**Time-Series Forecasting of Household Energy Consumption and Price Using Deep Learning Models**

Harshith Uppula, Himanshee, Mounica Ayalasomayajula, and Richa Sharma

Department of Applied Data Science, San Jose State University

DATA 270: Data Analyst Process

Dr. Eduardo Chan

November 1st, 2022.

**Assignment-6 (Data Collection Plan)**

**Requirements:** Our project is toanalyze time series data and forecast electricity consumption and price on the basis of that data. Hence, our major data collection requirement is to get data that is in the form of time series. Time series data is a series of data which is indexed with time order. It is collected over equally spaced time intervals. We needed the data of electricity consumption which is measured with a time lapse. This data can be in the form of txt, csv, excel, or json. It is a combination of continuous (numeric) and categorical (descriptive) features. One data set is in text format and we converted it to csv using python. The other data set is in csv format and we merged both to create our final dataset.

**Sources:** We have collected data from two sources. Below are the 2 repositories with the respective datasets.

1. Provided by Dua, D. and Graff, C. (2019), dataset of a household’s electricity consumption has the historical data measured in kW(kilowatt) for a household from December 2006 to November 2010 with one-minute interval sampling.
2. Home energy consumption dataset by Ewood (2022) has information of a household’s energy consumption by appliances.

The dataset we got from the first source, UCI ML repository, is publicly available. This dataset contains the historical data of energy consumption in kW(kilowatt) by a household from December 2006 to November 2010 with one-minute interval sampling. Figure 1 shows originally the UCI individual household electric power consumption dataset was in txt format and columns were separated by semicolons, which we converted into a CSV file for merging it with our other datasets. For converting the txt dataset into CSV file we have used python. The UCI single household dataset has 9 columns. This data source has the license to make this dataset publicly available. Their citation policy is to acknowledge the source in the work we are doing based on this dataset.

**Figure 1**

*Individual Household Electric Power Consumption Dataset raw Example*

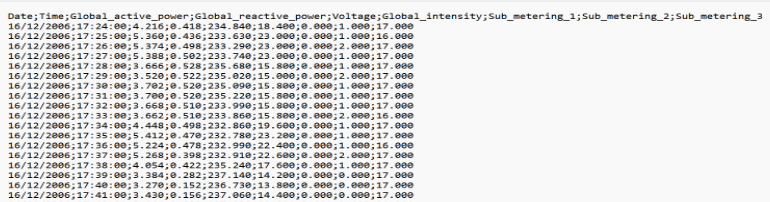
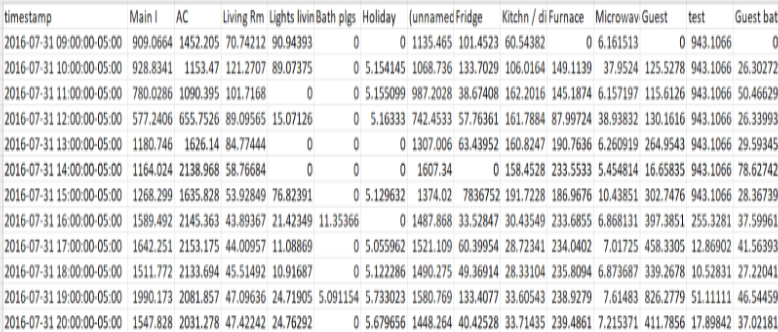


Figure 2 shows the second dataset from data.world is the home-energy-consumption dataset by ewood. It has information about a single home energy consumption in a home by appliances. And this dataset is in the same CSV format and it has 19 columns. It has electricity consumption recorded for each appliance.

**Figure 2**

*Raw Dataset Sample of Home-Energy-Consumption Dataset*



Both the sources have the required license to make these datasets publicly available. There are no restrictions on the usage of these datasets. We will be providing the sources names and links to both the data sources so that if anyone needs the data can access it from the original source and also to give due credit to the sources.

