Machine Learning for Solar Energy Prediction

Machine Learning: 19EC6CE1ML

BMS Collge of Engineering

Team Details: Priyanshu M; Roshan Nayak; K S Eshwar Subramanya Prasad; K Shivanithyanathan; Subramanya K; Tanay Somnani; Venkatesh Subramanya Iyer Giri

Mentor:

Renganayaki S, System Specification Engineer, Nokia

April 28, 2021

Overview

Problem Definition

- Renewable energy resources offer many advantages over traditional energy resources such as fossil fuel but the energy produced by them fluctuates with changing weather conditions.
- Power forecasts typically are derived from numerical weather prediction models, but statistical and machine learning techniques are increasingly being used to produce more accurate forecasts.
- Project aims at developing machine learning models which include regression, deep neural networks to evaluate the performance and obtain a statistical model to achieve the objective

Methodology

This project is decomposed in 3 parts:

- Data pre-processing: we processed the raw weather data files (input) and the power production data files (output) to get meaningful numeric value;
- Feature Selection: we run correlation analysis between the weather features and the energy output to discard useless features, also implement dimension reduction techniques such as Principal Component Analysis (PCA)
- Machine Learning : we compare the performances of our ML algorithms. Implemented models include Multivariate Regression with and without dimension reduction, Boosting Regression Trees, and artificial Neural Networks

Data Collection

- The Training dataset was taken from Kaggle AMS 2013-2014 Solar Energy Prediction Contest.
- The dataset is of size 2.84 GB. The data are in netCDF4 files with each file holding the grids for each ensemble member at every time step for a particular variable.
- Each netCDF4 file contains the latitude-longitude grid and time step values as well as metadata listing the full names of each variable and the associated units.
- Each netCDF4 file contains the total data for one of the model variables and are stored in a multidimensional array.

Data Preparation

- Weather features were contained in netCDF4 files stored in a multidimensional array.
- NetCDF libraries available in R language were used to convert netCDF4 format file into csv file for future data cleaning and processing.
- The dataset had many missing values for weather features and for solar output for daily and hourly dataset.
- Daily dataset was prepared by averaging over the entire day (for sunlit hours) to represent each data by single data instance.
- The feature Sky Condition was of type String which was converted to a numeric value.

Data Preparation

	A	B	C	D	E	F	G	H	I	J
1	Date	Cloud coverage	Visibility	Temperature	Dew point	Relative humidity	Wind speed	Station pressure	Altimeter	Solar energy
2	2/1/2016	0.1	9.45	3.11	0.32	79.46	4.7	29.23	30.02	20256
3	2/2/2016	0.8	3.94	6.99	6.22	93.6	13.29	28.91	29.7	1761
4	2/3/2016	0.87	8.7	1.62	0.02	85	16.73	29.03	29.82	2775
5	2/4/2016	0.37	10	-2.47	-5.89	74.52	9.46	29.46	30.26	28695
6	2/5/2016	0.52	9.21	-2	-4.15	82.03	5.92	29.55	30.35	9517
7	2/6/2016	0.13	8.12	0.91	-1.62	81.03	5.48	29.44	30.24	26973
8	2/7/2016	0.21	10	4.24	0.54	73.8	12.71	29.09	29.88	22365
9	2/8/2016	0.87	7.84	-3.33	-5.05	83.53	14.46	28.96	29.75	4995

Figure: Prepared Daily Dataset

	A	B	C	D	E	F	G	H	I	J	K
1	Date	Hour	Cloud covera	Visibility	Temperature	Dew point	Relative hum	Wind speed	Station press	Altimeter	Solar energy
2	31/01/2016	24	0.00	5.00	1.40	0.89	95.56	9.00	29.10	29.89	0.00
3	01/02/2016	1	0.00	7.88	1.16	0.62	91.04	7.04	29.11	29.90	0.00
4	01/02/2016	2	0.00	9.84	1.22	0.96	89.28	8.96	29.12	29.91	0.00
5	01/02/2016	3	0.00	9.84	1.02	0.61	89.12	6.36	29.14	29.93	0.00
6	01/02/2016	4	0.00	9.88	0.83	0.45	90.08	6.12	29.15	29.94	0.00
7	01/02/2016	5	0.00	9.84	0.77	0.10	85.44	5.08	29.16	29.95	0.00
8	01/02/2016	6	0.00	9.92	0.37	-0.01	89.12	4.72	29.19	29.98	0.00
9	01/02/2016	7	0.00	10.00	0.47	-0.04	90.08	6.00	29.20	29.99	84.29

