Power Consumption Analysis for House Holds

Team−30

Deepak Dalal
Yasha
Sukanthen SS
Anik pal

Contents

1. Introduction

1.1 Overview

As the popularity of home automation and the cost of electricity grow around the world, energy conservation has become a higher priority for many consumers. With a number of smart meter devices available for your home, you can now measure and record overall household power draw, and then with the output of a machine learning model, accurately predict individual appliance behaviour simply by analyzing meter data. For example, your electric utility provider might send you a message if it can reasonably assess that you left your refrigerator door open, or if the irrigation system suddenly came on at an odd time of day.

1.2 Purpose

The purpose of this project is to utilise the data collected by the smart sensors over the years , in our machine learning model to predict the global active power of the a household given the values of input variables including voltage, global intensity, active energy for kitchen(sub_metering_1),active energy for laundry(sub_metering_2), active energy for climate control systems(sub_metering_3) and the total reactive power consumed by the household(global_reactive_power). The prediction can help in adjusting and lowering the usage to reduce the power consumption and in effect the electricity bill of the household.

2. Literature survey

2.1 Existing Problem

India is the world's third largest producer and third largest consumer of electricity. The gross electricity consumption in 2018-19 was 1,181 kWh per capita. In 2015-16, electric energy consumption in agriculture was recorded as being the highest (17.89%) worldwide. Energy use can be viewed as a function of total GDP, structure of the economy and technology. The increase in household energy consumption is more significant than that in the industrial sector. To achieve reduction in electricity consumption, it is vital to have current information about household electricity use. Many households worry a lot due to the high electricity bill. Even they want to save the consumption of power. Due to the lack of information about individual appliances power consumption leads to huge wastage of power as well as money.

2.2 Proposed Solution

We are proposing a system that analyses electricity consumption of individual appliances, identify the energy consumption patterns apply statistical modelling and suggest the user about reduction of power consumption. The main objective of this project is to analyse and visualize the energy consumption. The regular consumption is captured through sensor and stored in database. Apply various exploratory data analysis using various libraries, visualize them using matplotlib and seaborn and suggest the user based on consumption. Web application to visualize the Analysis. Get insights from the data. suggests the user based on insights.

3. Theoretical Analysis

3.1 Modeling Methods

There are perhaps four classes of methods that might be interesting to explore on this problem; they are:

- Naive Methods.
- Classical Linear Methods.
- Machine Learning Methods.
- Deep Learning Methods.

Naive Methods

Naive methods would include methods that make very simple, but often very effective assumptions.

Some examples include:

- Tomorrow will be the same as today.
- Tomorrow will be the same as this day last year.
- Tomorrow will be an average of the last few days.

Classical Linear Methods

Classical linear methods include techniques are very effective for univariate time series forecasting.

Two important examples include:

- SARIMA
- ETS (triple exponential smoothing)

They would require that the additional variables be discarded and the parameters of the model be configured or tuned to the specific framing of the dataset. Concerns related to adjusting the data for daily and seasonal structures can also be supported directly.

Machine Learning Methods

Machine learning methods require that the problem be framed as a supervised learning problem.

This would require that lag observations for a series be framed as input features, discarding the temporal relationship in the data.

A suite of nonlinear and ensemble methods could be explored, including:

- k-nearest neighbors.
- Support vector machines
- Decision trees
- Random forest
- Gradient boosting machines

Careful attention is required to ensure that the fitting and evaluation of these models preserved the temporal structure in the data. This is important so that the method is not able to ‘cheat’ by harnessing observations from the future.

These methods are often agnostic to large numbers of variables and may aid in teasing out whether the additional variables can be harnessed and add value to predictive models.