**Table 1**

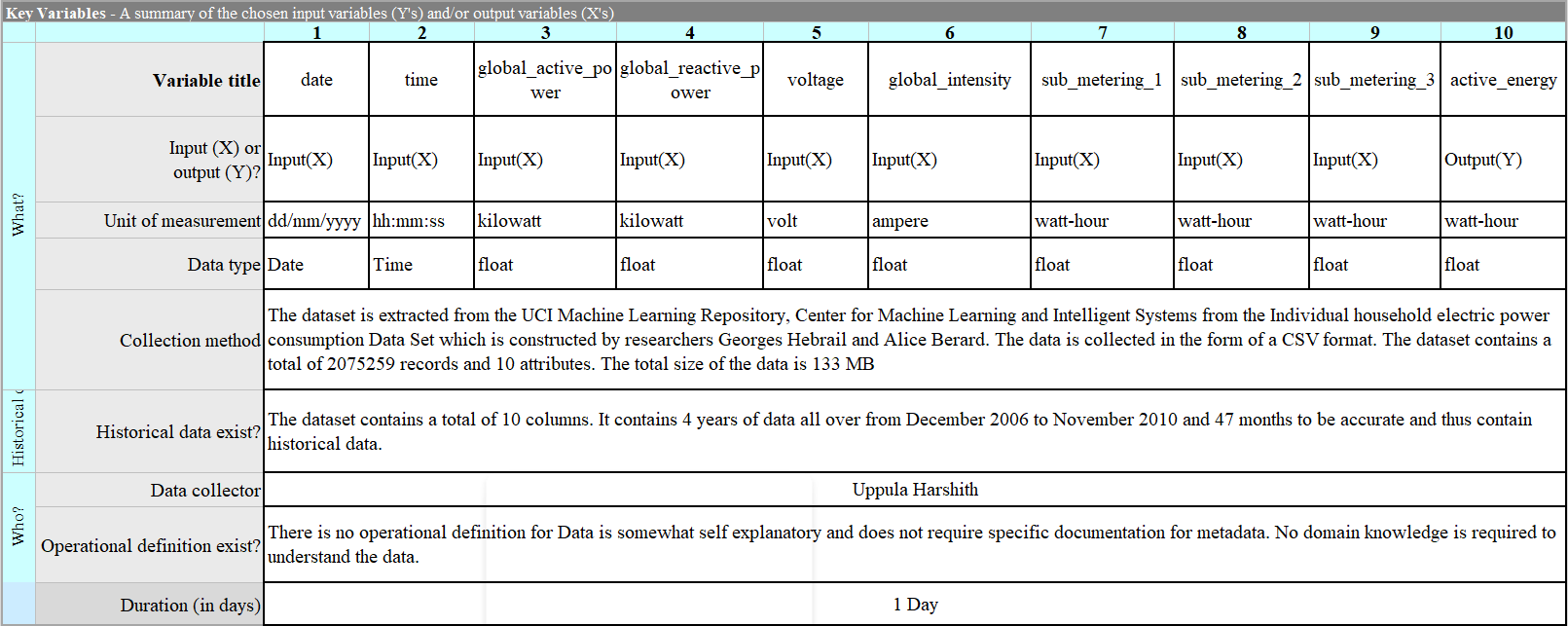
*Data Collection Plan for UCI Dataset.*

|  |  |
| --- | --- |
| *Description of Data* |  |
| Why are we collecting data? | Household energy consumption time-series data is required to tackle this business problem. This needs a household dataset which contains both historical and current data to predict the household electricity usage. |
| How will the data help? | The data extracted from the UCI Machine Learning Repository, Center for Machine Learning and Intelligent Systems from the Individual Household Electric Power Consumption Data Set is time-series and has 9 input columns including data and time. active\_energy is a target feature column which is helpful in predicting the future usage of household energy. |
| What should we do after collecting data? | In the given dataset, date and time are given as separate columns which will be difficult to make data preprocessing on the data. The two columns should be combined to form a single datetime column. |

Figure 3 shows our concise data collection plan for the UCI dataset which we have created using the CI toolkit where we have filled in the required details of features according to our dataset and have mentioned the collection method and whether historical data exists for our data and duration of data that we have collected and all the different type of variables present in our dataset.

