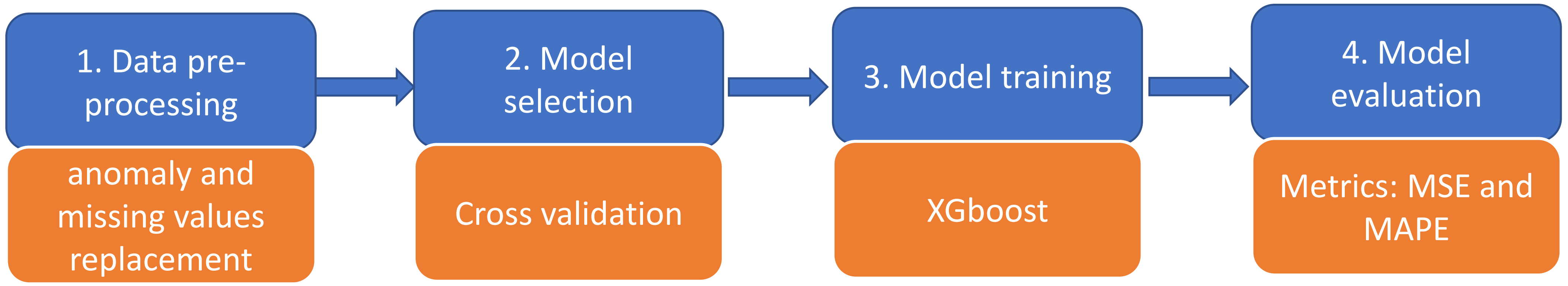
Predict dose rate of chemicals based on water parameters

Cheng Li

lichengxlxl@gmail.com (0420714881)

Overview of the proposed approach



Data Exploration

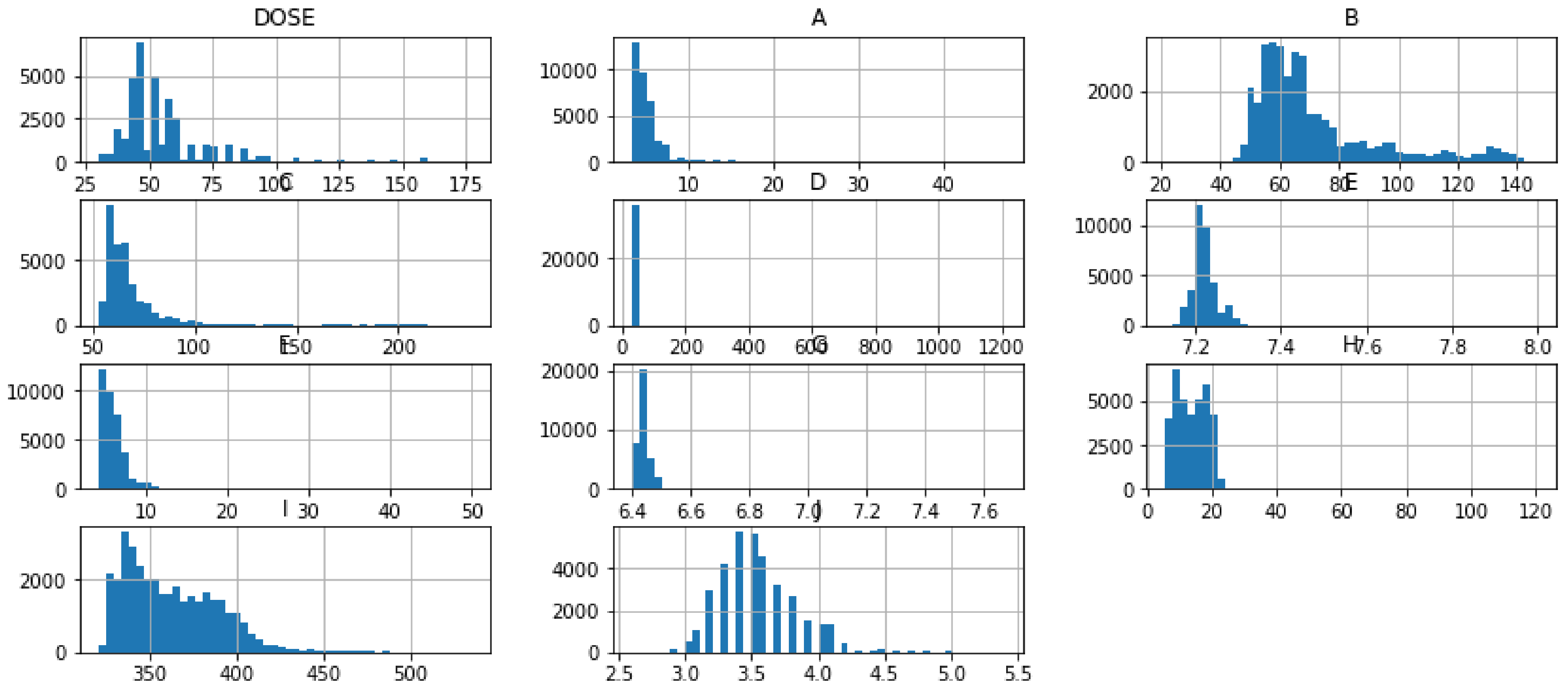
PLANT	No. rows	StartDate	EndDate
P01	5079	2009-06-28 12:00	2016-07-05 12:00
P02	61	2016-06-05 12:00	2016-07-05 12:00
P03	4820	2009-10-30 12:00	2016-07-05 12:00
P04	5204	2009-04-28 00:00	2016-07-05 12:00
P05	5035	2009-06-28 12:00	2016-07-05 12:00
P06	4706	2009-12-31 00:00	2016-07-05 12:00
P07	5068	2009-06-28 12:00	2016-07-05 12:00
P08	4833	2009-10-30 12:00	2016-07-05 12:00
P09	5071	2009-06-28 12:00	2016-07-05 12:00
P10	4835	2009-10-30 12:00	2016-07-05 12:00
ALL	44712		2016-07-05 12:00

