Journal of Energy and Power Engineering 16 (2022) 81-89
doi: 10.17265/1934-8975/2022.02.004

An Explanatory Study Approach, Using Machine Learning to Forecast Solar Energy Outcome

Agada Ihuoma Nkechi and Nagata Yasunori

Department of Electrical and Electronics Engineering, University of the Ryukyus, Okinawa 903-0213, Japan

Abstract: AI (artificial intelligence) techniques play a crucially important role in predicting the expected energy outcome and its performance, analysis, modeling and control of renewable energy. Solar energy usage has grown exponentially over the years. In the face of global energy consumption and increased depletion of most fossil fuel, the world is faced with the challenges of meeting the ever-increasing energy demands, also utility companies who provide solar energy have a challenge of unstable input of solar energy to the grid due to its intermittent nature, unlike other sources, hence the difference between expected generation and actual generation, demand and supply can lead to an unbalanced grid. Therefore, incorporating accurately machine learning technology to predict the expected outcome of solar energy from the intermittent solar radiation will be crucial to keep a balance grid operation between supply and demand, production planning and energy management especially during installations of a photovoltaic power plant. However, one of the major problems of forecasting is the algorithms used to control, model, and predict performances of the energy systems which are complicated and involve large computer power, differential equations, and time series. Also having unreliable data (poor quality) for solar radiation over a geographical location as well as insufficient long series can be a bottleneck to actualization. To overcome these problems, we employ the Anaconda Navigator (Jupyter Notebook) for machine learning which can combine large amounts of data with fast, iterative processing and intelligent algorithms allowing the software to learn automatically from patterns or features to predict the performance and outcome of Solar Energy which in turn enables the balance between supply and demand on loads, efficient operation of the utility company as well as enhances power production planning and management.

Key words: AI, backward elimination, data mining, machine learning, linear regression, solar energy.

Nomenclature

AI	Artificial Intelligence
PV	Photo-Voltaic
LR	Linear regression
BESS	Battery Energy Storage System
GUI	Graphical User Interface
NWS	National Weather Service
MLR	Multi Linear Regression
SE	Solar Energy
BE	Backward Elimination
ML	Machine Learning
SL	Supervised Learning
DM	Data Mining
BE	Bidirectional Elimination
FS	Forward Selection

1. Introduction

Data accumulation of solar energy generation has been on a steady rise at an almost unimaginable rate from a very wide variety of sources from general solar radiation emitted every day to its usage by solar converters for energy generation. The use of data mining cannot be overemphasized alongside advances in storage technology, which increasingly make it possible to store such vast amounts of data at relatively low cost whether in commercial data warehouses, scientific research laboratories or elsewhere. Such data can be critical to a company’s growth or decline knowledge that could lead to important discoveries in science, knowledge that could enable us accurately to predict the weather, solar outcomes, natural disasters and more. Yet the huge volumes involved mean that most of the data are merely stored never to be

Corresponding author: Agada Ihuoma Nkechi, PhD student, research fields: an explanatory study approach, using machine learning to forecast solar energy outcome.

examined in more than the most superficial way, if at all. It has rightly been said that the world is becoming “data rich but knowledge poor” [1].

Solar energy is increasing exponentially making it one of the most popular renewable forms of energy. The irradiance is a measure of the energy available to enter a solar PV (photo-voltaic) system, if the irradiance of a location is known over a period, it can be used to predict future solar energy generation at that location.

Our objective is to automate generating prediction models for smart homes that include on-site renewables. Prediction models are used by both the grid and the individual smart homes for advanced planning of electricity generation and consumption. Smart homes can use the models to potentially plan their consumption pattern to better match the power they generate on-site. The grid can use the models to plan generator dispatch schedules in advance as the fraction of renewables increases in the grid. Energy prediction must be done accurately to avoid shortage and surpluses. The more efficient the predictions are the more efficiently the utility companies can operate [2].

Nowadays solar panels are widely used to generate solar energy which is a very promising renewable energy source. With regards to solar electricity providers and a grid operator, it is critical to accurately predict solar power generation for supply-demand planning in an electrical grid, which directly affects their profit. Predicting solar output is, however, very difficult because solar power generation depends on numerous weather features which is uncontrollable. This research proposes the technology of data mining using the Anaconda Navigator [3] to be able to predict daily solar energy generation of any system. In this case we concentrated automatically generating models that correctly predict solar energy generation based on the previous NWS (National Weather Service) weather forecast. Also, degradation of the solar panels is taking into consideration as this affects the outcome of solar energy generation. The detailed model of the solar system which involves the panels, converters and load has been presented in the ICEE (International Conference of Electrical Engineers).

