Grid Forecaster

The Application of Machine Learning Models to Forecast Ireland's Electrical Power Demand & Generation from Renewables

Darragh Clabby
20142935
x20142935@student.ncirl.ie
National College of Ireland
Higher Diploma in Computing (AI/ML) 2020/2021

Contents

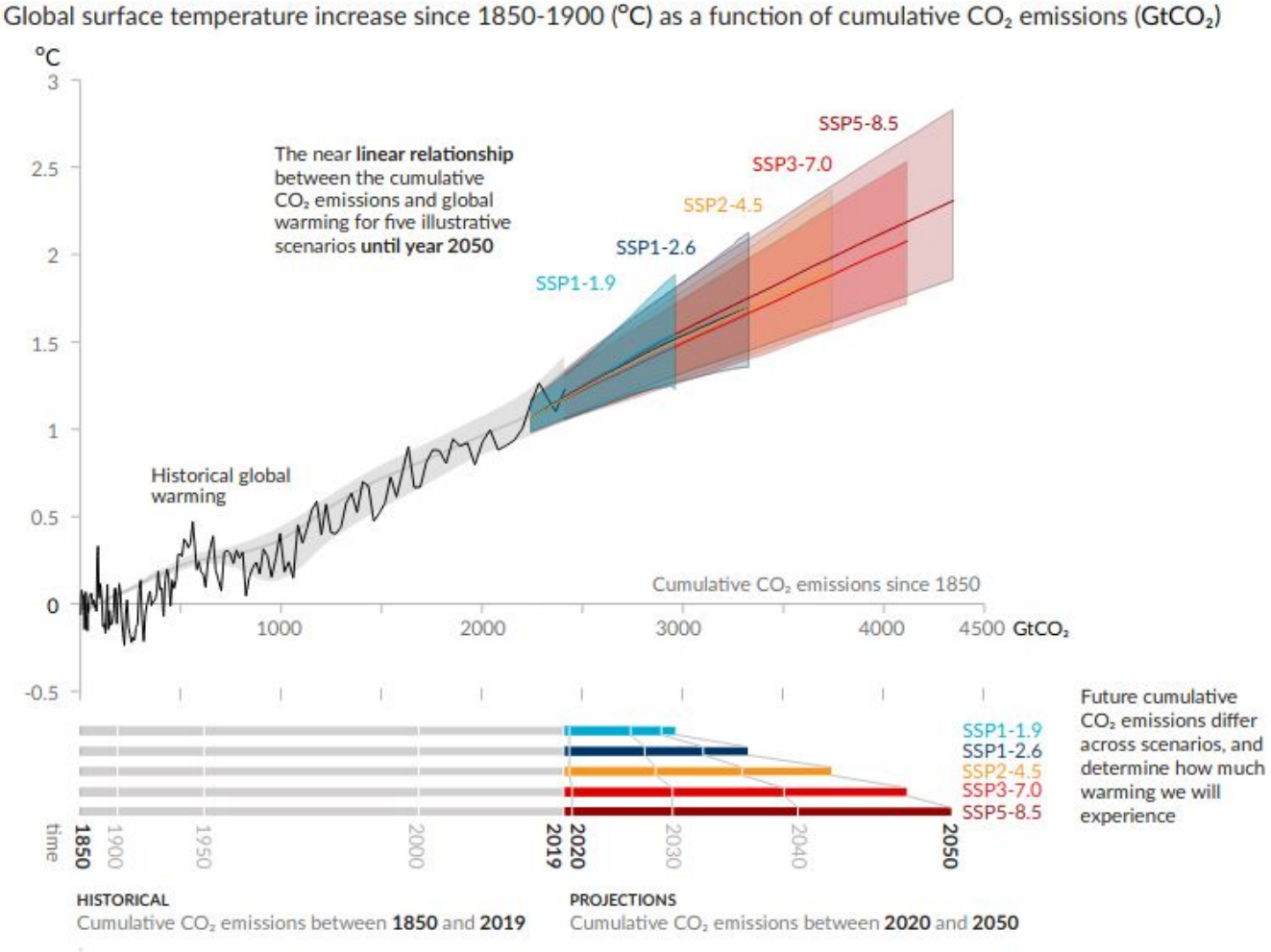
1. Introduction
2. Requirements
3. Design
4. Implementation
5. Testing
6. Evaluation
7. Conclusions
8. Further Work
9. Challenges & Personal Development

1. Introduction

Why Grid Forecaster?

- Personal interest in renewable energy;
- Accurate grid forecasting required to incorporate more renewable energy;
- Machine learning models can be applied to generate forecasts;
- Data available from Eirgrid.

