Precipitation Prediction in New York City

Zheyu Gu (zg2412)

Machine Learning for Environmental Engineering and Science

EAEE 4000 Project: Fall 2021

Professor: Pierre Gentine

Contents:

Introduction 3
Data preparation 3
Neural Network 7
Tree-based method 12
Random Forest 13
Xgboost 16
Comparison between different algorithms 19
Conclusion 20
Code 20
Reference 21

Introduction

Climate is a complex system, there are so many factors and it is difficult for people to analyze those nonlinear relations by themselves. Machine learning is good at finding the relations in system by applying complex mathematical calculation to big dataset. It's possible to quickly and automatically produce models that can analyze and deliver faster, more accurate results. And by building precise models, there are better chances of identifying profitable opportunities – or avoiding unknown risks. Over decades, people had developed a large number of machine learning algorithms. In this project I want to focus on tree-based method and neural network. I will use some climate parameters such as pressure, temperature or wind speed to predict the daily precipitation in New York City. There will be a lot of uncertainty forecasting the weather. What I am trying to do is more like finding the nonlinear relations between precipitation and all other parameters and see how the prediction fits the past actual data. Another purpose of the project is to see how is the performance of different machine learning methods when applying to climate data something mysterious and uncertain.

Data preparation

There are a lot of sources for climate data and I look for the data from NASA Prediction of Worldwide Energy Resource. The website provides solar and meteorological datasets from NASA research for support of renewable energy, building energy efficiency and agricultural needs. It is supported by NASA Earth Science's Applied Sciences Program. It has a data access viewer which is Responsive web mapping application providing data sub setting, charting, and visualization tools in an easy-to-use interface in Figure 1. In this interface, I just need to enter the date, location and select

parameters I am interested in, it will then allow me to download the data as excel sheet. In enter the location of New York City with Latitude of 40.7306 and longitude of -73.9352. For machine learning application, usually significant large datasets are required. As a result, I picked 12 parameters and their abbreviation and units are show in Table 1. To make sure I have enough data for a good prediction I downloaded daily data from January 1st 2000 until the same date in 2021.

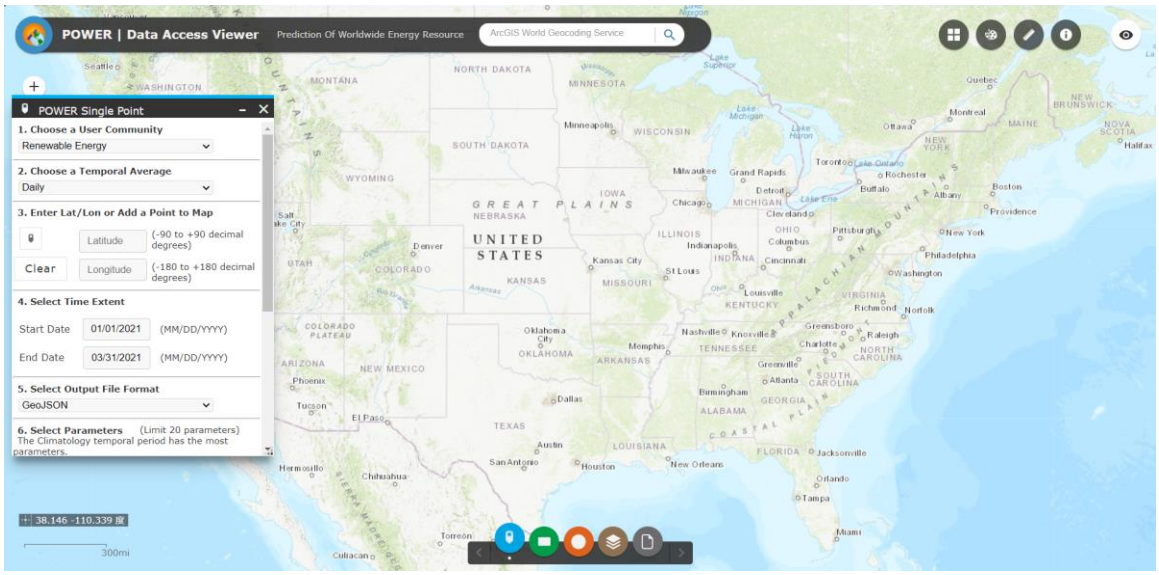

Figure 1 Data access viewer interface from NASA Prediction of World-Wide Energy Resource

Abbreviations	Parameters	Units
PRECTOTCORR	MERRA-2 Precipitation Corrected	mm
QV2M	MERRA-2 Specific Humidity at 2 Meters	g/kg
PS	MERRA-2 Surface Pressure	kPa
WS50M	MERRA-2 Wind Speed at 50 Meters	m/s
T2M	MERRA-2 Temperature at 2 Meters	°C
TS	MERRA-2 Earth Skin Temperature	°C
ALLSKY_SFC_SW_DWN	CERES SYN1deg All Sky Surface Shortwave Downward Irradiance	kW-hr/m^2/day
WS10M	MERRA-2 Wind Speed at 10 Meters	m/s
ALLSKY_SFC_UV_INDEX	CERES SYN1deg All Sky Surface UV Index	dimensionless
CLRSKY_SFC_PAR_TOT	CERES SYN1deg Clear Sky Surface PAR Total	W/m^2
ALLSKY_SFC_PAR_TOT	CERES SYN1deg All Sky Surface PAR Total	W/m^2
WS2M	MERRA-2 Wind Speed at 2 Meters	m/s

Table 1 Abbreviations of parameters and corresponding units

In Table 1 some interpretations look unfamiliar. MERRA is an abbreviation for Modern Era Retrospective-Analysis for Research and Applications. The dataset originates from the Global Modeling and Assimilation Office of NASA. Elevation from MERRA-2: Average for 0.5 x 0.625 degree lat/lon region = 10.17 meters. CERES stands for The Clouds and the Earth's Radiant Energy System. The CERES synoptic 1° (SYN1deg) product incorporates derived fluxes from the geostationary satellites (GEOs) to account for the regional diurnal flux variations in between Terra and Aqua CERES measurements. PAR is Photosynthetically active radiation.