The paper is organized as follows; Section II gives a review of the DC/AC micro-grid model conversion which consists of the photovoltaic system, a battery storage which is charged and discharged by the buck-boost converter during high and low solar radiation respectively and the boost converters which steps up input voltage from the PV to the required system voltage and converted by the inverter to be used by the load. Section III shows the behavior and mode of machine learning using linear regression, and its implementation. Section IV shows the data used for simulation, the simulation results and the forecasted result and Section V addresses the conclusion and the advantages of the proposed model to forecast solar energy outcome for the purpose of sustainability.

2. Review of the Hybrid DC/AC Solar Microgrid Model Conversion

Power electronics devices are used to convey and harness solar energy as it ensures maximum power tracking of current and voltage. The output active power of the PV is dependent on solar radiation and temperature for power generation. The lithium battery is used for power compensation during low solar radiation. This, in turn, smoothens the real power output and minimizes adverse impact on the grid. The system comprises the use of solar panels, boost converters to step up voltage and keep it constant, battery bank, buck-boost converter to charge and discharge the battery at high and low radiation respectively as seen in Fig. 1. The effectiveness of the proposed method is confirmed by simulation results on MATLAB®.

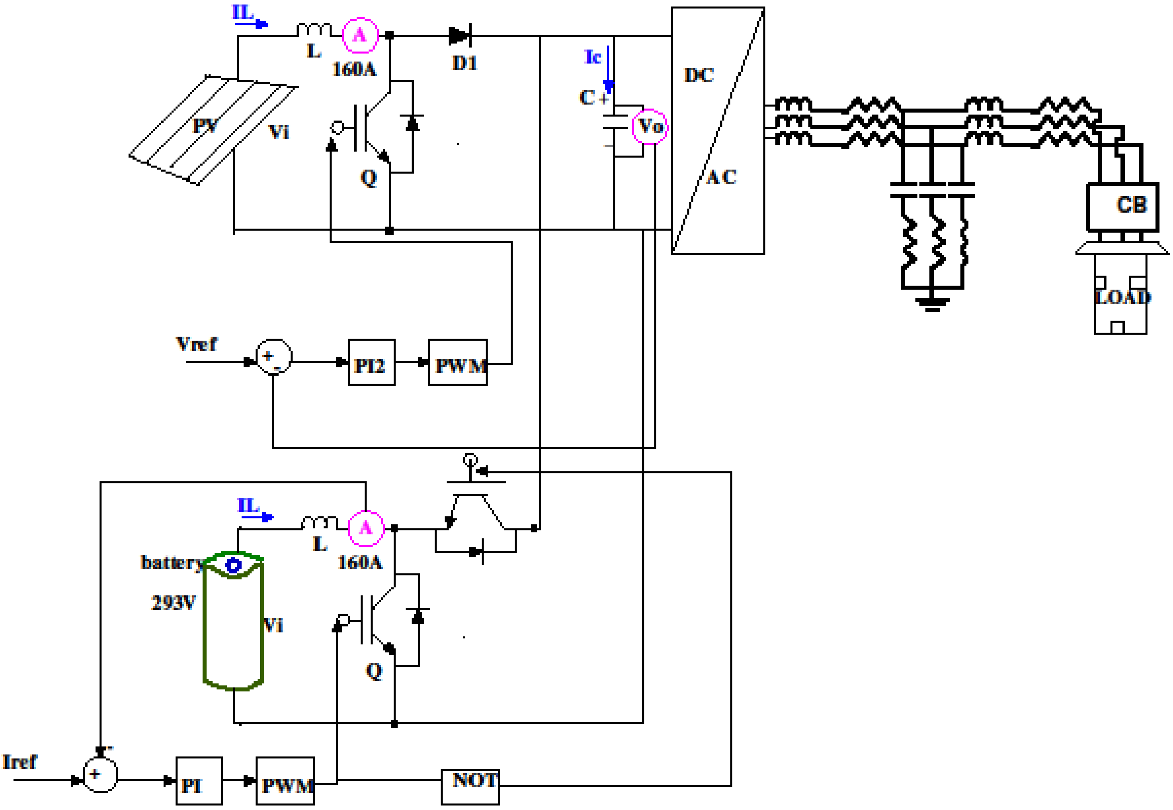


Fig. 1 **A detailed model of the micro-grid [9].**

3. The Machine Learning Tool, Anaconda Navigator

Anaconda Navigator is a desktop GUI (graphical user interface) included in anaconda distribution that allows you to launch applications and easily manage conda packages, environments, and channels without using command-line commands.

Linear Regression is used in this case, as it is known as the most basic and widely used technique for regression [4]. It models the relationship between the input and output variables using linear predictor functions whose unknown model parameters are estimated from the data using a least square approach. The parameter values can be estimated either by solving a set of linear equations or using an iterative method such as gradient descent [5].