Every tonne of CO_2 emissions adds to global warming



Global surface temperature will continue to increase until at least the mid-century under all emissions scenarios considered. Global warming of 1.5°C and 2°C will be exceeded during the 21st century unless **deep reductions** in carbon dioxide (CO2) and other greenhouse gas emissions occur in the **coming decades**.

IPCC, 2021: Summary for Policymakers. In: Climate Change 2021: The Physical Science Basis. Contribution of Working Group I to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change [Masson-Delmotte, V., P. Zhai, A. Pirani, S. L. Connors, C. Péan, S. Berger, N. Caud, Y. Chen, L. Goldfarb, M. I. Gomis, M. Huang, K. Leitzell, E. Lonnoy, J. B. R. Matthews, T. K. Maycock, T. Waterfield, O. Yelekçi, R. Yu and B. Zhou (eds.)]. Cambridge University Press. In Press.

2. Requirements
User Requirements

Consumer: “As an electricity consumer I want to avail of **new tariffs** that may be offered by electricity suppliers, and which are enabled by **smart meters**. The Grid Forecaster system will allow me to identify times when electricity costs are is likely to be lowest under these tariffs (e.g. low demand, high supply from wind) and **schedule charging** of my **electric vehicle** and/or **home battery** accordingly”;

Energy Trader: “As an energy trader I want to generate forecasts over a **longer horizon** than that presently available from Eirgrid”; “As an energy trader I want to use machine learning models to generate **more accurate** forecasts than those presently available from Eirgrid”;

Grid operator: “As a grid operator I want to use machine learning models to inform **grid operations planning**”;

Renewable energy project developer: “As a renewable energy project developer I want to understand how short term grid characteristics will affect my **project’s viability**”;

Energy sector researcher: “As an energy sector researcher I want to **understand** how machine learning models can be used to generate forecasts of Ireland’s electricity system”.

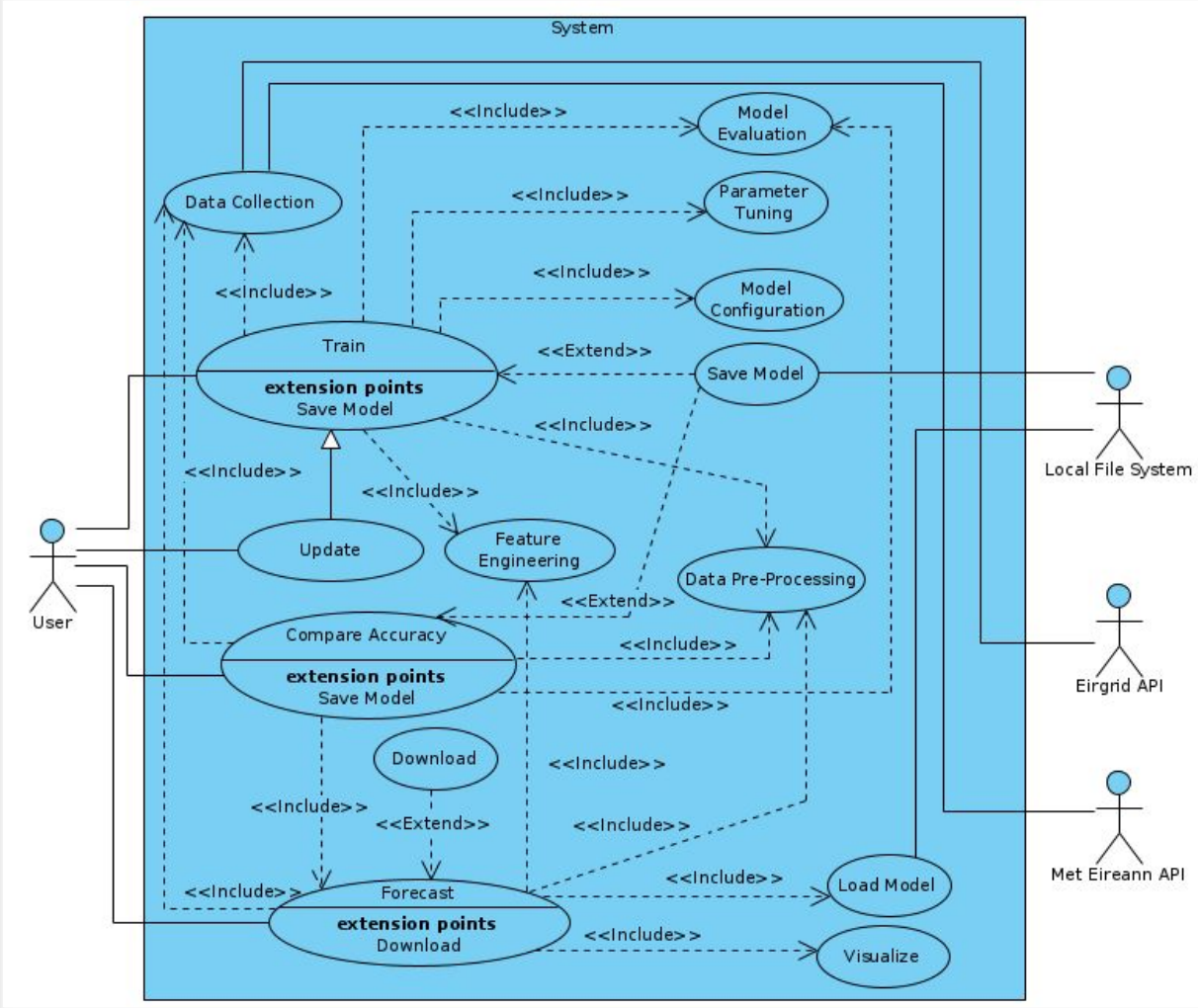
Machine learning researcher: “As a machine learning researcher I want to train **custom models** to generate forecasts of Ireland’s electricity system”;