Figure: Prepared Hourly Dataset

Exploratory Data Analysis

- Exploratory Data Analysis was performed on dataset for identifying the features, a number of observations, checking for null values or empty cells,etc.
- EDA was carried out for both daily data and hourly data to get the first impression of data before training

```
%matplotlib inline
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from matplotlib import pyplot
```

```
df = pd.read_csv(r"Downloads/hourly-dataset_final2.csv")
```

```
df.drop('Unnamed: 11',inplace= True,axis = 1)     #removing unnecessary columns
```

```
df.head()
```

	Date	Hour	Cloud coverage	Visibility	Temperature	Dew point	Relative humidity	Wind speed	Station pressure	Altimeter	Solar energy
0	31-01-2016	24	0.0	5.00	1.40	0.89	95.56	9.00	29.10	29.89	0.0
1	01-02-2016	1	0.0	7.88	1.16	0.62	91.04	7.04	29.11	29.90	0.0
2	01-02-2016	2	0.0	9.84	1.22	0.96	89.28	8.96	29.12	29.91	0.0
3	01-02-2016	3	0.0	9.84	1.02	0.61	89.12	6.36	29.14	29.93	0.0
4	01-02-2016	4	0.0	9.88	0.83	0.45	90.08	6.12	29.15	29.94	0.0

