Tuijin Jishu/Journal of Propulsion Technology
ISSN: 1001-4055
Vol. 44 No. 4 (2023)

Forecasting and Prediction of Solar Energy in Solar Photovoltaic Plants

Aatif Mohi Ud Din[1], Vivek Gupta[2]

[1]*M. Tech Scholar, Department of Electrical Engineering, Rayat Bahra University, Chandigarh, India*
[2]*Associate Professor, Department of Electrical Engineering, Rayat Bahra University, Chandigarh, India*

***Abstract*:**

An accurate solar energy projection is essential for a better the degree to which renewable energy is integrated into the functioning of the present power system. Due to the availability of data at previously unheard-of granularities, data-driven algorithms may be utilised to improve solar energy forecasts. In this study, two deep learning algorithms—the k Nearest Neighbor and Random Forest—are presented as the foundational models for the improved, globally applicable stackable ensemble technique. The results the core models are merged with a significant gradient boost technique, improving the precision of solar PV production predictions. Tests were conducted on the suggested model on four separate datasets of solar power in order to provide a full evaluation. To give more information on how the system learns, this study also used the shapely additive explanation framework. Comparing the predicted outcomes allowed for an evaluation of the suggested model's efficacy. of the model to those of individual KNN, RF, and Bagging. The recommended ensemble method offers the most consistency and stability across several case studies and surpasses existing models by 10% to 12% in terms of performance despite weather variations of R^2.

Keywords: *Photovoltaic, Prediction, KNN, RF, Machine learning.*

I. INTRODUCTION

The world is turning to in the hope to minimize the emission of greenhouse gases, other forms of renewable energy as a result of increased energy demand [1]. Even while the power system's high share of renewable energy sources has many advantageous economic and environmental consequences, their unpredictable and intermittent nature puts the reliability and security of the system at danger. Thanks to developments solar energy has been increasingly important in PV technology over the past decade. even if the cost of installing solar panels has grown, the effectiveness of energy conversion and generating electricity has dropped significantly [1]. Over the next ten years, it is projected that solar PV will continue to rise in popularity due to its availability of energy and low cost [2]. In order to integrate green energy supplies into the present power grid in a secure and steady manner, accurate forecasting techniques have become crucial [2]. A number of Optimizing supply consistency and lowering output variability are priorities for stakeholders in the energy sector. Since generating too much or too little electricity often results in fines, a transmission system administrator concentrates on the energy production from PV panels in the near term (0–5 hours) to establish the right balance for the whole grid. Electricity traders are more intrigued by long time horizons, usually day-ahead estimates, as the majority of energy is traded on the day-ahead market. For these operations to be profitable, the ability to predict the fluctuating solar PV panel energy output is required.

As more countries opt to invest steadily more in RES, the usage of solar PV panels is expected to increase. Therefore, accurate techniques for forecasting solar PV energy output will be more crucial. It is difficult to find a solution despite the obvious necessity for accurate and reliable estimations of PV panel energy production. The issue is now being studied while overcoming various obstacles. The weather's inherent fluctuation is one irritation that makes accurate forecasting challenging. With the surge in demand for PV power forecasting, the ability to predict using machine learning (ML) methodologies has lately gained favor compared to conventional time series forecasting models. The usefulness of ML techniques—which are not new—for prediction has been increased by the accessibility of high-quality data and advancements in computing power. The following is a fascinating area to explore when determining the quantity of solar electricity generated: How effective are machine learning algorithms for time series prediction in contrast to proven techniques?

demand. Most significantly, it is shown that when parts of the energy system are electrified (such as the construction of heat pumps), the level of energy self-sufficiency obtained by structures and communities is inadequate [3].

Forecast accuracy is the degree to which the projected value of the generation of PV panels is close to the actual (real) value. Solar PV projections are essential in the evolving energy strategy for controlling congestion, forecasting reserves, managing storage, trading energy amongst buildings, and grid integration [4]. But Data accessibility and use of smart meters has grown new possibilities for improving PV generation forecasting using data-driven artificial learning and deep learning systems. In the recent past, several research papers have attempted to include such data-driven algorithms.