2. Requirements

Functional Requirements

Use cases

1	Train a Model
1.1	Data Collection
1.2	Data Pre-Processing
1.3	Feature Engineering
1.4	Model Configuration
1.5	Model Training
1.6	Parameter Tuning
1.7	Model Evaluation
2	Forecast
3	Compare Accuracy
4	Visualization
5	Download
6	Update



3. Design
Design Overview

The system will be developed in two parts:

- a full version (available as source code);
  - All use cases implemented;
- a light version (GUI, available as a web application).
  - Restricted to Forecast and Model Evaluation (Test) use cases;

3. Design

System Architecture

Data downloaded & stored locally;

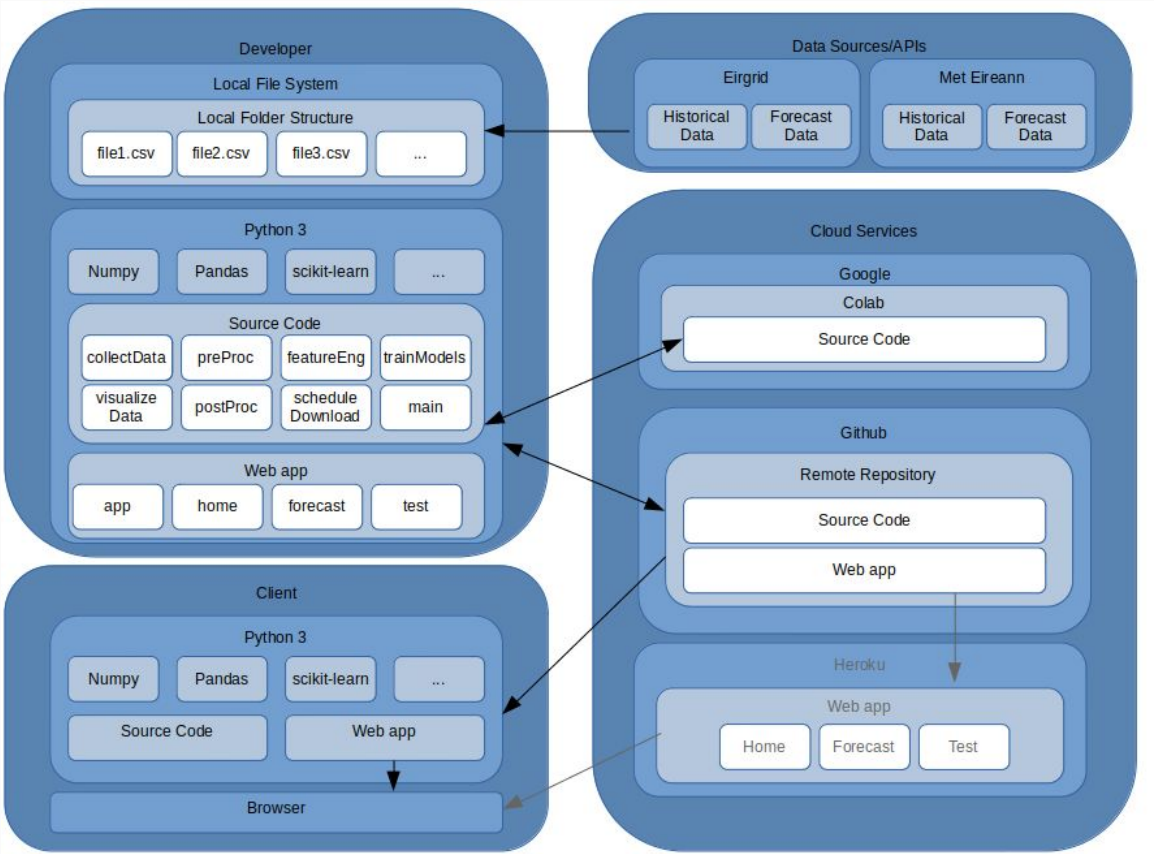
Source code (full version) & web app (light version) developed locally in Python 3;

Models trained either locally or on Google Colab;

Local Git repository pushed to Github for tracking & backup;

Clients can download source code from Github and run both full & light versions locally;

Web app will be deployed via suitable platform, e.g. Heroku (to be completed as part of future work).



