider the following factors:

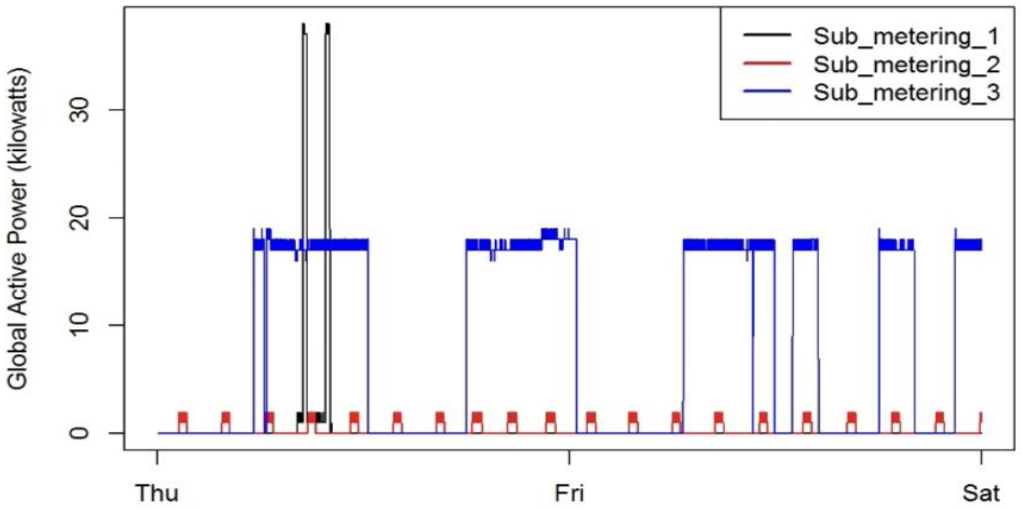
- ***The Type Of Energy Being Measured**: The dataset should contain data for the type of energy that you are interested in measuring, such as electricity, natural gas, or propane.*

- ***The Time Period Covered By The Dataset**: The dataset should cover the time period that you are interested in studying. For example, if you are interested in studying energy consumption patterns over the past year, then you will need a dataset that covers that time period.*

- ***The Frequency Of The Data*****:** *The dataset should be sampled at a frequency that is appropriate for your needs. For example, if you are interested in studying short-term energy consumption patterns, then you will need a dataset that is sampled at a high frequency, such as one minute.*

- ***The Format Of The Data*****:** *The dataset should be in a format that is compatible with your analysis tools. For example, if you are using a statistical software package, then you will need a dataset that is in a format that is compatible with that software package.*

```
## Generating Plot 3
with(data, {
    plot(Sub_metering_1 ~ Datetime, type = "l",
    ylab = "Global Active Power (kilowatts)", xlab = "")
    lines(Sub_metering_2 ~ Datetime, col = 'Red')
    lines(Sub_metering_3 ~ Datetime, col = 'Blue')
})
legend("topright", col = c("black", "red", "blue"), lty = 1, lwd = 2,
       legend = c("Sub_metering_1", "Sub_metering_2", "Sub_metering_3"))
```

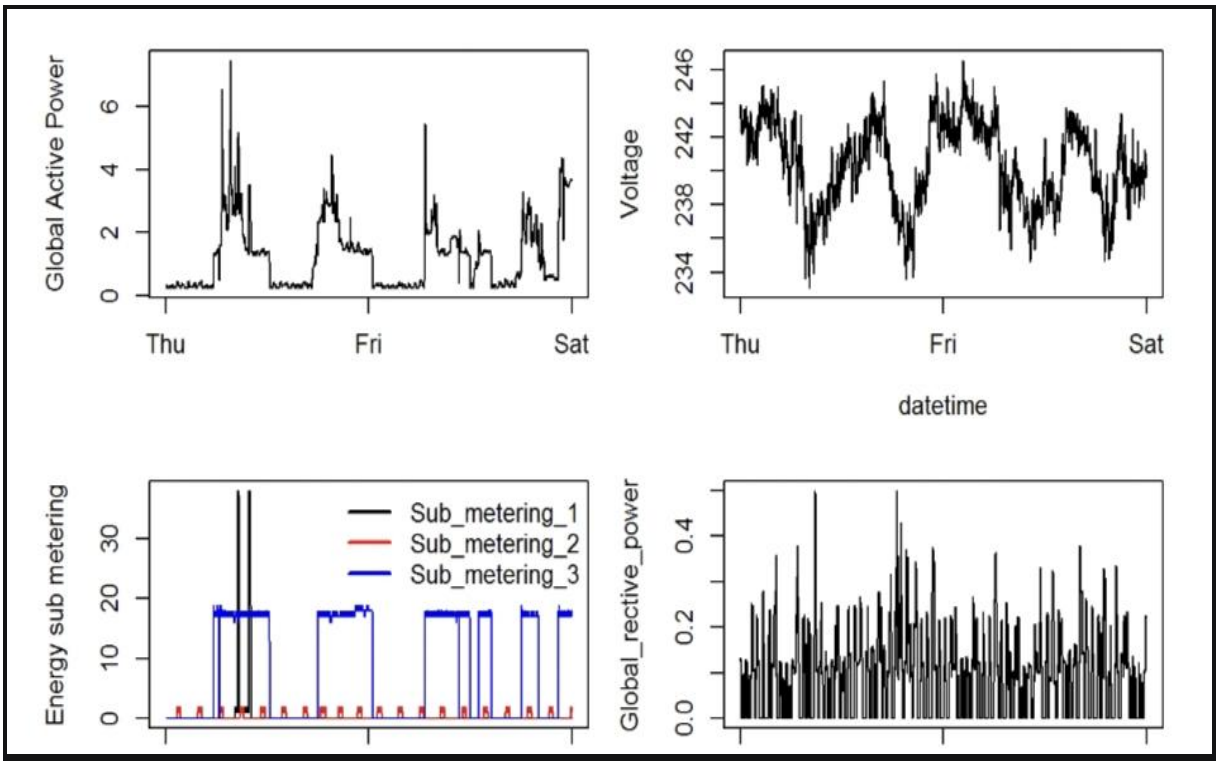


Once you have selected a dataset, you can begin to analyze the data to identify trends and patterns. You can also use the data to develop energy forecasting models or identify energy efficiency opportunities.

These are just a few examples of datasets that can be used to measure energy consumption. There are many other datasets available, both publicly and commercially.

In addition to the datasets listed, there are also a number of APIs that can be used to access energy consumption data. For example, the Google PowerMeter API allows developers to access real-time energy consumption data for homes and businesses.