A. Machine learning-based forecasting techniques

The research has suggested a number of forecasting techniques [7] as indicated in Table 1, based on neural networks for solar PV electricity. There is not even one approach that can effectively analyze several case studies, though. Changing between different solar PV plants (datasets) offers different kinds of variability for machine learning models. Different methods with the same weather and energy data can provide various results, as can the same approach with different internal hyperparameter values. The models are said to have volatility concerns, meaning that even a minor change in the reliability of the models and the anticipated values can both be significantly impacted by the input data [8]. Due to uncertainty in the PV module specifications, errors made during data collection, the effectiveness of the system's final product, and long-term effects, the modeling may not generate the same findings on a new dataset [9]

II. OBJECTIVES

The main objective is to contrast various approaches for projecting solar PV panel energy output. This may be achieved by applying proactively understand the link between diverse environmental variables and the electrical energy production of PV systems, use time series and deep learning approaches. Two ML algorithms, Random Forest and K Nearest Neighbour (KNN), are compared to traditional time series techniques using data from installed PV systems. In addition, feature engineering must be taken into account.

III. LITERATURE REVIEW

Hong et alpaper.'s [6] provides an overview of the various forecast utilized for energy forecasting. The paper demonstrates how Machine learning (ML) techniques have been applied to many time series. The paper assesses the techniques employed in the energy predictions contest that served as its foundation (Global Energy Predictions Competition 2014). The key findings are that anticipating load demand has made substantial use of both traditional time series approaches and machine learning. Similar to this, data on electricity price has been analyzed using a variety of time series and ML methodologies. The article also breaks down the methods used to be prepared for the solar PV and wind-generated energy output in the contest. The spectrum of reliable forecasting methods, in relation to load demand and energy costs approaches is relatively limited for the power production of wind turbines and solar PV panels. Although traditional time series have been used quite a bit, ML techniques have been applied quite a little.

IV. METHODOLOGY

The processes necessary for the creation and assessment of the suggested model a

A. *Data description*

The following subsection describes the two instances investigated and the input parameters impacting the modeling of solar PV production.

1. *Case study (I)*

Three solar farms on top of business structures in India provided the information for the cast study (I) at intervals of 15 minutes. Data for the study was acquired from 424 PV panels. The TRINA type solar PV cells have a high peak power output of 275-W. (Wp). On the roofs of buildings, A, B, and C, the panels are mounted in a flat-fix fusion south configuration with distributes of 205, 146, and 73, accordingly. To analyze the variation between the independent and consolidated forecasts, data were gathered for each of the building panels independently and then merged. Figure 2 displays the data for all solar PV systems' energy kWh, or kilowatt-hours, for producing Furthermore, it shows the intergenerational trend using the weakly rolling average approach. Seasonally, solar PV has different generating characteristics. The rolling mean reveals that from May through September, the production is higher in all test cases...

2. *Weather data*

The weather information used in this research was compiled using anticipated values from the Solargis weather data platform [5]. For past, present, and future time periods, the platform offers long-term meteorological data recordings. To calculate the level of uncertainty in the anticipated data, the Solargis data has been tested locally and at specific locations. The same site where the solar panels are was where the meteorological data was collected. The Solargis platform was chosen to obtain the weather forecast data. Additionally, the anticipated weather data should be taken into account for training since the model can adapt whether it is lower or higher than the original data. By doing this, the anticipated generation will neither be over or underestimated.

In this study,. the three climatic factors that were most closely linked to the generation of solar PV were determined to be global straight radiation levels, the temperature, and relative humidity. Based on a literature review, the climatic factors (Wind orientation, Wind acceleration, Dew point climate, and Air pressure) were selected because they have a substantial impact on solar PV production [7]. By providing the models with fewer meteorological input variables, this effort aimed to shorten the deep learning algorithms' calculation time while maintaining accurate findings.

