SOLAR

November 2, 2020

1 Solar Power Generation Forecast - Checkpoint 1

Kayla Garin CS 458 - Fall 2020

2 Introduction

The purpose of this project is to use the data from Global Energy Forecasting Competition 2014 to develop forcast models and predict a 24h ahead solar power generation on a rolling basis for three solar power plants located in a certain region of Australia. The data has been split into two files: solar_training.csv and solar_testing.csv.

3 Literature Review

https://www.brownanalytics.com/energy_forecasting/#std_pred Article on training data with Neural Networks. Similar to what needs to be accomplished in the project.

A Two-Step Approach to Solar Power Generation Prediction Based on Weather Data Using Machine Learning Seul-Gi Kim, Jae-Yoon Jung and Min Kyu Sim

Seasonal Self-evolving Neural Networks Based Short-term Wind Farm Generation Forecast

4 24 h Ahead Solar Power Generation Forcast Model

[14]:
```
#
#Kayla Garin
#CS 458
#Checkpoint 1 for Solar Power Generation Forcast Model
#

import pandas as pd
from matplotlib import pyplot
from datetime import datetime
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error

# datetime object containing current date and time for current 24h prediction
now = datetime.now()
dt_string = now.strftime("%Y%m%d %H:%M")
```

```
print("date and time =", dt_string) # printing for verification

# load dataset
data = pd.read_csv("solar_training.csv")
df = pd.DataFrame(data,
                 ␣
 ↪columns=['VAR78','VAR79','VAR134','VAR157','VAR164','VAR165','VAR166','VAR167',␣
 ↪'VAR169', 'VAR175', 'VAR178', 'POWER'])

test_data = pd.read_csv("solar_test.csv")


def rmse(actual, predicted):
    from sklearn.metrics import mean_squared_error
    from math import sqrt

    return sqrt(mean_squared_error(actual, predicted))

def add_date_time(_df, startDate):
    "Returns a DF with two new cols : the time and hour of the day"
    t = pd.date_range(start=startDate, periods=_df.shape[0], freq = 'H')
    t = pd.DataFrame(t)
    _df = pd.concat([_df, t], axis=1)
    _df.rename(columns={ _df.columns[-1]: "time" }, inplace = True)
    _df['year'] = _df['time'].dt.year

    return _df



df = add_date_time(df, '20120401')
df = df[~df.year.isin([2011])]

test_data = add_date_time(test_data, '20130701')
test_data = test_data[~test_data.year.isin([2012])]

from sklearn.metrics import mean_squared_error
model_instances, model_names, rmse_train, rmse_test = [], [], [], []


#x_train, y_train = df.drop(columns=['time']), df
#x_test, y_test = test_data.drop(columns=['time']), test_data

pastTwo = df[(df['time'] > '2012-04-01 00:00:00') & (df['time'] <= '2012-04-02␣
 ↪1:00:00')]
```

```
actualData = df[(df['time'] > '2012-04-02 00:00:00') & (df['time'] <=␣
 ↪'2012-04-03 1:00:00')]
pastTwo_target = actualData.POWER

pastTwo = pastTwo.drop(['time', 'POWER'], axis=1)


from sklearn.model_selection import LeaveOneOut
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score

loo = LeaveOneOut()
loo.get_n_splits(pastTwo)

import numpy as np
train_r2_scores = np.array([])
test_r2_scores = np.array([])
train_rmse_scores = np.array([])
test_rmse_scores = np.array([])
predicted_powers = np.array([])
actual_powers = np.array([])

# Train Linear Regression model
for train_index, test_index in loo.split(pastTwo):
    print("Hour index:{}".format(test_index))
    regr = LinearRegression()

    X_train, X_test = pastTwo.iloc[train_index], pastTwo.iloc[test_index]
    y_train, y_test = pastTwo_target.iloc[train_index], pastTwo_target.
 ↪iloc[test_index]
    regr.fit(X_train, y_train)
    y_train_pred = regr.predict(X_train)
    y_test_pred = regr.predict(X_test)

    actual_powers = np.append(actual_powers, y_test.values[0])
    predicted_powers = np.append(predicted_powers, y_test_pred[0])
    print("Actual Solar Generation: {}\tPredicted Solar Generation: {}".
 ↪format(y_test.values[0], y_test_pred[0]))
    print("MAE: {}".format(mean_absolute_error(actual_powers,predicted_powers)))
    print("RMSE: {}".format(rmse(actual_powers, predicted_powers)))


import matplotlib.pyplot as plt
```

```
date and time = 20201102 21:31
Hour index:[0]
Actual Solar Generation: 0.7714102559999999     Predicted Solar Generation:
```

```
0.8140897285608348
MAE: 0.04267947256083493
RMSE: 0.04267947256083493
Hour index:[1]
Actual Solar Generation: 0.613782051    Predicted Solar Generation:
0.6888350697645098
MAE: 0.058866245662672345
RMSE: 0.06105117936509055
Hour index:[2]
Actual Solar Generation: 0.554807692    Predicted Solar Generation:
0.3857310943427308
MAE: 0.09560302966087131
RMSE: 0.10960746458627565
Hour index:[3]
Actual Solar Generation: 0.458910256    Predicted Solar Generation:
0.37132781551998484
MAE: 0.09359788236565728
RMSE: 0.10453716176500395
Hour index:[4]
Actual Solar Generation: 0.19826923100000002    Predicted Solar Generation:
0.3734320631121255
MAE: 0.10991087231495092
RMSE: 0.12197876086685835
Hour index:[5]
Actual Solar Generation: 0.085064103    Predicted Solar Generation:
0.10020134044566387
MAE: 0.09411526650340307
RMSE: 0.11152221488008379
Hour index:[6]
Actual Solar Generation: 0.011474358999999998   Predicted Solar Generation:
-0.007058715770353885
MAE: 0.0833178105415389
RMSE: 0.10348685308565324
Hour index:[7]
Actual Solar Generation: 0.000128205    Predicted Solar Generation:
0.017179012134809213
MAE: 0.0750344351156977
RMSE: 0.09699061253218624
Hour index:[8]
Actual Solar Generation: 0.0    Predicted Solar Generation: -0.03034959445388008
MAE: 0.07006945281994019
RMSE: 0.09200152730523582
Hour index:[9]
Actual Solar Generation: 0.0    Predicted Solar Generation: 0.026686380781139718
MAE: 0.06573114561606014
RMSE: 0.08768733783030504
Hour index:[10]
Actual Solar Generation: 0.0    Predicted Solar Generation: 0.04780175211399751
```