**Figure 3**

*Data Collection Plan for UCI Dataset.*

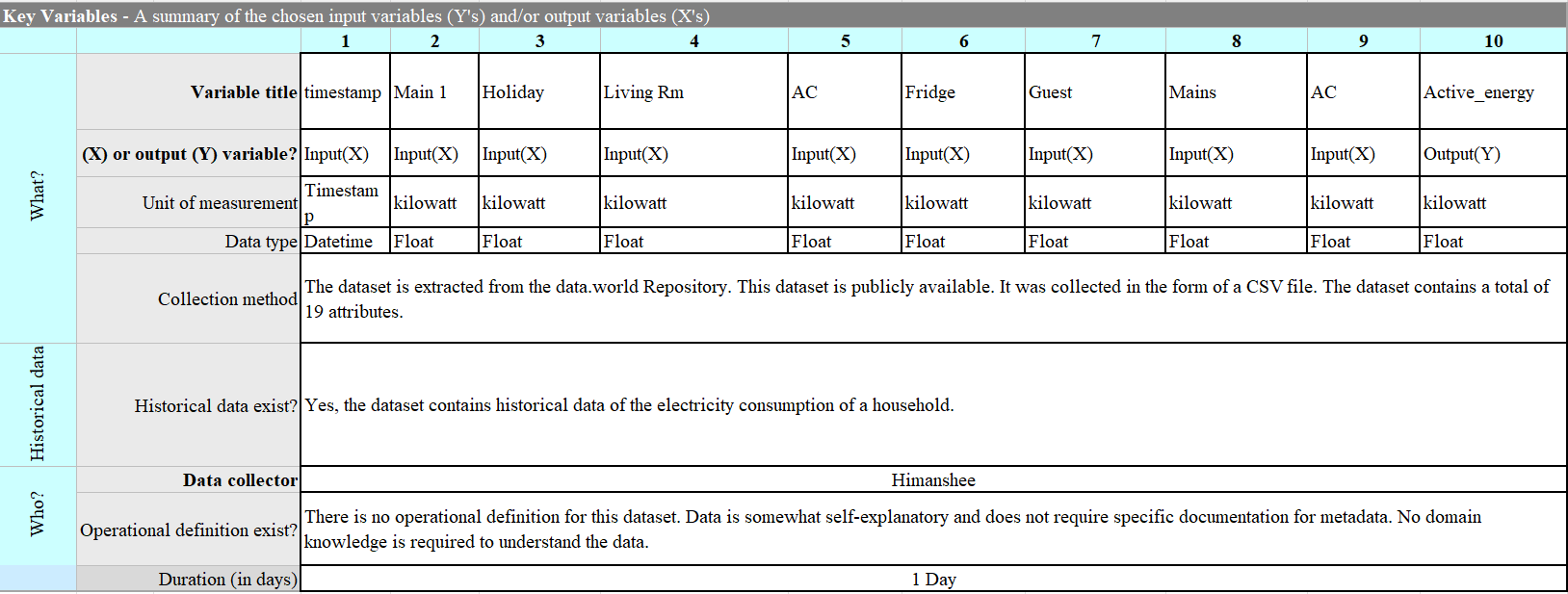
****Table 2**

*Data Collection Plan for data.world Dataset.*

|  |  |
| --- | --- |
| *Description of Data* |  |
| Why are we collecting data? | Since we needed time series data, this data matches our requirement and provides the data of electricity consumption of a household for each appliance. Hence, we collected this data. |
| How will the data help? | This data provides time spaced electricity consumption. Hence we can use it to merge with our available time series dataset. Increasing our dataset will help us create a better forecasting model. |
| What should we do after collecting data? | We are merging this dataset to our first dataset on the common column timestamp. The columns were of data type objects, we converted it to float. |

**Figure 4**

*Data Collection Plan for data.world Dataset.*



**Data Exploration Plan**

We will create our data quality report at the end of the EDA in the future which we will address the following issues as mentioned in our Table 3, and it is subjected to change as per our EDA requirements further on.

Table 3 shows our detailed EDA plan which we will be performing as part of our data exploration and data cleaning phases.