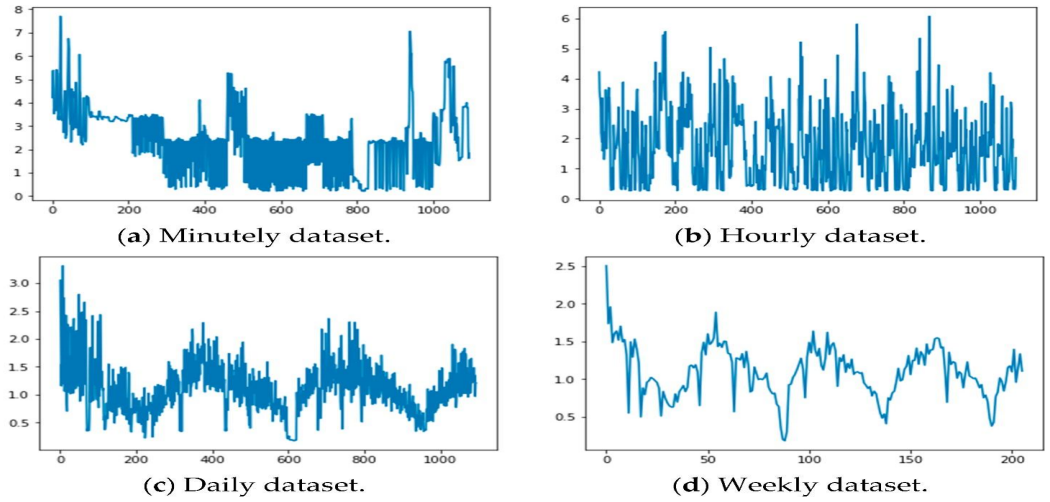
```
## Generating Plot 4
par(mfrow = c(2,2), mar = c(4,4,2,1), oma = c(0,0,2,0))
with(data, {
     plot(Global_active_power ~ Datetime, type = "l",
     ylab = "Global Active Power", xlab = "")
     plot(Voltage ~ Datetime, type = "l", ylab = "Voltage", xlab = "datetime")
     plot(Sub_metering_1 ~ Datetime, type = "l", ylab = "Energy sub metering",
          xlab = "")
     lines(Sub_metering_2 ~ Datetime, col = 'Red')
     lines(Sub_metering_3 ~ Datetime, col = 'Blue')
     legend("topright", col = c("black", "red", "blue"), lty = 1, lwd = 2,
            bty = "n",
            legend = c("Sub_metering_1", "Sub_metering_2", "Sub_metering_3"))
     plot(Global_reactive_power ~ Datetime, type = "l",
          ylab = "Global_rective_power", xlab = "datetime")
})
```

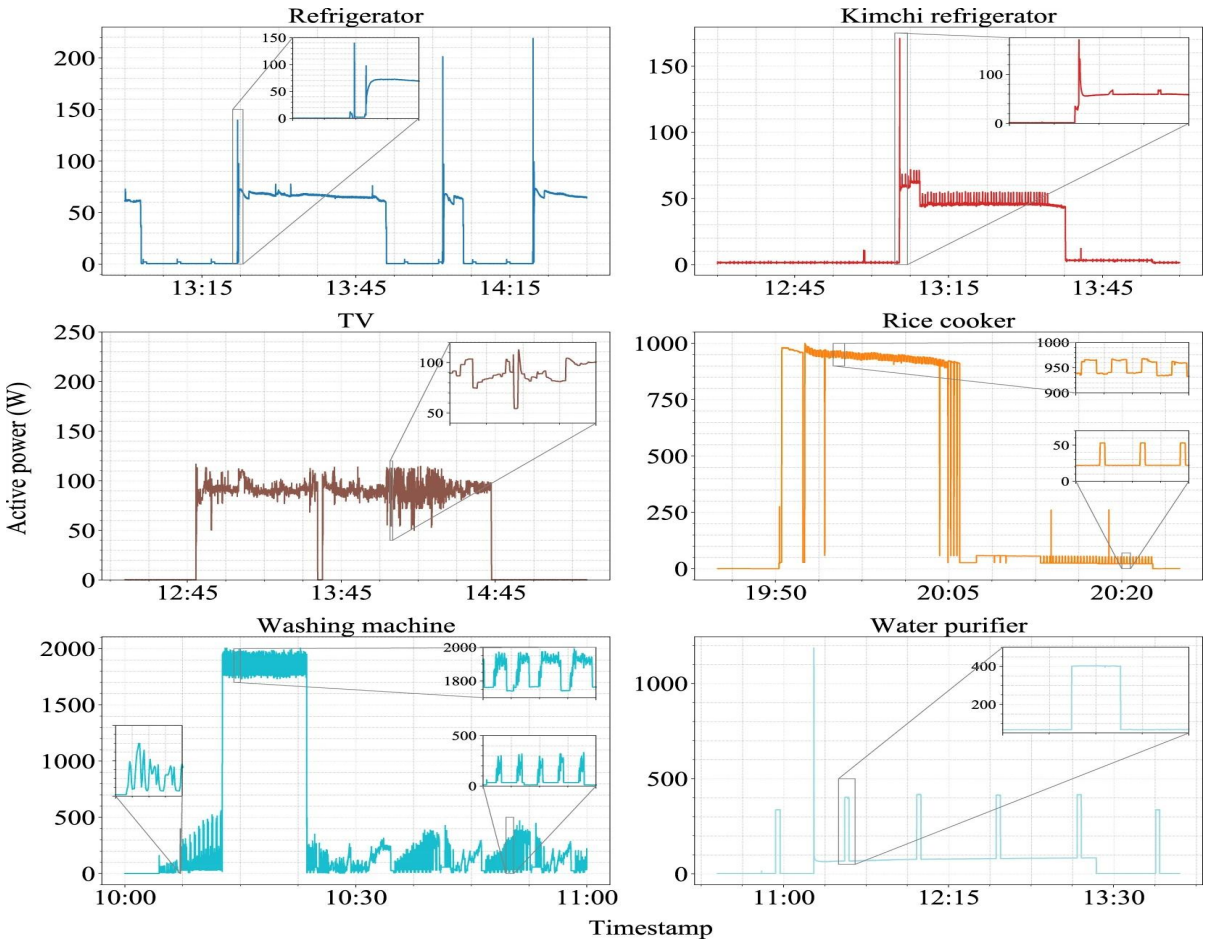


The dataset contains the following attributes:

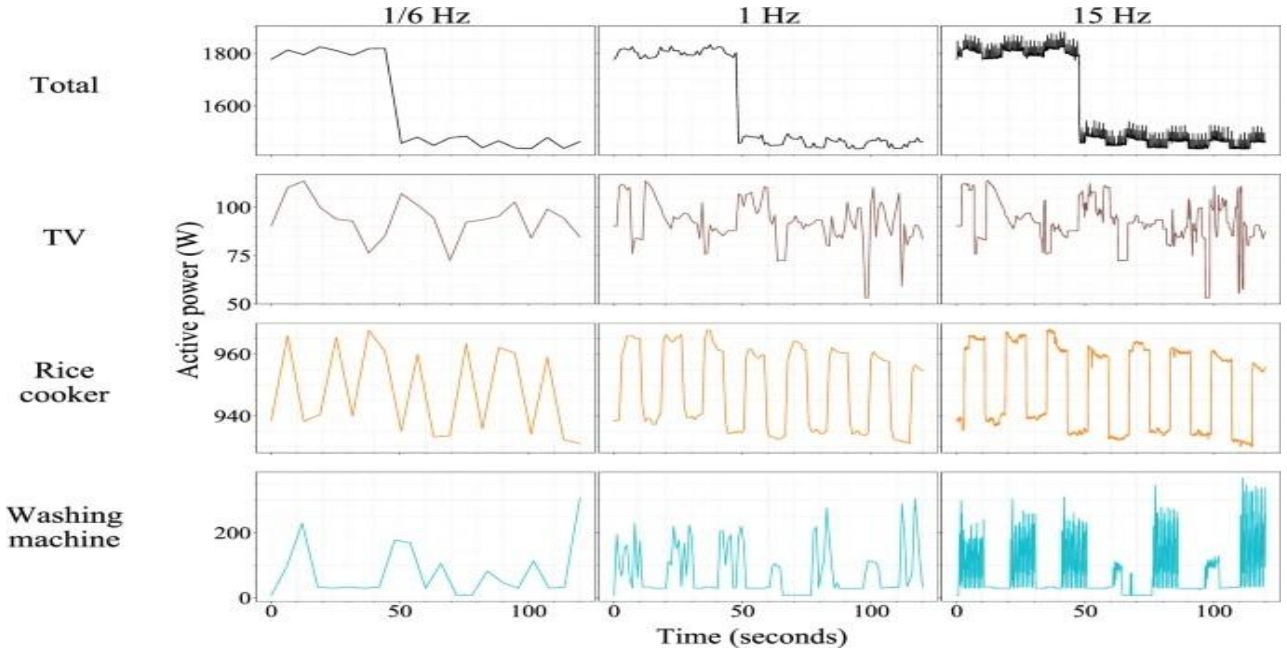
- ***Date****: The date of the measurement.*
- ***Time****: The time of the measurement.*
- ***Global Active Power****: The total active power consumption of the household in kilowatts (kW).*