The following figures show the pattern of daily precipitation range from January 1st to December 31st in 2018,2019 and 2020 respectively. It shows that the daily precipitations through the year share the similar patterns. This similar pattern is found in data of all selected years. Otherwise, they cannot be formulated together. For example, the precipitation for a drought year should be predicted separately since its relationships with other climate parameters will be different from a different year.

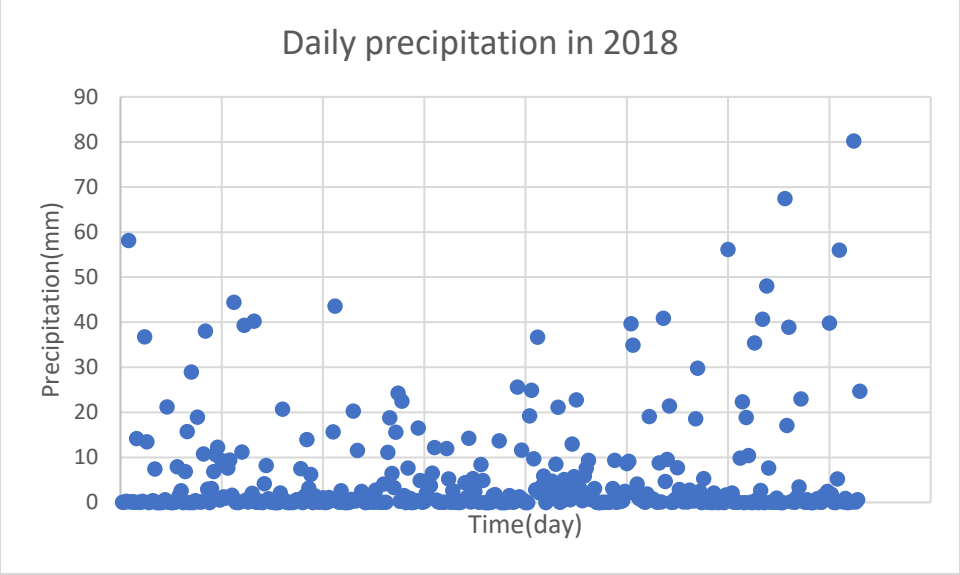

Figure 2 Daily precipitation in 2018

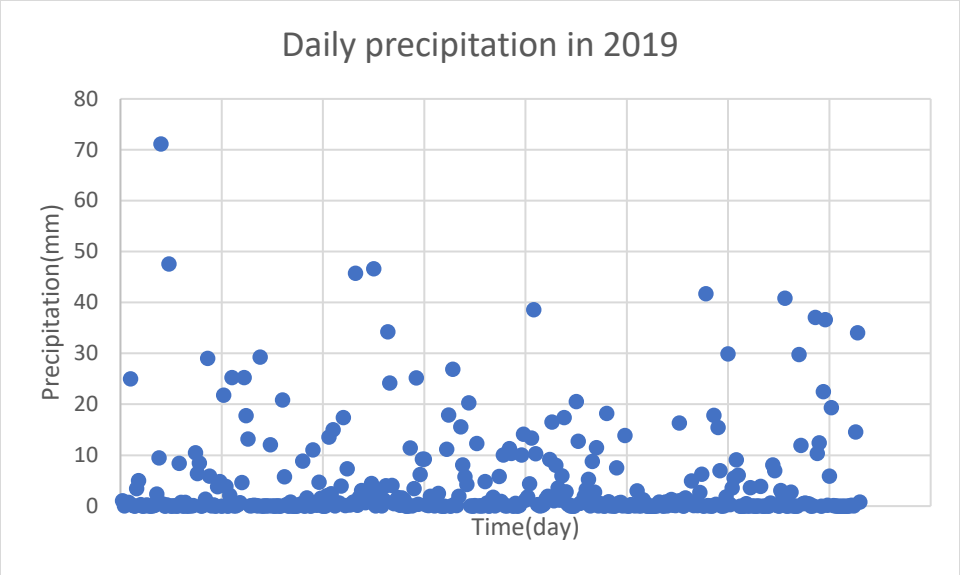

Figure 3 Daily precipitation in 2019

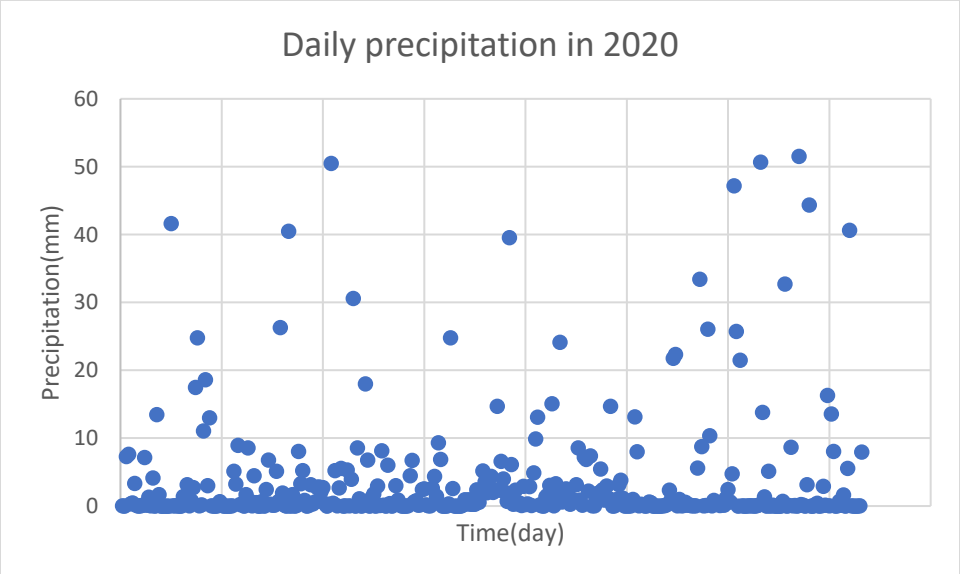

Figure 4 Daily precipitation in 2020

Neural Network

Overview of model:

Neural Network is a machine learning algorithm mimic the function of human brain. In our brain, we have neurons connect to each other by synapses and it can carry information when activated. The artificial neural network works in the similar way. There will be some nodes, hidden layers, input and output layer. The nodes in different layers will connect to node in subsequent nodes with some weights and bias. Each node has formula like this:

$\sum w_i x_i + \text{bias} = w_1x_1 + w_2x_2 + w_3x_3 + \text{bias}$

w is weight and x is input and in this project, it will be all parameters other than precipitation. Then it will be passed through an activation function. The nodes will be activated if the output value satisfied specific threshold value and passed to the next layer. The output of previous layer will be input of latter layer. The same procedure will be processed between each layer over and over and eventually pass to output layer. When we train the model, we want to evaluate the accuracy by a cost or a loss function. Mean squared error is the one people commonly used. Its formular is shown below.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

i is the index of the sample, Y-hat is the predicted outcome, Y is the actual value, and n is the number of samples. The purpose of this process is to have a prediction as close to actual value as possible. A minimized cost function indicates the best fit of predicted and actual value. In general, neural network can model the relations between

inputs and outputs that are complex and nonlinear. We are looking the best weights and bias that can minimize the error.

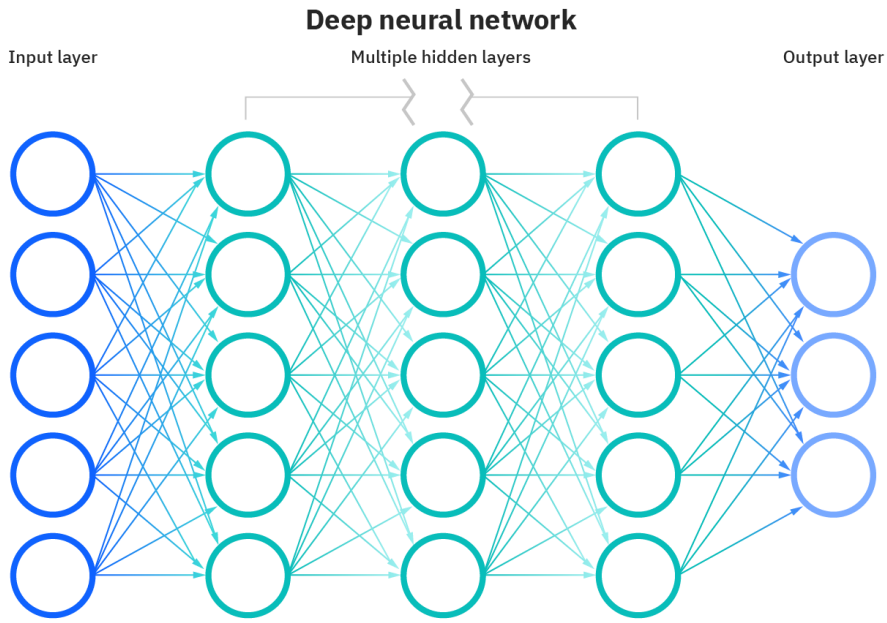

Figure 5 Structure of deep neural network

Neural Network is built by Keras in Jupyter Notebook. In my model, the first thing I did is to define my inputs and outputs. For this project, the inputs are all parameters other than precipitation: QV2M, PS, WS50M, T2M, TS, ALLSKY_SFC_SW_DWN, WS10M, ALLSKY_SFC_UV_INDEX, CLRSKY_SFC_PAR_TOT, ALLSKY_SFC_PAR_TOT, WS2M. The output is PRECTOTCORR. Then I normalized data because my parameters have different scale and ranges. It is done by subtract the mean of the parameter and divided by its standard deviation. At this point I have all my data to put in the model. I will need to construct the network. My initial hyper parameters include 3 hidden layers, 32 neurons, using ReLU

activation functions. For training, I will use mean absolute loss function and adam optimization. 80% of total available data will be used for training and the remaining 20% will be test data. I will change some hyper parameters and see which models have best R^2 or have the best fit of predicted and actual precipitation. To visualize the process of training, I will plot the error verses epoch diagram. Then in order to prevent overfitting, early stop will be used and the patience is set to 20.

Results:

The result of initial model is shown in this section. Figure 6 shows the process of training. The mean absolute value is 3.692. R^2 is 0.147. Figure 7 shows the actual verses predicted value of precipitation. Figure 8 shows the range of prediction error.