3. *Autoregressive features*

Autoregressive features are constructed through lag observations of time-series data [8]. Because values in time series data frequently correlate with one another at earlier time steps, lags are crucial. The accuracy of the models can be improved by incorporating integrated moving average (arima features, although employing a large amount of input variables might overfit the model during training [9]. Additionally, the inclusion of autoregressive elements constrained the models' prediction horizon steps.. Figure 1 shows the production case study

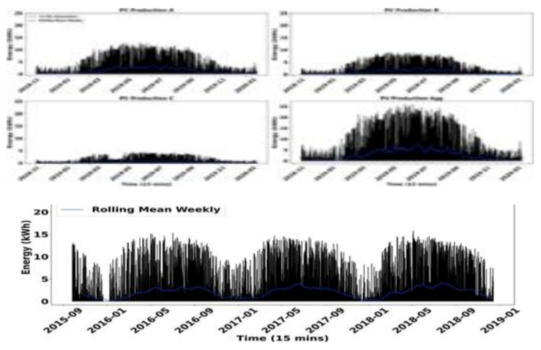


Figure 1 Production case study

The model can only forecast one time if the previous time step (15 or 1 hour) is supplied as an input, move ahead. In the same way, the model is able to predict when preceding day data is used. one day ahead.

Given these restrictions, modeling was done both ways to demonstrate the impact of the autoregressive characteristics on model predicting. Two self-regressive features—point forecasting (past hour for the case investigation II and previous 15 minutes for case study I) and multi-step—are based on the Pearson correlations function.

4. Calendar information

Since solar production data is a time-series with a regularity to time, it is necessary to determine this frequency prior to simulation. To enhance the forecasting outcome, categorical time factors were incorporated taking calendar data into consideration., including the day of the week, day of the month, and month amount. Eight input variables in total were thought of and are listed in Table 2.

B. Data preparation

This section provides a detailed a description of the data pre-processing techniques employed, as well as interpretive analysis of the data sets. that are accessible.

1. Data pre-processing

In order to use the power of machine learning to its fullest possibilities, data pre-processing is an essential stage in the modeling process. To fill in the missing values, search for outliers, and then adjust the features' scale to a range that is comparable, the collected dataset underwent pre-processing. The observations for the case study (I) are more likely to be comparable to those made before or after since the data was obtained throughout a 15-minute period. Therefore, the values that were absent were filled in using linear interpolation. Although the data was quite clean, the most crucial component of the data analysis was how to arrange the data properly. Data missing for a few days at the outset of 2016 in Case Study (II) were interpolated using the average from the prior.. Negative generation figures and the system's max capacity served as the basis for the outlier identification process. In order to search for outcomes of the system's maximum capacity and minus voltages, a visual inspection was conducted. However, no outliers in the dataset were found.

2. . Exploratory analysis

The output energy and global horizontal irradiance (GHI) exhibit a positive link, as demonstrated the distinct variables for both instance papers' coefficient matrix in Table. GHI is the most important component in solar PV projections. Positive correlations exist between each case study and the "Temp" parameter, albeit the link with the "GHI" is greater. However, each case study's output energy is inversely related to the 'RH' parameter...

A complete statistical breakdown provides a list of all the indicators utilized in the case study. The data's mean, median, standard deviation, kurtosis, and bias are all revealed by the statistical evaluation. The center peak is bigger and sharper when the kurtosis values are more than 3, which indicates that the data does not have a normal distribution [4]. The skewness numbers also show how seriously skewed the data is. in favour of the positive due to the rarity of high generation occurrences in the solar PV energy generation. Deep learning models outperform machine learning when data contains skewed dependent variables, Until the skewed values are changed in a suitable way [11].

3. Data scaling

Before being employed in data must be scaled for algorithms based on machine learning that employ the method of gradient descent minimization. To ensure the minima is determined by gradient descent seamlessly and that each step for every attribute updates at the same rate, the data is adjusted before being entered into the models. In order to obtain rapid results for the gradient training process of KNN theories, the training set was scaled using the Min-Max scaler into a range between 0 and 1. The MinMax scaler's limited range reduces the influence of outliers, resulting in a reduced standard deviation. In order to establish the Min-Max scaler,

¼ f fmin

$$\text{fscaled } \overline{\text{fmax fmin}} \qquad (1)$$

Here f_{min} and f_{max} are the characteristics' smallest and greatest values, respectfully.

C. Framework for stacked extension

The prospect of layering deep learning models utilizing a flexible approach that takes into account ensemble architecture is highlighted in this research. Numerous domains, including genomics [5], fault detection [4], and building energy forecasts [2, 4], have effectively employed stacked ensemble learning. The primary goal of stacking is to identify the optimal model combination for a given application. It has been suggested that choosing the right basis algorithms, specifying the leveling of the base models, and choosing the right meta learner are crucial for producing the stacking ensemble model with reduced bias and variance.