- ***Global Reactive Power****: The total reactive power consumption of the household in kilovolt-amperes reactive (kVAR).*
- ***Voltage****: The voltage of the electrical supply in volts (V).*
- ***Global Intensity****: The current drawn by the household in amperes (A).*
- ***Sub-metering 1****: The active power consumption of the kitchen appliances in kilowatts (kW).*
- ***Sub-metering 2****: The active power consumption of the laundry room appliances in kilowatts (kW).*
- ***Sub-metering 3****: The active power consumption of the other appliances in the household in kilowatts (kW).*



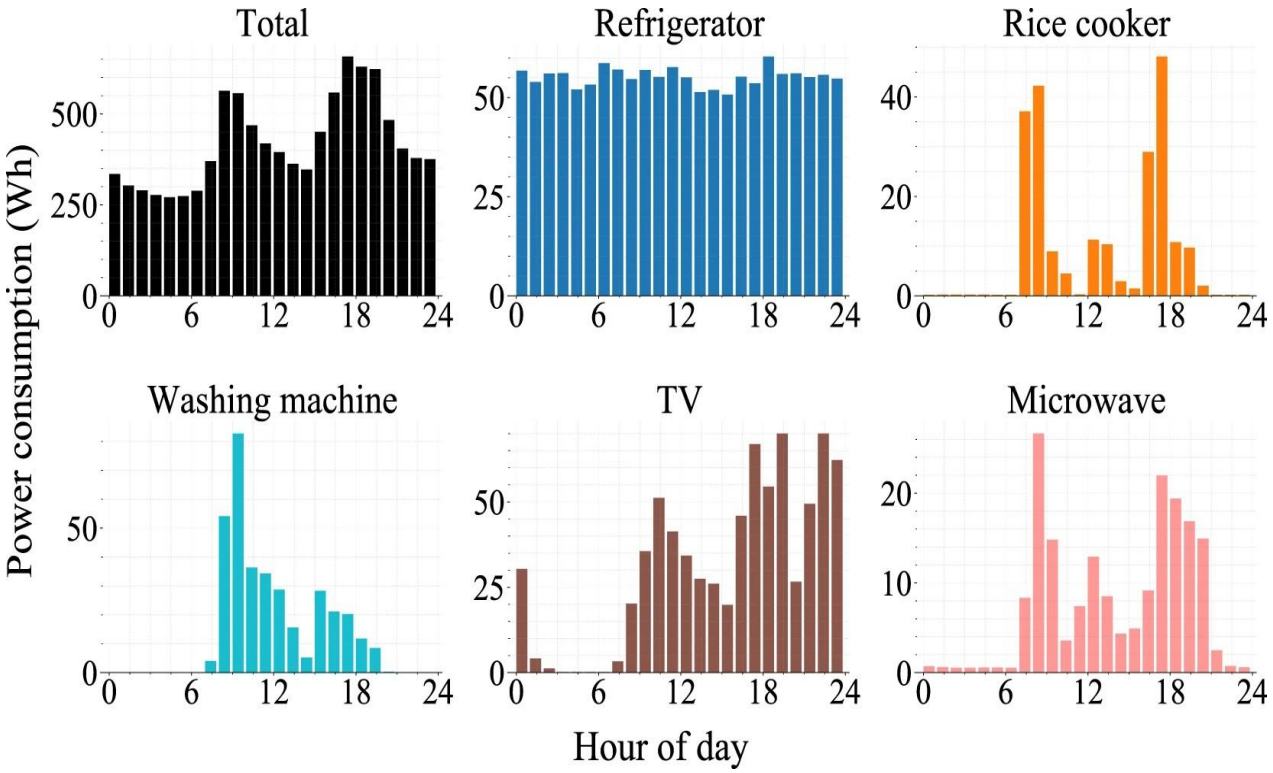


Data snippets from six different appliances



Power consumption measurements at sampling rates of 1/6, 1, and 15 Hz. The TV, rice cooker, and washing machine show distinct and visually distinguishable patterns at 15 Hz, but the patterns become less distinguishable at 1 Hz and become visually comparable at

1/6 Hz. Each plot shows 120 seconds of duration; 1/6 and 1 Hz data were generated by down-sampling (taking the first measurements of every 6 seconds and 1 second, respectively).



Hourly distribution of average power consumption

***CONCLUSION**:*

The Measure Energy Consumption Dataset is a valuable resource for researchers,policymakers, and energy consumers alike and practitioners working on energy forecasting, energy efficiency, and smart grid technologies. It can be used to develop and train machine learning models to predict energy consumption, identify patterns in energy usage, and develop energy-saving strategies.

It can be used to analyze energy consumption patterns, forecast future energy demand, develop energy efficiency strategies, and evaluate the impact of energy policies.

https://www.kaggle.com/code/ahmetyldrr/household-electric-power-predict-for-1000-data?scriptVersionId=131637822&cellId=2

LOADING AND PREPROCESSING:

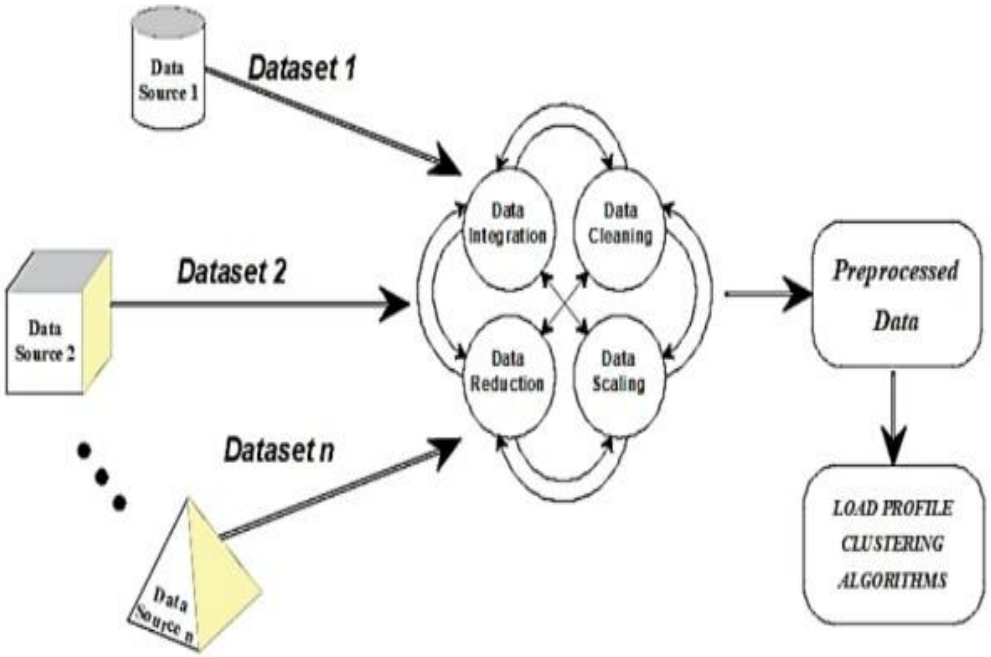


Figure 1 Data preprocessing steps and the place on load profile clustering work flow