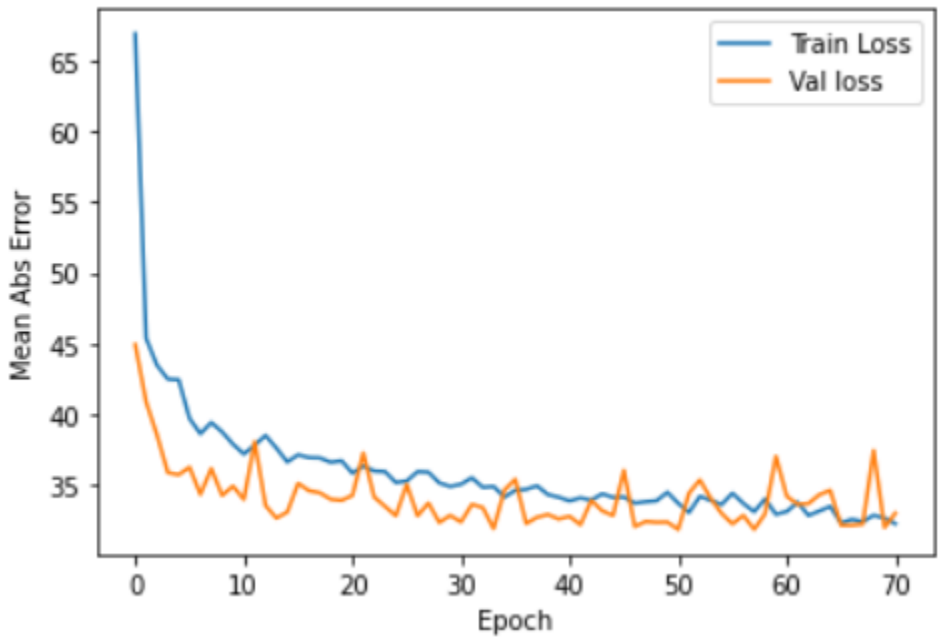

Figure 6 Mean absolute error vs Epoch

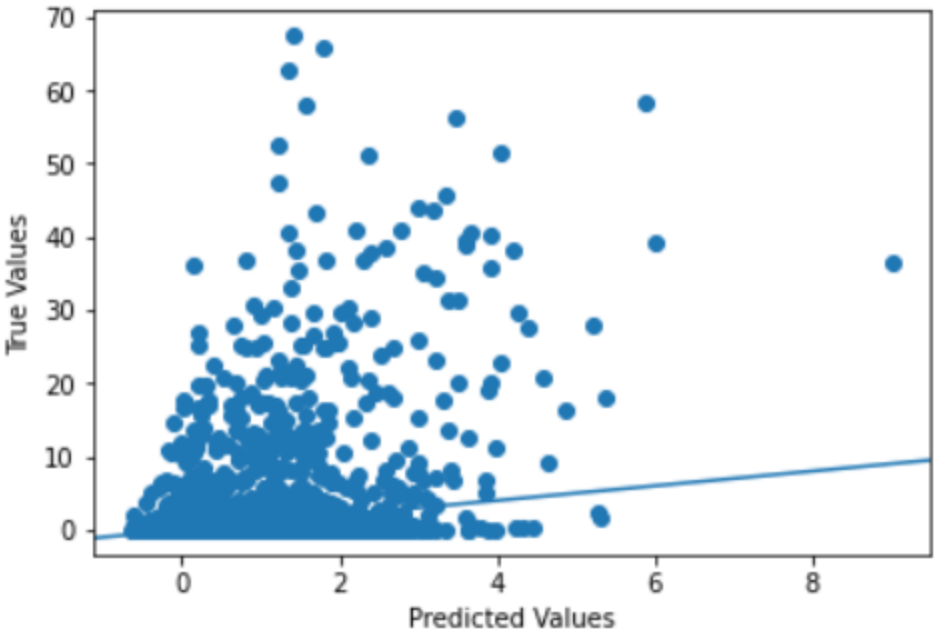

Figure 7 Actual vs predicted value

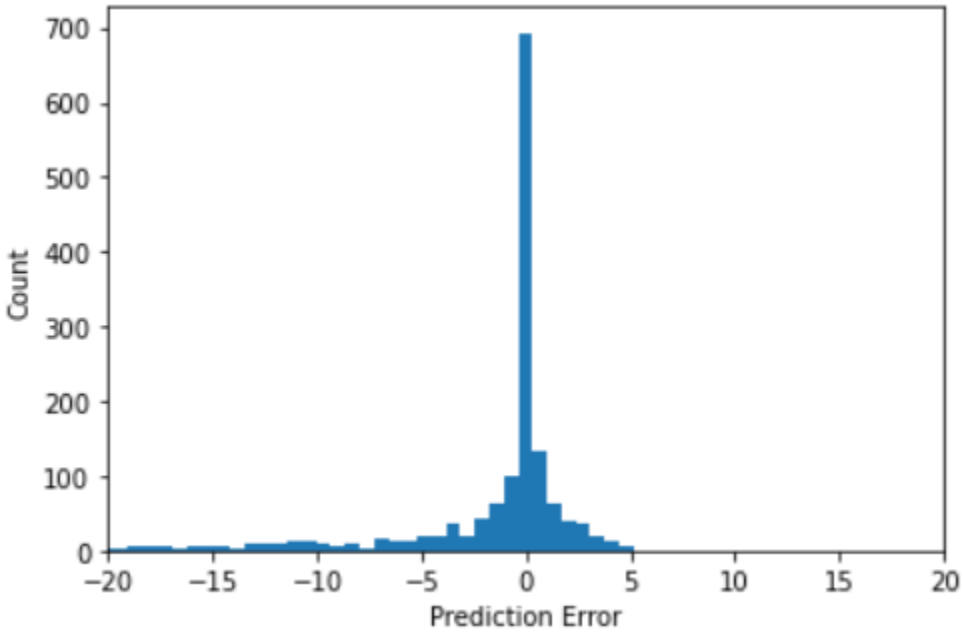

Figure 8 Range of error

I also changed the structure of neural network and the performance of each model is show below. In Table 2, number of hidden layers is changed and all other hyperparameters are kept constant. The results shows that when we have two hidden layers it will have the best performance.

Number of hidden layers	MAE	R^2
1	3.549	0.0786
2	3.498	0.171
3	3.692	0.147
4	3.559	0.0873

Table 2 Performance of models with different number of hidden layers

In Table 3, number of neurons in each layer is changed and number of hidden layers is set to be 2 for constant. It shows 128 and 256 neurons structure have relatively better performance than other models.