**Table 3**

*EDA Techniques Detailed Plan.*

|  |  |  |  |
| --- | --- | --- | --- |
| **Dataset** | **EDA Issues** | **Suggested Solutions** | **Tools Used** |
| **UCI** | Missing Values | Imputation based on the previous data. | Statistics using the NumPy function that takes the NumPy array of data and copy values from the previous 24 hours then we save the cleaned-up dataset to a separate file. |
| Renaming Columns | Rename the feature names for merging the datasets. | Using the function withColumnRenamed() of the PySpark. |
| Interpolation | Combining the date and time features into a new feature timestamp. | Using python library pandas having the interpolate() function. |
| Data Type Conversion | Conversion of datatype into the required data type formats. | Using PySpark libraries. |
| Correlation | Analyzing correlation between features using ACF (Autocorrelation Function) and PACF (Partial Autocorrelation Function). | Pair plot for identifying the relations between the features, to understand the importance of the features and pandas autocorrelated library. |
| Outliers | Analyzing the outliers of the dataset. | Box Plot for identifying the outliers to avoid noisy data. |
| Sampling/Resampling | Upsampling and Downsampling to understand additional information of the dataset. | Using the upsample and downsample functions. |
| Stationarity | Transformation for log or square root to stabilize non-constant variance and Differencing for  analyzing the difference in the current value and previous value. | Using Visualization, Statistics, ADF test we can analyze the stationarity of the dataset. |
| Augmented Dickey-Fuller (ADF) | Statistical test for null or alternate hypothesis. | Using the p value. |
| **API** | Missing Values | Imputation based on the previous data. | Statistics using the NumPy function that takes the NumPy array of data and copy values from the previous 24 hours then we save the cleaned-up dataset to a separate file. |
| Renaming Columns | Rename the feature names for merging the datasets. | Using the function withColumnRenamed() of the PySpark. |
| Interpolation | Combining the date and time features into a new feature timestamp. | Using python library pandas having the interpolate() function. |
| DataType Conversion | Conversion of datatype into the required data type formats. | Using PySpark libraries. |
|  | Correlation | Analyzing correlation between features. | Pair plot for identifying the relations between the features, to understand the importance of the features. |
| Outliers | Analyzing the outliers of the dataset. | Box Plot for identifying the outliers to avoid noisy data. |

***Roles and Responsibilities***

Table 4 displays the distribution of the roles and responsibilities among our team members.

**Table 4**

*Roles and Responsibilities.*

|  |  |  |
| --- | --- | --- |
| **Roles** | **Responsibilities** | **Member** |
| Missing Values | Imputation based on the previous data. | Mounica |
| Renaming Columns | Rename the feature names for merging the datasets. | Harshith |
| Interpolation | Combining the date and time features into new feature dates. | Richa |
| Data Type Conversion | Conversion of datatype into the required data type formats. | Himanshee |
| Correlation | Analyzing correlation between features using ACF (Autocorrelation Function) and PACF (Partial Autocorrelation Function). | Mounica |
| Outliers | Analyzing the outliers of the dataset. | Richa |
| Sampling/Resampling | Upsampling and Downsampling to understand additional information of the dataset. | Mounica |
| Stationarity | Transformation for log or square root to stabilize non-constant variance and Differencing for analyzing the difference in the current value and previous value. | Harshith/Himanshee |
| Augmented Dickey-Fuller (ADF) | Statistical test for null or alternate hypothesis. | Richa |

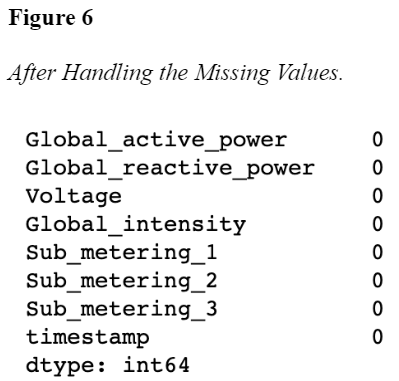
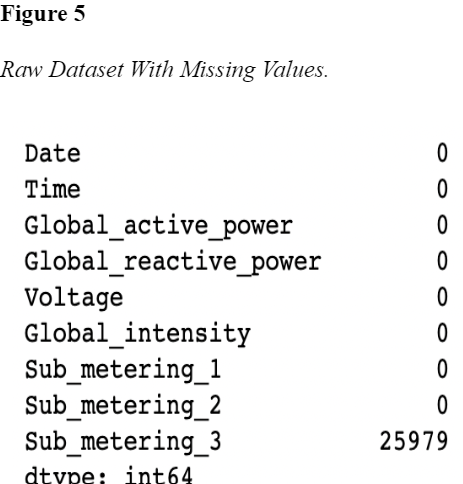
**Assignment-7 (Data Engineering)**

**Data Cleaning:**

**Incomplete & Missing data** - We have handledthe missing values by filling them with backward filling using fill\_missing() Numpy function.

Figure 5, shows how our dataset looks before handling missing values as the column ‘Sub\_metering\_3’ displays about 25k of missing data which needs to be handled in order to proceed further.