Figure: EDA using Pandas Library

Exploratory Data Analysis

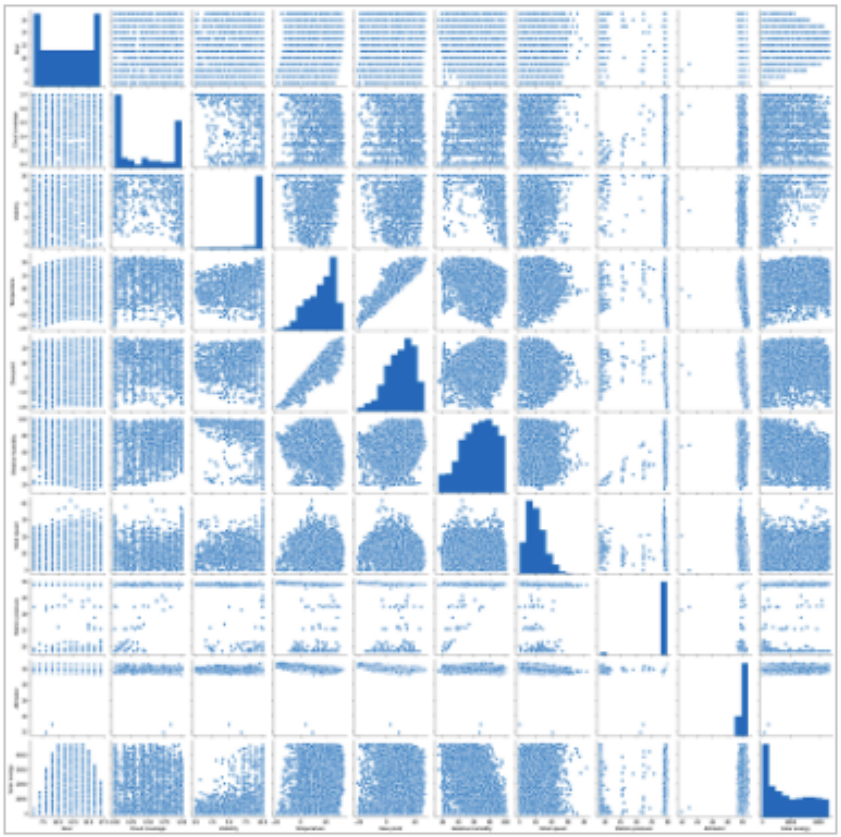

Figure: Pairplot

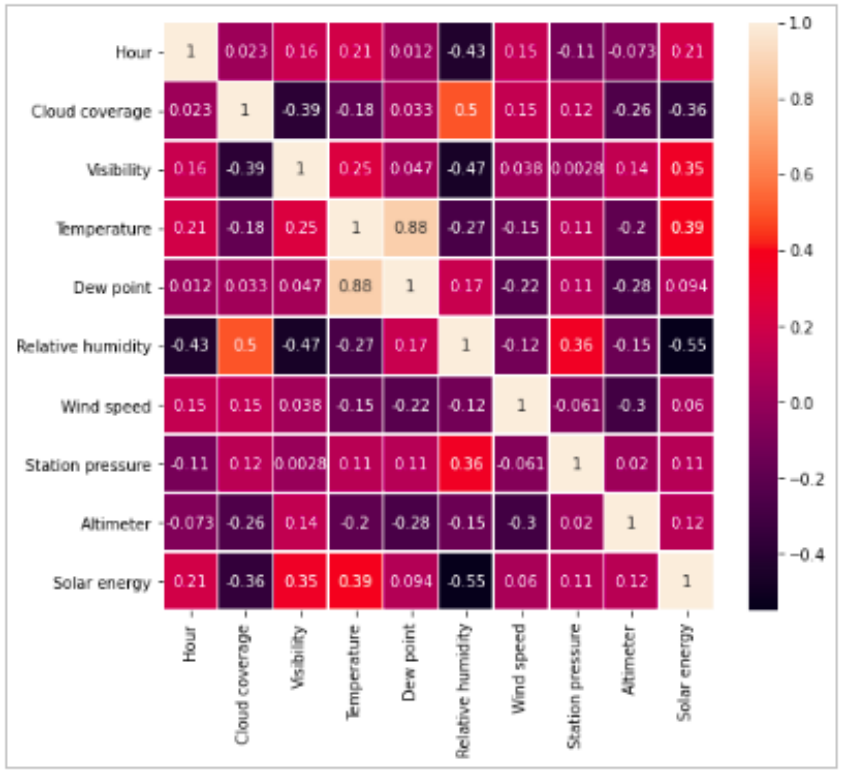

Figure: Correlation Heat Map

Dimensionality Reduction

- Dimensionality reduction refers to the transformation of data from a high-dimensional space into a low-dimensional space so that the low-dimensional representation retains some meaningful properties of the original data.
- Principal Component Analysis (PCA): PCA was used to provide low dimensional visualization (2D) from high dimensional space.

Principal Component Analysis (PCA)

```
scaler = MinMaxScaler()
#scalar = StandardScaler()

df = pd.DataFrame(scaler.fit_transform(df.iloc[:,1:]), columns=df.columns[1:])

x = df[['Cloud coverage','Visibility','Temperature','Dew point','Relative humidity','Wind speed',
        'Station pressure','Altimeter']]
pca = PCA().fit(x)
z = pca.transform(x)
print(z)
plt.scatter(z[:,0],z[:,1])
```