Deep Learning Methods

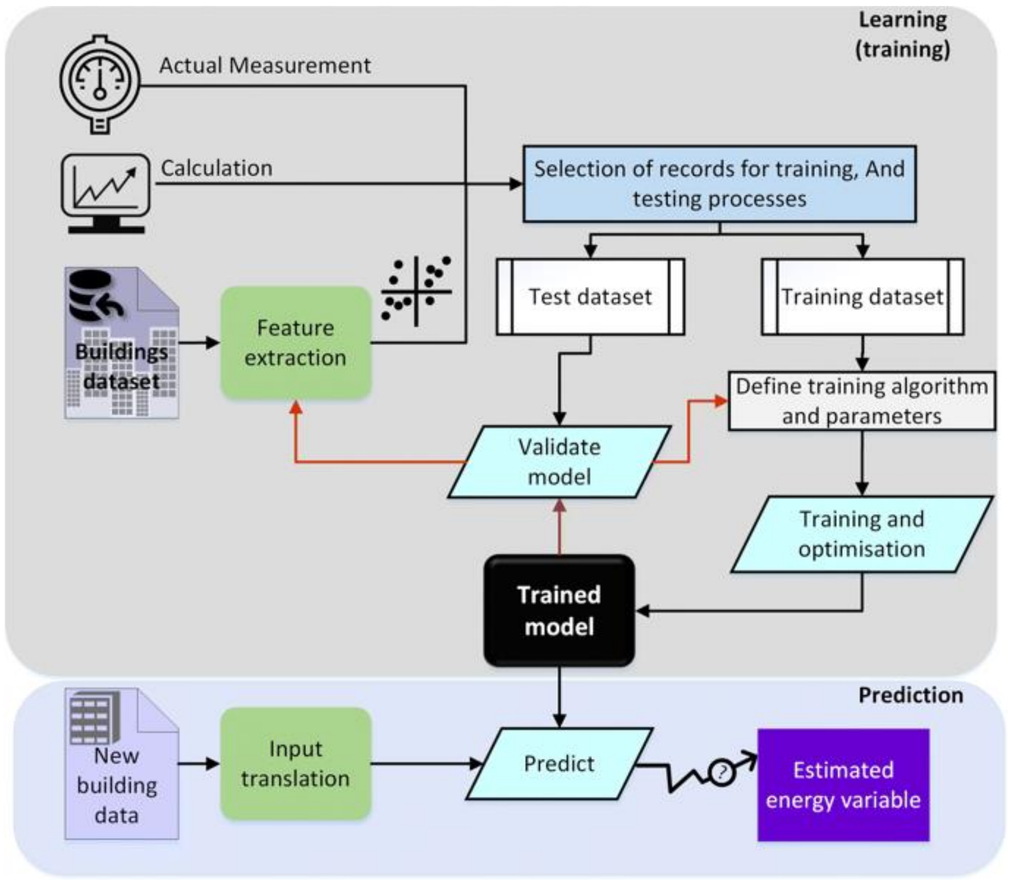
Generally, neural networks have not proven very effective at autoregression type problems.

Nevertheless, techniques such as convolutional neural networks are able to automatically learn complex features from raw data, including one-dimensional signal data. And recurrent neural networks, such as the long short–term memory network, are capable of directly learning across multiple parallel sequences of input data.

Further, combinations of these methods, such as CNN LSTM and ConvLSTM, have proven effective on time series classification tasks.

It is possible that these methods may be able to harness the large volume of minute–based data and multiple input variables.

3.2 Block diagram



3.3 Hardware/software designing

- Dataset preparation
- Dataset pre–processing
    1. Data visualisation
    2. Taking care of the missing data
    3. Label encoding and one–hot encoding (not indispensable)
    4. Data transformation

    5. Data splitting into train and test
- Model Building
    1. Training and testing the model
    2. Evaluation and the final predicting method should be such that the accuracy should probably be at most.
- Model Deployment
    1. Creating an HTML file
    2. Structuring a proper python code

4. Experimental Investigations

The Household Power Consumption dataset is a multivariate time series dataset that describes the electricity consumption for a single household over four years. The data was collected between December 2006 and November 2010 and observations of power consumption within the household were collected every minute.
It is a multivariate series comprised of seven variables (besides the date and time); they are:

- global_active_power: The total active power consumed by the household (kilowatts).
- global_reactive_power: The total reactive power consumed by the household (kilowatts).
- voltage: Average voltage (volts).
- global_intensity: Average current intensity (amps).
- sub_metering_1: Active energy for kitchen (watt-hours of active energy).
- sub_metering_2: Active energy for laundry (watt-hours of active energy).
- sub_metering_3: Active energy for climate control systems (watt-hours of active energy).

Active and reactive energy refer to the technical details of alternative current. In general terms, the active energy is the real power consumed by the household, whereas the reactive energy is the unused power in the lines. We can see that the dataset provides the active power as well as some division of the active power by main circuit in the house, specifically the kitchen, laundry, and climate control. These are not all the circuits in the household. The remaining watt-hours can be calculated from the active energy by first converting the active energy to watt-hours then

subtracting the other sub-metered active energy in watt-hours, as follows:
sub_metering_remainder = (global_active_power * 1000 / 60) - (sub_metering_1 + sub_metering_2 + sub_metering_3)

The dataset seems to have been provided without a seminal reference paper. Nevertheless, this dataset has become a standard for evaluating time series forecasting and machine learning methods for multi-step forecasting, specifically for forecasting active power. Further, it is not clear whether the other features in the dataset may benefit a model in forecasting active power.

The data is a multivariate time series and the best way to understand a time series is to create line plots. We can start off by creating a separate line plot for each of the eight variables. This gives us a really high level of the four years of one minute observations. We can see that something interesting was going on in ‘*Sub_metering_3* ‘ (environmental control) that may not directly map to hot or cold years. Perhaps new systems were installed. Interestingly, the contribution of ‘*sub_metering_4* ‘ seems to decrease with time, or show a downward trend, perhaps matching up with the solid increase in seen towards the end of the series for ‘*Sub_metering_3* ‘. These observations do reinforce the need to honor the temporal ordering of subsequences of this data when fitting and evaluating any model. We might be able to see the wave of a seasonal effect in the ‘*Global_active_power* ‘ and some other variates. There is some spiky usage that may match up with a specific period, such as weekends.