1. ***Data Collection:***
    - *Obtain energy consumption data from sources such as smart meters, sensors, or historical records. Ensure you have detailed records, including date and time.*

2. ***Data Cleaning:***
    - *Remove any duplicate entries, missing values, and outliers from the dataset to ensure data quality.*

3. ***Data Transformation:***
    - *Convert data into a consistent time series format, typically with timestamp and energy consumption values.*
    - *Aggregate data if necessary (e.g., hourly, daily) for analysis.*

4. ***Feature Engineering:***
    - *Create additional features that can impact energy consumption, such as temperature, humidity, occupancy, or day of the week.*

5. ***Data Normalization:***
    - *Normalize the data to a common scale, especially if you have multiple features with different units.*

6. ***Data Splitting:***

- *Divide the dataset into training, validation, and test sets for model evaluation. Ensure that the time series order is preserved.*

7. ***Data Visualization:***
   - *Visualize the data to identify patterns, trends, and correlations.*

8. ***Model Selection:***
   - *Choose an appropriate model for predicting energy consumption. This can include regression models, time-series models, or machine learning algorithms.*

9. ***Model Training:***
   - *Train the selected model using the training dataset.*

10. ***Validation:***
    - *Validate the model's performance on the validation dataset and fine-tune hyperparameters as needed.*

11. ***Testing:***
    - *Test the model on the test dataset to evaluate its generalization performance.*

12. ***Evaluation:***
    - *Measure the model's performance using relevant metrics (e.g., Mean Absolute Error, Root Mean Square Error, R-squared).*

13. ***Deployment:***
    - *Deploy the model for making predictions on new energy consumption data.*

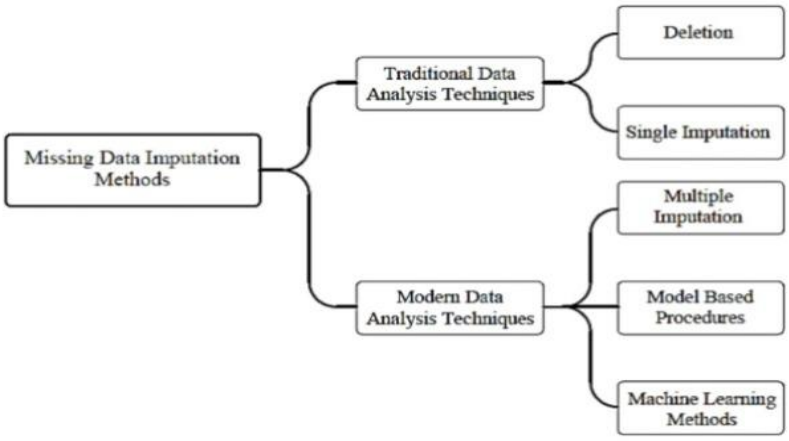
14. ***Monitoring:***
    - *Continuously monitor the model's performance and retrain it as needed to adapt to changing consumption patterns.*

LOADING THE DATASET

```
import pandas as pd

# Load the dataset from a CSV file
dataset = pd.read_csv('energy_consumption.csv')

# Print the dataset head
print(dataset.head())
```

PREPROCESSING THE DATASET

- ***Handling missing values:***



Classification of missing data imputation approaches

```
# Check for missing values
print(dataset.isnull().sum())

# Drop rows with missing values
dataset.dropna(inplace=True)
```

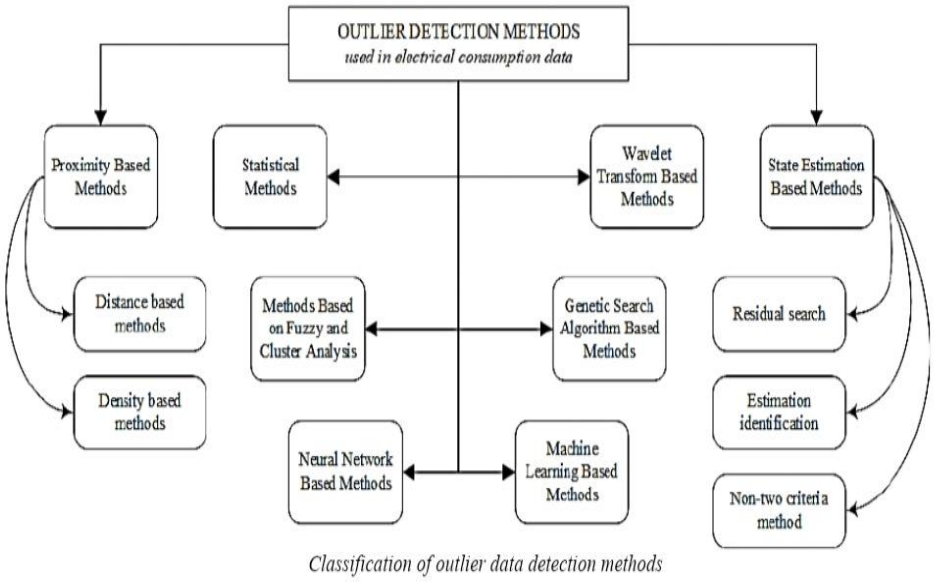
- ***Converting data types***

```
# Convert the 'date' column to datetime format
dataset['date'] = pd.to_datetime(dataset['date'])

# Convert the 'energy_consumption' column to float
dataset['energy_consumption'] = pd.to_numeric(dataset['energy_consumption'])
```

- ***Creating new features***

```
# Create a 'month' column from the 'date' column
dataset['month'] = dataset['date'].dt.month

# Create a 'year' column from the 'date' column
dataset['year'] = dataset['date'].dt.year
```

- ***Removing outliers***



Classification of outlier data detection methods

```
# Identify outliers using the boxplot method
import matplotlib.pyplot as plt

plt.boxplot(dataset['energy_consumption'])
plt.show()

# Remove outliers
dataset = dataset[~((dataset['energy_consumption'] < Q1 - 1.5 * IQR) |
(dataset['energy_consumption'] > Q3 + 1.5 * IQR))]
```

- ***Scaling the data***

```
from sklearn.preprocessing import StandardScaler

# Create a StandardScaler object.
scaler = StandardScaler()

# Scale the 'energy_consumption' column.
dataset['energy_consumption_scaled'] =
scaler.fit_transform(dataset[['energy_consumption']])
```

- ***Saving the preprocessed dataset***

```
dataset.to_csv('preprocessed_energy_consumption.csv', index=False)
```