Figure: PCA on Hourly dataset

Dimensionality Reduction: PCA

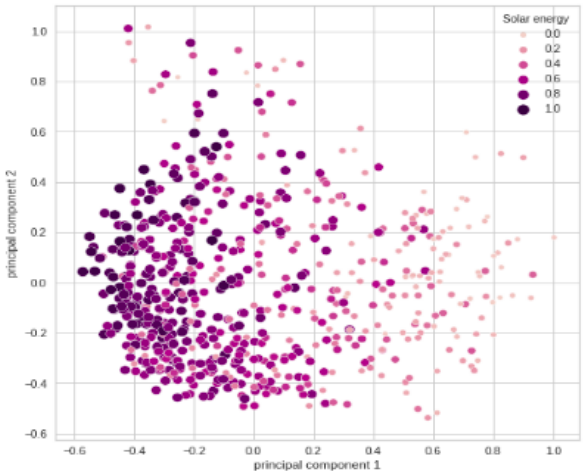

Figure: 2 component PCA visualization (colour coded on normalized Solar output)

```
from sklearn.model_selection import KFold

crossvalidation = KFold(n_splits=5, shuffle=True)
model = LinearRegression().fit(df_pca.iloc[:,0:2],df_pca.iloc[:,-1])

n_scores = cross_val_score(model,df_pca.iloc[:,0:2],df_pca.iloc[:,-1], scoring="r2"
                           ,cv=crossvalidation, n_jobs=1)

print(str(n_scores))
print('Mean r2_score: %.3f' % (np.mean(n_scores)))
```

```
[0.47294389 0.52498537 0.50114031 0.54833157 0.59913496]
Mean r2_score: 0.529
```

Figure: Two component PCA used as predictors to fit Multiple regression model

Model Selection

Several Models were build and model performance was analyzed

Multiple Linear Regression

- Multiple linear regression (MLR), also known simply as multiple regression, is a statistical technique that uses several explanatory variables to predict the outcome of a response variable.

Support Vector Regressor (SVR)

- SVR gives us the flexibility to define how much error is acceptable in our model and will find an appropriate line (or hyperplane in higher dimensions) to fit the data.
- We used RBF kernel which works best for regression.

Model Selection

Decision Trees

- Decision Tree is a decision-making tool that uses a flowchart-like tree structure or is a model of decisions and all of their possible results, including outcomes, input costs and utility.

Random forest

- Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean/average prediction (regression) of the individual trees.
- We used a model with 1000 trees.

Model Evaluation

Model	MSE (in kWh)
Multiple Linear Regression	1174947.56
Decision Trees	4861068.69
SVR	1674274.14
Random Forest	465756.90

Table: Model Comparison and Evaluation

- MLR assumes that there is a linear relationship between the independent features and the target variable. Hence it does not perform well if model has a non linear relation.
- Decision tree is more prone to overfitting if its not tuned well. Usually considering multiple DTs instead of just one work better. Hence we go for RF for best performing model

Energy Forecasting

- Forecasting in general is the process of making predictions based on past and present data and most commonly by analysis of trends.
- In Machine Learning we Forecast Time series data by training the model on past values. Using the technique we can forecast future data which can be helpful.
- In this project we have forecasted solar energy data by analyzing past solar energy values.
- We use LSTM i.e., Long short term memory. It is a type of RNN but it solves many drawbacks which RNN has.

Energy Forecasting

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 9, 50)             10400
_________________________________________________________________
lstm_1 (LSTM)                (None, 9, 50)             20200
_________________________________________________________________
lstm_2 (LSTM)                (None, 50)                20200
_________________________________________________________________
dense (Dense)                (None, 1)                 51
=================================================================
Total params: 50,851
Trainable params: 50,851
Non-trainable params: 0
```

Figure: Model Architecture: 3 LSTMs and one Dense layer for our model

```
Epoch 95/100
177/177 [==============================] - 2s 9ms/step - loss: 0.0098 - val_loss: 0.0098
Epoch 96/100
177/177 [==============================] - 1s 8ms/step - loss: 0.0100 - val_loss: 0.0095
Epoch 97/100
177/177 [==============================] - 1s 8ms/step - loss: 0.0100 - val_loss: 0.0097
Epoch 98/100
177/177 [==============================] - 1s 8ms/step - loss: 0.0096 - val_loss: 0.0099
Epoch 99/100
177/177 [==============================] - 1s 8ms/step - loss: 0.0092 - val_loss: 0.0100
Epoch 100/100
177/177 [==============================] - 1s 8ms/step - loss: 0.0093 - val_loss: 0.0106
```

Figure: Model Training: val loss and training loss for 100 epoch

Application Development

- We developed a web application to help the user to utilize our machine learning model and predict the total solar energy generation at a particular location, if they want to set up a solar energy plant.
- Currently we have a web page where the user has to input a city name and they will get the total solar energy generated at that city.

Data Collection using API

- Weatherbit: Weatherbit API is used to retrieve current weather observations from over 47,000 live weather stations. We provide the city name, from which the latitude and longitude is retrieved and used in the API, and the API provides the weather information required.
- Opentopodata: It is an elevation API, which is used for the "Altimeter" feature.The elevation is provided using the city name or the latitude and longitude.

Application Development