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. We can see some common gross patterns across the years, such as around Feb-Mar and around Aug-Sept where we see a marked decrease in consumption. We also seem to see a downward trend over the summer months (middle of the year in the northern hemisphere) and perhaps more consumption in the winter months towards the edges of the plots. These may show an

annual seasonal pattern in consumption. We can also see a few patches of missing data in at least the first, third, and fourth plots.

We can continue to zoom in on consumption and look at active power for each of the 12 months of 2007. This might help tease out gross structures across the months, such as daily and weekly patterns. We can see the sign−wave of power consumption of the days within each month. This is good as we would expect some kind of daily pattern in power consumption. We can see that there are stretches of days with very minimal consumption, such as in August and in April. These may represent vacation periods where the home was unoccupied and where power consumption was minimal.

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. 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. 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. 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. 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. 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.

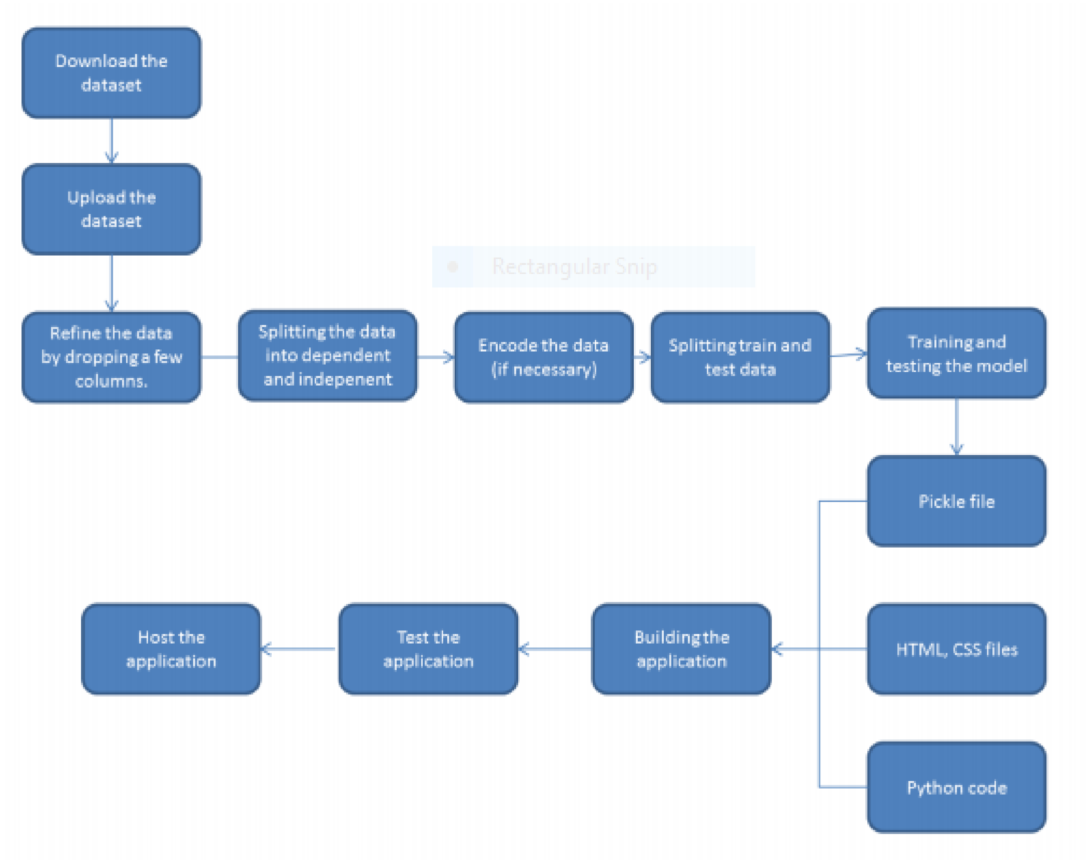
There does not appear to be a seminal publication for the dataset to demonstrate the intended way to frame the data in a predictive modeling problem. We are therefore left to guess at possibly useful ways that this data may be used. The data is only for a single household, but perhaps effective modeling approaches could be generalized across to similar households. Perhaps the most useful framing of the dataset is to forecast an interval of future active power consumption.

Four examples include:

- Forecast hourly consumption for the next day.
- Forecast daily consumption for the next week.
- Forecast daily consumption for the next month.
- Forecast monthly consumption for the next year.