3. Implementation
Full Version

Implemented by main.py:

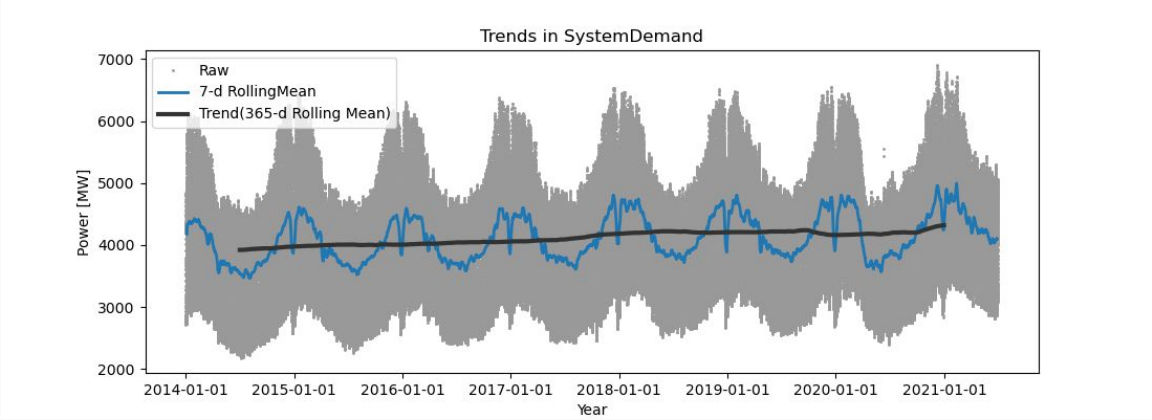
- Define inputs
- Load & clean grid data
  - Download
  - Clean
  - Detrend
- Feature Engineering
- Split into training & test sets
- Standardize features
- Save features & labels
- Train models
- Test models
- Integrate labels
- Plot time series
- Save models
- Generate Forecast

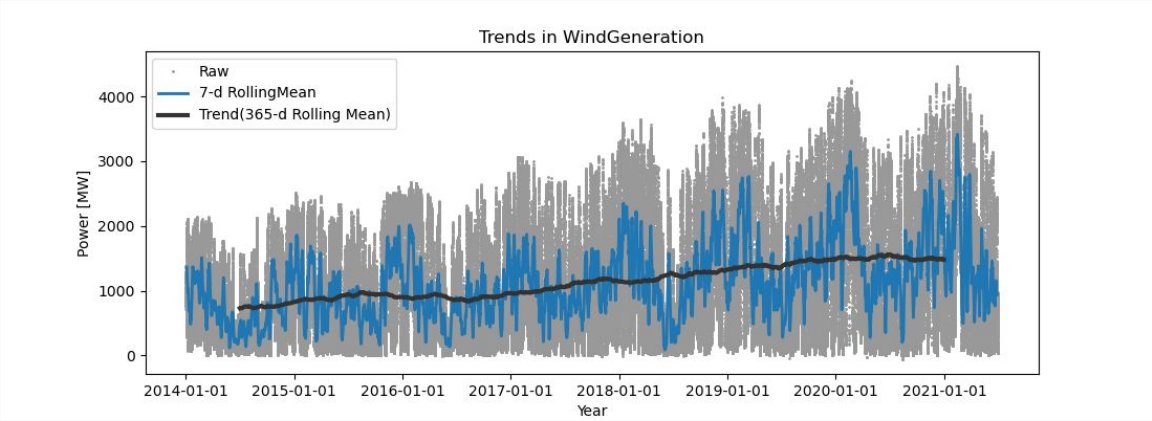
```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Mon Jul 26 18:54:04 2021

@author: dclabby

"""
import numpy as np
from datetime import datetime, timedelta #, date
import pickle

from preProc import loadEirgridData, cleanGridData, detrendData, readMetHistorical
from featureEng import gridFeatureEng, standardizeFeatures
from trainModels import trainModels, testModels, regression_results
from collectData import collectEirgridData
from visualizeData import plotTestPred, plotForecast
from postProc import intTestPreds, intForecast

# Inputs:
forecastVar = 'SystemDemand' #'WindGeneration' #        'SystemGeneration' #
modelsToTrain = {'LR':{},
                 'NN':{'tol':25, 'verbose':True, },
                 'KN':{'n_neighbors': 4},
                 'RF':{'n_estimators': 10}}
# modelsToTrain = {'LR':{},
#                  'NN':{'hidden_layer_sizes':[(100), (100, 100), (100, 100, 100)], 'tol':[25], 'verbose'
#                  'KN':{'n_neighbors': [4,8]},
#                  'RF':{'n_estimators': [10,20]}}
saveFeaturesLabels = False
saveModels = False
testDatePlot = datetime(2021,2,23)
generateForecast = False

# Load/Download Grid data
try:
    print('Attempting to load grid data from local file system...')
    gridData = loadEirgridData(forecastVar) # try to load data from previously downloaded files
    print('Grid data loaded successfully!')
except: # if files do not exist the data will need to be downloaded from the eirgrid dashboard
    print('Grid data not found on local file system. Downloading data from source...')
    collectEirgridData(datetime(2014,1,1), datetime(2021,6,30), forecastVar, saveFolder='data/Eirgrid')
    gridData = loadEirgridData(forecastVar)