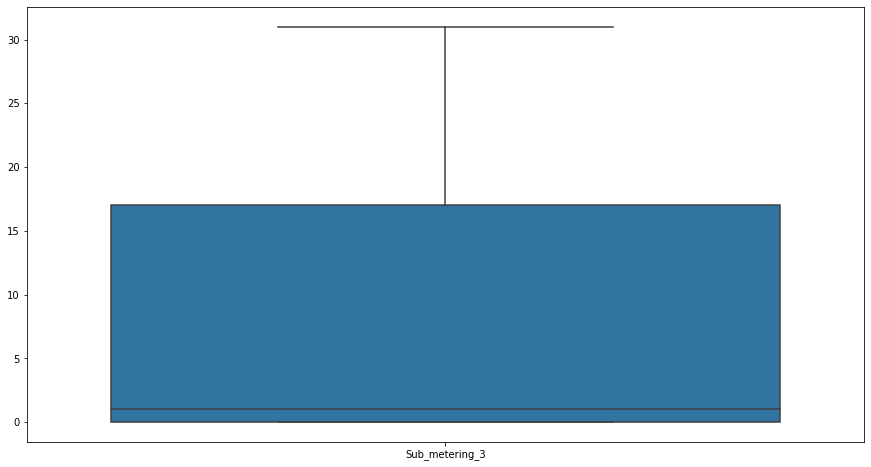
Figure 6 shows us how our data looks after handling the missing values by copying the values from the previous 24 hours of historical data. Now, after filling the missing values our data looks good for our further analysis and processing.



**Noisy Data -** We have plotted the boxplot for checking the mean and median values, analyzed the outliers, and using linear interpolation, backward filling we will be handling the replacement of data. Figure 7 shows the outliers of the column ‘Sub\_metering\_3’ using the box-plot. From this we can clearly observe that the outlier is quite different from the other data values present in the dataset.

**Figure 7**

*Analyzing the Outliers Using Box-Plot.*



**Inconsistent Data**- We have dealt with our inconsistent data using different techniques such as converting our dataset into required format, creating new features using feature engineering method and converting data type in the same required datatype for the better analysis etc. Figure 8 shows the conversion of the raw .txt file format.

**Figure 8**

*Raw Dataset in .txt Format.*

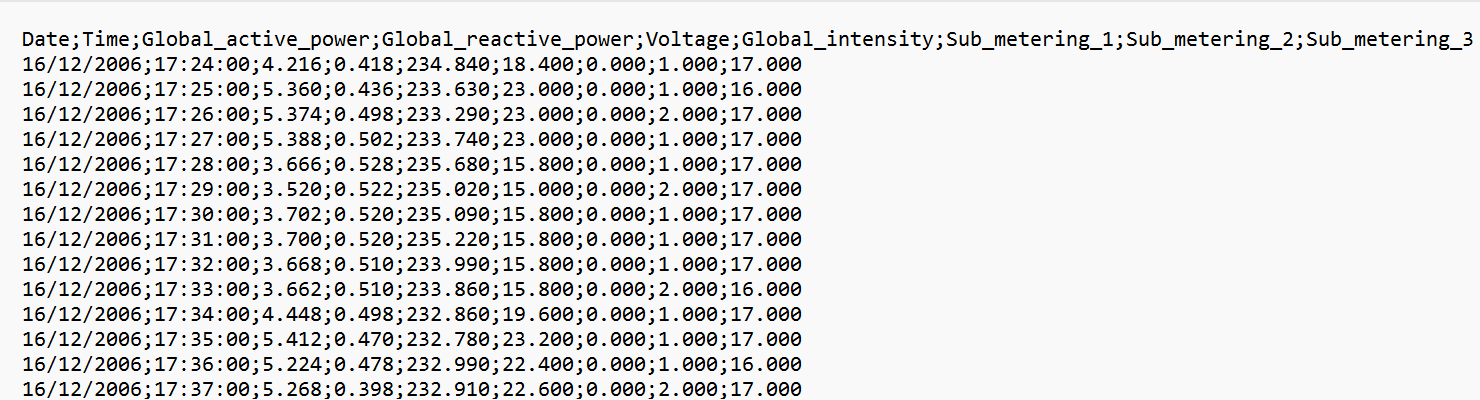


Figure 9 shows the converted .csv file format which will be used further on for our analysis.