3.1 Data Transformation for Solar Radiation Prediction

For this research the Jupyter Notebook is used for the data learning and predictions to forecast data to be used. Data come in, possibly from many sources. It is integrated and placed in some common data store. Part of it is then taken and pre-processed into a standard format. The “prepared data” are then passed to a data mining algorithm which produces an output in the form of rules or some other kind of “patterns”. These are then interpreted to give the forecasted value which is also known as useful knowledge as seen in Figs. 2 and 3 which explain the process of data transformation from its raw state and then previewed, identified, split, trained, saved and tested. The data are further committed to Git (global information tracker) which is a version control system, and finally to runtime model serving [6].

Jupyter is an electronic lab notebook to document procedures, data, calculations, and findings. It provides an interactive computational environment for developing data science applications.

Jupyter Notebooks combine software code, computational output, explanatory text, and rich content in a single document. It allows in-browser editing and execution of code and displays computation results which are saved in .ipynb extension.

4. Data Analysis and Predicted Result

For this research the Japan Meteorological Agency data on weather were used for analysis. The supervised learning approach is implemented here, as it deals with labels and features [7, 10].

4.1 Selection Process for Multiple Regression

The basis of a multiple linear regression is to assess whether one continuous dependent variable can be predicted from a set of independent or predictor variables, in simple terms, how much variance a continuous dependent variable is explained by a set of predictors. Certain regression selection approaches are helpful in testing predictors, thereby increasing the efficiency of analysis.

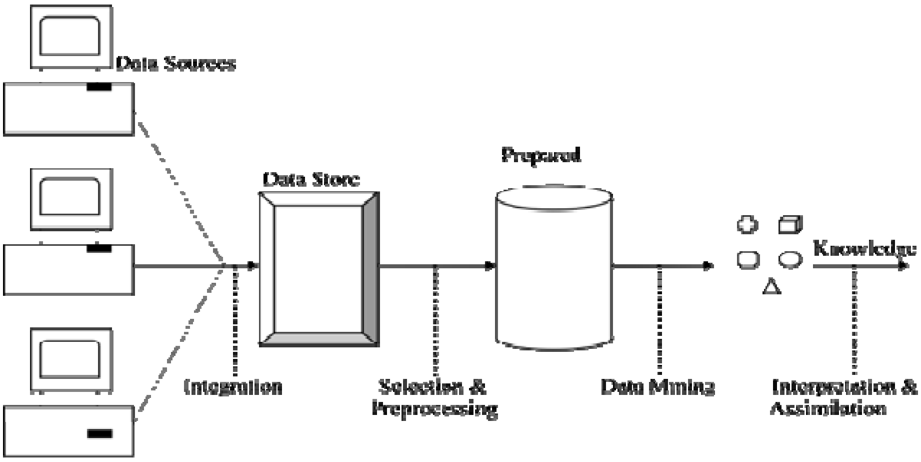


Fig. 2 A complete data processing unit [2].