Data statistics

- No. rows without missing values: 35852
- No. rows with missing values: 8860

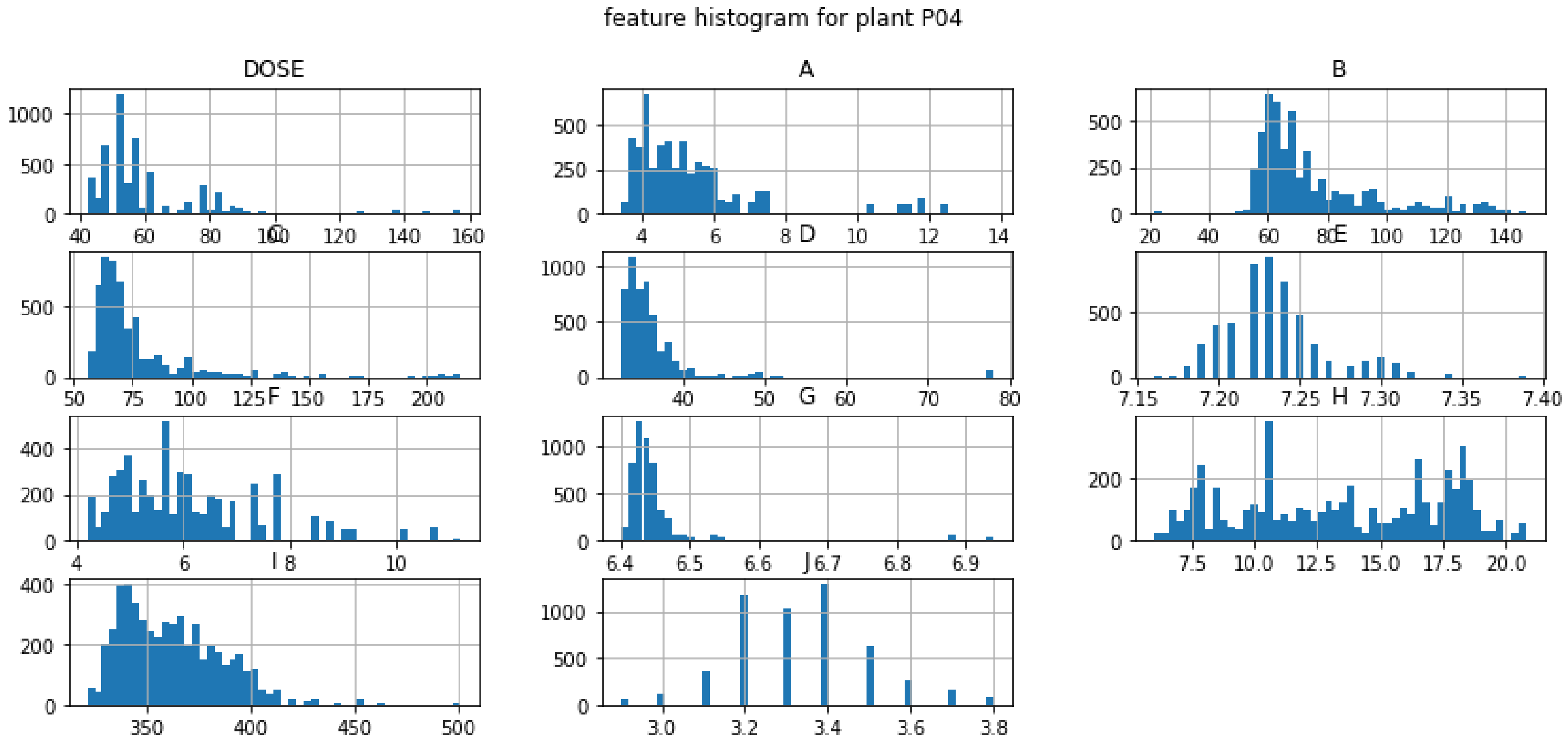
- It indicates that we may not filter out the rows with missing values

