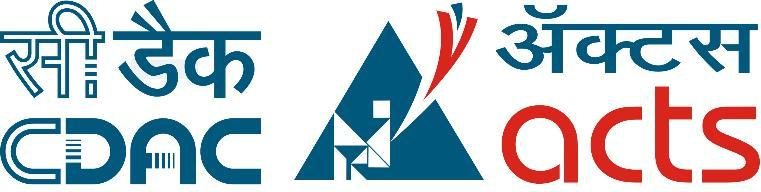
A

Project Report on

**Electricity Forecasting on the individual household based on activity patterns and environmental conditions**

Submitted in partial fulfilment for the award of

**PG DIPLOMA IN BIG DATA ANALYTICS**



CENTRE FOR DEVELOPMENT OF ADVANCED COMPUTING (C-DAC)

ADVANCED COMPUTING TRAINING SCHOOL (ACTS)

Knowledge Park, Bangalore - 560 038

Submitted By

ANSON ALEXANDER PRN: 180250125007

NEERAJ KRIPALANI PRN: 180250125035

NIKHIL HIREMATH PRN: 180250125046

OBIN JOSEPH PRN: 180250125047

Guided By

MR. MOHIT VED

**CERTIFICATE**

This is to certify that, the project report entitled

**Electricity Forecasting on the individual household based on activity patterns and environmental conditions**

Submitted by

ANSON ALEXANDER PRN: 180250125007

NEERAJ KRIPLANI PRN: 180250125035

NIKHIL HIREMATH PRN: 180250125046

OBIN JOSEPH PRN: 180250125047

Is the record of bonfire work carried out by them in partial fulfilment of the requirement for the award of **PG Diploma in Big Data Analytics** prescribed by **Centre for Development of Advanced Computing (C-DAC)**.

|  |  |
| --- | --- |
| Mr. Mohit Ved | Mrs M Savitri |
| Project Guide | Course Co-ordinator |

**ACKNOWLEDGEMENT**

We would like to express our sincere gratitude to all the people without whom this project would have been highly impossible.

We would like to devote our first vote of thanks to our guide Mr. Mohit Ved for his constant support and encouragement. He has a great hand in the firm foundation of this project. We are deeply in debt for his valuable suggestions, scholarly guidance and constructive criticisms along with constant encouragement at each and every step for successful completion of the project.

We would also like to thank our Project Co-ordinator Mrs. Janaki for inspiring us towards completion of this project

Last but not the least we would like to thank all those who assisted us directly or indirectly for their valuable time and help.

**ABSTRACT**

Electricity load forecasting is an important aspect of power systems planning and operation. At the utility scale, load forecasting is important for pricing and determination of the size of the spinning reserve. For this problem, forecasting is done on a large scale by aggregating the power consumed by many homes in a single neighbourhood. On a single home scale for grids with real time pricing, forecasting permits using energy storage systems to decrease cost of energy for the consumer. Knowledge of future power consumption along with future electricity prices can make it possible to decide when to engage a battery storage system as opposed to drawing power from the grid. The aim of the project is to carry out a short term forecast on electricity consumption of a single home.

|  |  |  |
| --- | --- | --- |
| **SR. NO.** | **DESCRIPTION** | **PAGE NO.** |
| 1. | Introduction | 6 |
| 2. | Objective | 7 |
| 3. | Data | 8 |
| 4. | Methodology | 9 |
| 5. | Accuracy Assessment | 17 |
| 6. | Visualisation | 18 |
| 7. | Conclusion | 23 |
| 8. | References | 23 |