1. Training

First, the information on solar productivity is given in the form m*n, where m denotes the quantity of characteristics and output, and n denotes the total amount of measurements. At this point, the data are divided into training and testing groups. Using k-folds cross-validations, KNN and Random Forest are trained on various data folds from the training set. An objective evaluation of the model's efficacy is provided by the k-fold crossvalidation. Seasonal variations in solar PV statistics may be found throughout the year depending on the weather. When contrast to a single hold-out model, K-fold cross validation of the training has reduced variance, which may be important if there are few data pointsThe data was divided into k identical portions using the Kfold algorithm. To forecast each of the test k-folds, the models are trained k times. Five folds of multiple validations are used to validate the models. The models are trained using the first four folds in the initial iteration, then tested using the final fold. The additional folds are utilized for training the models in the iteration that follows, and the fold after that is used for model testing. Prior to the forecasts for all five

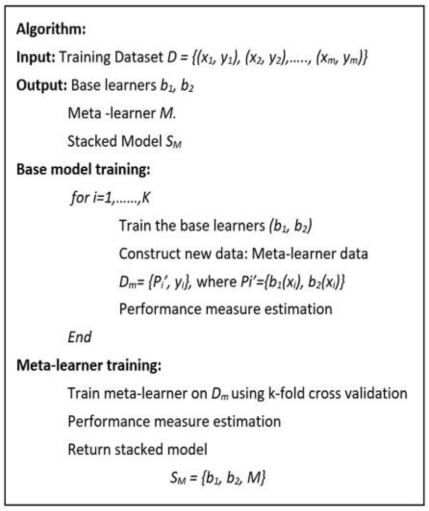
The context makes use of the cross-validation stacking structure to create second-level data for the meta learner. Similar to cross-validation, stacking addresses two crucial problems by producing out-of-sample predictions and capturing a variety of locations where each model works best.

The meta-learner is trained using base model assumptions in order to learn the best way to incorporating the result forecasts. The meta-learning model, which can identify the areas where each base model is doing well and poorly, is significantly responsible for the stacking algorithm's ability to generalise. Random Forest is taught as a meta-learner because of its quicker processing and potential for accurate prediction with a reduced training set in a high-dimensional setting.

Employing 5-fold cross validation upon the training data D as well as the cross-validated projected outputs, every single one of the basic algorithms (b1, b2) is trained.

Table 1 Statistical descriptions of the main indicators for case study (I).

Unit	kWh	kWh	kWh	kWh	C	W/m^2	g/m^3
Mean	1.31	0.91	0.42	2.64	10.10	110.99	80.07
Standard deviation	2.48	1.75	0.80	4.99	6.33	188.79	13.05
Kurtosis	4.13	4.38	5.33	4.14	0.11	3.26	0.29
Skewness	2.21	2.26	2.40	2.21	0.54	1.98	0.94



```
Algorithm:
Input: Training Dataset $D = \{(x_1, y_1), (x_2, y_2),\ldots., (x_m, y_m)\}$
Output: Base learners $b_1, b_2$
        Meta -learner $M$.
        Stacked Model $S_M$
Base model training:
        for i=1,……,K
                Train the base learners $(b_1, b_2)$
                Construct new data: Meta-learner data
                $D_m = \{P_i', y_i\}$, where $P_i' = \{b_1(x_i), b_2(x_i)\}$
                Performance measure estimation
        End
Meta-learner training:
        Train meta-learner on $D_m$ using k-fold cross validation
        Performance measure estimation
        Return stacked model
                $S_M = \{b_1, b_2, M\}$
```

Figure 2 RF algorithm.

2. Testing

Once the underlying models and the meta-learner is fully trained, the test data is used to get the final predictions. To provide a fair evaluation of the developed model, the test data is constructed using data that has not been viewed or utilized during development. Based on the test data, the trained base models provide projections, and the meta-learner uses the findings as inputs to generate the final forecast. The training and testing procedure using the stacked model's data partitioning is given...