```

3. Implementation
Full Version

Implemented by main.py:

- Define inputs
- **Load & clean grid data**
  - **Download**
  - **Clean**
  - **Detrend**
- Feature Engineering
- Split into training & test sets
- Standardize features
- Save features & labels
- Train models
- Test models
- Integrate labels
- Plot time series
- Save models
- Generate Forecast

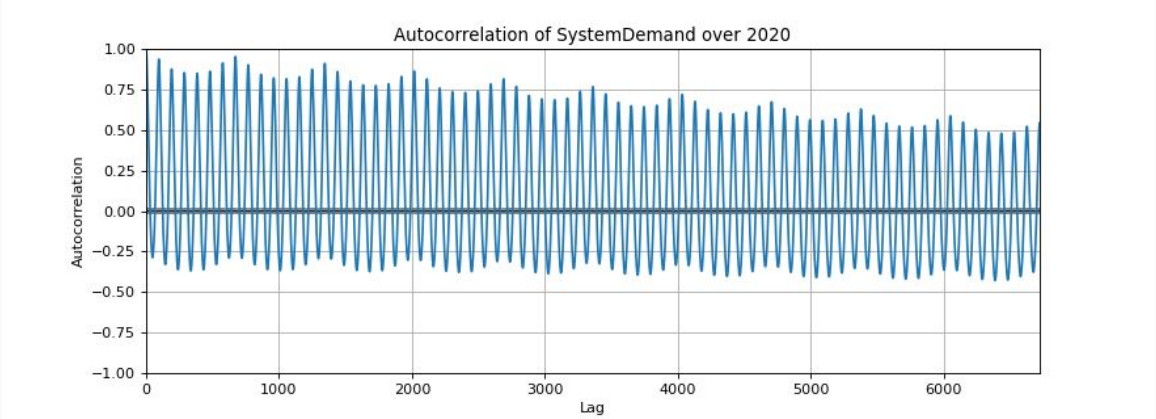


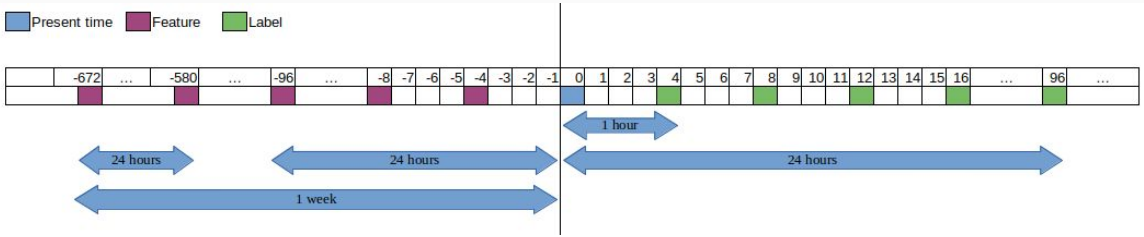


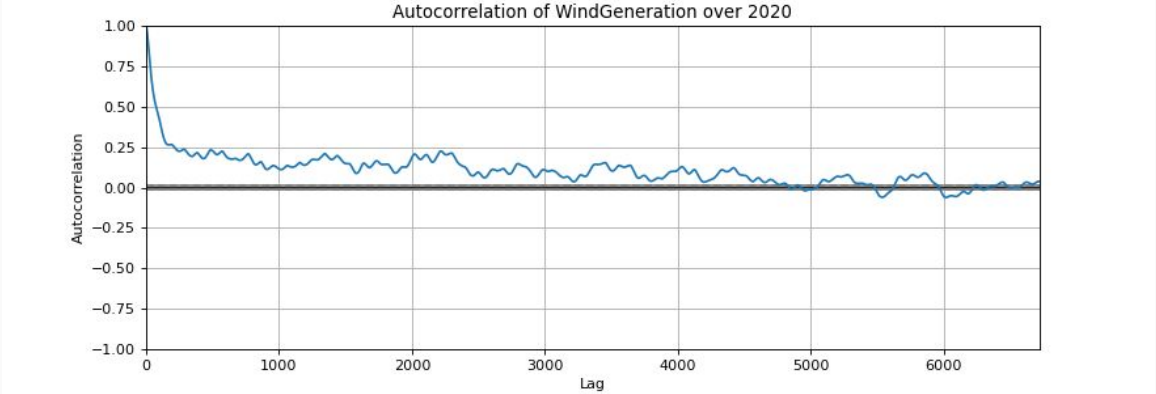
3. Implementation
Full Version

Implemented by main.py:

- Define inputs
- Load & clean grid data
  - Download
  - Clean
  - Detrend
- **Feature Engineering**
- Split into training & test sets
- Standardize features
- Save features & labels
- Train models
- Test models
- Integrate labels
- Plot time series
- Save models
- Generate Forecast



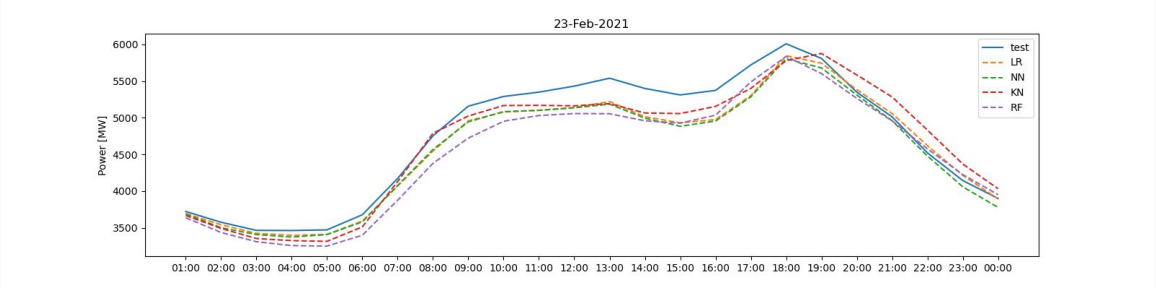


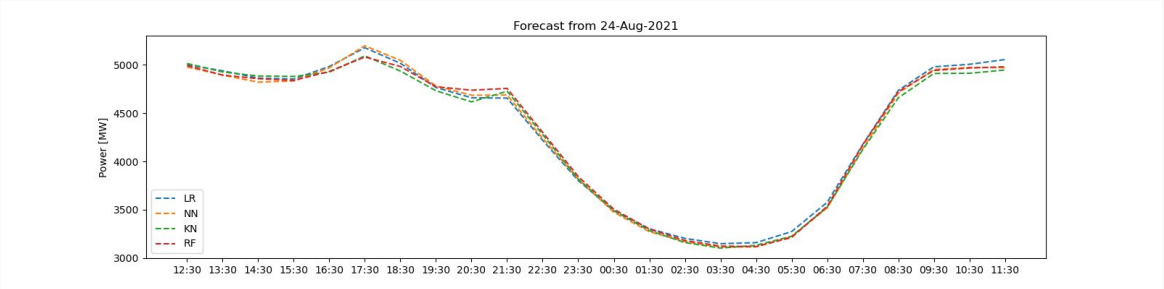


3. Implementation
Full Version

Implemented by main.py:

- Define inputs
- Load & clean grid data
  - Download
  - Clean
  - Detrend
- Feature Engineering
- Split into training & test sets
- Standardize features
- Save features & labels
- Train models
- Test models
- Integrate labels
- **Plot time series**
- Save models
- **Generate Forecast**





main.py
<Demonstration>

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Mon Jul 26 18:54:04 2021

@author: dclabby

"""
import numpy as np
from datetime import datetime, timedelta #, date
import pickle

from preProc import loadEirgridData, cleanGridData, detrendData, readMetHistorical
from featureEng import gridFeatureEng, standardizeFeatures
from trainModels import trainModels, testModels, regression_results
from collectData import collectEirgridData
from visualizeData import plotTestPred, plotForecast
from postProc import intTestPreds, intForecast