**TABLE OF CONTENTS**

**INTRODUCTION**

In many countries worldwide, electricity is now traded under market rules using spot and derivative contracts. However, electricity is a very special commodity. It is economically non-storable, and power system stability requires a constant balance between production and consumption. At the same time, electricity demand depends on whether (temperature, wind speed, precipitation, etc.) and the intensity of business and everyday activities. On the one hand, these unique and specific characteristics lead to price dynamics not observed in any other market, exhibiting seasonality at the daily, weekly and annual levels, and abrupt, short-lived and generally unanticipated price spikes. On the other hand, they have encouraged researchers to intensify their efforts in the development of better forecasting techniques. Load forecasting can be divided into three categories: short, medium, and long term. Short term forecasting corresponds to prediction of one hour to one week ahead. Medium-term refers to one week ahead to one year ahead while long-term forecasts are for more than a year. Here, we focus on developing regression models for short-term forecasting. There is not a lot of previous work on single home forecasting. We have utilized different sampling strategies including dividing the data by day and by time-series. Here we are implementing the same approach using time series analysis ARIMA (Autoregressive Integrated Moving Average), SARIMA (Seasonal Autoregressive Integrated Moving Average), and ARIMAX (Autoregressive Integrated Moving Average Extended) models to forecast electricity usage of this individual house hold.

**OBJECTIVE**

Our objective is developing regression models for short-term forecasting. There is not a lot of previous work on single home forecasting. We have utilized different sampling strategies including resampling the data by month, by week by day or by fifteen minutes interval time-series. We are implementing time series models like ARIMA (Autoregressive Integrated Moving Average), SARIMA (Seasonal Autoregressive Integrated Moving Average), and ARIMAX (Autoregressive Integrated Moving Average Extended) to forecast electricity usage of this individual house hold. Calculate the accuracy of each model.

**DATA**

The data sets were obtained from the U MASS repository. One dataset represents electricity consumption of individual devices like microwave oven, furnace, refrigerator, washing machine, garage, solar, generator, front room etc., for an interval of 15 minutes measured over 3 years from year 2014 to 2016, for a single home in United States of America (approximately one lakh thirty three thousand data points). Other dataset consists of weather data of the same area over same time period. It includes columns like temperature, humidity, wind speed, humidity, precipitation, apparent temperature, cloud etc. for every one hour interval.

**METHODOLOGY**

**1) PREPAIRNG THE DATA**

There were surprisingly few data points missing from either source over the entire data set, so we simply removed these points. Since the data set was very large, we chose to process it in Python using the PANDAS package. Import both the datasets in different data frame.

Weather dataset includes columns like temperature, humidity, wind speed, humidity, precipitation, apparent temperature, cloud etc. for every one hour interval. Whereas our electricity dataset had an interval of fifteen minutes. So, we applied interpolation method to get weather data in fifteen minute interval.Interpolation is the process of finding a value between two points on a line or curve. When graphical [data](https://searchdatamanagement.techtarget.com/definition/data) contains a gap, but data is available on either side of the gap or at a few specific points within the gap, interpolation allows us to estimate the values within the gap. After interpolation merge both data frames into a single data frame.

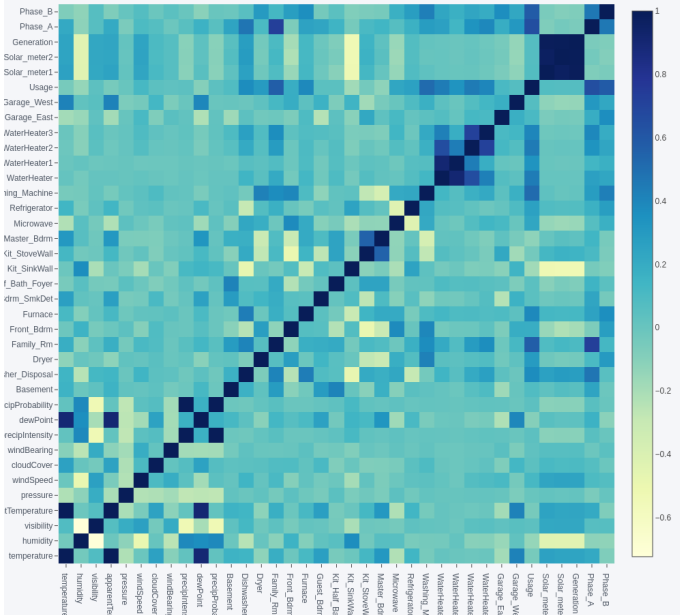
R**esampling** is the method that consists of drawing repeated [samples](http://www.statisticssolutions.com/sample-size-calculation-and-sample-size-justification/sampling/) from the original data samples. The method of Resampling is a nonparametric method of statistical inference. Resampling of data helps to represent and use the data in terms of month, week or day interval of time.

