

MACHINE LEARNING MODELS FOR ELECTRICITY CONSUMPTION FORECASTING

Submitted in partial fulfillment of the requirements for the award of Bachelor
of Engineering Degree in Computer Science and Engineering

By

**GURRALA SAIRAM REDDY (38110185)
KARRI HARSHA VINAY (38110236)**



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING

**SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited with Grade "A" by NAAC

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119, TAMILNADU**

MAY-2022



SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **GURRALA SAIRAM REDDY (Reg. No: 38110185)**, **KARRI HARSHA VINAY (Reg. No: 38110236)** who carried out the project entitled "**ELECTRICITY LOAD FORECASTING**" under my supervision from June 2021 to November 2021.

Internal Guide

Dr. M. MAHESWARI M.E., Ph.D.,

Head of the Department

Dr.L.LAKSHMANANM.E., Ph.D.,

Submitted for Viva voce Examination held on_____

Internal Examiner

External Examiner

DECLARATION

I, **GURRALA SAIRAM REDDY** hereby declare that the project report entitled **ELECTRICITY LOAD FORECASTING** done by me under the guidance of **Dr.M. MaheswariM.E., Ph.D.** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering.

DATE:25-02-2022

PLACE:CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph.D, Dean**, School of Computing, **Dr. S. Vigneshwari, M.E., Ph.D. and Dr. L. Lakshmanan, M.E., Ph.D., Heads of the Department of Computer Science and Engineering** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. M. Maheswari M.E., Ph.D.**, for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

Energy consumption prediction is an activity that helps energy supply firms to adjust to different behavior. Knowing the behavior of their customers to adapt their rates to consumption or knowing the intervals in which it will cause a greater demand for energy and having planned the adaptation of supply chains are just a few of the actions that organizations can do.

Forecasting is a technique for predicting future values of a time series based on present and previous values. Forecasting future consumptions based on various data and information available about customer behavior is known as load forecasting. Short-term load forecasting is useful for estimating load flows and avoiding overloads.

Timely implementations of such decisions lead to improvement of network reliability and to reduced occurrences of equipment failures or blackouts. The review made allowed us to observe that the data set using the Linear Regression and SVR has the best results provided.

Both schemes used similar parameters for training and testing simulations. After 10-time cross training validation and five averaged repeated runs with random permutation per data splitting, the proposed classifier shows better computation speed and higher classification accuracy than the conventional method.

However, when the number of its desired levels increases, its prediction accuracy seems to decrease and approaches the accuracy of the conventional method. The developed energy level prediction, which is computationally inexpensive and has a good classification performance, can serve as an alternative forecasting scheme.

	TABLE OF CONTENTS	
CHAPTER No.	TITLE	PAGE No
	ABSTRACT	V
	LIST OF FIGURES	viii
1	INTRODUCTION	1
	1.1 Overview	2
	1.2 Objective	2
2	LITERATURE SURVEY	3
3	AIM AND SCOPE OF PRESENT INVESTIGATION	6
	3.1 Aim	6
	3.2 SCOPE OF PRESENT INVESTIGATION	6
4	EXPERIMENTAL OR MATERIALS AND METHODS; ALGORITHMS USED	7
	4.1 Machine Learning Overview	7
	4.2 STEPS TO INSTALL PYTHON	7

	4.3 PYTHON LIBRARIES NEEDED	8
	4.4 MODULES IMPLEMENTED	9
	4.4.1 Data Preprocessing	10
	4.4.2 Data Validation/ Data Cleaning /Data Preparing process	11
	4.4.3 Exploration Data Analysis Of Visualization	12
	4.4.4 Comparing Algorithm with prediction in the form of best accuracy result	13
	4.5 ALGORITHMS AND TECHNIQUES	16
	4.5.1 Algorithm Explanation	16
	4.5.2 Used Python Packages	17
	4.5.3 Linear Regression	17
	4.5.4 Random Forest Classifier	18
	4.6SYSTEM ARCHITECTURE	20
	4.7 ALGORITHM DEPLOYMENT	20
5	RESULTS AND DISCUSSION,	25
6	SUMMARY AND CONCLUSION	26
	6.1 SUMMARY	27
	6.2 CONCLUSION	27
	REFERENCES	27
	APPENDIX	
	A. SCREENSHOTS	29
	B. SOURCE CODE	33

	C. PLAGIARISM REPORT	42
--	-----------------------------	-----------

LIST OF FIGURES

FIGURE NO.	FIGURE NAME	PAGE NO.
1	System Architecture	20
2	Methodology of ML Algorithms	23
3	Energy Demand as Per Requirement	29
4	Average energy Demand	29
5	Target Analysis	30
6	Distribution based on Years	30
7	Distribution on Weekdays	31
8	Ahead Forecasting for a Period	31
9	Actual VS Predicted (Demand)	32
10	30 days ahead Forecasting	32

CHAPTER-1

INTRODUCTION

Energy load forecasting is becoming one of the latest trends due to advancements in energy and power systems and management. As a result, emerging techniques in the field of artificial intelligence (AI) have recently come into play. One particular study reviews various prediction techniques for energy consumption prediction in buildings. Energy regression models are studied in Machine learning (ML) techniques such as artificial neural networks (ANNs) and support vector machines (SVMs) are employed to predict energy consumption and draw energy-saving mechanisms. Another study reviews the use of a probabilistic approach in load forecasting. Other studies analyze the effectiveness of AI tools applied in smart grid and commercial buildings.

Most of the studied AI tools focus primarily on actual value forecasting. For example, consumers may not know whether the next-hour forecasted load value based on these models is either high or low. The conventional way is to categorize the forecasted value into reasonable levels, such as low, average, or high, which consumers can understand. This study proposes an alternative method that can be applied when estimated levels instead of actual values are already sufficient for a load forecasting application.

Short-term forecasting of energy consumption load uses the most important historical data ranging from a few hours even up to a number of weeks before the forecasted day. Recently, short-term load forecasting research studies employed advanced machine learning such as artificial neural networks, fuzzy logic algorithms, and wavelet transform techniques integrated into a neural network system, and an extreme learning machine.

Studies on short-term forecasting also cover various settings such as residential, non-residential, and micro-grid buildings. In residential houses, typical energy consumption forecasting is driven by data generated from humidity and temperature sensors. Occupant behavior assessment can also predict building consumption.

A number of research papers study short- and long-term energy consumption both in residential and small commercial establishments. The emergence of algorithms and an increasing computational capability have encouraged the development of different prediction models. Stochastic models can reliably predict the energy consumption of buildings and identify areas of possible energy waste. Standard engineering regression and a statistical approach still have good prediction results.

A combination with genetic programming is also effective. Various machine learning tools such as support vector machines and neural network algorithms provide an acceptable energy prediction performance. Random forest outperforms other widely used classifiers such as artificial neural networks and support vector machines in energy consumption forecasting.

1.1 Overview:

Accurate forecasts of electricity demand inform investment decisions about power generation and supporting network infrastructure. Of major interest to energy policymakers, power utilities, and private investors alike, forecasts are also essential for development professionals

1.2 Objective:

Electricity plays an important role in the world. Forecasting the electricity to areas which contains high cost electrical machinery is most important aspect. If the electricity forecasting is below the limit it may cause load delays and short circuits which damages the machinery. In order to overcome this issue we developed a machine learning model to predict load usage based on present and past data so that we can forecast the power to areas using the predictions to prevent loss to machinery and several power accidents.

CHAPTER-2

LITERATURE SURVEY

[1] N. Kim (2018) considered Energy load statistics from the past as well as the future are critical for energy-related services and research. A strategy for predicting short-term electricity use is proposed in this research. The method suggests a Long-Short-Term-Memory network, which uses a sequence of previous consumption patterns to anticipate electricity usage for the month ahead. An experiment with an actual dataset is conducted for the results of the performance analysis indicates that suggested method performs well, with a prediction accuracy of roughly 82.5 %. With more training time and purposeful hyperparameter adjustment, the test accuracy can be enhanced.

[2] N. X. Tung (2020) stated that Short-term electrical load forecasting is critical for a country's operating energy. Forecasting accuracy is critical for electricity generation, yet it is dependent on the weather. The new models propose analysing the impact of temperature on anticipating electricity loads. For temperature-sensitive short-term power load forecasting, three approaches are analysed in depth. The developed models are then used to forecast power load in Vietnam.

[3] Goswami(2018) told that Energy is one of the most important commodities in present world. Temperature, seasons, humidity, day of the week, special holidays, and other factors all affect energy usage, and the most efficient use of energy necessitates the delivery of consistent, high-quality power. Tariffs play an important part in this. Tariffs can be used to make electricity trading more favourable during peak demand hours. As a result, we must plan carefully and judiciously for daily load profiles and precise peak load demand projections. Forecasting loads especially high load forecasting, can be done by a variety of ways. Auto Regression Technique has been proven to be a verified tool for predicting future data.

[4] S K Jha (2019) stated the accuracy of load demand forecasting is critical in the power sector's economic operation and planning. However, developing more precise load forecasting methods remains critical. In this paper, the Artificial Neural Network (ANN) with Levenberg-Marquardt (LM) training algorithm was used to forecast electricity load demand using three different Multi-Step-Ahead Load Forecasting (MSALF) strategies: Direct Strategy (DS), Recursive Strategy (RS), and DirRec Strategy (Direct-Recursive Strategy or DRS). The performance of three different MSALF techniques was evaluated. The DRS outperforms the DS and RS in terms of performance.

[5] M. Ali(2019)To reduce the huge disparity between electricity generation and consumption. The six-month load and price data from the New York load operator team is utilised to forecast in this article. For feature selection, the Decision Tree technique was employed, and the Recursive Feature Elimination is used for feature extraction. Redundancy is removed from selected features using the REF approach. For predicting, two classifiers are utilised after feature extraction. Support Vector Machine is one model, and KNN is another. These classifiers feature a variety of parameters, some of which have default values. This project involves load and pricing forecasts for the coming week. For load forecasting, modified SVM has an accuracy of 89% and modified KNN has an accuracy of 89 % is achieved for load forecasting.

[6] S. Katruksa (2019) presented a paper on Load Forecasting types based on process to medium energy supplying for Bangkok and Thailand electricity authorities, using geographic information system . This strategy can be used to increase the Metropolitan Electricity Authority's electricity load efficiency. The geographical forecast plays an important part in expanding regions of energy supplying, as investment decisions for new substations and power system services planning. Data seems for showing predicted of the MEA areas' density point is proportional to electricity demand.

[7] A.Inteha (2021) proposed that Load forecasting is an important aspect of the planning and operation of power systems. Among the various forecasting methodologies, hybrid models perform better. For post day load forecasting, a model of Genetic Algorithm and Gated Recurrent Unit network is proposed in this paper as STLTF. Using Genetic Algorithm, paper presents a method for opting time delays and neurons count. When we considered this with past work with same topic, the result is a significant improvement we got 0.83 MAPE and 68.46 RMSE values.

[8] H. Ur-Rehman (2019) considered DCNN for prediction of power load price estimation. The model presented in this paper has two types of process: Feature engineering and estimation. Feature Extraction and Feature Selection are two aspects of feature engineering. This research presents a method for FS that combines Random Forest and Recursive Feature Elimination. The designed method is useful for removing feature redundancy and reduce dimensionality. CNN and SVC are compared to DCNN performance.

CHAPTER-3

AIM AND SCOPE OF PRESENT INVESTIGATION

3.1 AIM:

We are going to create a Random forest model for Electricity Load Forecasting with more efficiency. We are going to use both regression and Random forest model. Improve the accuracy of the model by training it.

We have taken a real world scenario here. We have collected dataset of Spain electricity market for this project and taken the practical values to train the model.

3.2 SCOPE OF THE PRESENT INVESTIGATION:

Energy consumers may not know whether their next-hour forecasted load is either high or low based on the actual value predicted from their historical data. A conventional method of level prediction with a pattern recognition approach was performed by first predicting the actual numerical values using typical pattern-based regression models, then classifying them into pattern levels (e.g., low, average, and high).