Number of neurons	MAE	R^2
4	4.212	0.0540
8	3.610	0.0106
16	3.596	0.136
32	3.498	0.171
64	3.465	0.103
128	3.370	0.222
256	4.006	0.281

Table 3 Performance of models with different number of neurons

Tree-based method

Overview of model:

Tree-based is one of the supervised machine learning which can solve classification and regression tasks by building a structure like a tree. It starts at the top node and it will divide into two branches for each level. Eventually, it will reach the last level where the branches do not split anymore and that will be the decision or called leaves. In each level, there are conditions to decide which branch to go. The process repeated until the end and it tells us the prediction. The configuration of tree based model is shown in Figure 9.

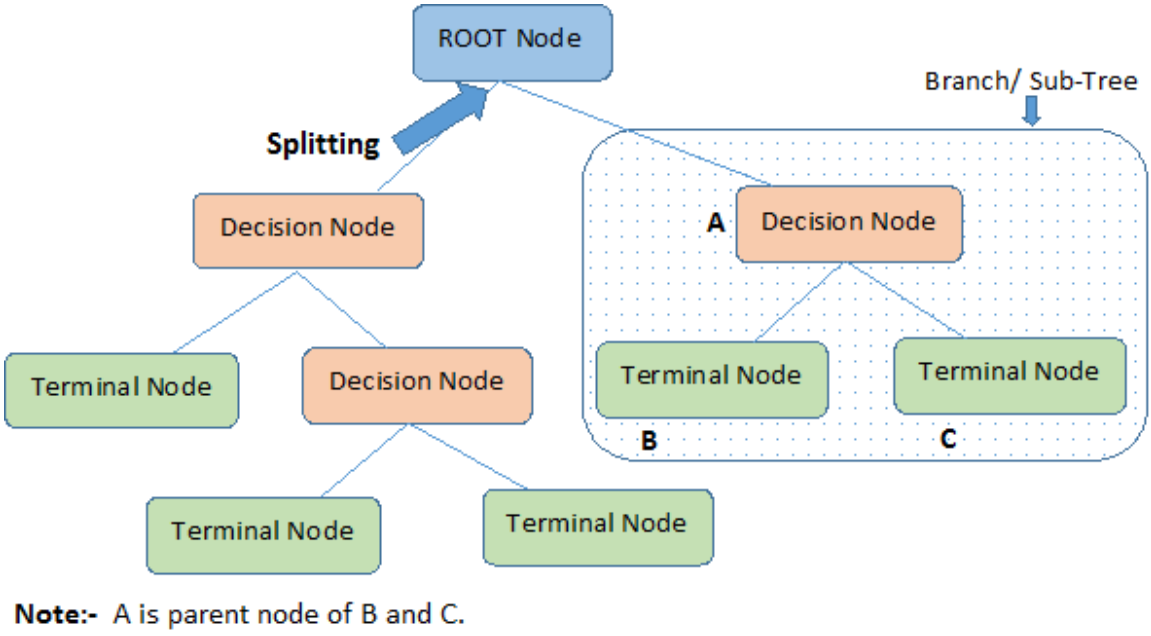

Figure 9 Structure of tree-based method

Random Forest

Random forest consists of a large number of decision tree models that work as an ensemble. For classification task, the output will be the one selected by most trees and for regression task, the prediction of individual tree is returned. One of the outstanding advantages of random forest is that the models or trees are uncorrelated. It can have ensemble predictions which are more accurate than any of the individual prediction. In other words, tree models protect each other from their own error. Bagging or bootstrap aggregation is an important feature of random forest. It allows each tree to randomly sample from dataset with replacement which results in different trees.

For the project, random forest model is built in Keras in Jupyter Notebook. The data preparation process is similar to neural network. Then I need to build the model. It is a decision tree using the CART algorithm and it is not sensitive to input data. As a result, there is no need to scale and normalize the data. The initial model set up includes 1000 estimators, mean squared error, max depth (10). Number of estimators show the number of trees to use and max depth indicates how may levels are in each tree.

Results:

After training, the MSE for initial model is 334.00, MAE is 17.761 and R^2 is -3.613. Figure 10 shows the actual verses predicted value of precipitation. Figure 11 shows the range of prediction error.