**2)** **CREATE A TRAINING DATASOURCE**

Divide the data set into train and test set. Ninety per cent of data as train and remain ten per cent as test. Now when the train and test dataset is ready we can use it to fit in a model, perform forecasting.

**3) CORRELATION COEFFICIENT**

Correlation is a statistical technique that can show whether and how strongly pairs of variables are related. Below correlation graph shows how all variable are related to each other. The main result of a correlation is called the correlation coefficient (or "r"). It ranges from -1.0 to +1.0. The closer r is to +1 or -1, the more closely the two variables are related. If r is close to 0, it means there is no relationship between the variables. If r is positive, it means that as one variable gets larger the other gets larger. If r is negative it means that as one gets larger, the other gets smaller (often called an "inverse" correlation).

****

**4) CREATE AN ARIMA MODEL AND GENERATE PREDICTIONS**

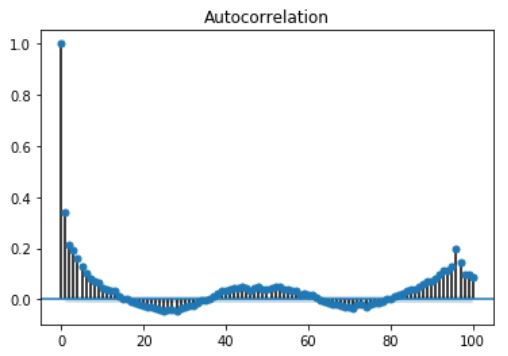
In statistics and econometrics, and in particular in time series analysis, an autoregressive integrated moving average (ARIMA) model is a generalization of an autoregressive moving average (ARMA) model. Both of these models are fitted to time series data either to better understand the data or to predict future points in the series (forecasting). ARIMA models are applied in some cases where data show evidence of non-stationary, where an initial differencing step (corresponding to the "integrated" part of the model) can be applied one or more times to eliminate the non-stationary. A time series is said to be **stationary** if its statistical properties such as mean, variance remain constant over time.

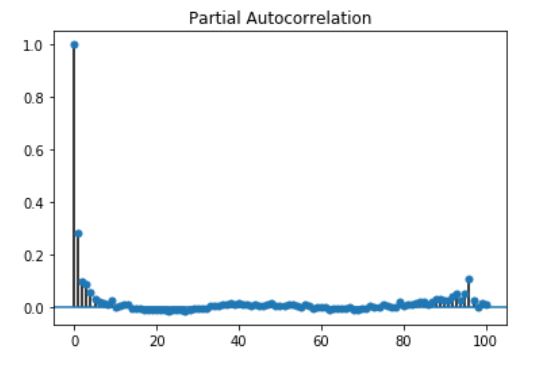
The AR part of ARIMA indicates that the evolving variable of interest is regressed on its own lagged (i.e., prior) values. The MA part indicates that the regression error is actually a linear combination of error terms whose values occurred contemporaneously and at various times in the past. The I (for "integrated") indicate that the data values have been replaced with the difference between their values and the previous values (and this differencing process may have been performed more than once). The purpose of each of these features is to make the model fit the data as well as possible.

Non-seasonal ARIMA models are generally denoted ARIMA (p, d, q) where parameters p, d, and q are non-negative integers, p is the order (number of time lags) of the autoregressive model, d is the degree of differencing (the number of times the data have had past values subtracted), and q is the order of the moving-average model.