A proposed prediction with pattern recognition scheme was developed to directly predict the desired levels using simpler classifier models without undergoing regression. The proposed pattern recognition classifier was compared to its regression method using a similar algorithm applied to a real-world energy dataset. A random forest (RF) algorithm which outperformed other widely used machine learning (ML) techniques in previous research was used in both methods.

Both schemes used similar parameters for training and testing simulations. After 10-time cross training validation and five averaged repeated runs with random permutation per data splitting, the proposed classifier shows better computation speed and higher classification accuracy than the conventional method. However, when the number of its desired levels increases, its prediction accuracy seems to decrease and approaches the accuracy of the conventional method.

CHAPTER 4

EXPERIMENTAL OR MATERIALS AND METHODS; ALGORITHMS USED

4.1 Machine Learning Overview:

Machine learning is a field of study that looks at using computational algorithms to turn empirical data into usable models. The machine learning field grew out of traditional statistics and artificial intelligences communities. Through their business processes immense amounts of data have been and will be collected. This has provided an opportunity to re-invigorate the statistical and computational approaches to autogenerate useful models from data. Machine learning algorithms can be used to (a) gather understanding of the cyber phenomenon that produced the data under study, (b) abstract the understanding of underlying phenomena in the form of a model, (c) predict future values of a phenomena using the above-generated model, and (d) detect anomalous behavior exhibited by a phenomenon under observation.

4.2 Steps to Download & Install Python:

Download the Latest version of the **Python** executable installer

(<https://www.python.org/downloads/>). Watch the PIP list where pip is the package installer for python. Now upgrade the pip and setup tools using the command

Pip install --upgrade pip and Pip install --upgrade setuptools

IDE INSTALLATION FOR PYTHON

IDE stands for Integrated Development Environment. It is a GUI(Graphical User Interface) where programmers write their code and produce the final products.

Best IDE is Pycharm. So download the pycharm new version and install the software (<https://www.jetbrains.com/pycharm/download/>)

PYTHON FILE CREATION

GO To FILE MENU > CREATE > NEW > PYTHON FILE > (Name Your Python

File as "HOUSE PRICE PPREDICTION" > SAVE

4.3 PYTHON LIBRARIES NEEDED

There are many libraries in python. In those, we only use a few main libraries needed

4.3.1 NUMPY LIBRARY

NumPy is an open-source numerical Python library. NumPy contains a multi-dimensional array and matrix data structures. It can be utilized to perform a number of mathematical operations on arrays such as trigonometric, statistical, and algebraic routines like mean, mode, standard deviation, etc...,

Installation- (<https://numpy.org/install/>)

Here we mainly use an array, to find the mean and standard deviation.

4.3.2 PANDAS LIBRARY

Pandas is a high-level data manipulation tool developed by Wes McKinney. It is built on the NumPy package and its key data structure is called the DataFrame. DataFrames allow you to store and manipulate tabular data in rows of observations and columns of variables. There are several ways to create a DataFrame.

Installation- (https://pandas.pydata.org/getting_started.html)

pip install PANDAS

Here we use pandas for reading the csv files, for grouping the data, for cleaning the data using some operations

4.3.3 MATPLOTLIB LIBRARY

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible. Use interactive figures that can zoom, pan, update, visualize etc.,

Installation- (<https://matplotlib.org/users/installing.html>)

pip install Matplotlib

Here we use pyplot mainly for plotting graphs.

matplotlib.pyplot is a collection of functions that make matplotlib work like MATLAB.

Each pyplot function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc.

4.3.4 SEABRONLIBRARY

Seaborn package was developed based on the Matplotlib library. It is used to create more attractive and informative statistical graphics. While seaborn is a different package, it can also be used to develop the attractiveness of matplotlib graphics.

Installation-(<https://seaborn.pydata.org/installing.html>)

pip install Seaborn

4.3.5 SCIKIT-LEARN LIBRARY

Scikit-learn is a free machine learning library for the Python. It features various algorithms like support vector machine, random forests, regression and k-neighbours, and it also supports Python numerical and scientific libraries like NumPy and SciPy Installation-(<https://scikit-learn.org/stable/install.html>)

Pip install Scikit-Learn

Here use scikit-learn's regression methods for prediction purpose.

4.4 Modules Implementation:

A modular design reduces complexity, facilitates change (a critical aspect of software maintainability), and results in easier implementation by encouraging parallel development of different part of system. Software with effective modularity is easier to develop because function may be compartmentalized and interfaces are simplified. Software architecture embodies modularity that is software is divided into separately named and addressable components called modules that are integrated to satisfy problem requirements.

Modularity is the single attribute of software that allows a program to be intellectually manageable. The five important criteria that enable us to evaluate a design method with respect to its ability to define an effective modular design are: Modular

decomposability, Modular Comps ability, Modular Understand ability, Modular continuity, Modular Protection.

The following are the modules of the project, which is planned in aid to complete the project with respect to the proposed system, while overcoming existing system and also providing the support for the future enhancement.

4.4.1 Data Pre-processing

Validation techniques in machine learning are used to get the error rate of the Machine Learning (ML) model, which can be considered as close to the true error rate of the dataset. If the data volume is large enough to be representative of the population, you may not need the validation techniques. However, in real-world scenarios, to work with samples of data that may not be a true representative of the population of given dataset. To finding the missing value, duplicate value and description of data type whether it is float variable or integer. The sample of data used to provide an unbiased evaluation of a model fit on the training dataset while tuning model hyper parameters.

The evaluation becomes more biased as skill on the validation dataset is incorporated into the model configuration. The validation set is used to evaluate a given model, but this is for frequent evaluation. It as machine learning engineers use this data to fine-tune the model hyper parameters. Data collection, data analysis, and the process of addressing data content, quality, and structure can add up to a time-consuming to-do list. During the process of data identification, it helps to understand your data and its properties; this knowledge will help you choose which algorithm to use to build your model.

A number of different data cleaning tasks using Python Pandas library and specifically, it focus on probably the biggest data cleaning task, missing values and it able to more quickly clean data. It wants to spend lesstime cleaning data, and more time exploring and modeling.

Some of these sources are just simple random mistakes. Other times, there can be a deeper reason why data is missing. It's important to understand these different types of missing data from a statistics point of view. The type of missing data will influence how to deal with filling in the missing values and to detect missing values, and do some basic imputation and detailed statistical approach for dealing with missing data. Before, joint into code, it's important to understand the sources of missing data. Here are some

typical reasons why data is missing:

- User forgot to fill in a field.
- Data was lost while transferring manually from a legacy database.
- There was a programming error.
- Users chose not to fill out a field tied to their beliefs about how the results would be used or interpreted.

Variable identification with Uni-variate, Bi-variate and Multi-variate analysis:

- import libraries for access and functional purpose and read the given dataset
- General Properties of Analyzing the given dataset
- Display the given dataset in the form of data frame
- show columns
- shape of the data frame
- To describe the data frame
- Checking data type and information about dataset
- Checking for duplicate data
- Checking Missing values of data frame
- Checking unique values of data frame
- Checking count values of data frame
- Rename and drop the given data frame
- To specify the type of values
- To create extra columns

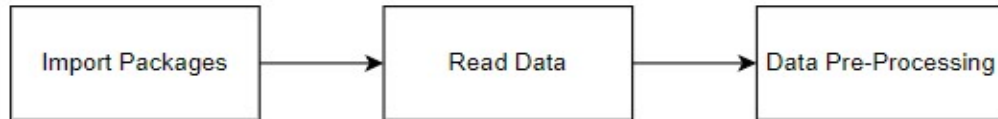
4.4.2 Data Validation/ Cleaning/Preparing Process

Importing the library packages with loading the given dataset. To analyzing the variable identification by data shape, data type and evaluating the missing values, duplicate values. A validation dataset is a sample of data held back from training your model that is used to give an estimate of model skill while tuning model's and procedures that you can use to make the best use of validation and test datasets when evaluating your models. Data cleaning / preparing by rename the given dataset and drop the column etc. to analyze the uni-variate, bi-variate and multi-variate process.

The steps and techniques for data cleaning will vary from dataset to dataset. The primary goal of data cleaning is to detect and remove errors and anomalies to increase

the value of data in analytics and decision making.

MODULE DIAGRAM



GIVEN INPUT EXPECT OUTPUT

input: data

output: removing noisy data

4.4.3 Exploration data analysis of visualization

Data visualization is an important skill in applied statistics and machine learning.

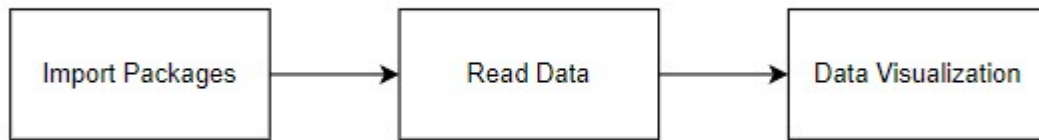
Statistics does indeed focus on quantitative descriptions and estimations of data. Data visualization provides an important suite of tools for gaining a qualitative understanding.

This can be helpful when exploring and getting to know a dataset and can help with identifying patterns, corrupt data, outliers, and much more. With a little domain knowledge, data visualizations can be used to express and demonstrate key relationships in plots and charts that are more visceral and stakeholders than measures of association or significance. Data visualization and exploratory data analysis are whole fields themselves and it will recommend a deeper dive into some the books mentioned at the end.

Sometimes data does not make sense until it can look at in a visual form, such as with charts and plots. Being able to quickly visualize of data samples and others is an important skill both in applied statistics and in applied machine learning. It will discover the many types of plots that you will need to know when visualizing data in Python and how to use them to better understand your own data.

- How to chart time series data with line plots and categorical quantities with bar charts.
- How to summarize data distributions with histograms and box plots.

MODULE DIAGRAM



GIVEN INPUT EXPECT OUTPUT

input: data

output: visualized data

4.4.4 Comparing Algorithm with prediction in the form of best accuracy result

It is important to compare the performance of multiple different machine learning algorithms consistently and it will discover to create a test harness to compare multiple different machine learning algorithms in Python with scikit-learn. It can use this test harness as a template on your own machine learning problems and add more and different algorithms to compare. Each model will have different performance characteristics. Using resampling methods like cross validation, you can get an estimate for how accurate each model may be on unseen data. It needs to be able to use these estimates to choose one or two best models from the suite of models that you have created. When have a new dataset, it is a good idea to visualize the data using different techniques in order to look at the data from different perspectives. The same idea applies to model selection. You should use a number of different ways of looking at the estimated accuracy of your machine learning algorithms in order to choose the one or two to finalize. A way to do this is to use different visualization methods to show the average accuracy, variance, and other properties of the distribution of model accuracies.

In the next section, you will discover exactly how you can do that in Python with scikit-learn. The key to a fair comparison of machine learning algorithms is ensuring that each algorithm is evaluated in the same way on the same data and it can achieve this by forcing each algorithm to be evaluated on a consistent test harness.

Pre-processing refers to the transformations applied to our data before feeding it to the

algorithm. Data Preprocessing is a technique that is used to convert the raw data into a clean data set. In other words, whenever the data is gathered from different sources it is collected in raw format which is not feasible for the analysis. To achieve better results from the applied model in the Machine Learning method of the data has to be in a proper manner. Some specified Machine Learning model needs information in a specified format, for example, the Random Forest algorithm does not support null values. Therefore, to execute random forest algorithm null values have to be managed from the original raw data set. And another aspect is that the data set should be formatted in such a way that more than one Machine Learning and Deep Learning algorithm are executed in a given dataset.

In the example below these 2 different algorithms are compared:

- Linear Regression
- Random Forest

The K-fold cross-validation procedure is used to evaluate each algorithm, importantly configured with the same random seed to ensure that the same splits to the training data are performed and that each algorithm is evaluated in precisely the same way. Before that comparing algorithm, Building a Machine Learning Model using install Scikit-Learn libraries. In this library package have to done preprocessing, linear model with logistic regression method, cross-validating by KFold method, ensemble with random forest method and tree with decision tree classifier. Additionally, splitting the train set and test set. To predicting the result by comparing accuracy.