The following will load the dataset, check for missing values, drop rows with missing values, convert datetime to datetime format, set datetime as index, resample to hourly frequency, create new features, and save the preprocessed dataset.

```
import pandas as pd
import numpy as np

# Load the dataset
df = pd.read_csv('energy_consumption.csv')

# Check for missing values
df.isnull().sum()

# Drop rows with missing values
df.dropna(inplace=True)

# Convert datetime to datetime format
df['datetime'] = pd.to_datetime(df['datetime'])

# Set datetime as index
df.set_index('datetime', inplace=True)

# Resample to hourly frequency
df = df.resample('H').mean()

# Create new features
df['energy_consumption_diff'] = df['energy_consumption'].diff()
df['energy_consumption_pct_change'] = df['energy_consumption_diff'] /
df['energy_consumption'].shift(1) * 100

# Save preprocessed dataset
df.to_csv('energy_consumption_preprocessed.csv')
```

The new features created are:

- ***energy_consumption_diff:****The difference in energy consumption between the current hour and the previous hour.*
- ***energy_consumption_pct_change:****The percentage change in energy consumption between the current hour and the previous hour.*
- ***energy_consumption_lag_1:*** *The energy consumption value from 1 hour ago.*
- ***energy_consumption_lag_2:****The energy consumption value from 2 hours ago.*
- *…*

- ***energy_consumption_lag_24:****The energy consumption value from 24 hours ago.*
- ***energy_consumption_rolling_mean_1:****The rolling mean of energy consumption over the past 1 hour.*
- *…*
- ***energy_consumption_rolling_mean_24:****The rolling mean of energy consumption over the past 24 hours.*
- ***energy_consumption_rolling_std_1:****The rolling standard deviation of energy consumption over the past 1 hour.*
- ***energy_consumption_rolling_std_2:****The rolling standard deviation of energy consumption over the past 2 hours.*
- *…*
- ***energy_consumption_rolling_std_24:****The rolling standard deviation of energy consumption over the past 24 hours.*

These features can be used to analyze energy consumption patterns and identify anomalies.

For example, you could use the energy_consumption_diff feature to identify periods of high energy consumption. You could use the energy_consumption_pct_change feature to identify periods of rapid change in energy consumption.

By analyzing these features, you can gain insights into how energy is being consumed and identify areas where energy consumption can be reduced.

TABLO 1 THE RAW DATA USED IN THE STUDIES AND THE DATA OBTAINED AFTER THE PREPROCESSING STEPS

	Source Data					Load Profile Clustering				
	No of Load	Type of Load	Voltage Level	Temporal Resolution	Period	No of Load	Considered Period	Temporal Resolution	Period	Analyzed Attributes
[4]	?	University Campus Buildings	LV	hourly	2 years	?	1 year	hourly	Daily	P
[32]	165	Mix	LV	15 min	6 months	165	6 months	15 m	Daily	P, VL,CD
[33]	245	HV-LV Substations	LV	hourly	4+ years	245	4+ years	hourly	Daily	P
[34]	234	Non-residential	MV	15 min	-	234	-	15 min	Daily	P
[35]	229	-	MV	15 min	6 months	208	6 months	15 min	Daily	P, CD,
[36]	-	Mix	MV	30 min	1 month	155	1 month	30 min	Daily	P
[37]	18098	Residential and Commercial	LV	hourly	396 days	1824	366 days	hourly	Daily	P, W, DL
[38]	1100	Residential	LV	10 min	8 months	952	8 months	10 min	Daily	P,W, SEC
[39]	218090	Residential	LV	1 hour	3.5 year	123150	1 year	hourly	Daily	P, Climate data
[40]	103	Residential	LV	1 min	1 years	103	Seasonal	hourly	Daily	P
[41]	1022	Mix	MV	15 min	1 year	1022	1 year	15 min	Daily	P
[42]	>197	Residential	LV	7-8 s	1 year	197	1 year	0.5 – 240 min	Daily	P
[43]	824	Substation	HV/LV	10 min	1 year	730	1 year	10 min	Daily	P, SI, CD
[44]	1072	Residential	LV	1 min	18 months	1072	18 months	10 min	Daily	P
[45]	4232	Residential	LV	30 min	18 months	3440	Work days in 5 years	hourly	Daily	P, SEC
[46]	1200	Residential	LV	1 day	1 month	938	1 month	1 day	Monthly	P
[47]	1	City		15 min	5 years	1	5 years	daily	Seasonal	P
[48]	100	HVAC units	LV	5 min	1 day	89	1 day	15 min	Daily	P
[49]	203	Feeders	HV	hourly	1 year	183	1 year	hourly	Daily	P
[50]	10	Research Institute Buildings	LV	various	3 years	10	1 year	hourly	Daily	P, W
[51]	114	Residential	LV	15 min	1 year	114	1 year	hourly	Daily	P,W
[52]	3000	Residential	LV	15 min	1 year	171	4 months	15 min	Daily	P
[53]	370	-	-	15 min	4 years	317	1 year	15 min	Daily	P
[54]	10 1000	Transformer Transformer	-	1 hour 1 hour	33 months 33 months	10 1000	33 months 33 months	hourly hourly	Daily Daily	P, W, C P

P	Active power consumption data	SI	System Information (Component's capacity, number of feeders, etc.)
W	Weather data	SEC	Socio-economic data (obtained via surveys)
C	Calendar data	VL	Voltage level
CD	Commercial Data (contracted power, activity code etc.)	DL	Day Length

TABLO 2 DATA PREPROCESSING STEPS USED IN STUDIES

	Data Integration	Data Cleaning													Data Reduction							Data Scaling	
		Outlier Detection								Missing Data Treatment					FS & IS				FE & IG				
											Data Imputation												
		PBM	SM	Fuzzy M.	NN M.	Genetic SA	Mac. L. M	Wavelet M.	S. Est. M.	Deletion	SI	MI	MBP	MLM	Filter	Wrapper	Hybrid	Discretization	PM	Transform.	No of New .F.	Linear Normalization	Z-Score
[4]					✓					✓	✓				✓					✓			
[32]			✓								✓				✓			✓				✓	
[33]			✓							✓					✓							✓	
[34]																					✓	✓	
[35]					✓					✓			✓							✓		✓	
[36]	✓		✓							✓													✓
[37]			✓							✓								✓		✓			✓
[38]			✓							✓		✓			✓								
[39]										✓								✓				✓	
[40]																					✓	✓	
[41]											✓											✓	
[42]										✓					✓							✓	
[43]	✓		✓																			✓	
[44]										✓													
[45]	✓									✓										✓		✓	
[46]					✓					✓													
[47]	✓		✓								✓									✓		✓	
[48]										✓	✓				✓			✓				✓	
[49]			✓							✓												✓	
[50]	✓										✓							✓		✓			
[51]	✓				✓					✓	✓				✓					✓			
[52]										✓	✓				✓						✓	✓	
[53]			✓							✓	✓							✓	✓	✓			✓
[54]	✓									✓	✓												