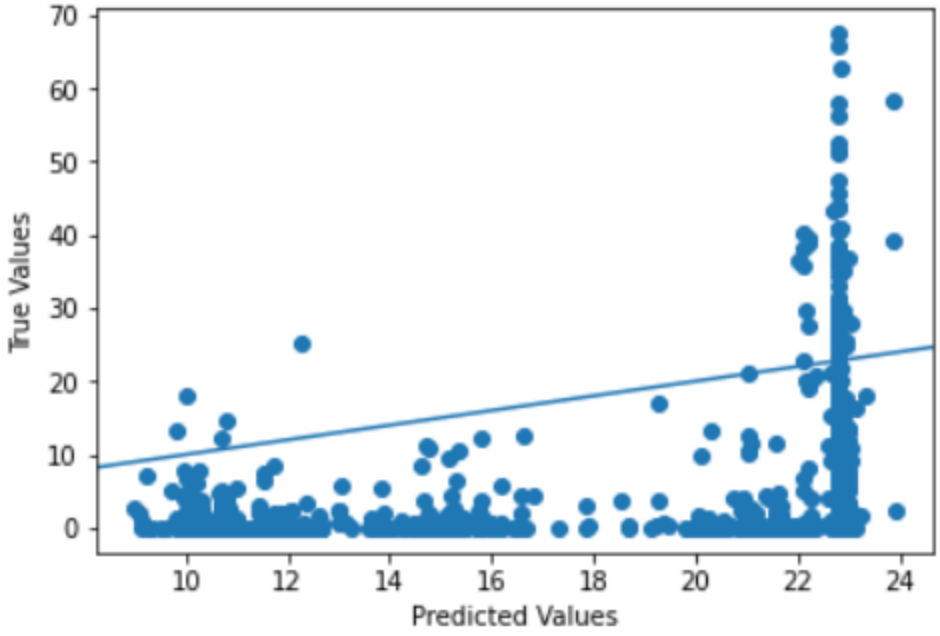

Figure 10 Actual vs predicted value

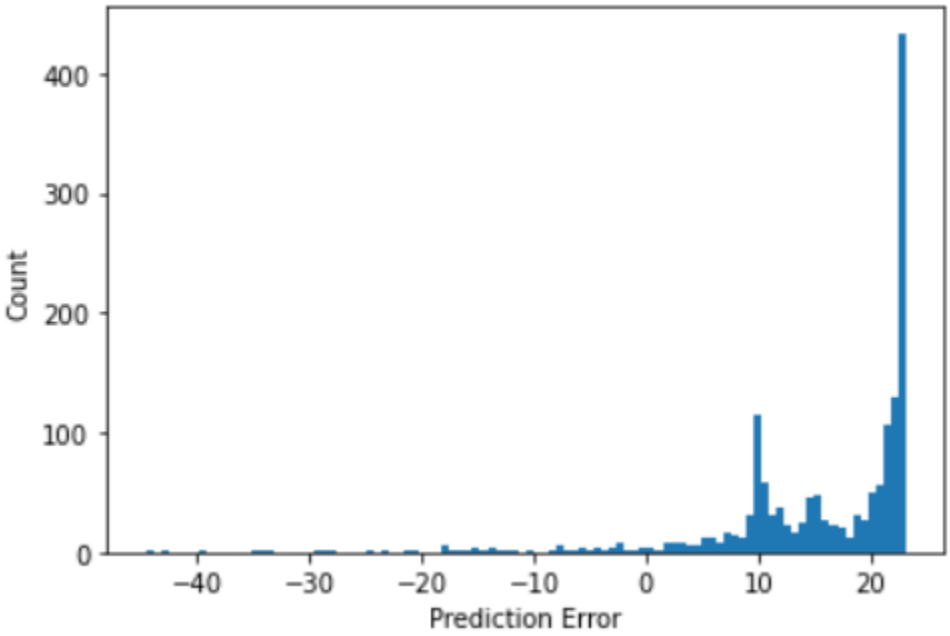

Figure 11 Range of error

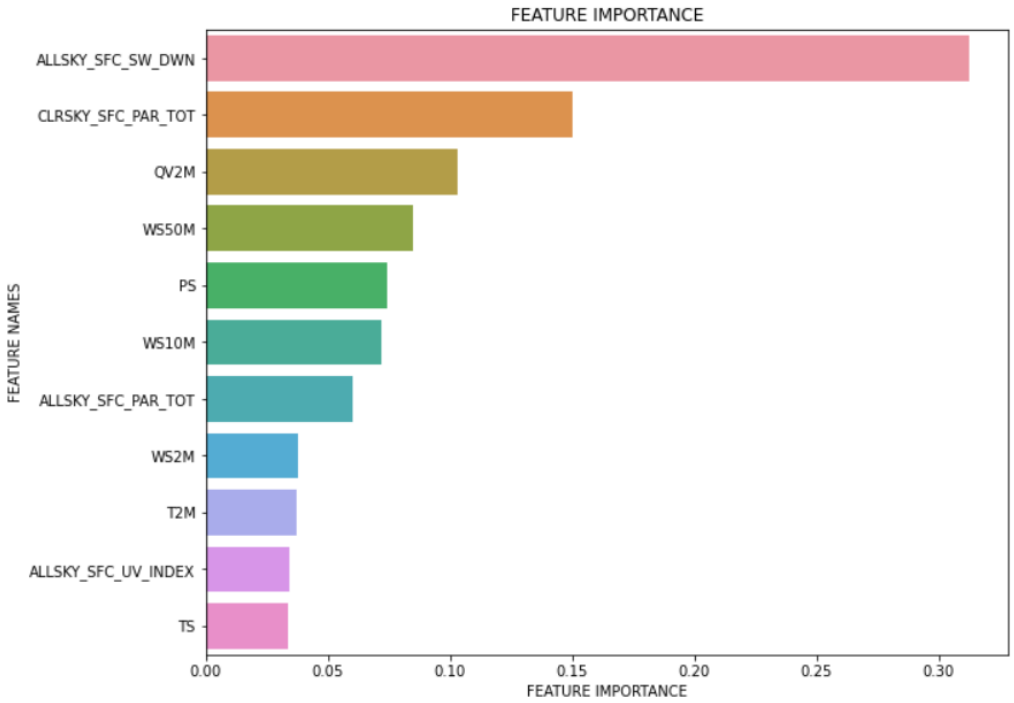

Figure 12 Feature importance

Figure 12 above shows the feature importance. It shows ALLSKY_SFC_SW_DWN (CERES SYN1deg All Sky Surface Shortwave Downward Irradiance) is relatively more important than other features.

I also changed the certain hyperparameters and the performance of each model is show below. In Table 4, the number of estimators is changed and all other hyperparameters are kept constant. With increase of number of estimators, the performance of model is getting better. The model with 400 estimators has the relatively better results.

Number of estimators	MSE	MAE	R^2
100	435.43	17.736	-5.013
200	392.83	16.919	-4.425
300	354.60	17.602	-3.897
400	339.25	16.795	-3.685

Table 4 Performance of models with different number of estimators

In Table 5, max depth is changed and number of estimators is set to be 1000 for constant. It shows when max depth=5, the model has the best performance.