False Positives (FP): A person who will pay predicted as defaulter. When actual class is no and predicted class is yes. E.g. if actual class says this passenger did not survive but predicted class tells you that this passenger will survive.

False Negatives (FN): A person who default predicted as payer. When actual class is yes but predicted class in no. E.g. if actual class value indicates that this passenger survived and predicted class tells you that passenger will die.

True Positives (TP): A person who will not pay predicted as defaulter. These are the correctly predicted positive values which means that the value of actual class is yes and the value of predicted class is also yes. E.g. if actual class value indicates that this passenger survived and predicted class tells you the same thing.

True Negatives (TN): A person who default predicted as payer. These are the correctly predicted negative values which means that the value of actual class is no and value of predicted class is also no. E.g. if actual class says this passenger did not survive and predicted class tells you the same thing.

Prediction result by accuracy:

Logistic regression algorithm also uses a linear equation with independent predictors to predict a value. The predicted value can be anywhere between negative infinity to positive infinity. It needs the output of the algorithm to be classified variable data. Higher accuracy predicting result is logistic regression model by comparing the best accuracy.

True Positive Rate(TPR) = $TP / (TP + FN)$

False Positive Rate(FPR) = $FP / (FP + TN)$

Accuracy: The Proportion of the total number of predictions that is correct otherwise overall how often the model predicts correctly defaulters and non-defaulters.

Accuracy calculation:

Accuracy = $(TP + TN) / (TP + TN + FP + FN)$

Accuracy is the most intuitive performance measure and it is simply a ratio of correctly predicted observation to the total observations. One may think that, if we have high accuracy then our model is best. Yes, accuracy is a great measure but only when you have symmetric datasets where values of false positive and false negatives are almost same.

Precision: The proportion of positive predictions that are actually correct.

Precision = $TP / (TP + FP)$

Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. The question that this metric answer is of all passengers that labeled as survived, how many actually survived? High precision relates to the low false positive rate. We have got 0.788 precision which is pretty good.

Recall: The proportion of positive observed values correctly predicted. (The proportion of actual defaulters that the model will correctly predict)

$$\text{Recall} = \text{TP} / (\text{TP} + \text{FN})$$

Recall(Sensitivity) - Recall is the ratio of correctly predicted positive observations to the all observations in actual class - yes.

F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if you have an uneven class distribution. Accuracy works best if false positives and false negatives have similar cost. If the cost of false positives and false negatives are very different, it's better to look at both Precision and Recall.

General Formula:

$$\text{F-Measure} = 2\text{TP} / (2\text{TP} + \text{FP} + \text{FN})$$

F1-Score Formula:

$$\text{F1 Score} = 2 * (\text{Recall} * \text{Precision}) / (\text{Recall} + \text{Precision})$$

4.5 ALGORITHM AND TECHNIQUES

4.5.1 Algorithm Explanation

In machine learning and statistics, classification is a supervised learning approach in which the computer program learns from the data input given to it and then uses this learning to classify new observation. This data set may simply be bi-class (like identifying whether the person is male or female or that the mail is spam or non-spam) or it may be multi-class too. Some examples of classification problems are: speech recognition, handwriting recognition, bio metric identification, document classification etc. In Supervised Learning, algorithms learn from labeled data. After understanding the data, the algorithm determines which label should be given to new data based on pattern and associating the patterns to the unlabeled new data.

4.5.2 Used Python Packages:

sklearn:

- In python, sklearn is a machine learning package which include a lot of ML algorithms.
- Here, we are using some of its modules like train_test_split, DecisionTreeClassifier or Logistic Regression and accuracy_score.

NumPy:

- It is a numeric python module which provides fast maths functions for calculations.
- It is used to read data in numpy arrays and for manipulation purpose.

Pandas:

- Used to read and write different files.
- Data manipulation can be done easily with data frames.

Matplotlib:

- Data visualization is a useful way to help with identify the patterns from given dataset.
- Data manipulation can be done easily with data frames.

4.5.3 Linear Regression

Linear regression is one of the easiest and most popular Machine Learning algorithms. It is a statistical method that is used for predictive analysis. Linear regression makes predictions for continuous/real or numeric variables such as sales, salary, age, product price, etc.

Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (x) variables, hence called as linear regression. Since linear regression shows the linear relationship, which means it finds how the value of the dependent variable is changing according to the value of the independent variable.

Types of Linear Regression

Linear regression can be further divided into two types of the algorithm:

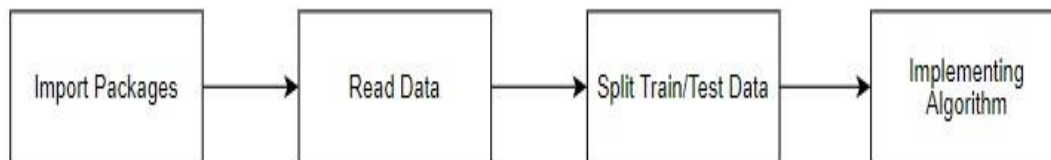
- Simple Linear Regression:

If a single independent variable is used to predict the value of a numerical dependent variable, then such a Linear Regression algorithm is called Simple Linear Regression.

- Multiple Linear regression:

If more than one independent variable is used to predict the value of a numerical dependent variable, then such a Linear Regression algorithm is called Multiple Linear Regression.

MODULE DIAGRAM



GIVEN INPUT EXPECT OUTPUT

input: data

output: getting accuracy

4.5.4 Random Forest Classifier

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks, that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of over fitting to their training set. Random forest is a type of supervised machine learning algorithm based on ensemble learning.

Ensemble learning is a type of learning where you join different types of algorithms or same algorithm multiple times to form a more powerful prediction model. The random forest algorithm combines multiple algorithm of the same type i.e. multiple decision *trees*, resulting in a *forest of trees*, hence the name "Random Forest". The

random forest algorithm can be used for both regression and classification tasks.

The following are the basic steps involved in performing the random forest algorithm:

- Pick N random records from the dataset.
- Build a decision tree based on these N records.
- Choose the number of trees you want in your algorithm and repeat steps 1 and 2.

In case of a regression problem, for a new record, each tree in the forest predicts a value for Y (output). The final value can be calculated by taking the average of all the values predicted by all the trees in forest. Or, in case of a classification problem, each tree in the forest predicts the category to which the new record belongs. Finally, the new record is assigned to the category that wins the majority vote.

GIVEN INPUT EXPECT OUTPUT

input: data

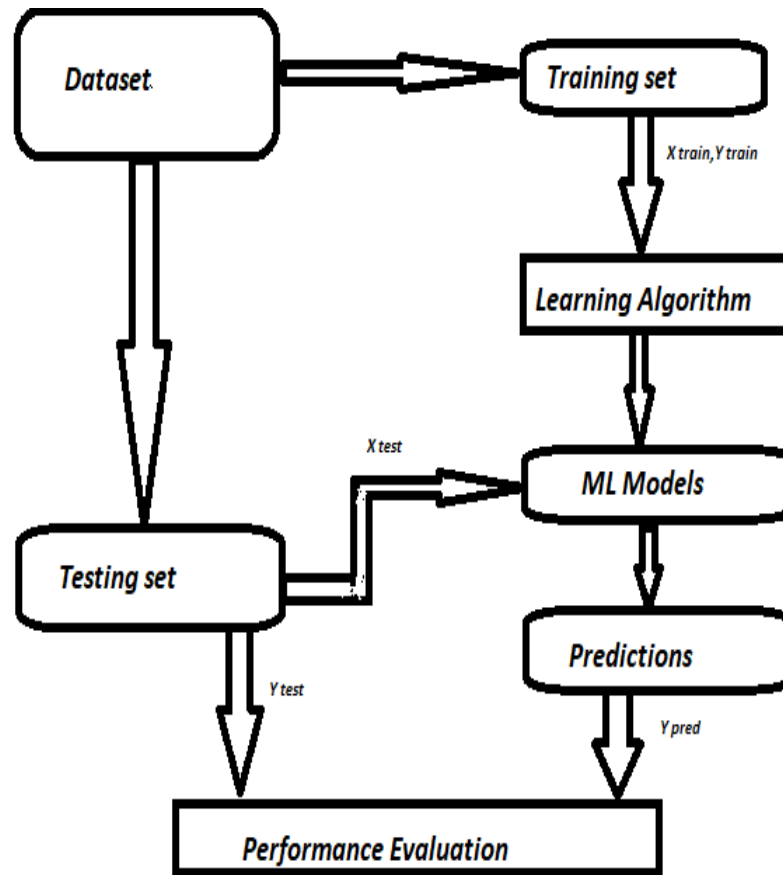
output: getting accuracy

Machine Learning Pipeline and Implementation:

The two main stages of an ML algorithm are the training and testing phases. First, the training phase creates the ML model using a training dataset based on the chosen ML classifier models. The three most commonly known ML models, namely, artificial neural network (ANN), support vector machine (SVM), and random forest (RF), are employed. The performance validation of the training stage guarantees the general performance of the classifier model and is used to avoid the overfitting issue. Then, verification is performed on the trained model in the testing stage using the testing dataset as input to the trained classifier. This testing dataset is the other partitioned data from the original dataset; therefore, it has identical characteristics to the training dataset. The original dataset is partitioned into 70% and 30% for training and testing, respectively.

Performance metrics are used to evaluate both stages. By comparing the performance of both training and testing, any overfitting issue can be determined. It occurs when the training performance is relatively higher than the testing performance.

4.6 SYSTEM ARCHITECTURE



4.7 Deployment of Algorithms

Algorithm 1 presents the implemented program ML classifier similar to the pseudocode of [21]. A k number of times of cross-validation was performed. In this study, 10-time cross-validation was used. This cycle was repeated for another 10 times. The overall average performance and random data shuffling were taken. This verification helped avoid any overfitting issue. This was performed by taking the loss function of random forest present the difference between the training and testing results. For the evaluation of the conventional method, which is a regression-type problem, the root-mean-square error (RMSE) function in Equation (1) is used. F-score, classification accuracy, and confusion matrix are the metrics used for the proposed classifier. F-score accuracy metrics in Equation (2) weigh the significance of both precision and recall performance of the ML model. Precision measures its positive predictive value, whereas

recall measures its sensitivity.

Algorithm 1. Machine Learning Implementation

Initialization In the initialization stage, data pre-processing is performed such as the loading and shuffle-splitting of the dataset into feature X and predictor y, and the importation of the necessary python-based libraries.