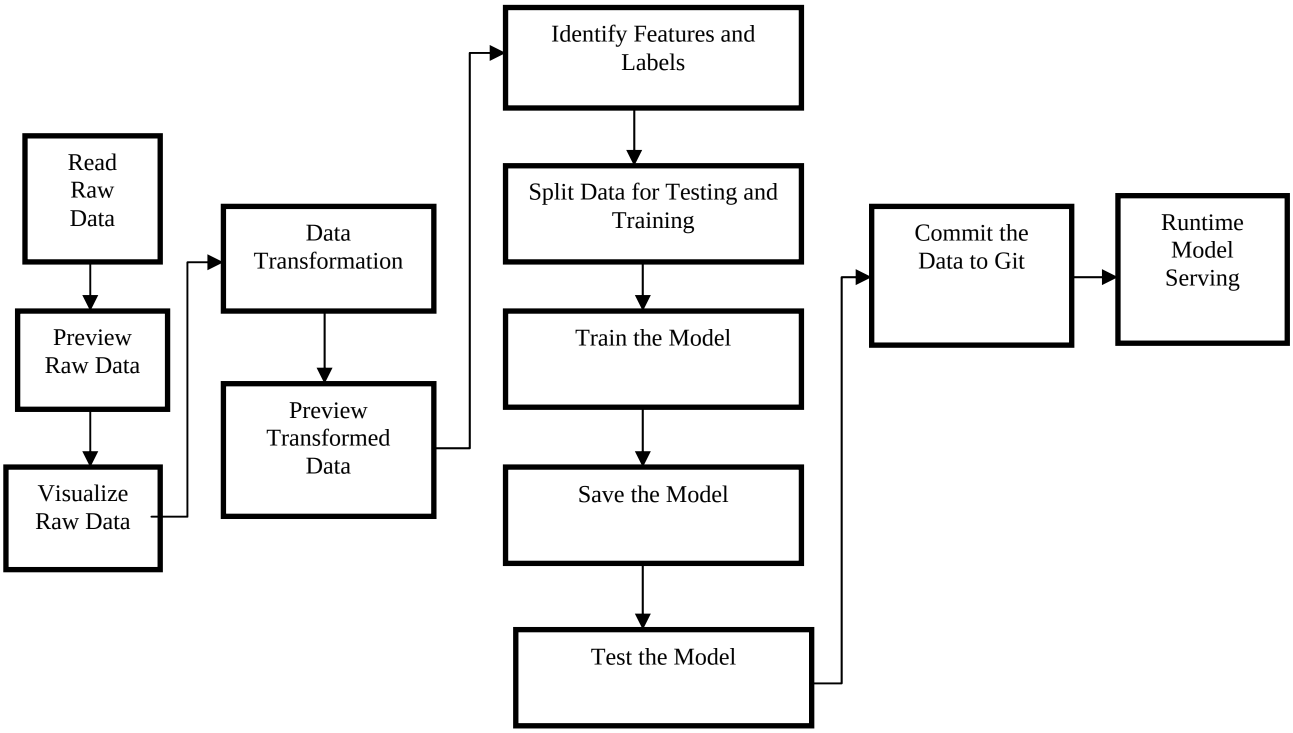


Fig. 3 The Notebook workflow for the solar outcome.

4.2 Entry Method

The standard method of entry is simultaneous; all independent variables are entered into the equation at the same time. This is an appropriate analysis when dealing with a small set of predictors and when it is hard to determine which independent variables will create the best prediction equation. Each predictor is assessed as though it were entered after all the other independent variables were entered, and assessed by what it offers to the prediction of the dependent variable that is different from the predictions offered by the other variables entered to the model [8].

4.3 P-Values and Statistical Significance

P-values are most often used by researchers to say whether a certain pattern they have measured is statistically significant. Statistical significance is another way of saying that the *p*-value of a statistical test is small enough to reject the null hypothesis of the test. How small is small enough? The most common threshold is $p < 0.05$; that is, when you would expect to find a test statistic as extreme as the one calculated by your test only 5% of the time. But the threshold depends on your field of study. Some fields prefer thresholds of 0.01, or even 0.001. The threshold value for determining statistical significance is also known as the alpha value.

4.4 Exploring the Raw Data for Analysis

A code cell was used to import the required python libraries and the raw files converted from .csv to a Data frame with a time series as seen in Fig. 4.

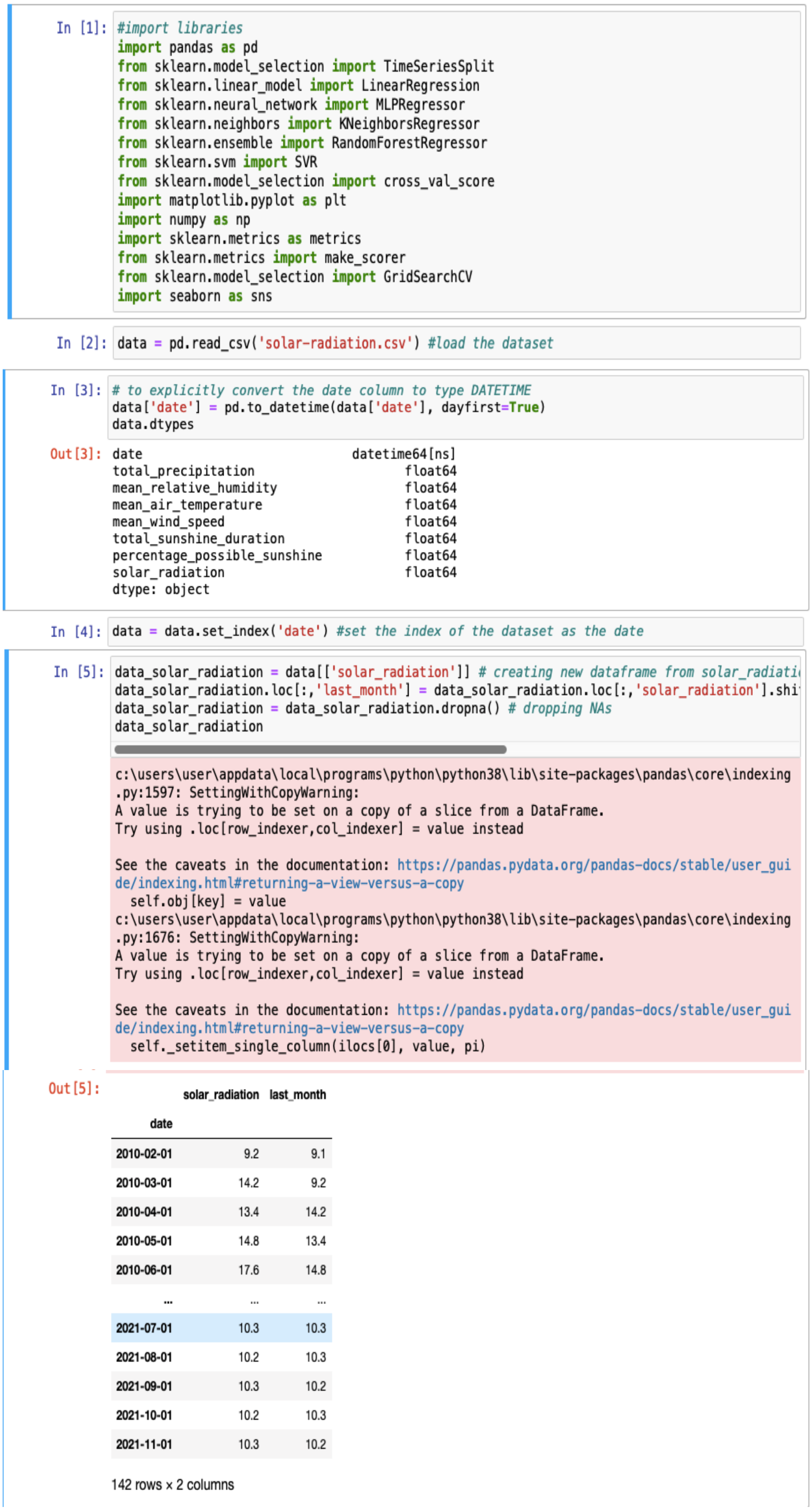
Libraries used for the analysis of the raw data include:

- NumPy: is for manipulating and creating vectors and matrices.
- Pandas: For analyzing, wrangling, and munging data.
- Matplotlib: Is for data visualization.
- Sklern: For supervised and unsupervised learning. This library provides various tools for model fitting, data preprocessing, model selection, and model evaluation. It has built-in machine learning algorithms and models called estimators.

4.5 The Use of Backward Elimination

Backward elimination is a feature selection technique while building a machine learning model. It is used to remove those features that do not have a significant effect on the dependent variable or prediction of output. There are various ways to build a model in Machine Learning, which are: All-in, Backward Elimination, Forward Selection, Bidirectional Elimination, Score Comparison. Above are possible methods for building the model in Machine learning, but we only used the Backward Elimination process as it is the fastest method.

Backward Elimination is important for multiple linear regression model. Unnecessary features increase the complexity of the model. Hence it is good to have only the most significant features and keep our model simple to get the better result. So, in order to optimize the performance of the model, we used the Backward Elimination method. This process is used to optimize the performance of the MLR (multi linear regression) model as it only includes the most effective feature and removes the least effective feature.



```
In [1]: #import libraries
        import pandas as pd
        from sklearn.model_selection import TimeSeriesSplit
        from sklearn.linear_model import LinearRegression
        from sklearn.neural_network import MLPRegressor
        from sklearn.neighbors import KNeighborsRegressor
        from sklearn.ensemble import RandomForestRegressor
        from sklearn.svm import SVR
        from sklearn.model_selection import cross_val_score
        import matplotlib.pyplot as plt
        import numpy as np
        import sklearn.metrics as metrics
        from sklearn.metrics import make_scorer
        from sklearn.model_selection import GridSearchCV
        import seaborn as sns
```

```
In [2]: data = pd.read_csv('solar-radiation.csv') #load the dataset
```

```
In [3]: # to explicitly convert the date column to type DATETIME
        data['date'] = pd.to_datetime(data['date'], dayfirst=True)
        data.dtypes

Out[3]: date                              datetime64[ns]
        total_precipitation                      float64
        mean_relative_humidity                   float64
        mean_air_temperature                     float64
        mean_wind_speed                          float64
        total_sunshine_duration                  float64
        percentage_possible_sunshine             float64
        solar_radiation                          float64
        dtype: object
```

```
In [4]: data = data.set_index('date') #set the index of the dataset as the date
```