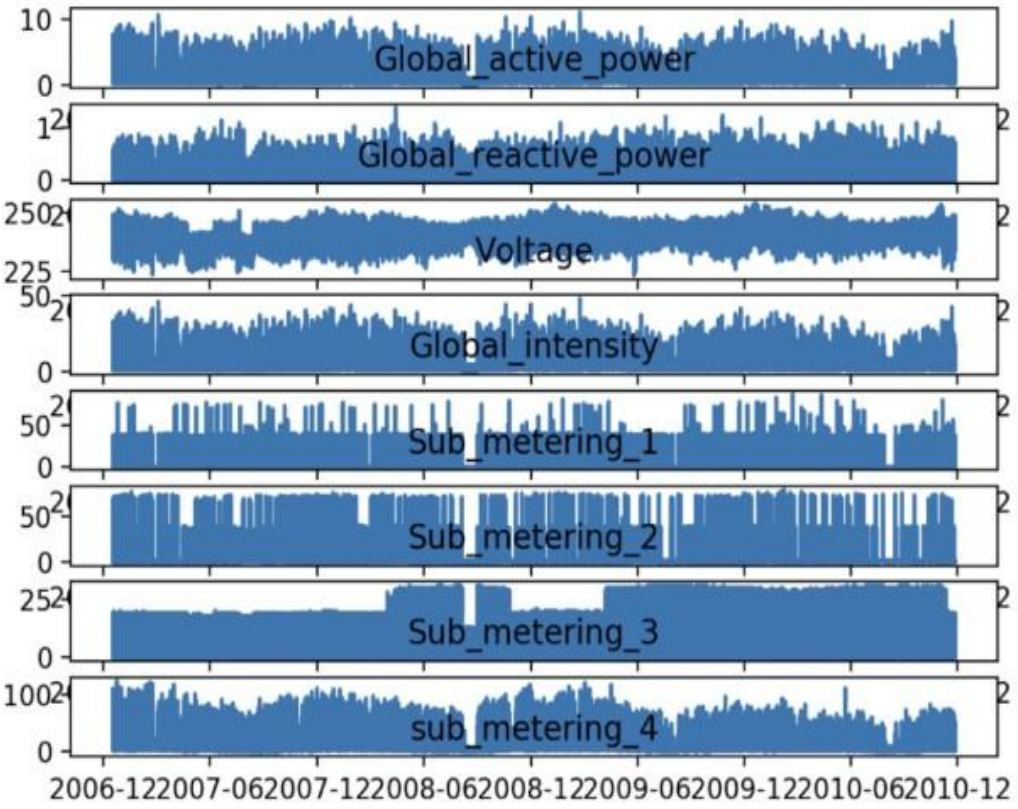
TABLO 3 METHODS USED IN THE DATA PREPROCESSING IN THE STUDIES

	Data Integration	Data Cleaning	Data Reduction	Data Transform
Listwise Deletion		[42],[33]		
Linear Regression		[33],[32],[37]		
Piecewise Aggregate Approximation			[42],[53]	
Linear normalization				[33],[32],[34] [35],[39],[41] [42],[43],[45] [47],[48],[49] [52],[54]
Principle Component Analysis			[33],[37],[34]	
Linear Interpolation		[4],[50]		
Cross-correlation Analysis			[50]	
Similar Day Approach		[50]		
Conditional permutation importance score			[50]	
Logistic Regression		[32]		
Piecewise Aggregate Approximation			[48],[53]	
Averaging		[48]		
Sense Checking Validity		[49],[43]		
Z-score				[37],[36],[53]
Nearest Neighbor interpolation		[38]		
Expectation Maximization		[38]		
Single Exponential Smoothing Technique		[52]		
Seasonal Auto Regressive Moving Average		[52]		
Savitzky–Golay Digital Filter		[52]		
Self Organizing Map		[52]		
Peak-Valley Attribute Analysis			[52]	
Sammon Map			[34]	
Curvilinear Component Analysis			[34]	
Symbolic Aggregation Approximation			[53]	
Extract, Transform and Load	[51]			
Recursive Feature Elimination			[51]	
Lasso Regularization			[51]	
Multilayer Perceptron Artificial Neural Network		[35]		
Forward Filling Method		[54]		
Sequential Backward Search			[54]	
Anderson-Darling test			[38]	
Durbin – Watson test			[38]	

Patterns in Observations Over Time

```
# line plots
from pandas import read_csv
from matplotlib import pyplot
# load the new file
dataset = read_csv('household_power_consumption.csv', header=0,
infer_datetime_format=True, parse_dates=['datetime'], index_col=['datetime'])
# line plot for each variable
pyplot.figure()
for i in range(len(dataset.columns)):
        pyplot.subplot(len(dataset.columns), 1, i+1)
        name = dataset.columns[i]
        pyplot.plot(dataset[name])
        pyplot.title(name, y=0)
pyplot.show()
```

Running the example creates a single image with eight subplots, one for each variable.



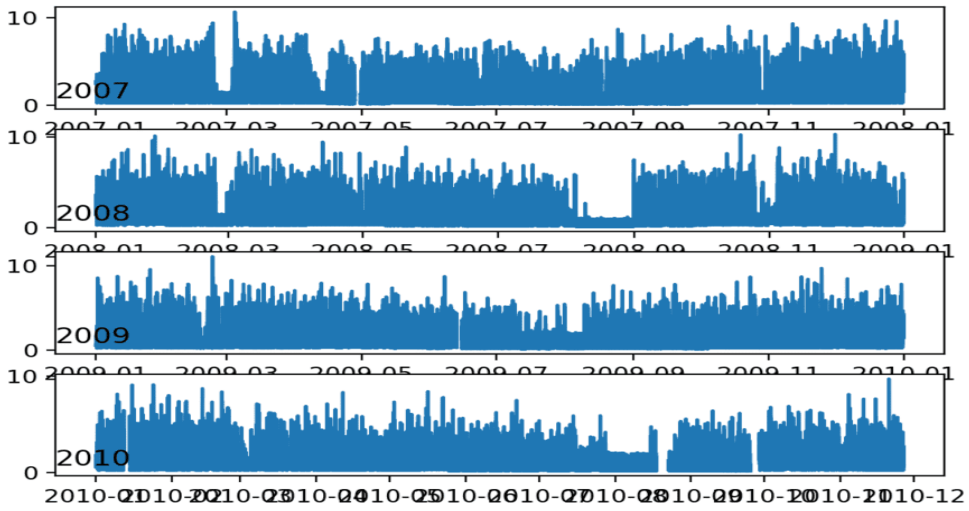
Let's zoom in and focus on the 'Global_active_power', or 'active power' for short.

We can create a new plot of the active power for each year to see if there are any common patterns across the years. The first year, 2006, has less than one month of data, so will remove it from the plot.

The complete example is listed below.

```
# yearly line plots
from pandas import read_csv
from matplotlib import pyplot
# load the new file
dataset = read_csv('household_power_consumption.csv', header=0,
infer_datetime_format=True, parse_dates=['datetime'], index_col=['datetime'])
# plot active power for each year
years = ['2007', '2008', '2009', '2010']
pyplot.figure()
for i in range(len(years)):
	# prepare subplot
	ax = pyplot.subplot(len(years), 1, i+1)
	# determine the year to plot
	year = years[i]
	# get all observations for the year
	result = dataset[str(year)]
	# plot the active power for the year
	pyplot.plot(result['Global_active_power'])
	# add a title to the subplot
	pyplot.title(str(year), y=0, loc='left')
pyplot.show()
```