Repeat n times the training and testing of the model

for i=1:n

Shuffle-splitting of dataset into training, validation, and testing datasets

k-time Training Cross-Validation

for j=1:k-time

Training of the model using an ML algorithm using the training dataset

Performance evaluation of the trained model using the validation dataset

Testing the model

Testing of the trained model using the testing dataset Performance evaluation of the tested model

Display Performance Results

Compute classification accuracy and F-score

Compute classification confusion matrix

Another measure, classification accuracy in (3), was also taken. This metric is the percent of correctly classified levels over the total number of taken levels.

$$\text{RMSE} = \sqrt{\sum_{i=1}^n \frac{(w^T x(i) - y(i))^2}{n}}$$

$$\text{F score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

$$\text{Accuracy} = \frac{\text{No. of correctly classified energy levels}}{\text{Total number of classified energy levels}}$$

Finally, a confusion matrix normalized between zero and one helps visualize the

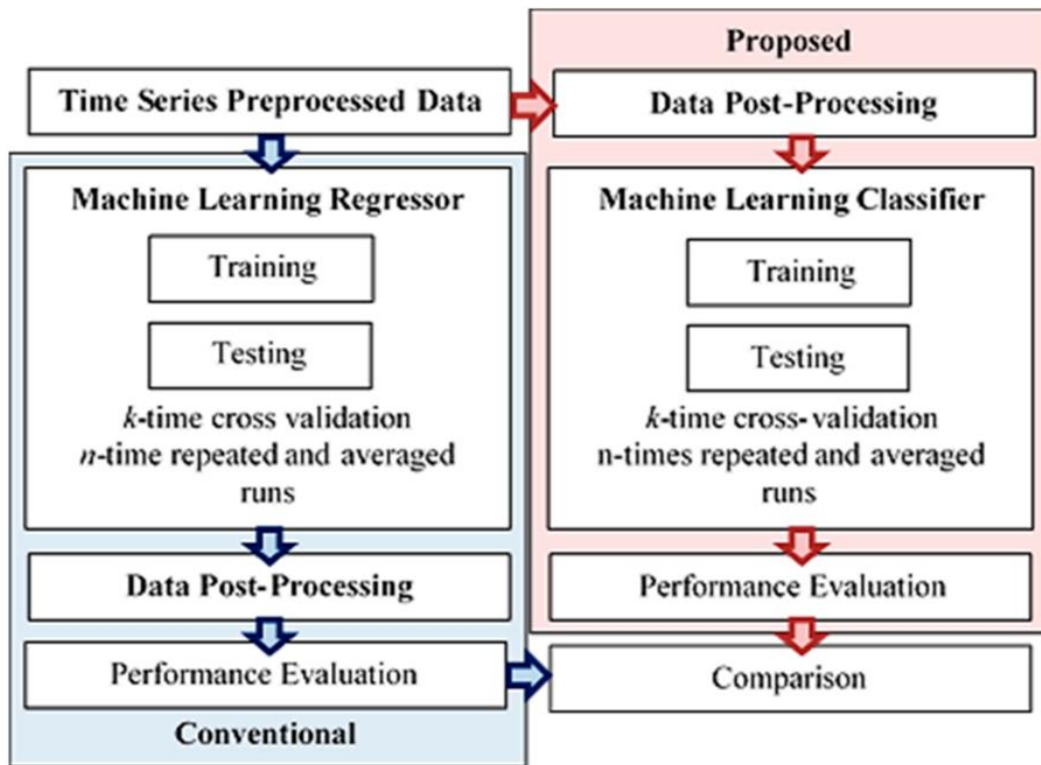
classification performance of the model. scikit-learn in [22] is an open-source platform that provides Python libraries and support. This was used to implement the three ML model classifiers—ANN, SVM, and RF

Algorithm 2: Random Forest Classifier

A decision tree was used as the predictive model. The model predicts from the subject observations up to the model decision on which the subject's target value is based. The subject observations are also called branches while subject's target values are also known as leaves. Bagging is a technique of estimated prediction by reducing its variance which is suitable for decision trees. For its regression application, a recursive fit of a similar regression tree was performed to produce bootstrap-sampled versions of training data taking its mean value. For classification, a predicted class was chosen by the majority vote from each committee of trees. Random forest (RF) is a modified bagging that produces a large collection of independent trees and averages their results. Each of the trees generated from bagging is identically distributed, making it hard to improve other than achieving variance reduction. RF performs the tree-growing process by random input variable selection, thereby improving bagging by the correlation reduction between trees without an excessive variance increase. Energy Data Processing This section presents the implementation of the proposed machine learning classifier using a real energy dataset. The dataset was processed according to the state-of-the-art data class interval method. It was then compared with the conventional forecasting technique. A 12-month energy dataset of from a large hypermarket was used in this study. An hourly energy consumption collected via a smart metering device and hourly temperature records were retrieved from meteorological sensors respectively. During the sunny days of the year between June to September, the energy load consumption is relatively high due to the prevalent use of air conditioning units in response to the high temperature. Symmetry 2019, 11, x FOR PEER REVIEW 4 of 9 other than achieving variance reduction. RF performs the tree-growing process by random input variable selection, thereby improving bagging by the correlation reduction between trees without an excessive variance increase. Energy Data Processing This section presents the implementation of the proposed machine learning classifier using a real energy dataset. The dataset was processed according to the state-of-the-art data class interval method. It was then compared with the conventional forecasting technique. A 12-month energy dataset from a large hypermarket was used in this study. Hourly energy consumption was collected via a smart metering device and hourly temperature records were retrieved from meteorological sensors. During the sunny days of the year between June to September, the energy load consumption is relatively high due to the prevalent use of air

conditioning units in response to the high temperature.

The conventional method and the new proposed scheme of predicting energy levels are shown in Figure. The conventional method of energy level prediction is performed by first predicting the actual numerical values using typical regression models and then classifying them into consumer-preferred levels (e.g., low, average, and high). Since the regression model becomes computationally as it's model becomes more complicated, a proposed prediction scheme was developed to directly predict the desired levels using simpler classifier models without undergoing regression.



Methodology and comparison of the conventional and the proposed time series machine learning classifiers (source: author's own conception).

In the proposed scheme, the energy consumption values are classified into ordinal bins using a general percentile statistic. Ordinal bin partitioning has an approximately equal number of data points as shown. For example, five bins representing five energy levels (very low, low, mid, high, very high) can be created using five even percentile ranges of the energy consumption data resulting in the [0.174, 0.366), [0.366, 0.634), [0.634, 0.782), [0.782, 0.874), [0.874, 1.36] energy value

ranges, respectively.

For prediction implementation, energy levels were converted into their respective ordinal values (1, 2, 3, 4, 5) for the machine learning implementation. The dataset contains three energy level cases—three, five, and seven classes, as shown in Table 1. The modified dataset can be found in [26]. With these, three prediction cases were conducted using a machine learning random forest classifier.

Three n-level cases for real dataset

n-Level Cases	Interval	Data Points
3-level	[0,0.525)	2927
	[0.525,0.807)	2917
	[0.807,1.36]	2940
5-level	[0,0.366)	1755
	[0.366,0.634)	1756
	[0.634,0.7816)	1759
	[0.7816,0.874)	1752
	[0.874, 1.36]	1762
7-level	[0, 0.3366)	1255
	[0.3366, 0.427)	1249
	[0.427, 0.675)	1260
	[0.675, 0.771)	1236
	[0.771, 0.827)	1271
	[0.827, 0.935)	1257
	[0.935, 1.36]	1256

CHAPTER-5

RESULTS AND DISCUSSION, PERFORMANCE ANALYSIS

The results of the prior proposed random forest classifier with the previously preprocessed energy data are presented in this section. The results were compared to those of a traditional forecasting classification algorithm.

To optimize the hyperparameters of both the traditional and proposed random forest classifiers, a brute-force simulation was used. Both the traditional and proposed classifiers have different training and testing loss functions. Based on three-level instances, it can be seen that the suggested RF classifier has a lower train–test difference than the conventional classifier, indicating a better model performance to avoid overfitting. Furthermore, when the parameters increase, the recommended method tends to converge with a train–test loss function difference of less than 2%, whereas the usual method deviates.

CHAPTER-6

SUMMARY AND CONCLUSIONS

6.1 Summary:

This project objective is to predict the Electricity Load Forecasting to a particular areas. So this Electricity Load Forecasting system will reduce the risks of Short circuits and Overloads which causes a more damage to Electronic things.

6.2 Conclusion:

Thus we can finally conclude that, A constructed random forest classifier was used to predict energy levels. Rather than employing the traditional method of regression-based load forecasting, the created classifier preprocessed numerical-valued input into levels and then forecasted them using a simpler classification process. With a smaller number of energy levels, both classifiers perform better.

Based on classification accuracy factors, the proposed classifier surpasses the conventional classifier. However, in terms of classification accuracy, the conventional classifier approaches that of the suggested method, although at the cost of calculation time. Not just for energy consumption forecasting, but also for other related applications, the proposed random forest classifier is an alternative to regression-based challenges. A constructed random forest classifier was used to predict energy levels.

REFERENCES:

1. A. Ganguly, K. Goswami and A. Kumar Sil, "WANN and ANN based Urban Load Forecasting for Peak Load Management," 2020 IEEE Calcutta Conference (CALCON), 2020, pp. 402-406, doi: 10.1109/CALCON49167.2020.9106520.
2. A. Inteha and Nahid-Al-Masood, "A GRU-GA Hybrid Model Based Technique for Short Term Electrical Load Forecasting," *2021 2nd International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST)*, 2021, pp. 515-519, doi: 10.1109/ICREST51555.2021.9331156.
3. A. Singh and K. B. Sahay, "Short-Term Demand Forecasting by Using ANN Algorithms," *2018 International Electrical Engineering Congress (iEECON)*, 2018, pp. 1-4, doi: 10.1109/IEECON.2018.8712265
4. G. Zhang and J. Guo, "A Novel Method for Hourly Electricity Demand Forecasting," in *IEEE Transactions on Power Systems*, vol. 35, no. 2, pp. 1351-1363, March 2020, doi: 10.1109/TPWRS.2019.2941277.
5. H. Ur-Rehman, S. Mujeeb and N. Javaid, "DCNN and LDA-RF-RFE Based Short-Term Electricity Load and Price Forecasting," *2019 International Conference on Frontiers of Information Technology (FIT)*, 2019, pp. 71-715, doi: 10.1109/FIT47737.2019.00023.
6. K. Goswami, A. Ganguly and A. K. Sil, "Day Ahead Forecasting and Peak Load Management using Multivariate Auto Regression Technique," *2018 IEEE Applied Signal Processing Conference (ASPCON)*, 2018, pp. 279-282, doi: 10.1109/ASPCON.2018.8748661
7. K. W. Yu, C. H. Hsu and S. M. Yang, "A Model Integrating ARIMA and ANN with Seasonal and Periodic Characteristics for Forecasting Electricity Load Dynamics in a State," *2019 IEEE 6th International Conference on Energy Smart Systems (ESS)*, 2019, pp. 19-24, doi: 10.1109/ESS.2019.8764179.
8. M. A. Boateng, F. K. Oduro-Gyimah and D. K. Ngala, "Bivariate Copula Modeling of Electricity Load, Case Study of Kwame Nkrumah University of Science and Technology," *2019 International Conference on Computing, Computational*

- Modelling and Applications (ICCMA), 2019, pp. 13-135, doi: 10.1109/ICCMA.2019.00010M. Ali, Z. A. Khan, S. Mujeeb, S. Abbas and N. Javaid, "Short-Term Electricity Price and Load Forecasting using Enhanced Support Vector Machine and K-Nearest Neighbor," *2019 Sixth HCT Information Technology Trends (ITT)*, 2019, pp. 79-83, doi: 10.1109/ITT48889.2019.9075063.
9. N. Kim, M. Kim and J. K. Choi, "LSTM Based Short-term Electricity Consumption Forecast with Daily Load Profile Sequences," *2018 IEEE 7th Global Conference on Consumer Electronics (GCCE)*, 2018, pp. 136-137, doi: 10.1109/GCCE.2018.8574484.
 10. N. X. Tung, N. Q. Dat, T. N. Thang, V. K. Solanki and N. T. N. Anh, "Analysis of temperature-sensitive on short-term electricity load forecasting," *2020 IEEE-HYDCON*, 2020, pp. 1-7, doi: 10.1109/HYDCON48903.2020.9242910.
 11. P. Yi, Z. Jianyong, Y. Yun, Z. Rui, Z. Cheng and S. Tian, "An Electricity Load Forecasting Approach Combining DBN-Based Deep Neural
 12. S. K. Jha, C. L. Dewangan and N. K. Verma, "Multi-Step Load Demand Forecasting Using Neural Network," *2019 20th International Conference on Intelligent System Application to Power Systems (ISAP)*, 2019, pp. 1-6, doi: 10.1109/ISAP48318.2019.9065953.
 13. S. Katruksa and S. Jiriwibhakorn, "Electricity Load Forecasting Based on a Geographic Information System," *2019 5th International Conference on Engineering, Applied Sciences and Technology (ICEAST)*, 2019, pp. 1-4, doi: 10.1109/ICEAST.2019.8802591.
 14. S. Khan, Z. A. Khan, Z. Noshad, S. Javaid and N. Javaid, "Short Term Load and Price Forecasting using Tuned Parameters for K-Nearest Neighbors," *2019 Sixth HCT Information Technology Trends (ITT)*, 2019, pp. 89-93, doi: 10.1109/ITT48889.2019.9075062.
 15. W. Kuo, T. Hsieh, H. Chen, C. Chi and Y. Huang, "A Novel Framework Short-Term Load Forecasting for Micro-grid Energy Management System," *2018 IEEE International Conference on Smart Energy Grid Engineering (SEGE)*, 2018, pp. 279-283, doi: 10.1109/SEGE.2018.8499492.