An importance concern here is how to determine the value of ‘p’ and ‘q’. We use two plots to determine these numbers. Let’s discuss them first.

1. **Autocorrelation Function (ACF):** It is a measure of the correlation between the time series with a lagged version of itself. For instance at lag 5, ACF would compare series at time instant ‘t1’…’t2’ with series at instant ‘t1-5’…’t2-5’ (t1-5 and t2 being end points).
2. **Partial Autocorrelation Function (PACF):** This measures the correlation between the TS with a lagged version of itself but after eliminating the variations already explained by the intervening comparisons. E.g. at lag 5, it will check the correlation but remove the effects already explained by lags 1 to 4.

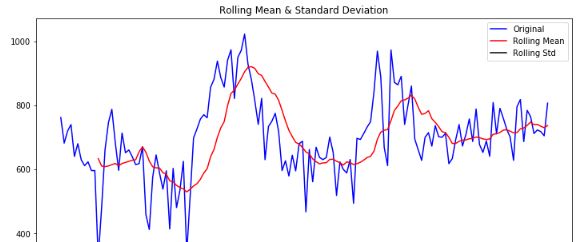
****

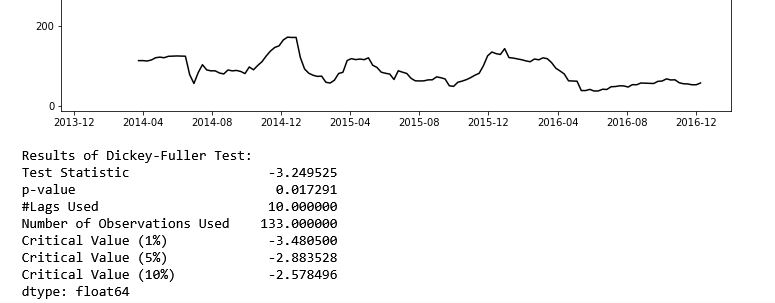


The lag value where the PACF chart crosses the upper confidence interval (a blue line seen near X axis) for the first time. If you notice closely, in this case p=2. The lag value where the ACF chart crosses the upper confidence interval for the first time. If you notice closely, in this case q=1.

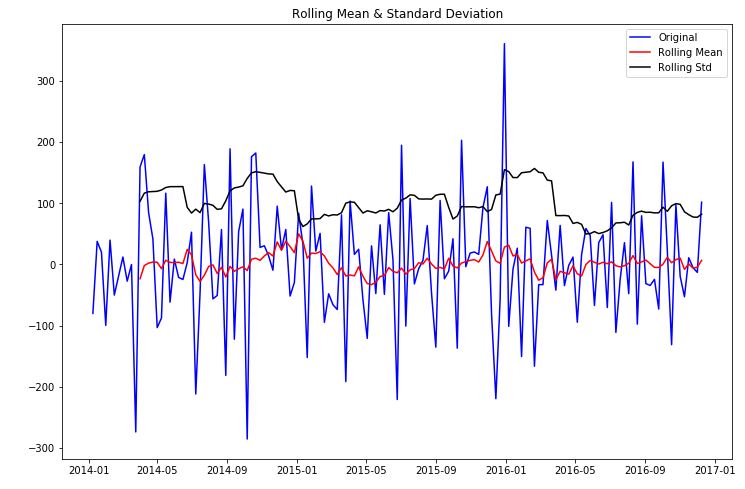
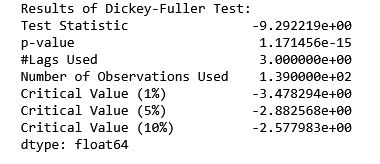
To know the value of d perform Dickey-Fuller test. In statistics, the Dickey–Fuller test tests the null hypothesis that a unit root is present in an autoregressive model. If the p value is greater than five percentages then, it fails to reject the null hypothesis (H0), the data has a unit root and is non-stationary.

If the p value is less than or equal to five percentage then, reject the null hypothesis (H0), the data does not have a unit root and is stationary.