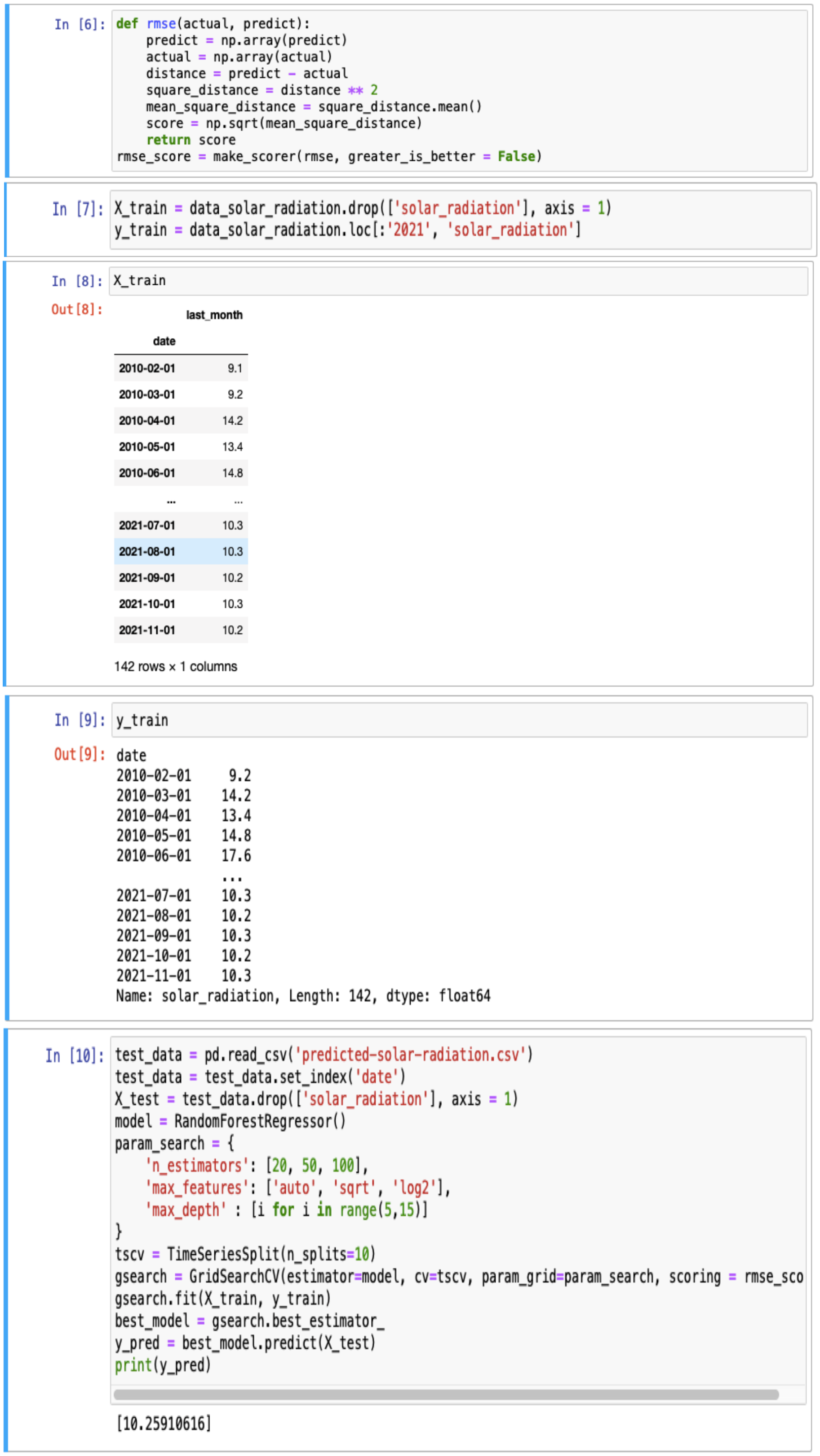
```
In [5]: data_solar_radiation = data[['solar_radiation']] # creating new dataframe from solar_radiati
        data_solar_radiation.loc[:,'last_month'] = data_solar_radiation.loc[:,'solar_radiation'].shi
        data_solar_radiation = data_solar_radiation.dropna() # dropping NAs
        data_solar_radiation
```

```
c:\users\user\appdata\local\programs\python\python38\lib\site-packages\pandas\core\indexing
.py:1597: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_gui
de/indexing.html#returning-a-view-versus-a-copy
  self.obj[key] = value
c:\users\user\appdata\local\programs\python\python38\lib\site-packages\pandas\core\indexing
.py:1676: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_gui
de/indexing.html#returning-a-view-versus-a-copy
  self._setitem_single_column(ilocs[0], value, pi)
```

Out[5]:

	solar_radiation	last_month
date		
2010-02-01	9.2	9.1
2010-03-01	14.2	9.2
2010-04-01	13.4	14.2
2010-05-01	14.8	13.4
2010-06-01	17.6	14.8
...	...	...
2021-07-01	10.3	10.3
2021-08-01	10.2	10.3
2021-09-01	10.3	10.2
2021-10-01	10.2	10.3
2021-11-01	10.3	10.2