**Figure 9**

*Converted Dataset into .csv Format.*

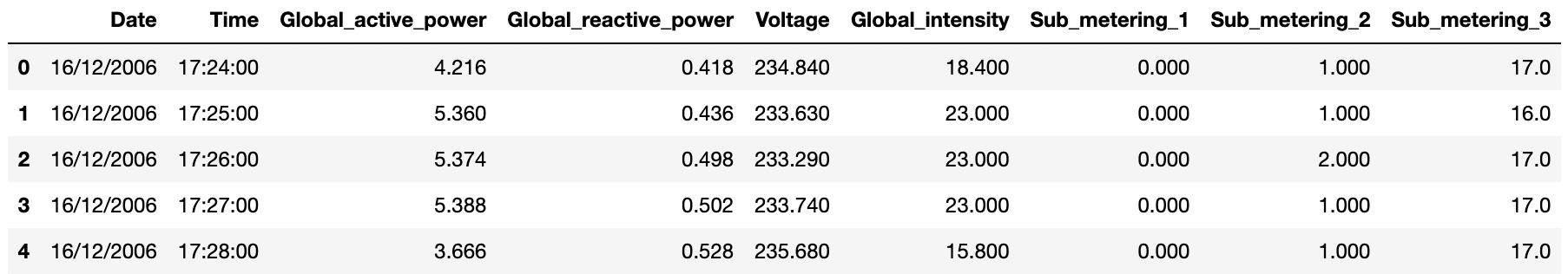
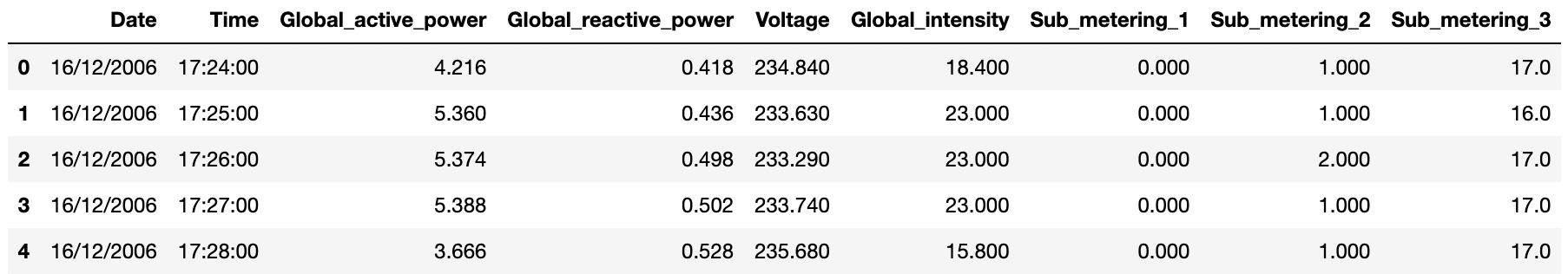


Figure 10 shows before combining the two columns ‘date’ and ‘time’ in our dataset into a single new column.

Figure 11 shows after combining the two columns ‘date’ and ‘time’ in our dataset into a single new column called ‘timestamp’.

**Figure 10**

*Before Combining the Columns Date and Time.*



**Figure 11**

*After Combining the Columns Date and Time Into one Column Timestamp.*

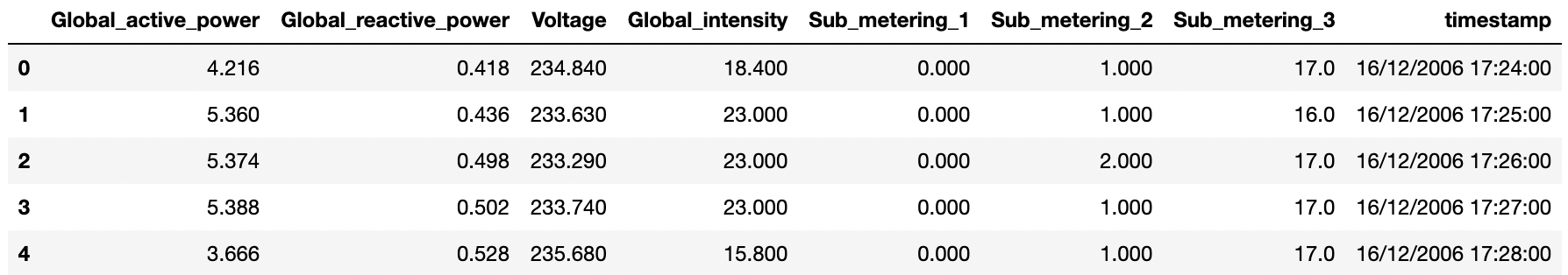
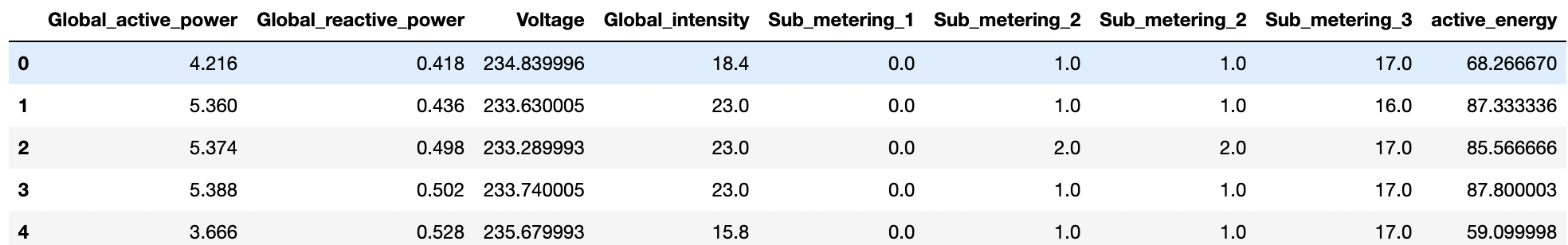