3. Performance measures

The coefficient of relationship (R2), root mean square error (RMSE), and the mean absolute error (MAE), three extensively used measurement tools for error in the literature [7], are utilized to assess the accuracy of the base models and ensemble approach. Despite the fact that Mean Absolute Percentage Error (MAPE) has been applied in a number of research, it is still crucial to confirm the validity of the original data before using MAPE. According to Makridakis [8], MAPE exhibits asymptotically asymmetric behavior, where errors above the starting value result in larger absolute percentage deviations than errors beneath the starting value.

D. Development setup

Python programming was used to run the simulation in a free cross-platform complete development platform called Spyder [9]. A python-based open-source kNN tool named Keras [5] was utilised to construct KNN and Random Forest due to its adaptable and modular design. Fast deep neural network testing is provided by Keras, which is built on the Tensorflow [1] framework. Differential programming uses Google's Tensorflow, an open-source, comprehensive machine learning framework, as its foundation. Using the Random Forest gradient boosting toolbox from Reference [2], the meta-learner for this study was developed. Finally, the SHAP analysis was built using the game-theoretic method from the GitHub package made by Slundberg et al. [3].

E. Raw Data

1. Power Data

The information on energy output was gathered using installations of small-scale solar PV panels with a peak power range of around 20kW to 150kW. Additionally, data on energy output was gathered around two to three years ago at intervals of 15 minutes at five distinct locations in India. Numerical Weather Prediction Data is shown in figure 3

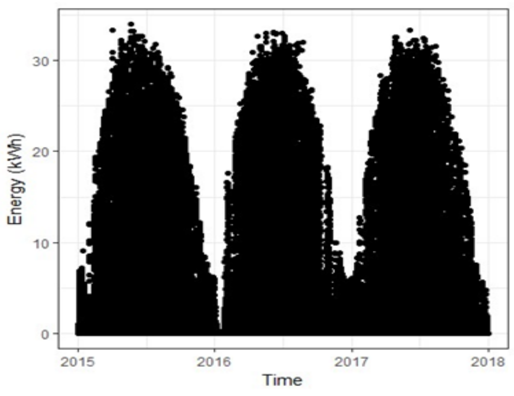


Figure 3 Numerical Weather Prediction Data

The weather information for the five separate websites was obtained from Meteomatics [3]. Table 4.2 lists the parameters of the obtained data, along with a brief explanation and the related units. See [3] for a more thorough discussion of meteorological variables and their accompanying definitions.

There are a few restrictions on the collected NWP data. One is that estimates for periods longer than six hours are not feasible, despite the fact that forecasts are released every six hours—at 6:00, 12:00, 18:00, and 24 hours—and that they cover six hours in ahead. Due to the inability to employ weather predictions that are unavailable, there is also a decline in the training set over longer time horizons. For instance, if one wanted to forecast between 11:00 and 13:00, they would need to use NWP data between 12:00 and 13:00, which is not available at 11:00 when the prediction is to be made because it becomes available at 12:00. Additionally, the data was gathered using the website's ZIP codes.

F. Forecast Models

This section details the precise actions taken for various models and their various projections.

1. Algorithm

Data should be filtered depending on availability for each prediction horizon; exclude observations made at night; the k half-year folds of the training information should be employed, and the collection of data is split into an instruction and a test set. For a KNN, make an argument set with K = (1,...,10) and carry out the following operations for each of the parameters p and each fold I in the set of K folds: as a validation set, fold I; [optional] Preparing the data and applying the model on the remaining K 1 folds; anticipate the values of the validation set;

calculate the mean RMSE across the validation sets to complete; choose the best parameters (those with the lowest RMSE); Train the model using the optimal hyperparameters given all the training data; use the test set, which is the data from the previous six months, to assess the model; and finally,

V. SIMULATION AND RESULTS.

A. Data Preprocessing

Initially, the data is imported from the dataset and saved in the *DF* variable. After that, it was necessary to format the float values in 4 points.

In [2]:

```
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
df = pd.read_csv('/Users/tahirshowkatbazaz/Downloads/solarpower.csv', sep=',')
pd.options.display.float_format = '{:.4f}'.format
```

My goal is to compare the performance of **kNN**, **Random Forest**, and **Ensemble Learning** models. For this, it is necessary to verify which types of data are present in the Dataset and exclude the columns with the most negligible impact on the prediction of solar energy generation. After this, the describe function is called, to show some important data about the data frame, like mean, max, min, std, and others.