Max depth	MSE	MAE	R^2
5	185.15	12.927	-1.557
10	334.00	17.761	-3.613
15	365.04	17.935	-4.041
20	323.43	17.981	-3.467

Table 5 Performance of models with different max depth

Xgboost

Xgboost is an open-source library providing a regularizing gradient boosting framework. It is an application of gradient boosted decision tree with high speed and performance. The name xgboost, though, actually refers to the engineering goal to push the limit of computations resources for boosted tree algorithms. Which is the reason why many people use xgboost — Tianqi Chen. The features of algorithm include implementation with automatic process with missing data value, support of parallelization of tree structure and people can boost a prepared model on new data. Xgboost is similar to random forest but it used boosting instead of bagging.

In Keras in Jupyter Notebook, the model of xgboost is similar to random forest. The only difference is the parameter set up. The parameter of my initial model include: XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1, colsample_bynode=1, colsample_bytree=1, gamma=0, gpu_id=-1, importance_type='gain', interaction_constraints='', learning_rate=0.300000012, max_delta_step=0, max_depth=20, min_child_weight=1, missing=nan, monotone_constraints='()', n_estimators=2000, n_jobs=8, num_parallel_tree=1,

random_state=0, reg_alpha=0, reg_lambda=1, scale_pos_weight=1, subsample=1, tree_method='exact', validate_parameters=1, verbosity=None)

Results:

After training, MSE is 92.206, MAE is 8.003 and R^2 is -0.273. Figure 13 shows the actual verses predicted value of precipitation. Figure 14 shows the range of prediction error.