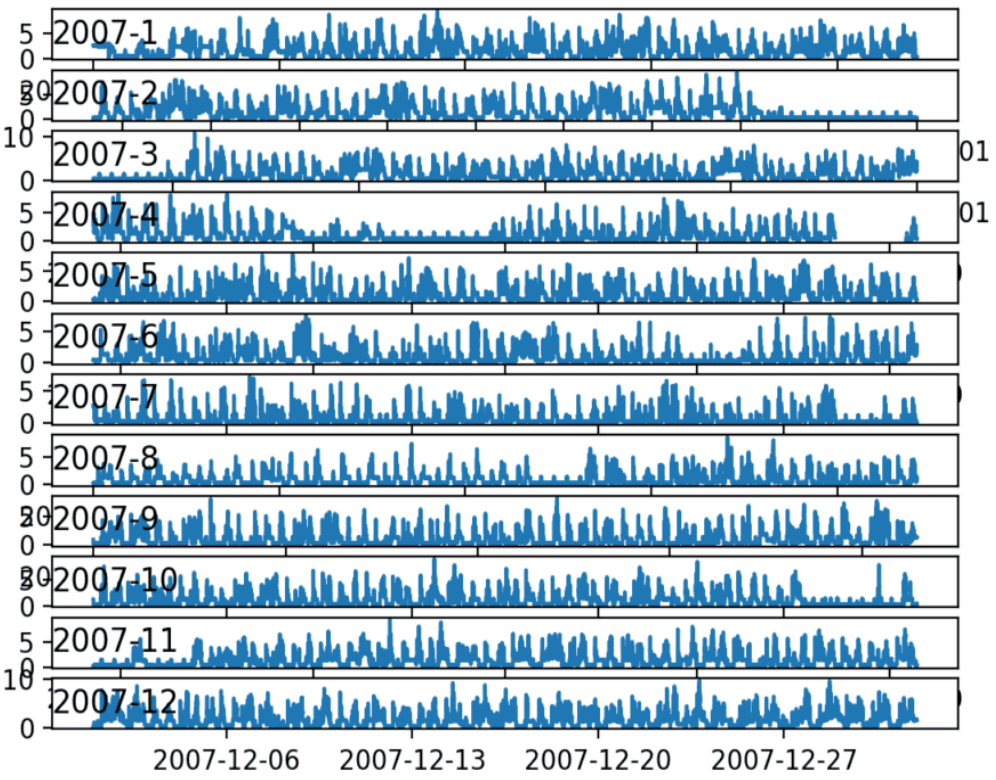
Running the example creates one single image with four line plots, one for each full year (or mostly full years) of data in the dataset.



We can continue to zoom in on consumption and look at active power for each of the 12 months of 2007.

```
# monthly line plots
from pandas import read_csv
from matplotlib import pyplot
# load the new file
dataset = read_csv('household_power_consumption.csv', header=0,
infer_datetime_format=True, parse_dates=['datetime'], index_col=['datetime'])
# plot active power for each year
months = [x for x in range(1, 13)]
pyplot.figure()
for i in range(len(months)):
	# prepare subplot
	ax = pyplot.subplot(len(months), 1, i+1)
	# determine the month to plot
	month = '2007-' + str(months[i])
	# get all observations for the month
	result = dataset[month]
	# plot the active power for the month
	pyplot.plot(result['Global_active_power'])
	# add a title to the subplot
	pyplot.title(month, y=0, loc='left')
pyplot.show()
```

Running the example creates a single image with 12 line plots, one for each month in 2007.



Finally, we can zoom in one more level and take a closer look at power consumption at the daily level.

We would expect there to be some pattern to consumption each day, and perhaps differences in days over a week.

The complete example is listed below.

```
# daily line plots
from pandas import read_csv
from matplotlib import pyplot
# load the new file
dataset = read_csv('household_power_consumption.csv', header=0,
infer_datetime_format=True, parse_dates=['datetime'], index_col=['datetime'])
# plot active power for each year
days = [x for x in range(1, 20)]
pyplot.figure()
for i in range(len(days)):
        # prepare subplot
        ax = pyplot.subplot(len(days), 1, i+1)
```

```
        # determine the day to plot
        day = '2007-01-' + str(days[i])
        # get all observations for the day
        result = dataset[day]
        # plot the active power for the day
        pyplot.plot(result['Global_active_power'])
        # add a title to the subplot
        pyplot.title(day, y=0, loc='left')
pyplot.show()
```

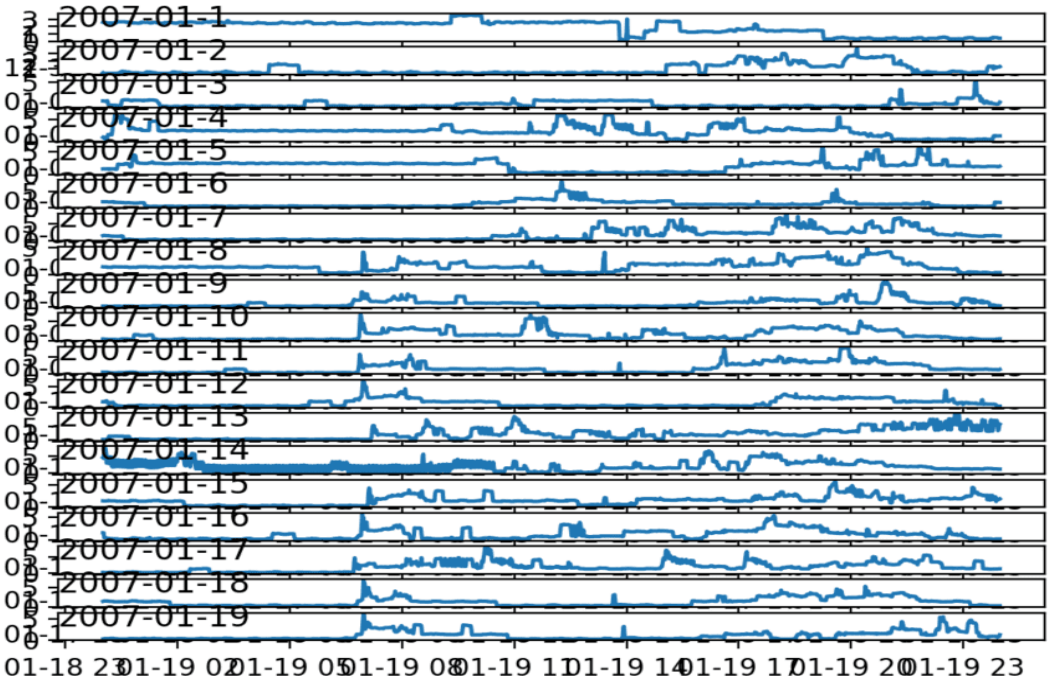
Running the example creates a single image with 20 line plots, one for the first 20 days in January 2007.

There is commonality across the days; for example, many days consumption starts early morning, around 6-7AM.

Some days show a drop in consumption in the middle of the day, which might make sense if most occupants are out of the house.

We do see some strong overnight consumption on some days, that in a northern hemisphere January may match up with a heating system being used.

Time of year, specifically the season and the weather that it brings, will be an important factor in modeling this data, as would be expected.



Time Series Data Distributions

Another important area to consider is the distribution of the variables.

For example, it may be interesting to know if the distributions of observations are Gaussian or some other distribution.

We can investigate the distributions of the data by reviewing histograms.

We can start-off by creating a histogram for each variable in the time series.