Figure 12 shows the additional created column ‘active\_energy’ which is a calculated field using feature engineering of the columns.

**Figure 12**

*Created a new Feature ‘active\_energy’.*



**Data Transformation:**

To prepare our data for forecasting, we have performed three data transformation techniques:

1) Data Standardization, 2. Data Reduction, 3. Data Regularization.

Below are the details of these transformations that we performed.

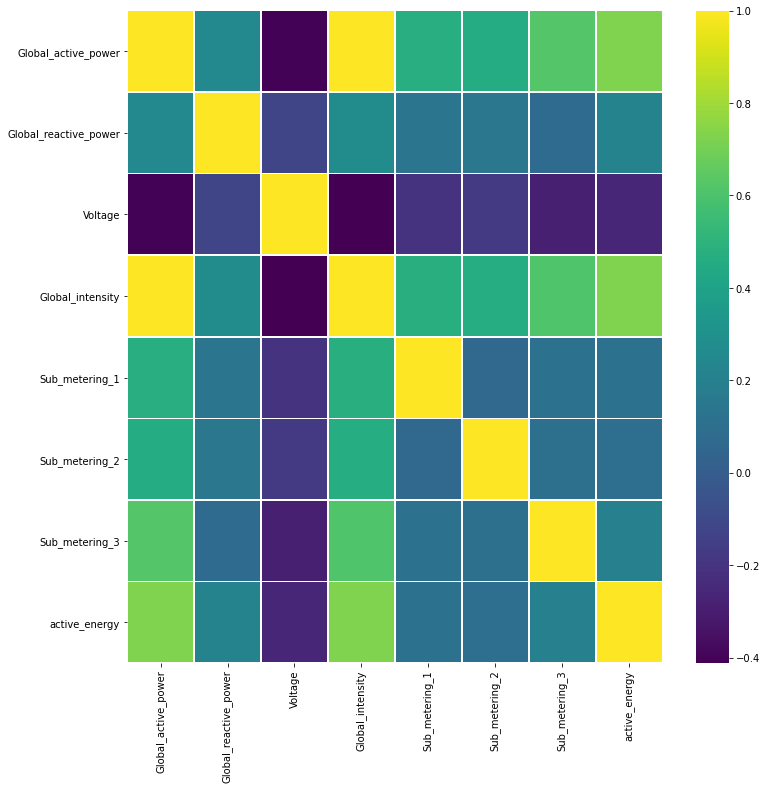
**Data Standardization**

The process of standardization rescales the attribute values in a manner that makes the mean value zero and the standard deviation 1. It brings down the column values on the same scale without losing the differences in the range of these values.

Figure 13 shows the heatmap of correlation between the features before scaling of the data. As the visualization of high dimensional data seems difficult, we have decided to use PCA for finding our principal components. But, before proceeding with PCA we need to standardize and scale our data so that each feature has a single unit variance.