APPENDIX:

Screen Shots:

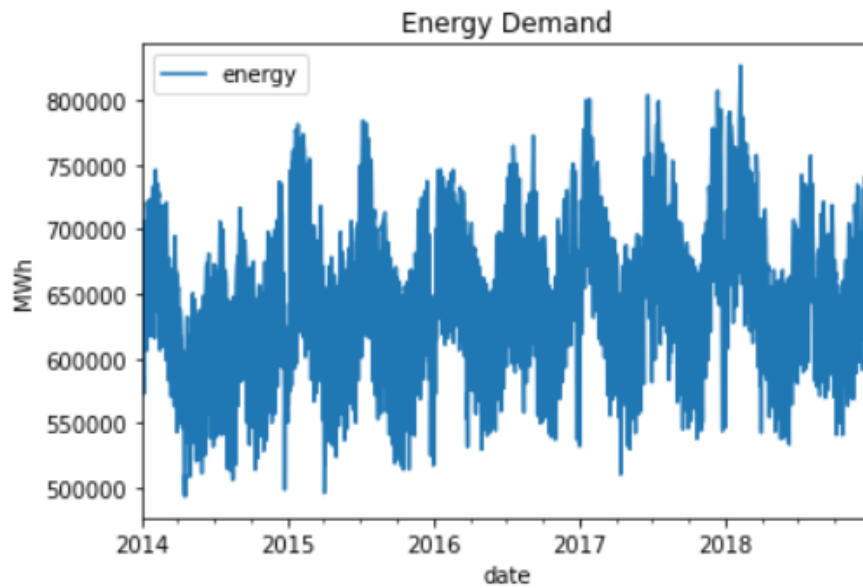


Fig 1: Energy Demand

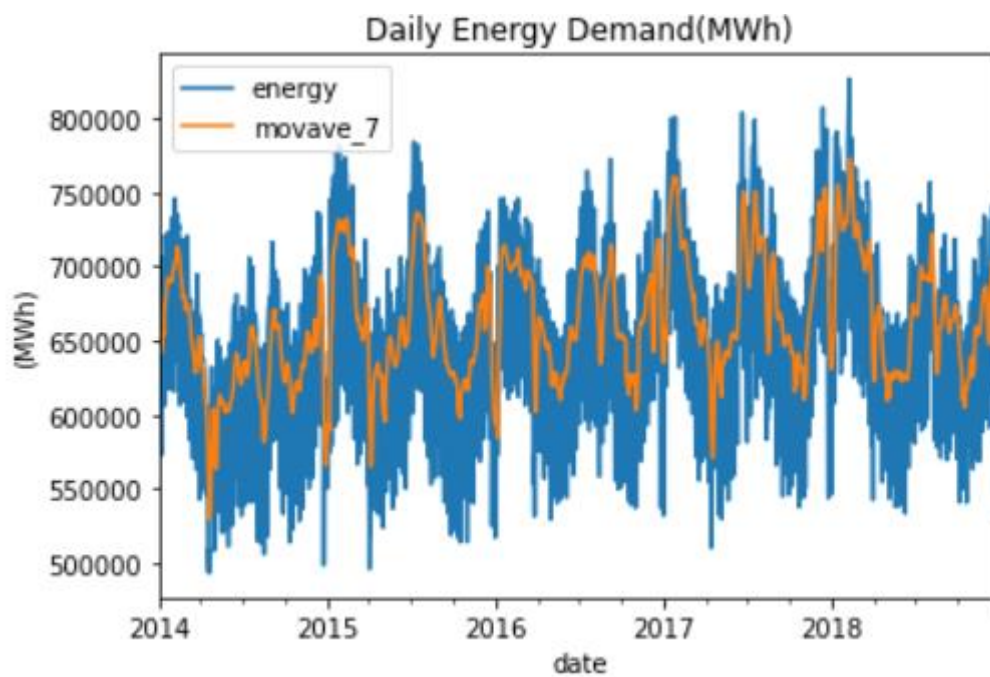


Fig 2: Average Energy Demand

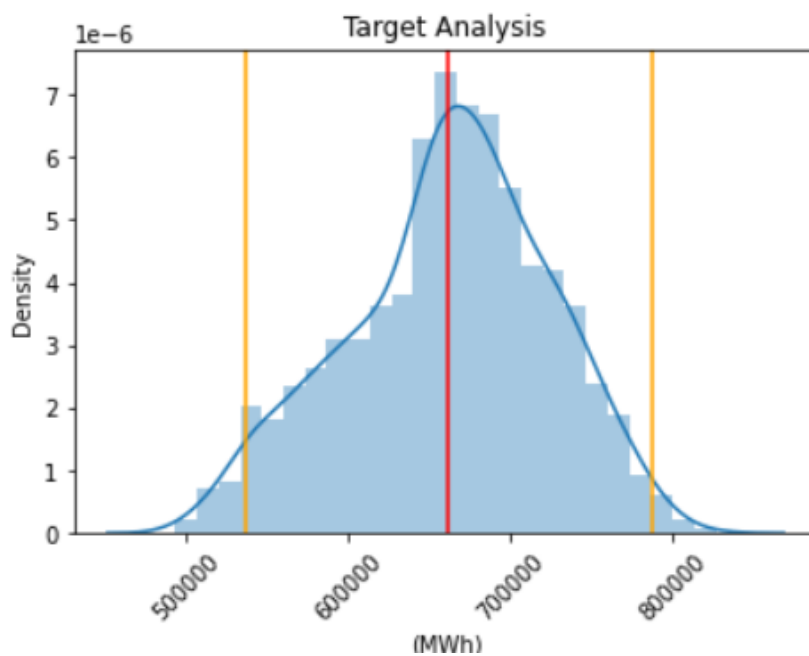


Fig 3:Target Analysis

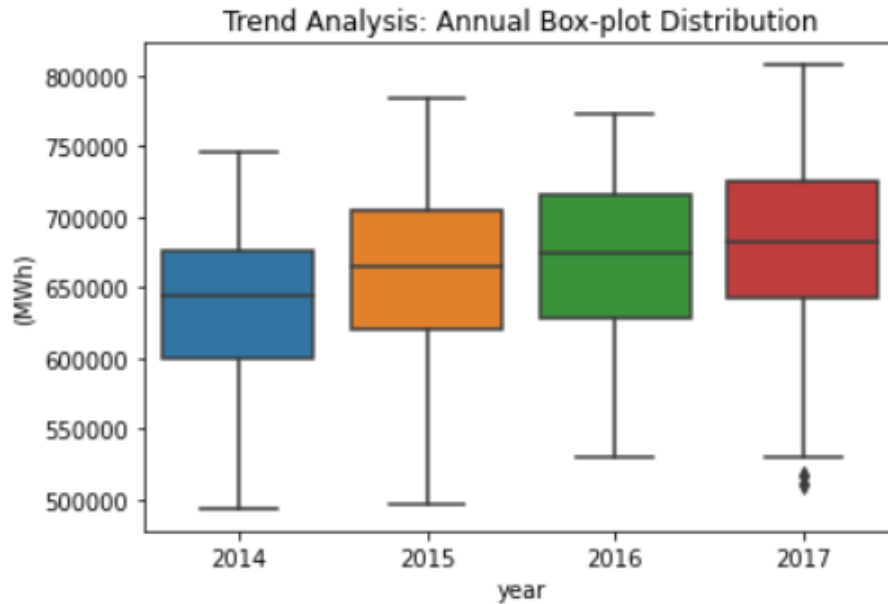


Fig 4:Distribution based on Years

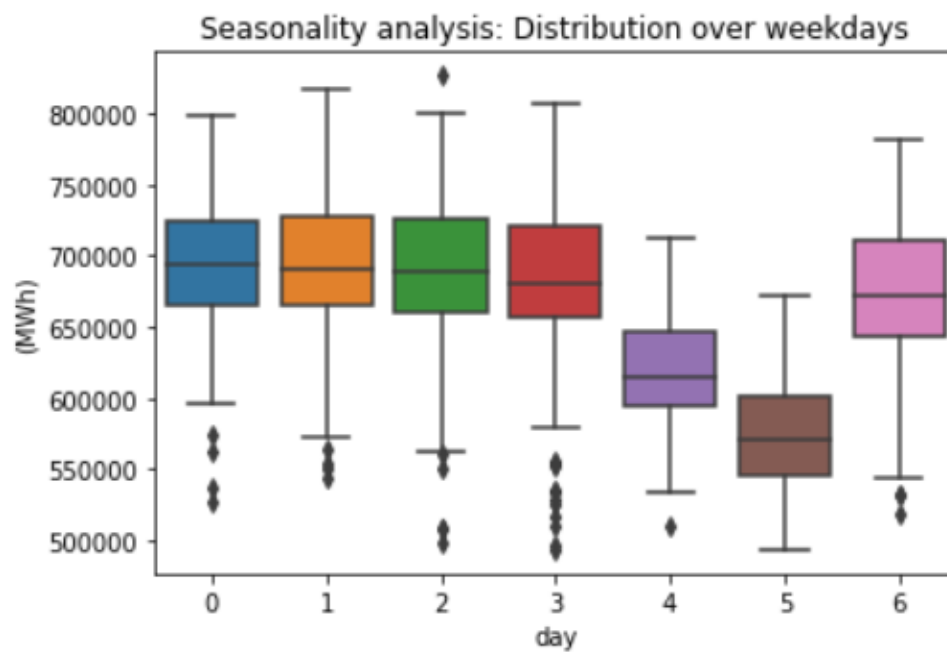


Fig 5: Distribution on Weekdays

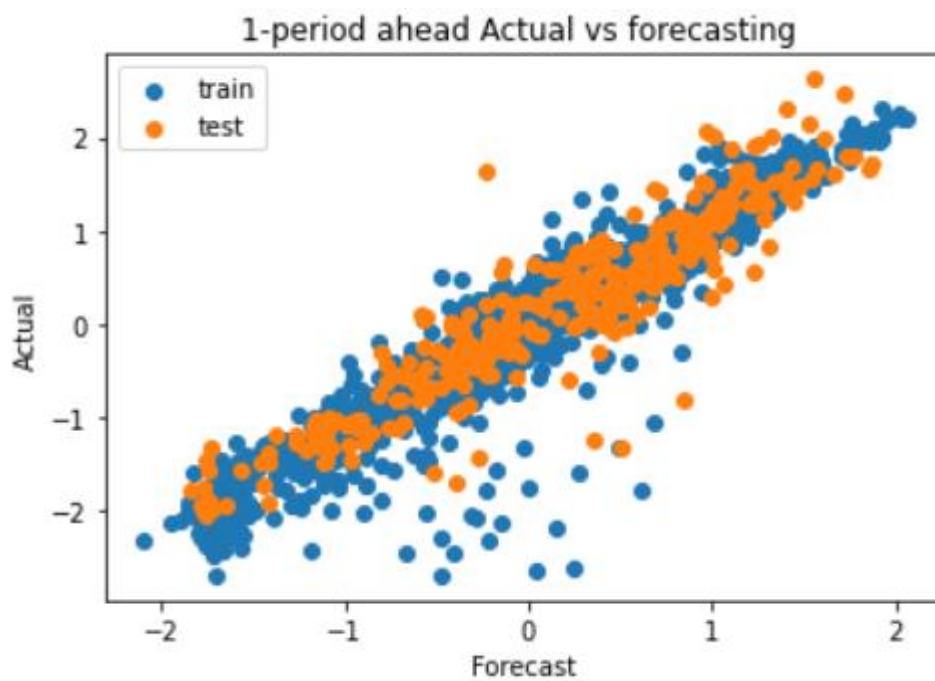


Fig 6: Ahead forecasting for a period

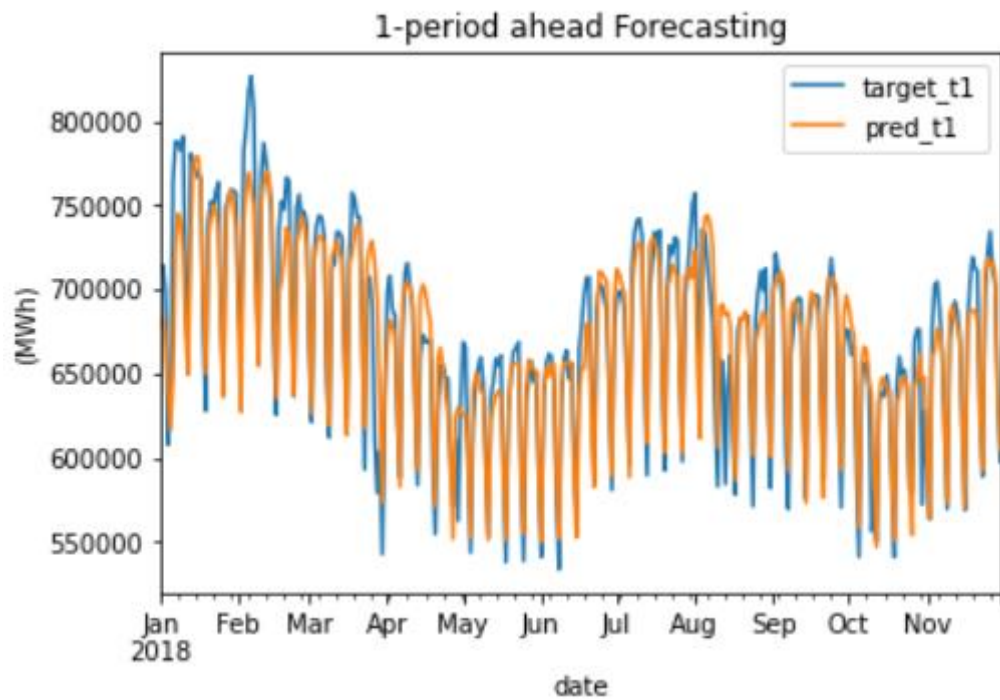


Fig 7: Actual VS Predicted

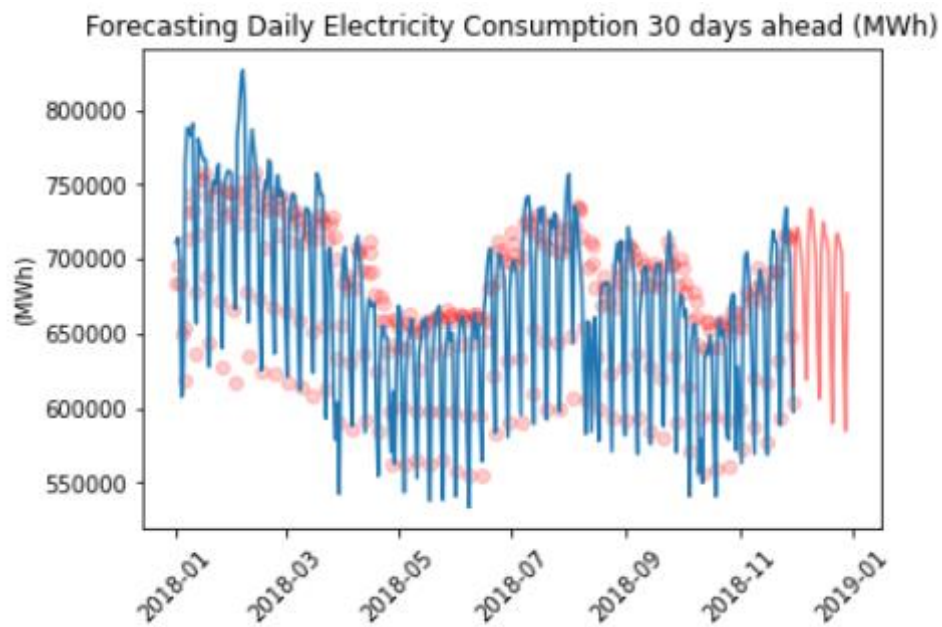


Fig 8: 30 days a head forecasting

B. SOURCE CODE:

```
import pandas as pd
import datetime as dt
import numpy as np
import datetime as dt
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import skew, kurtosis, shapiro

data = pd.read_csv("/content/spain_energy_market.csv", sep=",", parse_dates=["datetime"])
data = data[data["name"]=="Demanda programada PBF total"]
data["date"] = data["datetime"].dt.date
data.set_index("date", inplace=True)
data = data[["value"]]
data = data.asfreq("D")
data = data.rename(columns={"value": "energy"})
data.info()
data[:5]
data.plot(title="Energy Demand")
plt.ylabel("MWh")
plt.show()

len(pd.date_range(start="2014-01-01", end="2018-12-31"))
data["year"] = data.index.year
data["qtr"] = data.index.quarter
data["mon"] = data.index.month
data["week"] = data.index.week
data["day"] = data.index.weekday
data["ix"] = range(0, len(data))
data[["movave_7", "movstd_7"]] = data.energy.rolling(7).agg([np.mean, np.std])
data[["movave_30", "movstd_30"]] = data.energy.rolling(30).agg([np.mean, np.std])
data[["movave_90", "movstd_90"]] = data.energy.rolling(90).agg([np.mean, np.std])
data[["movave_365", "movstd_365"]] = data.energy.rolling(365).agg([np.mean, np.std])

plt.figure(figsize=(20, 16))
data[["energy", "movave_7"]].plot(title="Daily Energy Demand in Spain (MWh)")
plt.ylabel("(MWh)")
plt.show()
mean = np.mean(data.energy.values)
std = np.std(data.energy.values)
```

```

skew = skew(data.energy.values)
ex_kurt = kurtosis(data.energy)
print("Skewness: {} \nKurtosis: {}".format(skew, ex_kurt+3))
def shapiro_test(data, alpha=0.05):
    stat, pval = shapiro(data)
    print("H0: Data was drawn from a Normal Distribution")
    if (pval<alpha):
        print("pval {} is lower than significance level: {}, therefore
null hypothesis is rejected".format(pval, alpha))
    else:
        print("pval {} is higher than significance level: {}, therefore
null hypothesis cannot be rejected".format(pval, alpha))

shapiro_test(data.energy, alpha=0.05)
sns.distplot(data.energy)
plt.title("Target Analysis")
plt.xticks(rotation=45)
plt.xlabel(" (MWh) ")
plt.axvline(x=mean, color='r', linestyle='-'
', label="\mu: {0:.2f}%".format(mean))
plt.axvline(x=mean+2*std, color='orange', linestyle='-')
plt.axvline(x=mean-2*std, color='orange', linestyle='-')
plt.show()
data_rolling = data.energy.rolling(window=90)
data['q10'] = data_rolling.quantile(0.1).to_frame("q10")
data['q50'] = data_rolling.quantile(0.5).to_frame("q50")
data['q90'] = data_rolling.quantile(0.9).to_frame("q90")

data[["q10", "q50", "q90"]].plot(title="Volatility Analysis: 90-
rolling percentiles")
plt.ylabel(" (MWh) ")
plt.show()
data.groupby("qtr")["energy"].std().divide(data.groupby("qtr")["energy"
].mean()).plot(kind="bar")
plt.title("Coefficient of Variation (CV) by qtr")