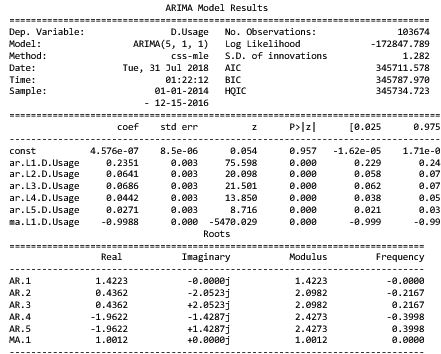


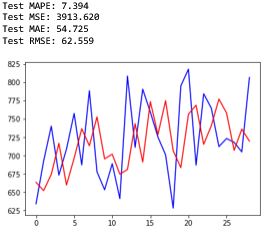


As visible in above picture p value is less than five percentages, so we can reject the null hypothesis and therefore the data is stationary. If we try to difference the data we get results as shown below

Fit the ARIMA model and predict the results:

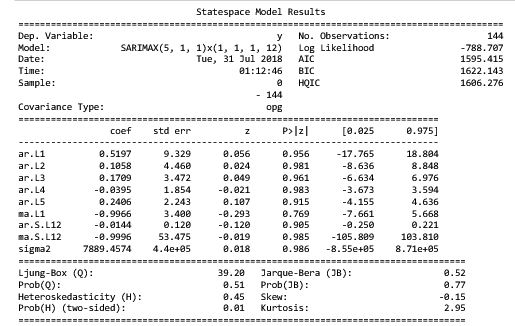


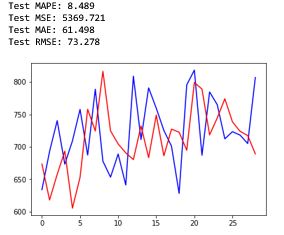


In above graph blue line is actual value and red line is predicted value. ARIMA model gave a Mean Absolute Percentage error of 7.354 per cent.

**5) CREATE AN SARIMAX MODEL AND GENERATE PREDICTIONS**

When dealing with seasonal effects, we make use of the seasonal ARIMA, which is denoted as ARIMA (p, d, q) (P, D, Q)s. Here, (p, d, q) are the non-seasonal parameters described above, while (P, D, Q) follow the same definition but are applied to the seasonal component of the time series. The term s is the periodicity of the time series (4 for quarterly periods, 12 for yearly periods, etc.).



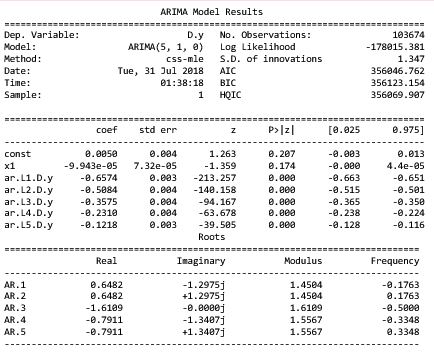


In above graph blue line is actual value and red line is predicted value. ARIMA model gave a Mean Absolute Percentage error of 8.489 per cent.

**6) CREATE AN ARIMAX MODEL**

Autoregressive Integrated Moving Average with Explanatory Variable. Its multivariate regression model .It considers explanatory variable when forecasting the dependent variable. Endog contains the dependent variable is use exog contains all the Explanatory Variable.

model3=sm.tsa.ARIMA(endog=dataset['Usage'].values,exog=dataset['temperature'].values,order=[5,1,0])



**ACCURACY AND ERROR**

The **mean absolute percentage error (MAPE)**, also known as mean absolute percentage deviation (MAPD), is a measure of prediction accuracy of a forecasting method in statistics, for example in trend estimation. It usually expresses accuracy as a percentageThe difference between At and Ft is divided by the actual value At again. The absolute value in this calculation is summed for every forecasted point in time and divided by the number of fitted points n. Multiplying by 100% makes it a percentage error.