**Figure 13**

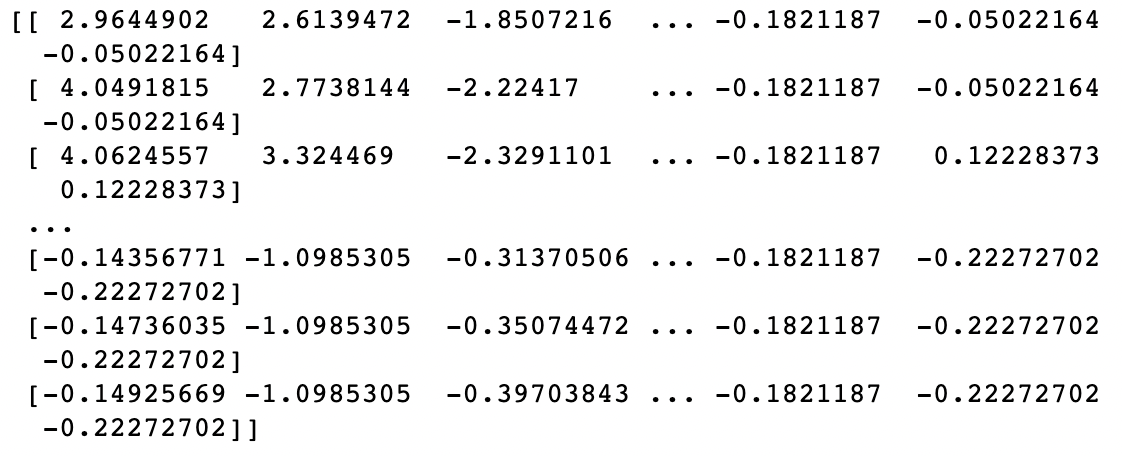
*Heatmap of Correlation Between the Features Before Scaling.*



We have performed data standardization in python using the scikit-learn object [StandardScaler](http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html). Figure 14 shows the scaled values of our data which has been handled using the StandardScaler function which standardizes all the features which will help in our PCA (Principal Component Analysis) and hereby reducing the high dimensional data for better visualization and processing.

**Figure 14**

*Standardization of the Data.*



**Data Reduction**

When we deal with a large dataset, modeling it and evaluation of results of such a large dataset becomes problematic. Hence we use data reduction techniques to create a reduced sample of the actual dataset which is smaller in volume but is the exact representation of the actual dataset. The data reduction techniques that we have used are:

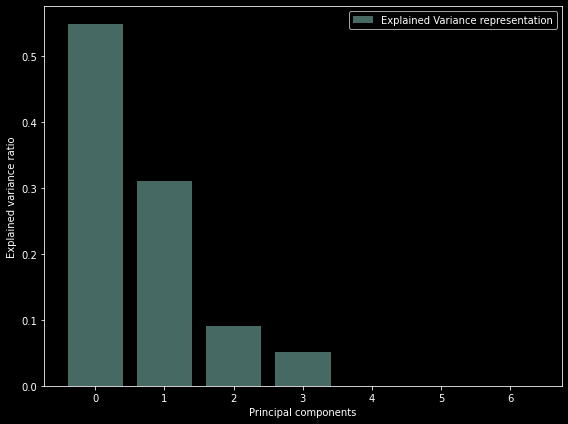
**PCA(Principal Component Analysis)**

As we have noticed that it is difficult to visualize high dimensional data, we have decided to use PCA for finding our principal components.

Figure 15 shows us the principal components to be considered from our scaled data. The graph clearly shows us the feature reduction of our dataset from seven columns to four principal component columns which are the most correlated features of our data. The first component has the largest explained variance ratio in the given dataset thus making it of utmost importance followed by the next highest principal component column and so on. The variance ratio diminishes in the rest of the columns eventually leading us to the most important columns in our data.

**Figure 15**

*PCA of the Data Reduction.*



**Data Regularization**

One of the major concerns of any Machine Learning model is overfitting. In overfitting, data tries to find patterns even in the noise in the data. Data regularization is a technique that can prevent overfitting by reducing the coefficient estimates towards zero. It balances the variance and bias in the model by significantly reducing the variance without increasing the bias.

We have evaluated two of the common regularization techniques which are L1 and L2. The lasso regression model(L1) and ridge regression model(L2) on our dataset using repeated 10-fold cross validation, calculated the average mean absolute error on the dataset. Figure 16 shows the lasso regression model MAE score which is around 56% and the standard deviation score around 0% whereas Figure 17 shows us the MAE score of the ridge regression model which is about 9% and the standard deviation score around 9%.

Hence, considering the MAE scores of the models, we can conclude that the ridge regression model(L2) suits better for our dataset.

**Figure 16**

*Lasso Regression Evaluation Lasso before reduction*



**Figure 17**

*Ridge Regression Model Evaluation. Ridge before reduction*

**

**References**

Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

*Home Energy Consumption - dataset by ewood*. (n.d.). Home Energy Consumption - Dataset by Ewood | data.world. Retrieved November 1, 2022, from <https://data.world/ewood/home-energy-consumption>