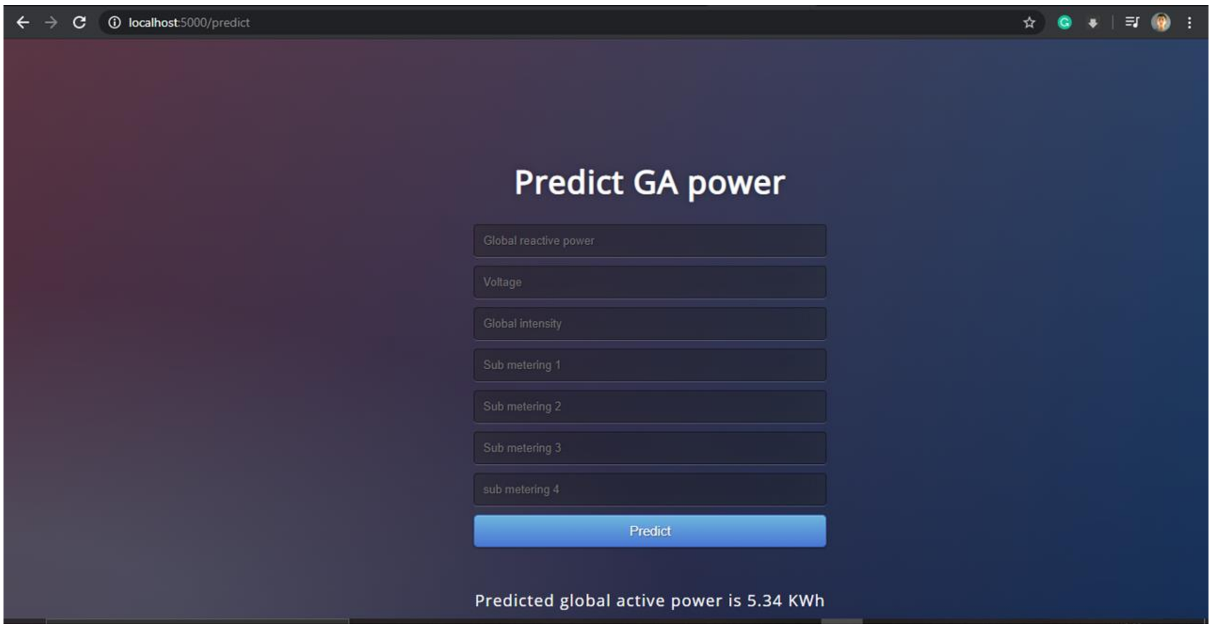
Generally, these types of forecasting problems are referred to as multi-step forecasting. Models that make use of all of the variables might be referred to as a multivariate multi-step forecasting model. Each of these models is not limited to forecasting the minutely data, but instead could model the problem at or below the chosen forecast resolution. Forecasting consumption in turn, at scale, could aid in a utility company forecasting demand, which is a widely studied and important problem.

5. Flow chart


Download the dataset
Upload the dataset
Refine the data by dropping a few columns.
Splitting the data into dependent and indepenent
Encode the data (if necessary)
Splitting train and test data
Training and testing the model
Pickle file
HTML, CSS files
Python code
Building the application
Test the application
Host the application
Rectangular Snip

6. Result

In this project, we discovered a household power consumption dataset for multi–step time series forecasting and better understood the raw data using exploratory analysis. The household power consumption dataset described electricity usage for a single house over four years. We explored and understood the dataset using a suite of line plots for the series data and histogram for the data distributions. We used the new understanding of the problem to consider different framings of the prediction problem, ways the data may be prepared, and modelling methods that may be used. Finally, we chose multi–linear regression model as it offered highest accuracy. Then we implemented our model to predict the household’ s global active power usage given the consumption by various appliances, voltage, global intensity etc. using a local host web application. We have provided an end–to–end demonstration of how you can use machine learning to determine the operating status of home appliances accurately, based on only smart power readings.



Predicted global active power is 5.34 KWh

7. Advantages and disadvantages

Advantages and disadvantages of multi linear regression learning model in predicting the global active power usage in our project were

Advantages

1. It' s very simple to understand.
2. Good interpretability.
3. High accuracy and cheap computational cost.
4. Space complexity is very low it just needs to save the weights at the end of training. hence, it's a high latency algorithm.
5. Feature importance is generated at the time model building. With the help of hyperparameter lamba, you can handle features selection hence we can achieve dimensionality reduction.
6. Ground for more complex machine learning algorithms.

Disadvantages

1. The algorithm assumes data is normally distributed in real they are not.
2. Before building the model we need to avoid multicollinearity. In our case we needed to ignore any such multicollinearity between input variables.
3. Prone to outliers. By its definition, linear regression only models' relationships between dependent and independent variables that are linear. It assumes there is a straight–line relationship between them

which is incorrect sometimes. Linear regression is very sensitive to the anomalies in the data (or outliers).

4. Take for example most of your data lies in the range 0-10. If due to any reason only one of the data items comes out of the range, say for example 15, this significantly influences the regression coefficients.
5. Oversimplify or fail in non-linear problems (only do well in linear modeling).
6. Sensitive to outliers and noises. If we have a number of parameters than the number of samples available then the model starts to model the noise rather than the relationship between the variables.