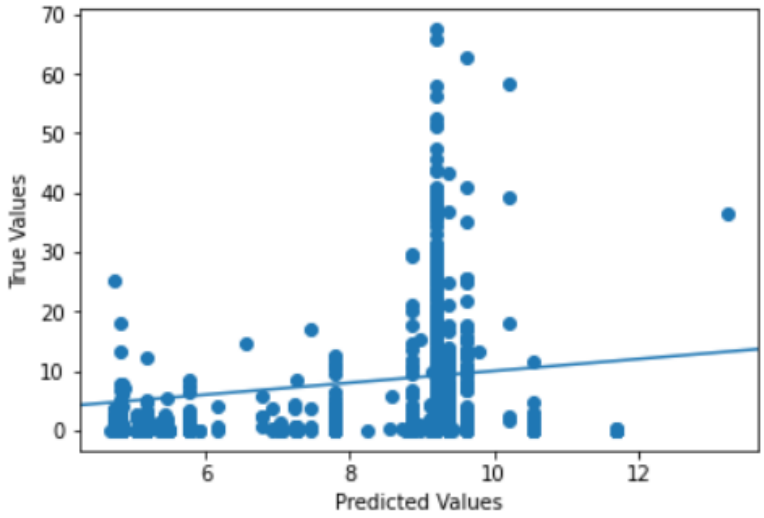

Figure 13 Actual vs predicted value

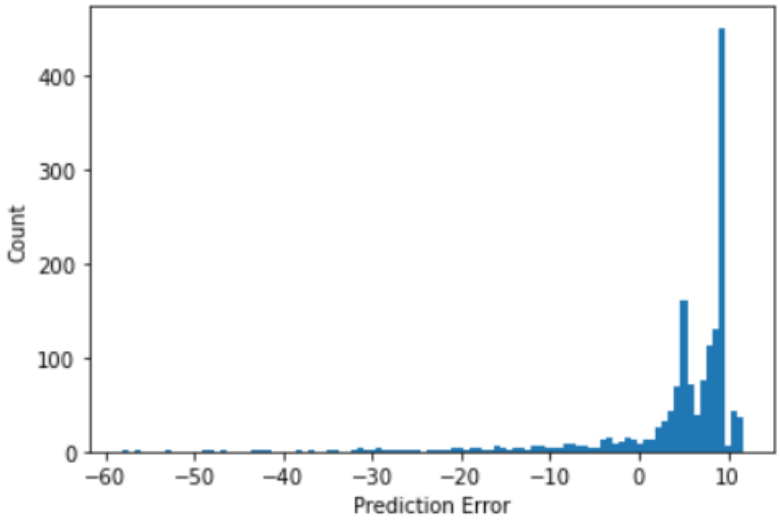

Figure 14 Range of error

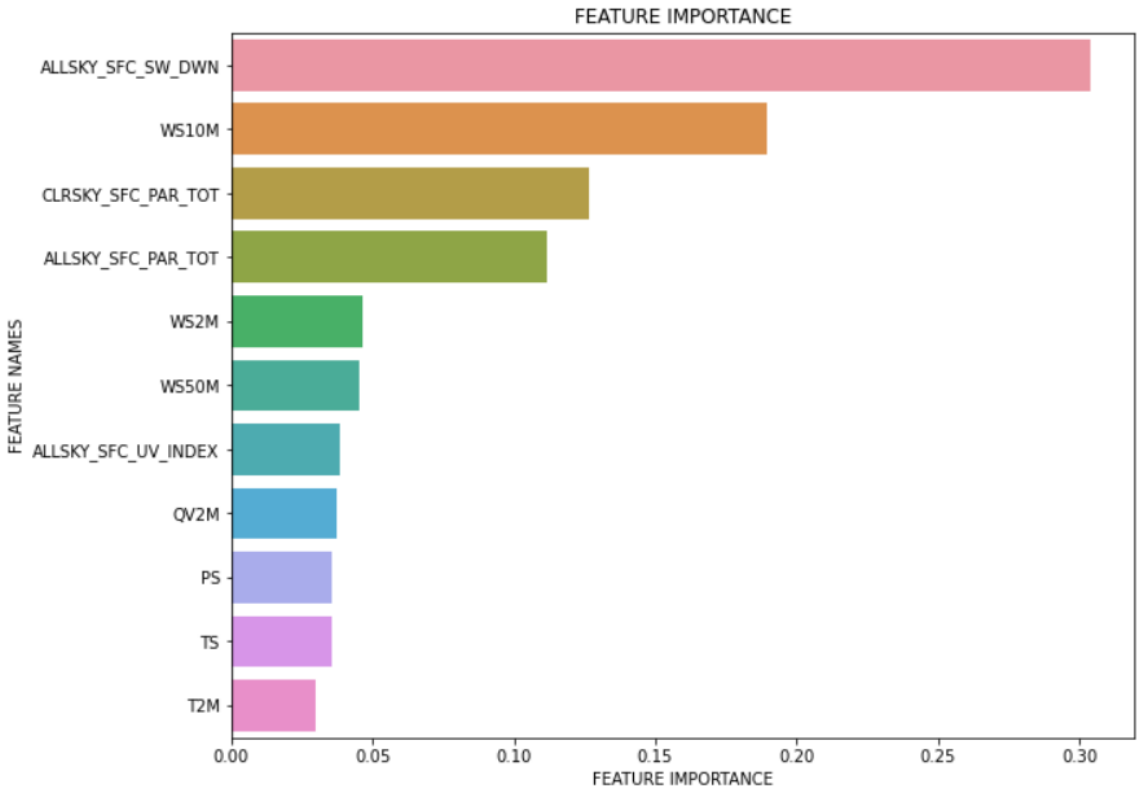

Figure 15 Feature importance

Figure 12 above shows the feature importance. Again, it has ALLSKY_SFC_SW_DWN (CERES SYN1deg All Sky Surface Shortwave Downward Irradiance) as relatively more important feature than the rest.

The same process done in random forest model will be repeated here. I changed the certain hyperparameters and the performance of each model is show below. In Table 6, the number of estimators is changed and all other hyperparameters are kept constant. It shows that by changing the number of estimators in with xgboost model, it will not affect the results a lot.

Number of estimators	MSE	MAE	R^2
100	162.883	8.003	-1.250
200	162.897	8.003	-1.250

300	162.897	8.003	-1.250
400	162.897	8.003	-1.250

Table 6 Performance of models with different number of estimators

In Table 7, max depth is changed and other hyperparameters remain constant. When we have max depth 20, the performance of the model is relatively better than others.

Max depth	MSE	MAE	R^2
5	462.065	20.238	-5.381
10	118.527	9.711	-0.637
15	162.897	11.838	-1.250
20	92.206	8.003	-0.273

Table 7 Performance of models with different max depth

Comparison between different algorithms

The best performed model for each algorithm are listed in Table 8. Although R^2 in all three algorithms are relatively low. It is obviously that neural network have better performance than random forest and xgboost. The range of prediction diagram also show that neural network has most error with positive and negative 5 while random forest and xgboost has their error around 20 and 10 respectively. The feature importance for random forest and xgboost are generally similar. They both have ALLSKY_SFC_SW_DWN as the most important feature.

Model name	MAE	R^2
Neural network	3.370	0.222
Random forest	12.927	-1.557
Xgboost	8.003	-0.273

Table 8 Comparison between different algorithms

Conclusion

The goal of this study is to apply machine learning in climate system, precipitation in particular. Also, I want to compare the performance of different algorithms. In general, increase of number of neurons and hidden layers can improve the result in neural network. However, I did not consider the cost for that and in realty, it is not the strategy people will follow. It is the same for tree-based methods. Blindly Increasing number of estimators and depth are not recommended. In this project neural network have relative higher R^2 and lower MAE than random forest and xgboost method. The possible explanation is that neural network is good at finding relations in big dataset while random forest and xgboost are faster and more effective in smaller dataset and tree-based methods are not sensitive to the input data. In all the models, R^2 value are all very low and it might because of the parameters I selected. Some parameters do not relate strongly with output, the precipitation. As result, it is hard to make prediction based on that. On the other hand, this project shows the difficulty of weather forecast in reality. I assumed precipitation over decade share the same pattern which means I did not take in account of extreme weather. Some future improvement can be consideration of feature importance first and do not include parameters that have poor connection with precipitation. More work is need to fully understand the algorithms.

Code

All code and data sheet is available at the following Github repository: https://github.com/zheyugu/Zheyu-Gu

Reference

Brownlee, J. (2021, February 16). *A gentle introduction to XGBoost for applied machine learning*. Machine Learning Mastery. Retrieved December 18, 2021, from https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/

Doelling, D. R., Sun, M., Nguyen, L. T., Nordeen, M. L., Haney, C. O., Keyes, D. F., & Mlynczak, P. E. (2016, March 1). *Advances in geostationary-derived longwave fluxes for the Ceres Synoptic (syn1deg) product*. AMETSOC. Retrieved December 18, 2021, from https://journals.ametsoc.org/view/journals/atot/33/3/jtech-d-15-0147_1.xml

NASA. (n.d.). *NASA Power*. NASA. Retrieved December 18, 2021, from https://power.larc.nasa.gov/data-access-viewer/

Neural circuits. Centre of Excellence for Integrative Brain Function. (2017, November 14). Retrieved December 18, 2021, from https://www.brainfunction.edu.au/research/research-themes/neural-circuits/

Neural networks. IBM. (n.d.). Retrieved December 18, 2021, from https://www.ibm.com/topics/neural-networks

Tree-based machine learning algorithms: Compare and contrast. Analytics Vidhya. (2021, April 15). Retrieved December 18, 2021, from https://www.analyticsvidhya.com/blog/2021/04/distinguish-between-tree-based-machine-learning-algorithms/

Yiu, T. (2021, September 29). *Understanding random forest*. Medium. Retrieved December 18, 2021, from https://towardsdatascience.com/understanding-random-forest-58381e0602d2