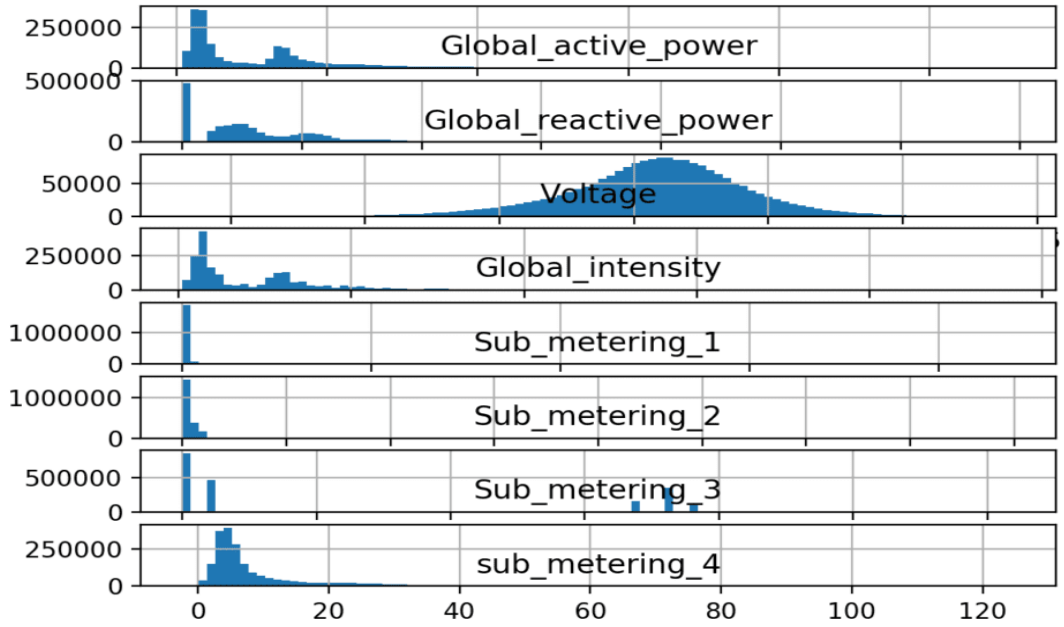
The complete example is listed below.

```
# histogram plots
from pandas import read_csv
from matplotlib import pyplot
# load the new file
dataset = read_csv('household_power_consumption.csv', header=0,
infer_datetime_format=True, parse_dates=['datetime'], index_col=['datetime'])
# histogram plot for each variable
pyplot.figure()
for i in range(len(dataset.columns)):
	pyplot.subplot(len(dataset.columns), 1, i+1)
	name = dataset.columns[i]
	dataset[name].hist(bins=100)
	pyplot.title(name, y=0)
pyplot.show()
```

Running the example creates a single figure with a separate histogram for each of the 8 variables.

We can see that active and reactive power, intensity, as well as the sub-metered power are all skewed distributions down towards small watt-hour or kilowatt values.

We can also see that distribution of voltage data is strongly Gaussian.



The distribution of active power appears to be bi-modal, meaning it looks like it has two mean groups of observations.

We can investigate this further by looking at the distribution of active power consumption for the four full years of data.

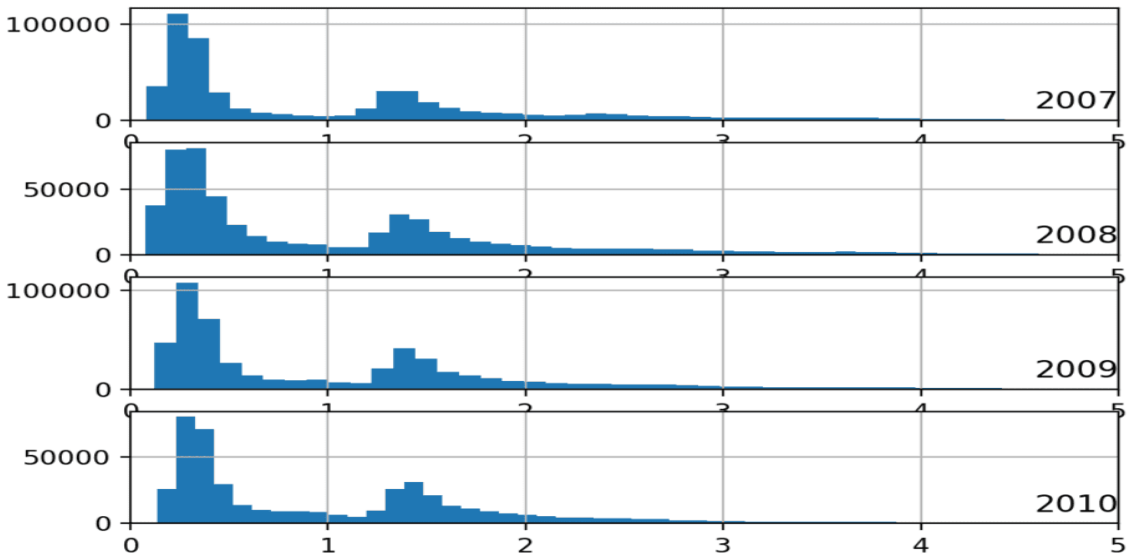
```
# yearly histogram plots
from pandas import read_csv
from matplotlib import pyplot
# load the new file
dataset = read_csv('household_power_consumption.csv', header=0,
infer_datetime_format=True, parse_dates=['datetime'], index_col=['datetime'])
# plot active power for each year
years = ['2007', '2008', '2009', '2010']
pyplot.figure()
for i in range(len(years)):
	# prepare subplot
	ax = pyplot.subplot(len(years), 1, i+1)
	# determine the year to plot
	year = years[i]
	# get all observations for the year
	result = dataset[str(year)]
	# plot the active power for the year
	result['Global_active_power'].hist(bins=100)
	# zoom in on the distribution
```

```
        ax.set_xlim(0, 5)
        # add a title to the subplot
        pyplot.title(str(year), y=0, loc='right')
pyplot.show()
```

Running the example creates a single plot with four figures, one for each of the years between 2007 to 2010.

We can see that the distribution of active power consumption across those years looks very similar. The distribution is indeed bimodal with one peak around 0.3 KW and perhaps another around 1.3 KW.

There is a long tail on the distribution to higher kilowatt values. It might open the door to notions of discretizing the data and separating it into peak 1, peak 2 or long tail. These groups or clusters for usage on a day or hour may be helpful in developing a predictive model.



It is possible that the identified groups may vary over the seasons of the year.

We can investigate this by looking at the distribution for active power for each month in a year.

```
# monthly histogram plots
from pandas import read_csv
from matplotlib import pyplot
# load the new file
dataset = read_csv('household_power_consumption.csv', header=0,
infer_datetime_format=True, parse_dates=['datetime'], index_col=['datetime'])
```

```
# plot active power for each year
months = [x for x in range(1, 13)]
pyplot.figure()
for i in range(len(months)):
	# prepare subplot
	ax = pyplot.subplot(len(months), 1, i+1)
	# determine the month to plot
	month = '2007-' + str(months[i])
	# get all observations for the month
	result = dataset[month]
	# plot the active power for the month
	result['Global_active_power'].hist(bins=100)
	# zoom in on the distribution
	ax.set_xlim(0, 5)
	# add a title to the subplot
	pyplot.title(month, y=0, loc='right')
pyplot.show()
```

Running the example creates an image with 12 plots, one for each month in 2007.

We can see generally the same data distribution each month. The axes for the plots appear to align (given the similar scales), and we can see that the peaks are shifted down in the warmer northern hemisphere months and shifted up for the colder months.

We can also see a thicker or more prominent tail toward larger kilowatt values for the cooler months of December through to March.