8. Applications

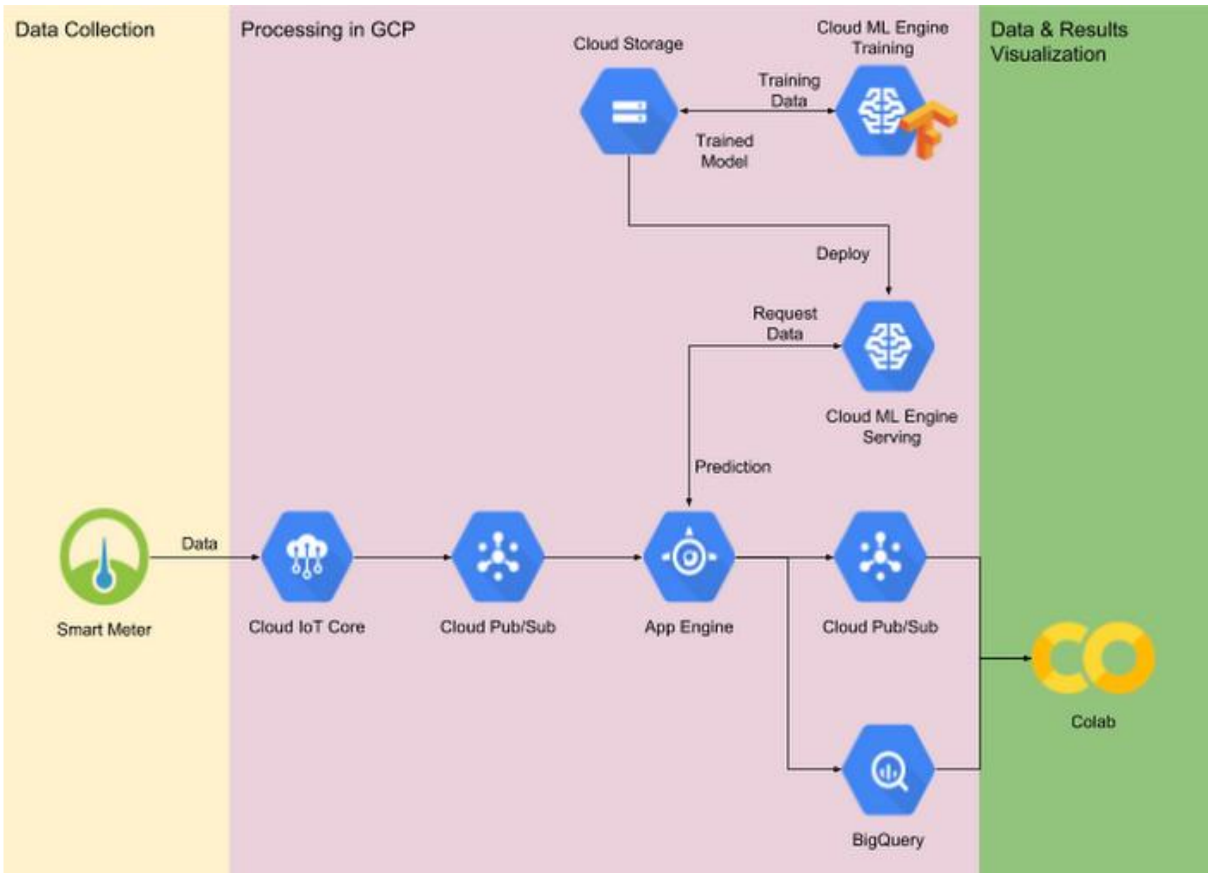
The proposed solution to reduce electricity consumption and bill in a household using machine learning can be extended and used in all the homes by using suitable sensors to collect the data every minute from various appliances. The web application product can then be used to learn from the data and predict the global active power for any sort of logical values of the input variables. This proposed solution is more applicable and beneficial in India with high ratio of middle class and poor people who find it difficult to pay increasing electricity bills. This can extend to industries as well. Also, people willing to save environment by saving the electric power will find it beneficial.

9. Conclusion

We successfully forecasted the global active power consumption using multi-linear regression model. Forecasting consumption in turn, at scale, could aid in a utility company forecasting demand, which is a widely studied and important problem.

10. Future Scope

This system can be used in all the households in the country in the coming future time to limit and reduce the electricity bill of the people. We can save the consumption of power and therefore avoid wastage of power as well as money that occurs due to the lack of information about the consumption of power by the individual appliances. Several products including Cloud IoT Core, Cloud Pub/Sub, Cloud ML Engine, App Engine and BigQuery can be orchestrated to support the whole system, in which each product solves a specific problem required to implement this demo, such as data collection/ingestion, machine learning model training, real time serving/prediction, etc.