plt.show()
data.groupby("mon")["energy"].std().divide(data.groupby("mon")["energy"
].mean()).plot(kind="bar")
plt.title("Coefficient of Variation (CV) by month")
plt.show()
data[["movstd_30", "movstd_365"]].plot(title="Heteroscedasticity analys
is")
plt.ylabel(" (MWh) ")
plt.show()

```

```

data[["movave_30", "movave_90"]].plot(title="Seasonal Analysis: Moving
Averages")
plt.ylabel(" (MWh) ")
plt.show()
sns.boxplot(data=data, x="qtr", y="energy")
plt.title("Seasonality analysis: Distribution over quaters")
plt.ylabel(" (MWh) ")
plt.show()
sns.boxplot(data=data, x="day", y="energy")
plt.title("Seasonality analysis: Distribution over weekdays")
plt.ylabel(" (MWh) ")
plt.show()
data_mon = data.energy.resample("M").agg(sum).to_frame("energy")
data_mon["ix"] = range(0, len(data_mon))
data_mon[:5]
sns.regplot(data=data_mon, x="ix", y="energy")
plt.title("Trend analysis: Regression")
plt.ylabel(" (MWh) ")
plt.xlabel("")
plt.show()
sns.boxplot(data=data["2014": "2017"], x="year", y="energy")
plt.title("Trend Analysis: Annual Box-plot Distribution")
plt.ylabel(" (MWh) ")
plt.show()
data["target"] = data.energy.add(-mean).div(std)
sns.distplot(data["target"])
plt.show()
features = []
corr_features=[]
targets = []
tau = 30 #forecasting periods

for t in range(1, tau+1):
    data["target_t" + str(t)] = data.target.shift(-t)
    targets.append("target_t" + str(t))

for t in range(1, 31):
    data["feat_ar" + str(t)] = data.target.shift(t)
    #data["feat_ar" + str(t) + "_lagly"] = data.target.shift(350)
    features.append("feat_ar" + str(t))
    #corr_features.append("feat_ar" + str(t))
    #features.append("feat_ar" + str(t) + "_lagly")

for t in [7, 14, 30]:

```

```

    data[["feat_movave" + str(t), "feat_movstd" + str(t), "feat_movmin"
+ str(t) , "feat_movmax" + str(t)]] = data.energy.rolling(t).agg([np.me
an, np.std, np.max, np.min])
    features.append("feat_movave" + str(t))
    #corr_features.append("feat_movave" + str(t))
    features.append("feat_movstd" + str(t))
    features.append("feat_movmin" + str(t))
    features.append("feat_movmax" + str(t))

months = pd.get_dummies(data.mon,
                        prefix="mon",
                        drop_first=True)

months.index = data.index
data = pd.concat([data, months], axis=1)

days = pd.get_dummies(data.day,
                      prefix="day",
                      drop_first=True)

days.index = data.index
data = pd.concat([data, days], axis=1)

features = features + months.columns.values.tolist() + days.columns.val
ues.tolist()
corr_features = ["feat_ar1", "feat_ar2", "feat_ar3", "feat_ar4", "feat_
ar5", "feat_ar6", "feat_ar7", "feat_movave7", "feat_movave14", "feat_mo
vave30"]
corr = data[["target_t1"] + corr_features].corr()

top5_mostCorrFeats = corr["target_t1"].apply(abs).sort_values(ascending
=False).index.values[:6]

# Plot heatmap of correlation matrix
sns.heatmap(corr, annot=True)
plt.title("Pearson Correlation with 1 period target")
plt.yticks(rotation=0); plt.xticks(rotation=90) # fix ticklabel direct
ions
plt.tight_layout() # fits plot area to the plot, "tightly"
plt.show() # show the plot
sns.pairplot(data=data[top5_mostCorrFeats].dropna(), kind="reg")
plt.title("Most important features Matrix Scatter Plot")
plt.show()
data_feateng = data[features + targets].dropna()
nobs= len(data_feateng)
print("Number of observations: ", nobs)