# Inputs:
forecastVar = 'SystemDemand' #'WindGeneration' #        'SystemGeneration' #
modelsToTrain = {'LR':{},
                 'NN':{'tol':25, 'verbose':True,},
                 'KN':{'n_neighbors': [4,8]},
                 'RF':{'n_estimators': [10,20]}}
# modelsToTrain = {'LR':{},
#                  'NN':{'hidden_layer_sizes':[(100), (100, 100), (100, 100, 100)], 'tol':[25], 'verbose'
#                  'KN':{'n_neighbors': [4,8]},
#                  'RF':{'n_estimators': [10,20]}}
saveFeaturesLabels = False
saveModels = False
testDatePlot = datetime(2021,2,23)
generateForecast = False

# Load/Download Grid data
try:
    print('Attempting to load grid data from local file system...')
    gridData = loadEirgridData(forecastVar) # try to load data from previously downloaded files
    print('Grid data loaded successfully!')
except: # if files do not exist the data will need to be downloaded from the eirgrid dashboard
    print('Grid data not found on local file system. Downloading data from source...')
    collectEirgridData(datetime(2014,1,1), datetime(2021,6,30), forecastVar, saveFolder='data/Eirgrid')
    gridData = loadEirgridData(forecastVar)

```

3. Implementation
Light Version

Implemented by app.py:

- Uses flask to create dynamic web app;
- Renders the following pages:
  - Home
  - Forecast
  - Test
- Home displays information about Grid Forecaster, and provides links to the source data & source code;
- Forecast allows user to generate a forecast for the next 24 hours;
- Test allows users to assess a model and compare time series of a model's forecast to corresponding measured data.

Welcome to Grid Forecaster 1.0

Grid Forecaster uses machine learning algorithms to generate forecasts for Ireland's electricity grid. This GUI provides the ability to forecast and test pre-trained models. At present, forecasts are available for total system power demand and generated wind power, across the island of Ireland (ROI and NI). The available models include:

- Linear Regressor
- Multi-Layer Perceptron Neural Network
- K Neighbore Regressor
- Random Forest Regressor

Models were tarined using data from Eirgrid's Smart Grid Dashboard. Further details and source code are available from github.com/dclabby/gridForecaster

---

4. Testing

SCENARIO ID			1	SCENARIO DESCRIPTION		Use Case 1: Train a Model				
S.NO	ID	TEST CASE DESCRIPTION	PRECONDITION	TEST DATA	EXPECTED RESULT	POSTCONDITION	ACTUAL RESULT	STATUS	DEFECT ID	COMMENTS
1	1.6	Multiple models will be trained with default parameters	Tester edits modelsToTrain and saveModels appropriately. Tester saves script.	modelsToTrain = {LR:{},NN:{},RF:{},KN:{},SV:{}}	All models will be trained without hyperparameter tuning	Script completes without error; models trained and stored in models dictionary;	Error displayed during training of support vector regressor: "got an array of shape {} instead.".format(shape)) ValueError: y should be a 1d array, got an array of shape (209660, 24) instead.	Fail	1	SV does not handle multiple label regression. Test was repeated with SV omitted and expected result was achieved. Recommend developing wrapper for SV to allow for multi-label regression

SCENARIO ID			2	SCENARIO DESCRIPTION		Use Case 2: Forecast				
S.NO	ID	TEST CASE DESCRIPTION	PRECONDITION	TEST DATA	EXPECTED RESULT	POSTCONDITION	ACTUAL RESULT	STATUS	DEFECT ID	COMMENTS