11. References/ Bibliography

https://archive.ics.uci.edu/ml/datasets/Individual+household+electric+power+consumption

https://machinelearningmastery.com/how-to-load-and-explore-household-electricity-usage- data/

https://cloud.google.com/blog/products/ai-machine-learning/monitoring-home-appliances-from-power-readings-with-ml

12. Appendix

12.1 Data Visualization and ML jupyter Source code

```
#Importing the Libraries

    import numpy as np
    import pandas as pd
    from matplotlib import pyplot as plt

    # load all data

dataset = pd.read_csv(r"G:\smartbridge\Team project (Electricity
consumption)\household_power_consumption.txt", sep=';',
header=0, low_memory=False, infer_datetime_format=True,
parse_dates={'datetime':[0,1]}, index_col=['datetime'])

# summarize

dataset.shape
dataset.head()


    # mark all missing values

    dataset.replace('?', np.nan, inplace=True)

    # add a column for the remainder of sub metering

    values = dataset.values.astype('float32')
    dataset['sub_metering_4'] = (values[:,0] * 1000 / 60) -
(values[:,4] +
    values[:,5] + values[:,6])

    # save updated dataset

    dataset.to_csv('household_power_consumption.csv')

    # load the new file

     dataset = pd.read_csv('household_power_consumption.csv',
header=0,
     infer_datetime_format=True, parse_dates=['datetime'],
     index_col=['datetime'])

    dataset.head()
```

```
        dataset.shape

        #Patterns in Observations Over Time

         # line plot for each variable

          plt.figure(figsize=(20,20))
         for i in range(len(dataset.columns)):
          plt.subplot(len(dataset.columns), 1, i+1)
          name = dataset.columns[i]
          plt.plot(dataset[name])
          plt.title(name, y=0)
           plt.show()


years = ['2007', '2008', '2009', '2010']
plt.figure(figsize=(20,20))
for i in range(len(years)):
   # prepare subplot
   ax = plt.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
   plt.plot(result['Global_active_power'])
   # add a title to the subplot
   plt.title(str(year), y=0, loc='left')
plt.show()

# plot active power for each year

months = [x for x in range(1, 13)]
plt.figure(figsize=(20,20))
for i in range(len(months)):
   # prepare subplot
   ax = plt.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
   plt.plot(result['Global_active_power'])
```

```
    # add a title to the subplot
    plt.title(month, y=0, loc='left')
plt.show()

# plot active power for each year

days = [x for x in range(1, 21)]
plt.figure(figsize=(20,20))
for i in range(len(days)):
    # prepare subplot
    ax = plt.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
    plt.plot(result['Global_active_power'])
    # add a title to the subplot
    plt.title(day, y=0, loc='left')
plt.show()

# histogram plot for each variable

plt.figure(figsize=(20,20))
for i in range(len(dataset.columns)):
    plt.subplot(len(dataset.columns), 1, i+1)
    name = dataset.columns[i]
    dataset[name].hist(bins=100)
    plt.title(name, y=0)
plt.show()

# plot active power for each year

years = ['2007', '2008', '2009', '2010']
plt.figure(figsize=(20,20))
for i in range(len(years)):
    # prepare subplot
    ax = plt.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
        plt.title(str(year), y=0, loc='right')
    plt.show()

    # plot active power for each year

    months = [x for x in range(1, 13)]
    plt.figure(figsize=(20,20))
    for i in range(len(months)):
        # prepare subplot
        ax = plt.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
        plt.title(month, y=0, loc='right')
    plt.show()

    #Data Preprocessing

    dataset.isnull().any()
    dataset['Global_active_power'].fillna(dataset['Global_active_power'].m
ean(),inplace = True)
    dataset['Global_reactive_power'].fillna(dataset['Global_reactive_power'
].mean(),inplace = True)
    dataset['Voltage'].fillna(dataset['Voltage'].mean(),inplace = True)
    dataset['Global_intensity'].fillna(dataset['Global_intensity'].mean(),inpla
ce = True)
    dataset['Sub_metering_1'].fillna(dataset['Sub_metering_1'].mean(),in
place = True)
    dataset['Sub_metering_2'].fillna(dataset['Sub_metering_2'].mean(),in
place = True)
    dataset['Sub_metering_3'].fillna(dataset['Sub_metering_3'].mean(),in
place = True)
```

```
   dataset['sub_metering_4'].fillna(dataset['sub_metering_4'].mean(),inp
lace = True)

   dataset.isnull().any()

   #Defining input and output varibles

   x = dataset.iloc[:,1:8].values
   y = dataset.iloc[:,0:1].values

   x.shape
   y.shape
   dataset.describe()

   #split the data in to train and test

   from sklearn.model_selection import train_test_split
   x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.2,
random_state = 0)

   x_train.shape
   y_train.shape
   x_test.shape
   y_test.shape

   #MultiLinear Regression learning model

   from sklearn.linear_model import LinearRegression
   mlr  = LinearRegression()
   mlr.fit(x_train,y_train)

   y_pred = mlr.predict(x_test)

   from sklearn.metrics import r2_score
   accuracy = r2_score(y_test,y_pred)

   accuracy

   mlr.predict([[0.5,239,25,0,2.5,16.5,65]])

   #Saving the model by serializing it using pickle.

   import pickle
```

```
pickle.dump(mlr,open('MLR_model.pkl','wb'))

#Decision Tree Regression learning Model

from sklearn.tree import  DecisionTreeRegressor
dtr = DecisionTreeRegressor(random_state = 0)
dtr.fit(x_train,y_train)

ydtr = dtr.predict(x_test)
accuratdtr = r2_score(y_test,ydtr)
accuratdtr
dtr.predict([[0.5,239,25,0,2.5,16.5,65]])
```