142 rows × 2 columns



In [6]:
```
def rmse(actual, predict):
    predict = np.array(predict)
    actual = np.array(actual)
    distance = predict - actual
    square_distance = distance ** 2
    mean_square_distance = square_distance.mean()
    score = np.sqrt(mean_square_distance)
    return score
rmse_score = make_scorer(rmse, greater_is_better = False)
```

In [7]:
```
X_train = data_solar_radiation.drop(['solar_radiation'], axis = 1)
y_train = data_solar_radiation.loc[:'2021', 'solar_radiation']
```

In [8]:
```
X_train
```

Out[8]:

	last_month
date	
2010-02-01	9.1
2010-03-01	9.2
2010-04-01	14.2
2010-05-01	13.4
2010-06-01	14.8
...	...
2021-07-01	10.3
2021-08-01	10.3
2021-09-01	10.2
2021-10-01	10.3
2021-11-01	10.2

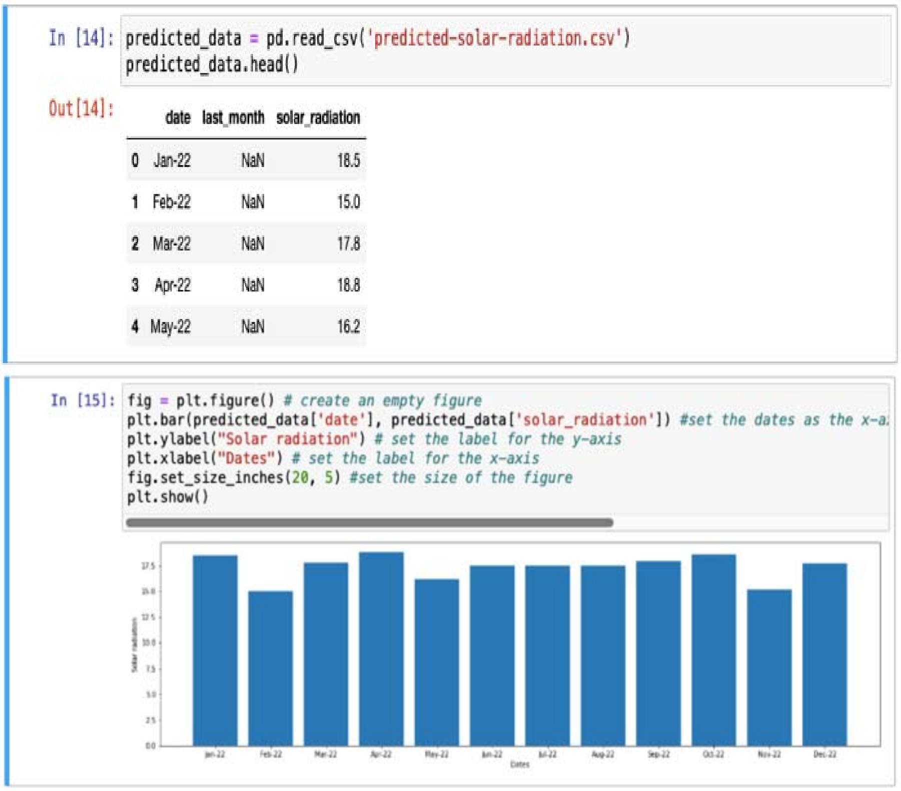
142 rows × 1 columns

In [9]:
```
y_train
```

Out[9]:
```
date
2010-02-01     9.2
2010-03-01    14.2
2010-04-01    13.4
2010-05-01    14.8
2010-06-01    17.6
              ...
2021-07-01    10.3
2021-08-01    10.2
2021-09-01    10.3
2021-10-01    10.2
2021-11-01    10.3
Name: solar_radiation, Length: 142, dtype: float64
```

In [10]:
```
test_data = pd.read_csv('predicted-solar-radiation.csv')
test_data = test_data.set_index('date')
X_test = test_data.drop(['solar_radiation'], axis = 1)
model = RandomForestRegressor()
param_search = {
    'n_estimators': [20, 50, 100],
    'max_features': ['auto', 'sqrt', 'log2'],
    'max_depth' : [i for i in range(5,15)]
}
tscv = TimeSeriesSplit(n_splits=10)
gsearch = GridSearchCV(estimator=model, cv=tscv, param_grid=param_search, scoring = rmse_sco
gsearch.fit(X_train, y_train)
best_model = gsearch.best_estimator_
y_pred = best_model.predict(X_test)
print(y_pred)
```

```
[10.25910616]
```



```
In [14]: predicted_data = pd.read_csv('predicted-solar-radiation.csv')
         predicted_data.head()

Out[14]:
            date   last_month   solar_radiation
         0  Jan-22        NaN              18.5
         1  Feb-22        NaN              15.0
         2  Mar-22        NaN              17.8
         3  Apr-22        NaN              18.8
         4  May-22        NaN              16.2
```

```
In [15]: fig = plt.figure() # create an empty figure
         plt.bar(predicted_data['date'], predicted_data['solar_radiation']) #set the dates as the x-a
         plt.ylabel("Solar radiation") # set the label for the y-axis
         plt.xlabel("Dates") # set the label for the x-axis
         fig.set_size_inches(20, 5) #set the size of the figure
         plt.show()
```

Fig. 4 Data frame converted from .csv.