```

```

X_train = data_feateng.loc["2014":"2017"][features]
y_train = data_feateng.loc["2014":"2017"][targets]

X_test = data_feateng.loc["2018"][features]
y_test = data_feateng.loc["2018"][targets]

n, k = X_train.shape
print("Total number of observations: ", nobs)
print("Train: {}, \nTest: {}".format(X_train.shape, y_train.shape,
                                     X_test.shape, y_test.shap
e))

plt.plot(y_train.index, y_train.target_t1.values, label="train")
plt.plot(y_test.index, y_test.target_t1.values, label="test")
plt.title("Train/Test split")
plt.xticks(rotation=45)
plt.show()

from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error

reg = LinearRegression().fit(X_train, y_train["target_t1"])
p_train = reg.predict(X_train)
p_test = reg.predict(X_test)

RMSE_train = np.sqrt(mean_squared_error(y_train["target_t1"], p_train))
RMSE_test = np.sqrt(mean_squared_error(y_test["target_t1"], p_test))

print("Train RMSE: {}\nTest RMSE: {}".format(RMSE_train, RMSE_test) )
from sklearn.model_selection import TimeSeriesSplit, ParameterGrid

splits = TimeSeriesSplit(n_splits=3, max_train_size=365*2)
for train_index, val_index in splits.split(X_train):
    print("TRAIN:", len(train_index), "TEST:", len(val_index))
    y_train["target_t1"][train_index].plot()
    y_train["target_t1"][val_index].plot()
    plt.show()

from sklearn.ensemble import RandomForestRegressor

splits = TimeSeriesSplit(n_splits=3, max_train_size=365*2)
rfr = RandomForestRegressor()
# Create a dictionary of hyperparameters to search
rfr_grid = {"n_estimators": [500],
            'max_depth': [3, 5, 10, 20, 30],
            'max_features': [4, 8, 16, 32, 59],
            'random_state': [123]}

```

```

rfr_paramGrid = ParameterGrid(rfr_grid)

def TimeSplit_ModBuild(model, paramGrid, splits, X, y):
    from sklearn.model_selection import TimeSeriesSplit
    from sklearn.metrics import mean_squared_error

    #Loop over each time split and for each
    for train_index, val_index in splits.split(X_train):
        _X_train_ = X.iloc[train_index]
        _y_train_ = y.iloc[train_index]
        _X_val_ = X.iloc[val_index]
        _y_val_ = y.iloc[val_index]

        train_scores = []
        val_scores = []
        #models = []

        # Loop through the parameter grid, set the hyperparameters, and
        save the scores
        for g in paramGrid:
            model.set_params(**g)
            model.fit(_X_train_, _y_train_)
            p_train = model.predict(_X_train_)
            p_val = model.predict(_X_val_)
            score_train = np.mean(mean_squared_error(_y_train_, p_train
))

            score_val = np.mean(mean_squared_error(_y_val_, p_val))
            train_scores.append(score_train)
            val_scores.append(score_val)
            #models.append(model)
            best_idx = np.argmin(val_scores)

        print("Best-Fold HyperParams:: ", paramGrid[best_idx])
        print("Best-Fold Train RMSE: ", train_scores[best_idx])
        print("Best-Fold Val RMSE: ", val_scores[best_idx])
        print("\n")

    #Return most recent model
    return train_scores, val_scores, best_idx

CV_rfr_tup = TimeSplit_ModBuild(rfr, rfr_paramGrid, splits, X_train, y_
train["target_t1"])
best_rfr_idx = CV_rfr_tup[2]
best_rfr_grid = rfr_paramGrid[best_rfr_idx]

```

```

best_rfr = RandomForestRegressor().set_params(**best_rfr_grid).\
    fit(X_train.loc["2016":"2017"], y_train.loc["2016":"2017", "target_
t1"])
importances = best_rfr.feature_importances_

# Get the index of importances from greatest importance to least
sorted_index = np.argsort(importances)[::-1]
sorted_index_top = sorted_index[:10]
x = range(len(sorted_index_top))

# Create tick labels
labels = np.array(features)[sorted_index_top]
plt.bar(x, importances[sorted_index_top], tick_label=labels)
plt.title("Feature importance analysis")
# Rotate tick labels to vertical
plt.xticks(rotation=45)
plt.show()
p_train = best_rfr.predict(X_train)
train_resid_1step = y_train["target_t1"]- p_train

p_test = best_rfr.predict(X_test)
test_resid_1step = y_test["target_t1"]- p_test
test_df = y_test[["target_t1"]]*std+mean
test_df["pred_t1"] = p_test*std+mean
test_df["resid_t1"] = test_df["target_t1"].add(-test_df["pred_t1"])
test_df["abs_resid_t1"] = abs(test_df["resid_t1"])
test_df["ape_t1"] = test_df["resid_t1"].div(test_df["target_t1"])

test_MAPE = test_df["ape_t1"].mean()*100
print("1-period ahead forecasting MAPE: ", test_MAPE)
test_df[["target_t1", "pred_t1"]].plot()

plt.title("1-period ahead Forecasting")
plt.ylabel(" (MWh) ")
plt.legend()
plt.show()
plt.scatter(y=y_train["target_t1"],x=p_train, label="train")
plt.scatter(y=y_test["target_t1"],x=p_test, label="test")
plt.title("1-period ahead Actual vs forecasting ")
plt.ylabel("Actual")
plt.xlabel("Forecast")
plt.legend()
plt.show()
test_resid_1step.plot.hist(bins=10, title="Test 1-
step ahead residuals distribution")

```

```

plt.xlabel("Residuals")
plt.show()
test_resid_1step.plot(title="Test 1-step ahead residuals time series")
plt.ylabel("Residuals")
plt.show()
plt.scatter(x=y_test["target_t1"].values, y=test_resid_1step.values)
plt.title("Test 1-step ahead residuals vs Actual values")
plt.ylabel("Residuals")
plt.xlabel("Actual values")
plt.show()
multi_rfr = RandomForestRegressor().set_params(**best_rfr_grid).\
    fit(X_train.loc["2016":"2017"], y_train.loc["2016":"2017"])
p_train = multi_rfr.predict(X_train)
train_resid_1step = y_train- p_train
p_test = multi_rfr.predict(X_test)
test_resid_1step = y_test- p_test
periods = [1, 7, 14, 30]

ytest_df = y_test*std+mean
ptest_df = pd.DataFrame(data=p_test*std+mean, index=test_df.index, columns=["pred_t" + str(i) for i in range(1, 31)])
test_df = pd.concat([ytest_df, ptest_df], axis=1)

test_MAPE = []

for t in periods:
    test_df["resid_t" + str(t)] = test_df["target_t" + str(t)].add(-
test_df["pred_t" + str(t)])
    test_df["abs_resid_t" + str(t)] = abs(test_df["resid_t" + str(t)])
    test_df["ape_t" + str(t)] = test_df["abs_resid_t" + str(t)].div(test_df["target_t" + str(t)])
    test_MAPE.append(round(test_df["ape_t" + str(t)].mean(), 4)*100)

print("MAPE test: ", test_MAPE)
mape_df = pd.DataFrame(index=periods, data={"test_MAPE": test_MAPE})
mape_df.plot(kind="bar", legend=False)
plt.title("Mean Absolute Percent Error in Test")
plt.xlabel("Forecasting Period")
plt.ylabel("%")
plt.xticks(rotation=0)
plt.show()
for t in periods:
    test_df[["target_t" + str(t), "pred_t" + str(t)]]).plot(x="pred_t" + str(t), y="target_t" + str(t) ,kind="scatter")
    plt.title("{}-period(s) ahead forecasting".format(t))

```

```

plt.xlabel("Forecasted (MWh)")
plt.ylabel("Actual values (MWh)")
plt.xticks(rotation=45)
plt.show()
test_df.index = test_df.index.date
forecast_range = pd.date_range(start=np.max(test_df.index.values), periods=tau, freq="D")
len(forecast_range)
forecast = []
for t in range(0, tau):
    #print(-(t+1), (t))
    forecast = p_test[-(t+1):, (t)]*std+mean
test_df["target_t1"].plot()
plt.scatter(x=test_df.index, y=test_df["pred_t1"], c="r", alpha=0.2, label="test preds")
plt.plot(forecast_range, forecast, c="r", alpha=0.5, label="forecasting")
plt.ylabel("(MWh)")
plt.xticks(rotation=45)
plt.title("Forecasting Daily Electricity Consumption (MWh) in Spanish Market (2018)")
plt.show()

```

MACHINE LEARNING MODELS FOR ELECTRICITY CONSUMPTION FORECASTING

Sairam Reddy G., Harsha Vinod K.,
Students, Department of CSE, School
of Computing
Sathyabama Institute of Science and Technology,
Chennai, India
sairamnani24@gmail.com,
princeharsha352@gmail.com

M. Maheshwari, M. Seetha,
Associate Professors, Department of CSE, School
of Computing
Sathyabama Institute of Science and Technology,
Chennai, India

ABSTRACT

Energy consumption prediction is an activity that helps energy supply firms to adjust to different behavior. Knowing the behavior of their customers to adapt their rates to consumption or knowing the intervals in which it will cause a greater demand for energy and having planned the adaptation of supply chains are just a few of the actions that organizations can do. Forecasting is a technique for predicting future values of a time series based on present and previous values. Forecasting future consumptions based on various data and information available about customer behavior is known as load forecasting. Short-term load forecasting is useful for estimating load flows and avoiding overloads. Timely implementations of such decisions lead to improvement of network reliability and to reduced occurrences of equipment failures or blackout. The review made allowed to observe that for the data set using the Linear Regression and SVR has the best results provided.

Keywords: Random Forest, Machine Learning, Energy Forecasting, Linear Regression, Support Vector Regression, K-nearest Neighbors

INTRODUCTION

Forecasting short term load flows can help to estimate load flows and to make decisions that may help to avoid overloading. Appropriate decisions may increase network reliability and reduce equipment failures or blackout. Load forecasting is estimating future consumptions based on various data and information available as per consumer behaviour from past and future.

RELATED WORK

[1] N. Kim (2018) considered Energy load statistics from the past as well as the future are critical for energy-related services and research. A strategy for predicting short-term electricity use is proposed in this research. The method suggests a Long-Short-Term-Memory network, which uses a sequence of previous consumption patterns to anticipate electricity usage for the month ahead. An experiment with an actual dataset is conducted for the results of the performance analysis indicates that suggested method performs well, with a prediction accuracy of roughly 82.5 %. With more training time and purposeful hyperparameter adjustment, the test accuracy can be enhanced.

[2] N. X. Tung (2020) stated that Short-term electrical load forecasting is critical for a country's operating energy. Forecasting accuracy is critical for electricity generation, yet it is dependent on the weather. The new models propose analysing the impact of temperature on anticipating electricity loads. For temperature-sensitive short-term power load forecasting, three approaches are analysed in depth. The developed models are then used to forecast power load in Vietnam.

[3] Goswami(2018) told that Energy is one of the most important commodities in present world. Temperature, seasons, humidity, day of the week, special holidays, and other factors all affect energy usage, and the most efficient use of energy necessitates the delivery of consistent, high-quality power. Tariffs play an important part in this. Tariffs can be used to make electricity trading more favourable during peak demand hours. As a result, we must plan carefully and judiciously for daily load profiles and precise peak load demand projections. Forecasting loads especially high load forecasting, can be done by a variety of ways.

Auto Regression Technique has been proven to be a verified tool for predicting future data.

[4] S K Jha (2019) stated the accuracy of load demand forecasting is critical in the power sector's economic operation and planning. However, developing more precise load forecasting methods remains critical. In this paper, the Artificial Neural Network (ANN) with Levenberg-Marquardt (LM) training algorithm was used to forecast electricity load demand using three different Multi-Step-Ahead Load Forecasting (MSALF) strategies: Direct Strategy (DS), Recursive Strategy (RS), and Direct-Recursive Strategy (Direct-Recursive Strategy or DRS). The performance of three different MSALF techniques was evaluated. The DRS outperforms the DS and RS in terms of performance.

[5] M. Ali(2019)To reduce the huge disparity between electricity generation and consumption. The six-month load and price data from the New York load operator team is utilised to forecast in this article. For feature selection, Decision Tree technique was employed, and the Recursive Feature Elimination is used for feature extraction. Redundancy is removed from selected features using the RFE approach. For predicting, two classifiers are utilised after feature extraction. Support Vector Machine is one model, and KNN is another. These classifiers feature a variety of parameters, some of which have default values. This project involves load and pricing forecasts for the coming week. For load forecasting, modified SVM has an accuracy of 89% and modified KNN has an accuracy of 89 % is achieved for load forecasting.