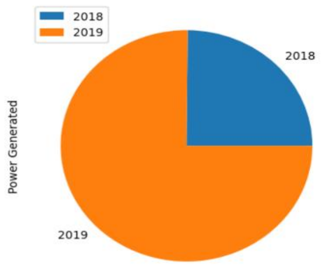


Figure 4 Pie chart of the power generated in year 2018 and 2019.

Most of the data present in the Dataset is from the year 2019. Also, the features that have the greatest impact on the prediction of solar energy generation are **Relative Humidity**, **Distance to Solar Noon** and **Is Daylight**.

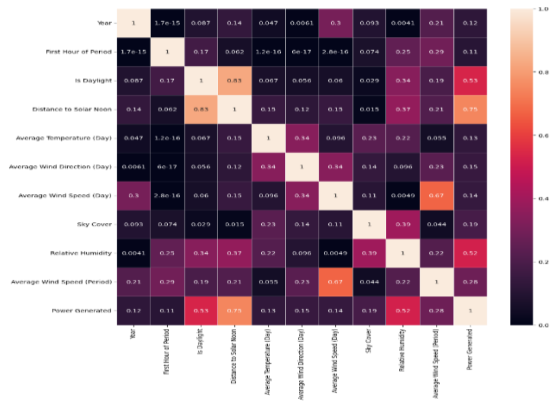


Figure 5 correlation matrix

Uneven value scales can be seen throughout the distribution visualisation phase. If these scale irregularities are not corrected, the models' accuracy may suffer. This is due to the fact that columns with bigger scales frequently have an influence on model outcomes. As a result, it was essential to distribute the column weight while performing a distribution of values on a scale between zero and one.

In [9]:

df.hist(bins=50,figsize=(20,15))

plt.show()

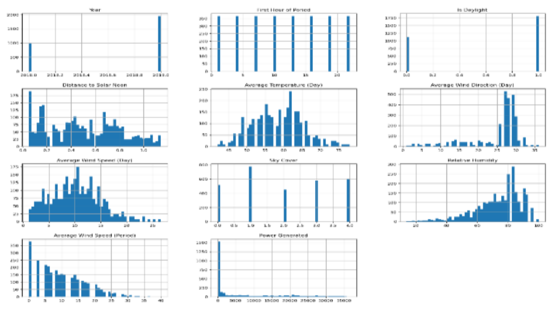


Figure 6 Power and intensity plots over the time

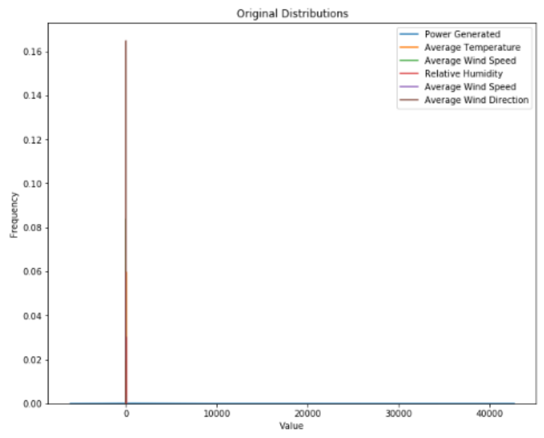


Figure 7 power distribution

The MinMaxScaler function of the Sklearn package, which scales the resources utilised in a range of values, was used to accomplish the distribution (usually 0 and 1). Due to the Dataset's exclusive usage of numeric columns, this technique was employed. It is possible to determine the scale of the columns following the distribution.

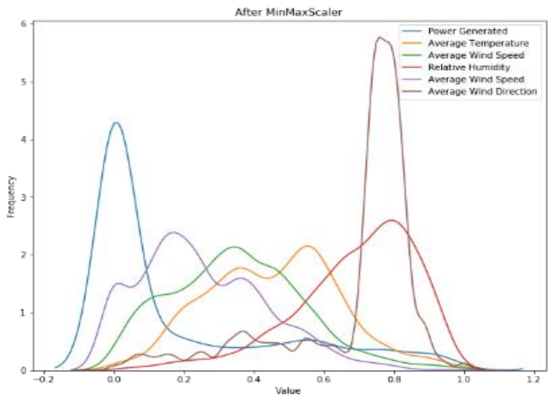


Figure 8 power generated

B. Models and Hyper parametrization

Now, we develop three models that, when combined with the other attributes, forecast solar energy output with at least 75% accuracy. For this, the dataset was divided into X and Y, where X represented the prediction data I used and Y represented the outcome I wanted to foresee. They are split into training (70%) and testing (30%), respectively, for this.