In statistics, the **mean squared error (MSE)** or mean squared deviation (MSD) of an estimator (of a procedure for estimating an unobserved quantity) measures the average of the squares of the errors—that is, the average squared difference between the estimated values and what is estimated. MSE is a risk function, corresponding to the expected value of the squared error loss.

In statistics, **mean absolute error (MAE)** is a measure of difference between two continuous variables.

**Root Mean Square Error (RMSE)** measures **how much error there is between two data sets**. In other words, it compares a predicted value and an observed or known value

**VISUALISATION**

**Plotting Graphs explored for live data visualization**

**Matplotlib**

 Matplotlib is probably the single most used Python package for 2D-graphics. It provides both a very quick way to visualize data from Python and publication-quality figures in many formats. We are going to explore matplotlib in interactive mode covering most common cases.

*pyplot* provides a procedural interface to the matplotlib object-oriented plotting library. It is modeled closely after Matlab. Therefore, the majority of plotting commands in pyplot have Matlab analogs with similar arguments. Important commands are explained with interactive examples.

**from** **matplotlitb** **import** pyplot **as** plt

**Toolkits**

Several toolkits are available which extend matplotlib functionality. Some are separate downloads; others ship with the matplotlib source code but have external dependencies.

* Base map: map plotting with various map projections, coastlines, and political boundaries
* Cartopy: a mapping library featuring object-oriented map projection definitions, and arbitrary point, line, polygon and image transformation capabilities.
* Excel tools: utilities for exchanging data with Microsoft Excel
* GTK tools: interface to the GTk+ library
* Qt interface
* Mplot3d: 3-D plots
* Natgrid: interface to the natgrid library for gridding irregularly spaced data.
* matplotlib2tikz: export to Pgfplots for smooth integration into LaTeX documents

**Matplotlib.animation**

The easiest way to make a live animation in matplotlib is to use one of the Animation classes. The animation tools center around the matplotlib.animation. Animation base class, which provides a framework around which the animation functionality is built.

FuncAnimation

Makes an animation by repeatedly calling a function func.

**PyQtGraph**

PyQtGraph is a pure-python graphics and GUI library built on PyQt4 / PySide and numpy. It is intended for use in mathematics / scientific / engineering applications. Despite being written entirely in python, the library is very fast due to its heavy leverage of numpy for number crunching and Qt's GraphicsView framework for fast display. PyQtGraph is distributed under the MIT open-source license.

PyQtGraph makes heavy use of the Qt GUI platform for its high-performance graphics and numpy for heavy number crunching. In particular, pyqtgraph uses Qt’s GraphicsView framework which is a highly capable graphics system on its own, we bring optimized and simplified primitives to this framework to allow data visualization with minimal effort.

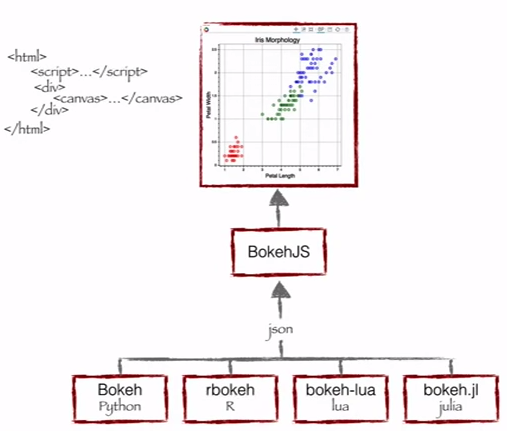
Amongst the core features of pyqtgraph are:

* Basic data visualization primitives: Images, line and scatter plots
* Fast enough for real-time update of video/plot data
* Interactive scaling/panning, averaging, FFTs, SVG/PNG export
* Widgets for marking/selecting plot regions
* Widgets for marking/selecting image region-of-interest and automatically slicing multi-dimensional image data
* Framework for building customized image region-of-interest widgets
* Docking system that replaces/complements Qt’s dock system to allow more complex (and more predictable) docking arrangements
* ParameterTree widget for rapid prototyping of dynamic interfaces (Similar to the property trees in Qt Designer and many other applications)

**Bokeh**

Bokeh is an interactive visualization library that targets modern web browsers for presentation. Its goal is to provide elegant, concise construction of versatile graphics, and to extend this capability with high-performance interactivity over very large or streaming datasets. Bokeh can help anyone who would like to quickly and easily create interactive plots, dashboards, and data applications.