[6] S. Katruksa (2019) presented a paper on Load Forecasting types based on process to medium energy supplying for Bangkok and Thailand electricity authorities, using geographic information system. This strategy can be used to increase the Metropolitan Electricity Authority's electricity load efficiency. The geographical forecast plays an important part in expanding regions of energy supplying, as investment decisions for new substations and power system services planning. Data seems for showing predicted of the MEA areas' density point is proportional to electricity demand.

[7] A.Inteha (2021) proposed that Load forecasting is an important aspect of the planning and operation

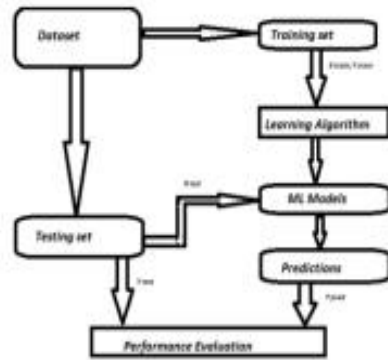
of power systems. Among the various forecasting methodologies, hybrid models perform better. For post day load forecasting, a model of Genetic Algorithm and Gated Recurrent Unit network is proposed in this paper as STLFG. Using Genetic Algorithm, paper presents a method for optimizing time delays and neurons count. When we considered this with past work with same topic, the result is a significant improvement we got 0.83 MAPE and 68.46 RMSE values.

[8] H. Ur-Rehman (2019) considered DCNN for prediction of power load price estimation. The model presented in this paper has two types of process: Feature engineering and estimation. Feature Extraction and Feature Selection are two aspects of feature engineering. This research presents a method for FS that combines Random Forest and Recursive Feature Elimination. The designed method is useful for removing feature redundancy and reduce dimensionality. CNN and SVC are compared to DCNN performance.

PROPOSED SYSTEM

A simple prediction model based on classification method uses previous data to make predictions. We also used the classification method to perform a process of power consumption prediction in this work. Creating a Random Forest model for forecasting electricity demand. Random Forest is a well-known machine learning algorithm that uses the supervised learning method. Random Forest is a classifier that combines a number of decision trees on different subsets of a dataset and averages the results to increase the dataset's predicted exactness.

ARCHITECTURE DIAGRAM



Machine Learning Methodology

This section gives you a glimpse of information about the working of machine learning algorithms and their implementation.

Machine learning implementation

The schooling and checking out stages of an ML set of rules are the 2 maximum crucial steps. The schooling segment first produces the ML version the use of a schooling dataset primarily based totally at the ML classifier fashions which have been chosen. Random forest (RF), the maximum famous system studying version, is used. The schooling stage's overall performance validation guarantees the classifier version's general overall performance and allows to keep away from the overfitting problem. The skilled version is then examined the use of the checking out dataset as enter to the skilled classifier withinside the checking out stage. This checking out dataset is the authentic dataset's different partitioned data, consequently it stocks the identical capabilities because the schooling dataset. For schooling and checking out, the authentic dataset is split into 70% and 30 %, respectively.

Random Forest Classifier

In the search for missing data, the search for false data, and the eradication of weeds are the three main components of coordinated media in this class. Finally, the data cleaning framework was transformed into an effective data mining framework.

Machine learning algorithm

At this stage, the data identified with the examination tree is the choice tree and is extricated from the informational index. The meteorological informational index had ten attributes that pre-owned two elements later on. Because of the idea of cloud information, all qualities are the equivalent, and the vast majority of the qualities that are absent in sunlight-based information are not utilized in the investigation.

Energy Data Preprocessing

The data for this research got from the daily power market demand in MWh in Spain. The case data was collected from 2014- 2018. Datetime, id, name, geoid, geo-name, and value are the six columns of the dataset. For this study, we collected data for 40,212 days.

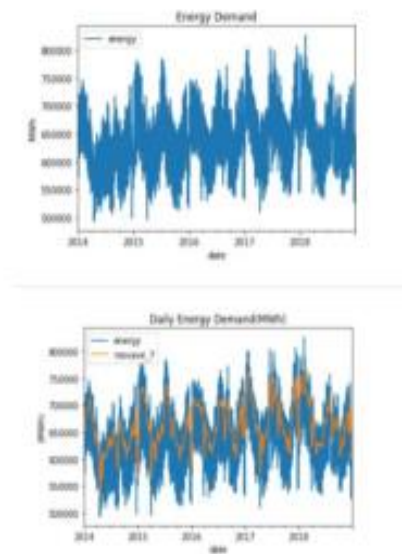


Fig 1&2: Energy Demand per Year

Results and Discussion

The results of the prior proposed random forest classifier with the previously preprocessed energy data are presented in this section. The results were compared to those of a traditional forecasting classification algorithm.

To optimize the hyperparameters of both the traditional and proposed random forest classifiers, a brute-force simulation was used. Both the traditional and proposed classifiers have different training and testing loss functions. Based on three-

level instances, it can be seen that the suggested Random Forest model has less train and test variation than the conventional classifier, indicating a better model performance to avoid overfitting. Furthermore, when the parameters increase, the recommended method tends to converge with a train-test loss function difference of less than 2%, whereas the usual method deviates. Thus we got an accuracy of 84 % and MSE of 0.7 which is too low.



Fig 3: Forecasting 1 period ahead

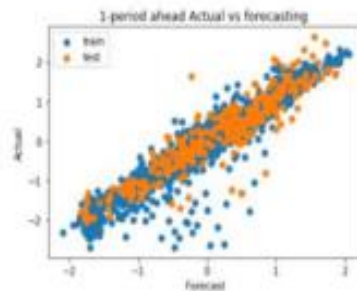


Fig 4: Actual vs Forecasting

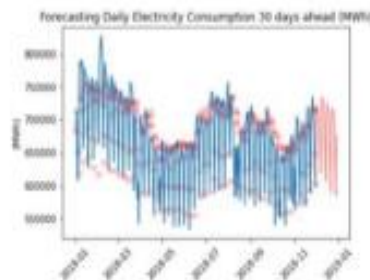


Fig 5: Forecasting 30 days ahead

Conclusion

A random forest classifier model was used to predict energy levels. Rather than employing the traditional method of regression-based load forecasting, the created classifier pre numerical inputs into levels and then forecasted them using a simpler classification process. With a smaller number of energy levels, both classifiers perform better.

Based on the classification accuracy, the proposed classifier outperforms the traditional classifier. With regard to the classification accuracy, however, the conventional classifier is close to the proposed method, albeit at the cost of computing time. The proposed random forest classifier is an alternative to regression-based work not only for energy consumption forecasting, but also for other related applications. A random forest classifier was used to predict energy levels.

REFERENCES

1. N. Kim, M. Kim and J. K. Choi, "LSTM Based Short-term Electricity Consumption Forecast with Daily Load Profile Sequences," 2018 IEEE 7th Global Conference on Consumer Electronics (GCCE), 2018, pp. 136-137, doi: 10.1109/GCCE.2018.8574484.
2. N. X. Tung, N. Q. Dat, T. N. Thang, V. K. Solanki and N. T. N. Anh, "Analysis of temperature-sensitive on short-term electricity load forecasting," 2020 IEEE-HYDICON, 2020, pp. 1-7, doi: 10.1109/HYDICON48903.2020.9242910.
3. K. Goswami, A. Ganguly and A. K. Sil, "Day Ahead Forecasting and Peak Load Management using Multivariate Auto Regression Technique," 2018 IEEE Applied Signal Processing Conference (ASPCON), 2018, pp. 279-282, doi: 10.1109/ASPCON.2018.8748661.
4. P. Yi, Z. Jianyong, Y. Yun, Z. Rui, Z. Cheng and S. Tian, "An Electricity Load Forecasting Approach Combining DBN-Based Deep Neural
5. M. Ali, Z. A. Khan, S. Mujeeb, S. Abbas and N. Javaid, "Short-Term Electricity Price and Load Forecasting using Enhanced Support Vector Machine and K-Nearest Neighbor," 2019 Sixth HCT Information Technology Trends (ITT), 2019, pp. 79-83, doi: 10.1109/ITT48889.2019.9075063.
6. S. Katruksa and S. Jiriwibhakorn, "Electricity Load Forecasting Based on a

- Geographic Information System*," 2019 5th International Conference on Engineering, Applied Sciences and Technology (ICEAST), 2019, pp. 1-4, doi: 10.1109/ICEAST.2019.8802591.
7. A. Inteha and Nahid-Al-Masood, "A GRU-GA Hybrid Model Based Technique for Short Term Electrical Load Forecasting," 2021 2nd International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST), 2021, pp. 515-519, doi: 10.1109/ICREST51555.2021.9331156.
 8. H. Ur-Rehman, S. Mujeeb and N. Javaid, "DCNN and LDA-RF-RFE Based Short-Term Electricity Load and Price Forecasting," 2019 International Conference on Frontiers of Information Technology (FIT), 2019, pp. 71-715, doi: 10.1109/FIT47737.2019.00023.
 9. A. Singh and K. B. Sahay, "Short-Term Demand Forecasting by Using ANN Algorithms," 2018 International Electrical Engineering Congress (IEECON), 2018, pp. 1-4, doi: 10.1109/IEECON.2018.8712265.
 10. W. Kuo, T. Hsieh, H. Chen, C. Chi and Y. Huang, "A Novel Framework Short-Term Load Forecasting for Micro-grid Energy Management System," 2018 IEEE International Conference on Smart Energy Grid Engineering (SEGE), 2018, pp. 279-283, doi: 10.1109/SEGE.2018.8499492.
 11. S. Khan, Z. A. Khan, Z. Noshad, S. Javaid and N. Javaid, "Short Term Load and Price Forecasting using Tuned Parameters for K-Nearest Neighbors," 2019 Sixth HCT Information Technology Trends (ITT), 2019, pp. 89-93, doi: 10.1109/ITT48889.2019.9075062.
 12. G. Zhang and J. Guo, "A Novel Method for Hourly Electricity Demand Forecasting," in *IEEE Transactions on Power Systems*, vol. 35, no. 2, pp. 1351-1363, March 2020, doi: 10.1109/TPWRS.2019.2941277.
 13. S. K. Jha, C. L. Dewangan and N. K. Verma, "Multi-Step Load Demand Forecasting Using Neural Network," 2019 20th International Conference on Intelligent System Application to Power Systems (ISAP), 2019, pp. 1-6, doi: 10.1109/ISAP48318.2019.9065953.

9%

SIMILARITY INDEX

5%

INTERNET SOURCES

7%

PUBLICATIONS

2%

STUDENT PAPERS

PRIMARY SOURCES

1

Sonu Kumar Jha, Chaman Lal Dewangan, Nishchal K. Verma. "Multi-Step Load Demand Forecasting Using Neural Network", 2019 20th International Conference on Intelligent System Application to Power Systems (ISAP), 2019

Publication

2%

2

Submitted to ABES Engineering College

Student Paper

2%

3

"Program", 2019 20th International Conference on Intelligent System Application to Power Systems (ISAP), 2019

Publication

1%

4

www.easychair.org

Internet Source

1%

5

Hammad Ur-Rehman, Sana Mujeeb, Nadeem Javaid. "DCNN and LDA-RF-RFE Based Short-Term Electricity Load and Price Forecasting", 2019 International Conference on Frontiers of

1%

6 ebin.pub
Internet Source

1%

7 iopscience.iop.org
Internet Source

1%

8 Azfar Inteha, Nahid-Al-Masood. "A GRU-GA Hybrid Model Based Technique for Short Term Electrical Load Forecasting", 2021 2nd International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST), 2021
Publication

1%

9 Mughees Ali, Zahoor Ali Khan, Sana Mujeeb, Shahid Abbas, Nadeem Javaid. "Short-Term Electricity Price and Load Forecasting using Enhanced Support Vector Machine and K-Nearest Neighbor", 2019 Sixth HCT Information Technology Trends (ITT), 2019
Publication

1%

10 Chenthur Pandian, S.. "Fuzzy approach for short term load forecasting", Electric Power Systems Research, 200604

<1%