12.2 HTML source code

```
<!DOCTYPE html>
<html >
<!--From https://codepen.io/frytyler/pen/EGdtg-->
<head>
  <meta charset="UTF-8">
  <title>ML API</title>
  <link href='https://fonts.googleapis.com/css?family=Pacifico'
rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Arimo'
rel='stylesheet' type='text/css'>
<link href='https://fonts.googleapis.com/css?family=Hind:300'
rel='stylesheet' type='text/css'>
<link
href='https://fonts.googleapis.com/css?family=Open+Sans+Conde
nsed:300' rel='stylesheet' type='text/css'>
<link rel="stylesheet" href="{{ url_for('static',
filename='css/style.css') }}">

</head>

<body>
 <div class="login">
     <h1>Predict GA power</h1>

      <!-- Main Input For Receiving Query to our ML -->
     <form action="{{ url_for('predict')}}"method="post">
     <input type="text" name="Global_reactive_power"
placeholder="Global reactive power" required="required" />
         <input type="text" name="Voltage"
placeholder="Voltage" required="required" />
         <input type="text" name="Global_intensity"
placeholder="Global intensity" required="required" />
         <input type="text" name="Sub_metering_1"
placeholder="Sub metering 1" required="required" />
         <input type="text" name="Sub_metering_2"
placeholder="Sub metering 2" required="required" />
         <input type="text" name="Sub_metering_3"
placeholder="Sub metering 3" required="required" />
          <input type="text" name="sub_metering_4"
placeholder="sub metering 4" required="required" />

         <button type="submit" class="btn btn-primary btn-
block btn-large">Predict</button>
    </form>

   <br>
   <br>
   {{ prediction_text }}

 </div>


</body>
</html>
```

12.3 Flask app source code

```
import numpy as np
from flask import Flask, request, render_template
import pickle

app = Flask(__name__) #Initialize the flask App
model = pickle.load(open('MLR_model.pkl', 'rb'))

@app.route('/')
def home():
    return render_template('index.html')
```

```
@app.route('/predict',methods=['POST'])
def predict():
    '''
    For rendering results on HTML GUI
    '''
    int_features = [float(x) for x in request.form.values()]
    final_features = [np.array(int_features)]
    prediction = model.predict(final_features)

    output = np.round(prediction[0], 2)
    output=str(output[0])+" KWh"

    return render_template('index.html',
prediction_text='Predicted global active power is
{}'.format(output))

if __name__ == "__main__":
    app.run(debug=True)
```