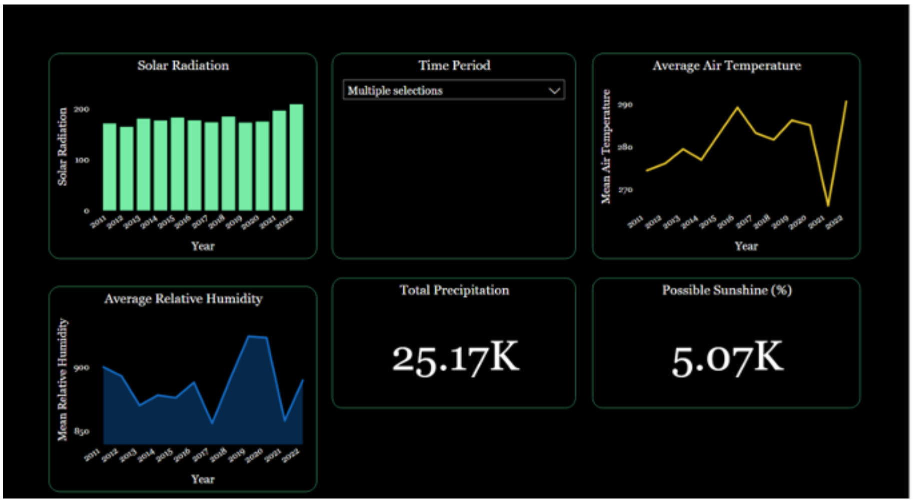


Fig. 5 A virtual representation of the predicted output using power bi.

5. Conclusion

The most important features were total sunshine duration and solar radiation, although various temperature related variables contributed towards solar irradiance prediction accuracy. The prediction of solar radiation is very important to determine the amount of energy that can be generated in a day. Linear regression tells us exactly the outcome expected. After importing all libraries needed, the data are cleaned and trained. Linear regression models have predictive power but also many shortcomings like low values were overestimated while high values were underestimated, also residuals violate homoscedasticity and normality. Further work is needed like the use of distance-weighted average of weather forecasts (at multiple grid points) as the weather forecast variable features.

References

[1] Kim, J. G., Kim, D. H., Yoo, W. S., Lee, J. Y., and Kim, Y. B. “Daily Prediction of Solar Power Generation Based on Weather Forecast Information in Korea.” *IET Renewable Power Generation* 11 (10): 1268-73.

[2] www.newenergysolar.com.au.

[3] https://docs.anaconda.com/anaconda/navigator/.

[4] https://developers.redhat.com/articles/2021/05/21/introduction-machine-learning-jupyter-notebooks#getting_started_with_jupyter_notebooks.

[5] https://www.data.jma.go.jp/obd/stats/etrn/view/monthly_s3_en.php?block_no=47936&view=11.

[6] Montgomery, D. C., Peck, E. A., and Vining, G. G. 2015. *Introduction to Linear Regression Analysis*. New York, NJ: John Wiley & Sons.

[7] Jawaid, F., and Junejo, K. N. “Predicting Daily Mean Solar Power Using Machine Learning Regression Techniques.” In *Proceedings of the 2016 6th Int. Conf. Innov. Comput. Technol. INTECH 2016*, pp. 355-60.

[8] Chuluunsaikhan, T., Nasridinov, A., Choi, W. S., Choi, D. B., Choi, S. H., and Kim, Y. M. 2021. “Predicting the Power Output of Solar Panel Based on Weather and Air Pollution Features Using Machine Learning.” *Journal of Korea Multimedia Society* 24: 222-32.

[9] Nkechi, A. I., Howlader, A. M., and Yona, A. 2018. “Integration of Photovoltaic Energy to the Grid, Using the Virtual Synchronous Generator Control Technique.” *Journal of Energy and Power Engineering* 12: 329-39. doi: 10.17265/1934-8975/2018.07.001.

[10] Devendra, K., Ashiwani, K., and Lokesh, K. Y. 2014. “Unit Commitment of Thermal Power Plant in Integration with Wind and Solar Plant Using Genetic Algorithm.” *International Journal of Engineering Research & Technology* 3 (7): 664-9.