1. Testing k Factors

Running a test with several values and calculating the percentage results is one method of determining the optimal value of K. It's almost like a "brute force test," as each potential value must be thoroughly validated separately.

Our measure shows the relative error between the anticipated and actual values to be 6.36%, with the least error occurring when we have k = 6 (RMSLE of 0.0636).

With the use of a function referred to as the "elbow function," it is feasible to recognise this outcome visually. When K climbs to 5 and then increases once again when it reaches 7, it is possible to spot a decline in RMSLE. As a result, it is feasible to develop a model that employs a value within this range, making it possible to increase the model's hit rate. However, employing hyper-parameterization with GridSearchCV is another approach to determining the optimal value of K. as shown in figure 9.

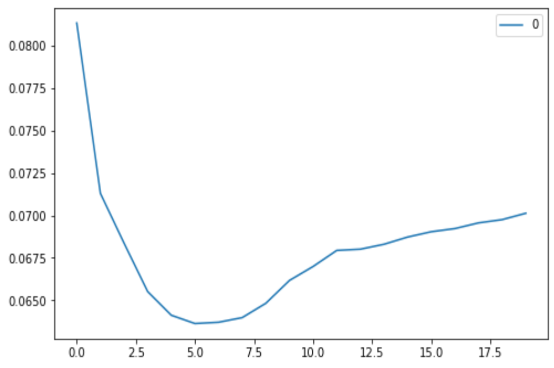


Figure 9 k factor

2. Tuning Hyperparameters

A hyperparameter is a parameter that is used to regulate learning but is not directly learnt by models. Since different constraints, weights, or learning rates that generalise various data patterns may be required for the same sort of machine learning model. Last but not least, these measurements—known as hyperparameters—need to be changed in order for the model to best answer the machine learning challenge.

The scikit-learn package, which carried out a parameter scan of a manually chosen subset of the learning algorithm's hyperparameter space, was used to carry out the modification on the hyperparameters.

C. Models

1. K-Nearest Neighbor

It's time to train the models and evaluate the outcomes once you've determined the ideal value for K. We'll also utilise the score function to assess how accurate our models are using the scoring function.:

- KNN: 90.55%;
- Random Forest: 92.03%;
- Ensemble: 92.34%.

D. Random Forest

In [25]:

```
forest_model = RandomForestRegressor(random_state=1)

forest_model.fit(X_train, y_train)

y_pred_rf = forest_model.predict(X_test)

forest_model.score(X_test, y_test)
```

Out[25]:

```
0.9203515129942916
```

1. Ensemble Stacking

In [26]:

```
estimators=[("knn", knn), ("rf", forest_model)]

ensemble_stack = StackingRegressor(estimators=estimators)

ensemble_stack.fit(X_train, y_train)


y_pred_stacking = ensemble_stack.predict(X_test)

ensemble_stack.score(X_test, y_test)
```

```
Out[26]:

0.9234901930691495
```

E. Validating the Models with Metrics

Here, the metrics' test results and forecast outcomes are compared so that we can see how each of them performed. Because there were negative values among the reported data, the RMSLE measure cannot be used in the Ensemble Learning model.

Table 2 prediction table

	Prediction-KNN	**Prediction-RF**	**Prediction-Stacking**
Test			
0.0000	0.0000	0.0000	-0.0014
0.0000	0.0000	0.0000	-0.0014
0.0000	0.0000	0.0000	-0.0014
0.0000	0.0000	0.0000	-0.0014
0.0124	0.0036	0.0128	0.0092
0.0990	0.1530	0.0898	0.1052
0.0000	0.0000	0.0000	-0.0014
0.3662	0.3510	0.4078	0.3955
0.0622	0.3106	0.3085	0.3103
0.0000	0.0000	0.0000	-0.0014
0.3806	0.4374	0.4273	0.4321
0.5980	0.5572	0.5718	0.5716
0.7560	0.4372	0.5381	0.5158
0.0000	0.0000	0.0000	-0.0014
0.2127	0.4332	0.4375	0.4387