```python
@app.route('/predict',methods=['POST'])
def home():
    cityname = request.form['cityname']
    lat,long = latandlong(cityname)
    X = getWeatherData(lat,long)
    X = getCorrectUnit(X)
    X = list(X)
    Xmax = [1.0,10.0,28.18,25.02,97.85,24.83,29.87,30.67,2017,12,31]
    Xmin = [0.0,1.15,-16.06,-18.72,21.25,1.03,8.59,29.48,2016,1,1]
    XN= [x-xmn for x, xmn in zip(X, Xmin)]
    XD = [xmx-xmn for xmx, xmn in zip(Xmax, Xmin)]
    scaled =  [xn / xd for xn, xd in zip(XN, XD)] #scaled=(X-Xmin)/(Xmax-Xmin)
    scaled = list([scaled])
    pred = model.predict(scaled)
    print(pred)
    return render_template('after.html',data=pred)
```

Figure: data retrieved from the API by using Flask

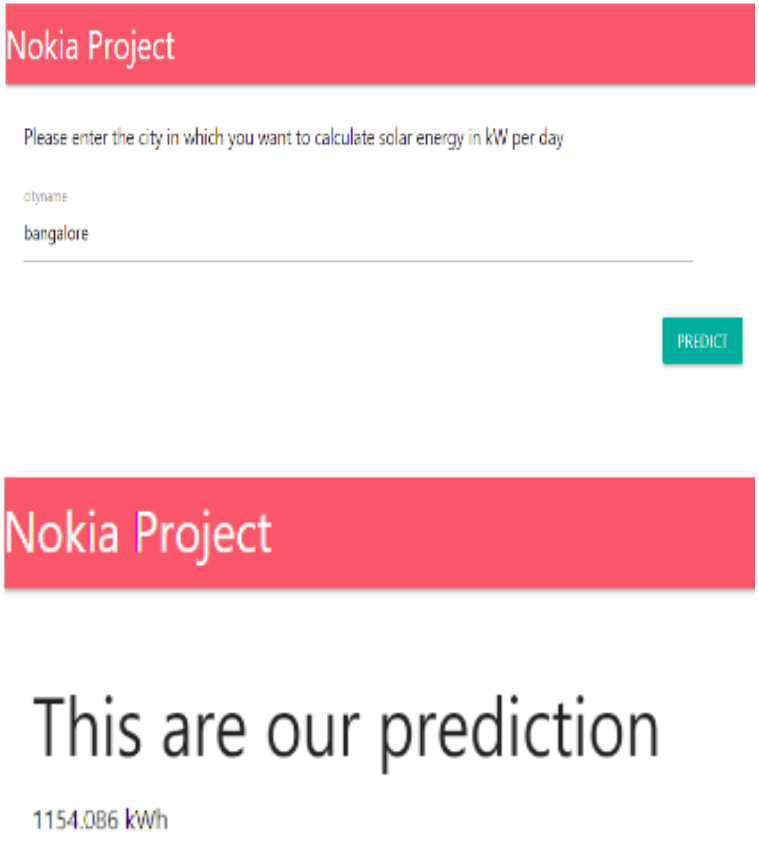

Figure: Front end screenshots

Planned Activities

- Work on Boosting techniques and Ensemble techniques for improving model performance
- Working on Developing model using neural networks
- Application Development: Data visualization of Hourly prediction from the model and provide useful summary for energy generation per year, per month and per day estimation.

The End