MAE: 0.06410120075223626
RMSE: 0.08483979537009124
Hour index:[11]
Actual Solar Generation: 0.0   Predicted Solar Generation: -0.04358118688402968
MAE: 0.06239119959655238
RMSE: 0.08219642100534205
Hour index:[12]
Actual Solar Generation: 0.0   Predicted Solar Generation:
-0.024526223489477417
MAE: 0.059478509126777386
RMSE: 0.07926419074680913
Hour index:[13]
Actual Solar Generation: 0.0   Predicted Solar Generation:
-0.045503165987980765
MAE: 0.058480270331149053
RMSE: 0.07734297145397928
Hour index:[14]
Actual Solar Generation: 0.0   Predicted Solar Generation: -0.0903209550773183
MAE: 0.06060298264756034
RMSE: 0.07827514215582267
Hour index:[15]
Actual Solar Generation: 0.0   Predicted Solar Generation:
-0.0006741405054846439
MAE: 0.0568574300136806
RMSE: 0.07578976788547005
Hour index:[16]
Actual Solar Generation: 0.0   Predicted Solar Generation:
-0.004080281337099478
MAE: 0.05375289185623465
RMSE: 0.07353353430835434
Hour index:[17]
Actual Solar Generation: 0.0   Predicted Solar Generation: 0.038480718982927264
MAE: 0.05290443780771757
RMSE: 0.07203503664782689
Hour index:[18]
Actual Solar Generation: 0.0   Predicted Solar Generation: 0.06798852879496842
MAE: 0.05369833733336236
RMSE: 0.07182774611786319
Hour index:[19]
Actual Solar Generation: 0.015 Predicted Solar Generation: 0.06452070760086581
MAE: 0.05348945584673753
RMSE: 0.07087932618767
Hour index:[20]
Actual Solar Generation: 0.086282051   Predicted Solar Generation:
0.10554911941678569
MAE: 0.05185981835007316
RMSE: 0.06929880007953808
Hour index:[21]

```
Actual Solar Generation: 0.31525641     Predicted Solar Generation:
0.20186565899893072
MAE: 0.05465667892511843
RMSE: 0.0718920465806912
Hour index:[22]
Actual Solar Generation: 0.514551282    Predicted Solar Generation:
0.5086698476014
MAE: 0.05253601611961763
RMSE: 0.07032250280730265
Hour index:[23]
Actual Solar Generation: 0.734615385    Predicted Solar Generation:
0.5812519777346523
MAE: 0.05673715741735639
RMSE: 0.07562549887244491
Hour index:[24]
Actual Solar Generation: 0.67525641     Predicted Solar Generation:
0.7254597409235015
MAE: 0.056475804357602195
RMSE: 0.07477474449271293
```

5 Methods

In order to train the data, I trained a linear regression model by taking the variables of data from 24 hours prior to the desired data and the power from my desired time frame.

6 Evaluation Results

(RMSE and MAE are printed with each hour result) note: Issues displaying chart with predicted and actual results. Will fix in part 2

The main pro of the prediction model is that it is more on the simple side. At first, I tried to find ways to predict the power by predicting what the variables would be 24h in the future and then using a formula to calculate the power. The con of the prediction model is that it relies on data that the predicted amount solar generation can be extrememly off from the actual amount of solar generation because it calculates the power based on past conditions rather than predicted current ones.

7 Conclusion

In order to more accurately predict the data, it might be more effective to predict the variables. I also need to implement a way to choose a desired date and timeframe. For the intial experimentation results, I only predicted 24h based on the first 24 of the .csv file.