Bokeh is a Python library for interactive visualization that targets web browsers for representation. This is the core difference between Bokeh and other visualization libraries. Look at the snapshot below, which explains the process flow of how Bokeh helps to present data to a web browser.



**Benefits of Bokeh:**

* Bokeh allows you to build complex statistical plots quickly and through simple commands
* Bokeh provides you output in various medium like html, notebook and server
* We can also embed Bokeh visualization to flask and django app
* Bokeh can transform visualization written in other libraries like matplotlib, seaborn, ggplot
* Bokeh has flexibility for applying interaction, layouts and different styling option to visualization

**Seaborn**

Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

eaborn is a library for making statistical graphics in Python. It is built on top of matplotlib and closely integrated with pandas data structures.

Here is some of the functionality that seaborn offers:

* A dataset-oriented API for examining relationships between multiple variables
* Specialized support for using categorical variables to show observations or aggregate statistics
* Options for visualizing univariate or bivariate distributions and for comparing them between subsets of data
* Automatic estimation and plotting of linear regression models for different kinds dependent variables
* Convenient views onto the overall structure of complex datasets
* High-level abstractions for structuring multi-plot grids that let you easily build complex visualizations
* Concise control over matplotlib figure styling with several built-in themes
* Tools for choosing color palettes that faithfully reveal patterns in your data

**Plotly**

Plotly, also known by its URL, Plot.ly, is a technical computing company headquartered in Montreal, Quebec, that develops online data analytics and visualization tools. Plotly provides online graphing, analytics, and statistics tools for individuals and collaboration, as well as scientific graphing libraries for Python, R, MATLAB, Perl, Julia, Arduino, and REST.

Plotly was built using Python and the Django framework, with a front end using JavaScript and the visualization library D3.js, HTML and CSS. Files are hosted on Amazon S3

***Initialization for Offline Plotting***

* Plotly Offline allows you to create graphs offline and save them locally. There are also two methods for plotting offline: plotly.offline.plot() and plotly.offline.iplot().
* Use plotly.offline.plot() to create and standalone HTML that is saved locally and opened inside your web browser.
* Use plotly.offline.iplot() when working offline in a Jupyter Notebook to display the plot in the notebook.

***Initialization for Online Plotting***

Plotly provides a web-service for hosting graphs! Create a free account to get started. Graphs are saved inside your online Plotly account and you control the privacy. Public hosting is free, for private hosting, check out our paid plans.

**CONCLUSION**

We have demonstrated that we can predict some aspects of power variability of single-home power consumption using basic regression models. We tested data with ARIMA, SARIMA, ARIMAX model. We obtained good results in early morning hours, and larger errors at times when the inhabitants may be more active. Furthermore, we found that all the models had similar results, indicating that the errors may be due to a lack of clean structure in our data despite our efforts to separate temporally. We attribute this to the inherent unpredictability of a single family’s actions and consequent power usage. In the future, to improve our single-home model we ideally would have an increased number of relevant features and the data to account for different classes, including running separate regression models on each hour of the day, weekdays vs. weekends, and different seasons.

**REFERENCES**

Data: http://traces.cs.umass.edu/index.php/Smart/Smart

<https://www.analyticsvidhya.com/blog/2016/02/time-series-forecasting-codes-python/>

<https://machinelearningmastery.com/arima-for-time-series-forecasting-with-python/>

<https://www.digitalocean.com/community/tutorials/a-guide-to-time-series-forecasting-with-arima-in-python-3>