2	2.6	Use app.py to generate a forecast without connection to internet	Tester disconnects internet connection	Forecast Variable='SystemDemand'; Forecast Model='LR'	A warning message will be displayed to inform the user that connection to the Eirgrid dashboard failed and that it will retry in 10s	Forecast generated when tester reconnects internet	Warning message was displayed in console running app.py. The script completed and generated a forecast on reconnection of the internet	Fail	2	Message displayed in console, not in GUI. Download completed once connection re-established but browser timed out so forecast was not displayed.

6. Evaluation

Forecast variables:

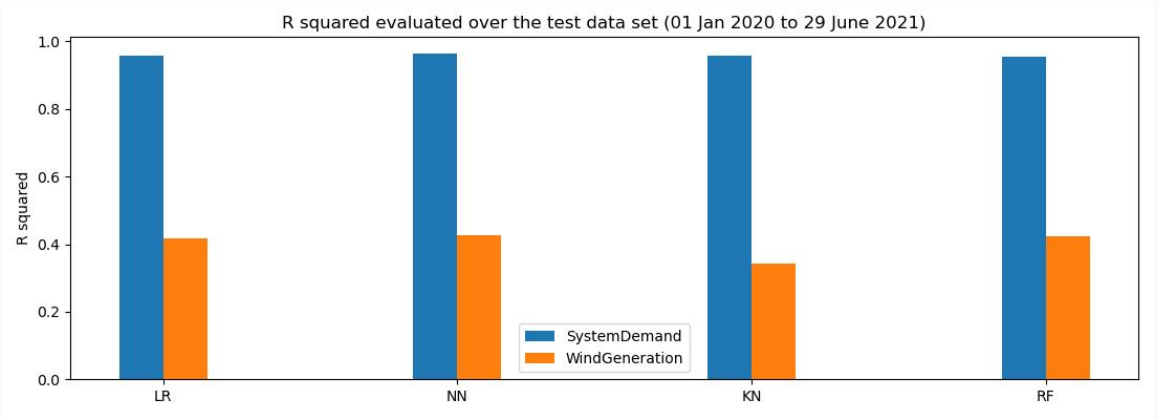
- System Demand
- Wind Generation

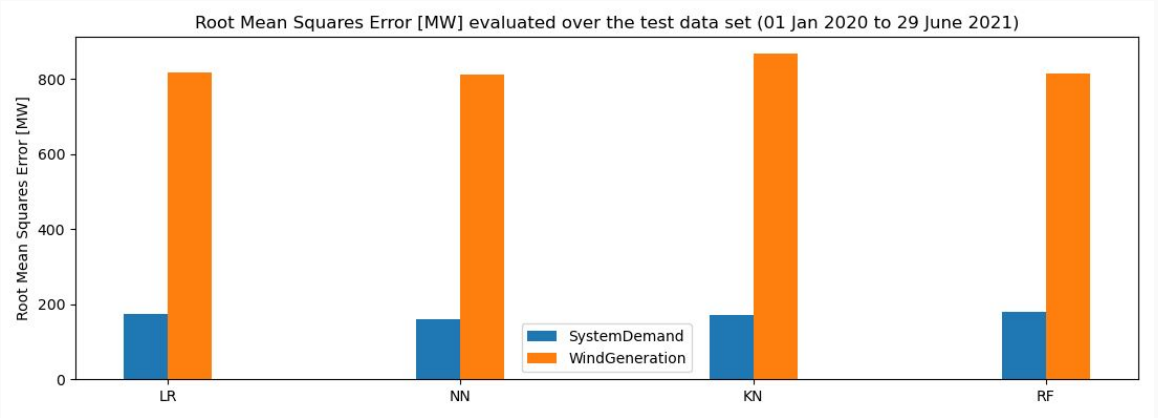
Models evaluated:

- Linear Regressor
- MLP Regressor (Neural Network)
- K Neighbors Regressor
- Random Forest Regressor

Evaluation metrics:

- **R^2 (Coefficient of Determination)**
- Explained Variance
- Mean Absolute Error
- Mean Squared Error
- **Root Mean Squared Error**





7. Conclusions

- Strong autocorrelation of system demand with **previous day** and **previous week** was observed;
- **Excellent accuracy** of system demand forecast was achieved:
  - **R^2 > 0.95** for all models;
  - **RMSE < 180MW** for all models;
- No significant autocorrelation observed for wind generation. Implies **other variables** required (e.g. weather data: wind speed, direction, etc.);
- Relatively **poor accuracy** observed for forecasts of wind generation:
  - **R^2 < 0.43** for all models;
  - **RMSE > 810MW** for all models;
- Given high accuracy for system demand, it significant improvement on wind forecast expected with incorporation of weather data.

8. Further Work

- Improve accuracy of wind forecasts by incorporating meteorological data:
  - Historical weather data has been downloaded & processed;
  - Forecast weather data automatically scheduled for download every day at 00:05 (script running on Raspberry Pi);
- Evaluate Eirgrid forecasts and compare to Grid Forecaster
  - Eirgrid forecasts also scheduled for automatic download each day
- Deploy web application:
  - Light version was intended to be deployed as web app via Heroku. However, time constraints meant that this was not possible.

9. Challenges & Personal Development
Challenges

- Time constraints:
  - Insufficient time to fully develop feature engineering specific to wind. Resulted in poor accuracy of wind forecasts since inappropriate feature structure was used;
  - Some functionality remains to be completed, e.g. deployment of web app, comparison of forecast accuracy to Eirgrid forecasts.
- Google Colab
  - File sync issues between Colab & Drive
  - Hardware accelerator (facilitates use of GPUs, TPUs) not discovered until end of project

9. Challenges & Personal Development
Personal Development

Personal Development

- Developed an end to end ML system
- Improved familiarity with Python & libraries (e.g. scikit-learn)
- Gained experience with Google Colab and learned to overcome challenges
- Used Flask to develop a GUI for python scripts
- Developed a dynamic website and used chartjs to display time series plots

Thanks!

Contact:

Darragh Clabby
x20142935@student.ncirl.ie

Source Code:
github.com/dclabby/gridForecaster