12.4 CSS file source code

```
@import url(https://fonts.googleapis.com/css?family=Open+Sans);
.btn { display: inline-block; *display: inline; *zoom: 1; padding:
4px 10px 4px; margin-bottom: 0; font-size: 13px; line-height:
18px; color: #333333; text-align: center;text-shadow: 0 1px
1px rgba(255, 255, 255, 0.75); vertical-align: middle;
background-color: #f5f5f5; background-image: -moz-linear-
gradient(top, #ffffff, #e6e6e6); background-image: -ms-linear-
gradient(top, #ffffff, #e6e6e6); background-image: -webkit-
gradient(linear, 0 0, 0 100%, from(#ffffff), to(#e6e6e6));
background-image: -webkit-linear-gradient(top, #ffffff,
#e6e6e6); background-image: -o-linear-gradient(top, #ffffff,
#e6e6e6); background-image: linear-gradient(top, #ffffff,
#e6e6e6); background-repeat: repeat-x; filter:
progid:dximagetransform.microsoft.gradient(startColorstr=#ffffff,
endColorstr=#e6e6e6, GradientType=0); border-color: #e6e6e6
#e6e6e6 #e6e6e6; border-color: rgba(0, 0, 0, 0.1) rgba(0, 0,
0, 0.1) rgba(0, 0, 0, 0.25); border: 1px solid #e6e6e6; -
webkit-border-radius: 4px; -moz-border-radius: 4px; border-
radius: 4px; -webkit-box-shadow: inset 0 1px 0 rgba(255,
255, 255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.05); -moz-box-
shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px
rgba(0, 0, 0, 0.05); box-shadow: inset 0 1px 0 rgba(255, 255,
255, 0.2), 0 1px 2px rgba(0, 0, 0, 0.05); cursor: pointer;
*margin-left: .3em; }
.btn:hover, .btn:active, .btn.active, .btn.disabled, .btn[disabled] {
background-color: #e6e6e6; }
.btn-large { padding: 9px 14px; font-size: 15px; line-height:
normal; -webkit-border-radius: 5px; -moz-border-radius: 5px;
border-radius: 5px; }
.btn:hover { color: #333333; text-decoration: none;
background-color: #e6e6e6; background-position: 0 -15px; -
webkit-transition: background-position 0.1s linear; -moz-
transition: background-position 0.1s linear; -ms-transition:
background-position 0.1s linear; -o-transition: background-
position 0.1s linear; transition: background-position 0.1s linear;
}
.btn-primary, .btn-primary:hover { text-shadow: 0 -1px 0
rgba(0, 0, 0, 0.25); color: #ffffff; }
.btn-primary.active { color: rgba(255, 255, 255, 0.75); }
.btn-primary { background-color: #4a77d4; background-image:
-moz-linear-gradient(top, #6eb6de, #4a77d4); background-
image: -ms-linear-gradient(top, #6eb6de, #4a77d4);
background-image: -webkit-gradient(linear, 0 0, 0 100%,
from(#6eb6de), to(#4a77d4)); background-image: -webkit-
linear-gradient(top, #6eb6de, #4a77d4); background-image: -
o-linear-gradient(top, #6eb6de, #4a77d4); background-image:
linear-gradient(top, #6eb6de, #4a77d4); background-repeat:
repeat-x; filter:
progid:dximagetransform.microsoft.gradient(startColorstr=#6eb6de,
endColorstr=#4a77d4, GradientType=0);  border: 1px solid
#3762bc; text-shadow: 1px 1px 1px rgba(0,0,0,0.4); box-
shadow: inset 0 1px 0 rgba(255, 255, 255, 0.2), 0 1px 2px
rgba(0, 0, 0, 0.5); }
.btn-primary:hover, .btn-primary:active, .btn-primary.active,
.btn-primary.disabled, .btn-primary[disabled] { filter: none;
background-color: #4a77d4; }
.btn-block { width: 100%; display:block; }

* { -webkit-box-sizing:border-box; -moz-box-sizing:border-
box; -ms-box-sizing:border-box; -o-box-sizing:border-box;
box-sizing:border-box; }

html { width: 100%; height:100%; overflow:hidden; }
```

```
body {
	width: 100%;
	height:100%;
	font-family: 'Open Sans', sans-serif;
	background: #092756;
	color: #fff;
	font-size: 18px;
	text-align:center;
	letter-spacing:1.2px;
	background: -moz-radial-gradient(0% 100%, ellipse cover, 
rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%),-moz-linear-
gradient(top,  rgba(57,173,219,.25) 0%, rgba(42,60,87,.4) 
100%), -moz-linear-gradient(-45deg,  #670d10 0%, #092756 
100%);
	background: -webkit-radial-gradient(0% 100%, ellipse 
cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -
webkit-linear-gradient(top,  rgba(57,173,219,.25) 
0%,rgba(42,60,87,.4) 100%), -webkit-linear-gradient(-45deg, 
#670d10 0%,#092756 100%);
	background: -o-radial-gradient(0% 100%, ellipse cover, 
rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -o-linear-
gradient(top,  rgba(57,173,219,.25) 0%,rgba(42,60,87,.4) 100%), 
-o-linear-gradient(-45deg,  #670d10 0%,#092756 100%);
	background: -ms-radial-gradient(0% 100%, ellipse cover, 
rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), -ms-
linear-gradient(top,  rgba(57,173,219,.25) 0%,rgba(42,60,87,.4) 
100%), -ms-linear-gradient(-45deg,  #670d10 0%,#092756 
100%);
	background: -webkit-radial-gradient(0% 100%, ellipse 
cover, rgba(104,128,138,.4) 10%,rgba(138,114,76,0) 40%), 
linear-gradient(to bottom,  rgba(57,173,219,.25) 
0%,rgba(42,60,87,.4) 100%), linear-gradient(135deg,  #670d10 
0%,#092756 100%);
	filter: progid:DXImageTransform.Microsoft.gradient( 
startColorstr='#3E1D6D', endColorstr='#092756',GradientType=1 );

}
.login {
	position: absolute;
	top: 40%;
	left: 50%;
	margin: -150px 0 0 -150px;
	width:400px;
	height:400px;
}

.login h1 { color: #fff; text-shadow: 0 0 10px rgba(0,0,0,0.3);
letter-spacing:1px; text-align:center; }

input {
	width: 100%;
	margin-bottom: 10px;
	background: rgba(0,0,0,0.3);
	border: none;
	outline: none;
	padding: 10px;
	font-size: 13px;
	color: #fff;
	text-shadow: 1px 1px 1px rgba(0,0,0,0.3);
	border: 1px solid rgba(0,0,0,0.3);
	border-radius: 4px;
	box-shadow: inset 0 -5px 45px rgba(100,100,100,0.2), 0
1px 1px rgba(255,255,255,0.2);
	-webkit-transition: box-shadow .5s ease;
	-moz-transition: box-shadow .5s ease;
	-o-transition: box-shadow .5s ease;
	-ms-transition: box-shadow .5s ease;
	transition: box-shadow .5s ease;
}
input:focus { box-shadow: inset 0 -5px 45px
rgba(100,100,100,0.4), 0 1px 1px rgba(255,255,255,0.2); }
```