VI. CONCLUSION

This study recommends combining KNN with Random Forest to improve deep learning systems. The proposed ensemble strategy outperformed the individual deep learning approaches because it combined strong basis learners as opposed to poor base students. The KNN model specifically incorporated the expected solar PV's dependency., whereas Random Forest extracted the repeated patterns based on the data. A boosting algorithm is used to combine the forecasts from each base learner. technique known as Random Forest was applied. The Random Forest metal learner used the outputs from the basis models to evaluate the testing data and come to conclusions. The meta learner wasn't taught until the foundation models had been trained by the ensemble. The Random Forest builds trees in a sequential fashion as a meta-learner to reduce the flaws of the prior trees. As a consequence of learning from its predecessors, each tree modifies the residual mistakes. In order to control the uncertainty, the Random Forest was able to successfully mix the output from these potent learners from each of the basic learners, which each provided essential data for prediction. In order to better comprehend how the meta-learner gains from its projections of the underlying method, this work has linked comprehensible machine learning with the modeling.

By resolving the flaws in the previous models and choosing the strongest characteristics for the result estimation, the meta learner reduces variance and bias. There can always be uncertainties in data or irreducible ambiguity.

have a detrimental influence on a deep learning model's performance. More data might not make uncertainty smaller, and it might make it harder for the models to converge. To lessen the uncertainty of the forecast, increase measurement accuracy or create models that don't entirely rely on the input features. The suggested Composite model can manage forecast uncertainty from multiple approaches and does not completely rely on the input characteristics. By generating forecasts that are less reliant on the training data's quirks, individual model optimization, the training run's providence.

By effectively implementing the proposed stacking ensemble method, problems from the health, control engineering, and financial markets sectors may be generalised. Additional study on these techniques is required in order to employ DSE approaches more often and to a larger variety of datasets in order to provide consistent results.

REFERENCES

[1]. . IRENA. Renewable power generation costs in 2017. Technical report, International Renewable Energy Agency, Abu Dhabi, January 2018.

[2]. Jose R. Andrade and Ricardo J. Bessa. Improving renewable energy forecasting with a grid of numerical weather predictions. *IEEE Transactions on Sustainable Energy*, 8(4):1571–1580, October 2017.

[3]. Rich H. Inman, Hugo T.C. Pedro, and Carlos F.M. Coimbra. Solar forecasting methods for renewable energy integration. *Progress in Energy and Combustion Science*, 39(6):535 – 576, 2013.

[4]. J. Antonanzas, N. Osorio, R. Escobar, R. Urraca, F.J. Martinez-de Pison, and F. Antonanzas-Torres. Review of photovoltaic power forecasting. *Solar Energy*, 136:78–111, October 2016.

[5]. V Kostylev, A Pavlovski, et al. Solar power forecasting performance–towards industry standards. In *1st international workshop on the integration of solar power into power systems, Aarhus, Denmark*, 2011.

[6]. Tao Hong, Pierre Pinson, Shu Fan, Hamidreza Zareipour, Alberto Troccoli, and Rob J. Hyndman. Probabilistic energy forecasting: Global energy forecasting competition 2014 and beyond. *International Journal of Forecasting*, 32(3):896 – 913, 2016.

[7]. Gordon Reikard. Predicting solar radiation at high resolutions: A comparison of time series forecasts. *Solar Energy*, 83(3):342 – 349, 2009.

[8]. Peder Bacher, Henrik Madsen, and Henrik Aalborg Nielsen. Online short-term solar power forecasting. *Solar Energy*, 83(10):1772 – 1783, 2009.

[9]. Hugo T.C. Pedro and Carlos F.M. Coimbra. Assessment of forecasting techniques for solar power production with no exogenous inputs. *Solar Energy*, 86(7):2017 – 2028, 2012.

[10]. Federica Davò, Stefano Alessandrini, Simone Sperati, Luca Delle Monache, Davide Airoldi, and Maria T. Vespucci. Post-processing techniques and principal component analysis for regional wind power and solar irradiance forecasting. *Solar Energy*, 134:327 – 338, 2016