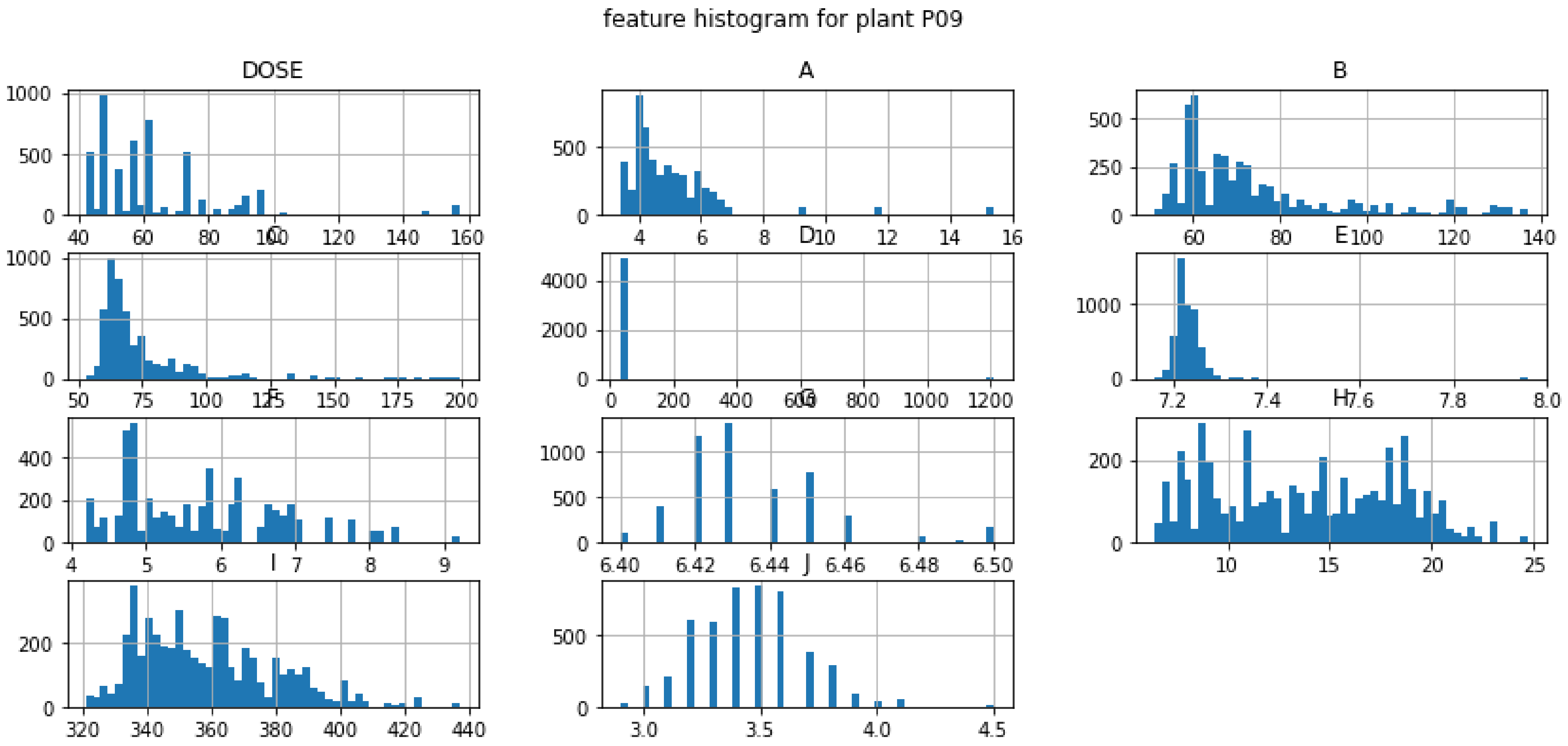
Feature distribution for all data without missing values



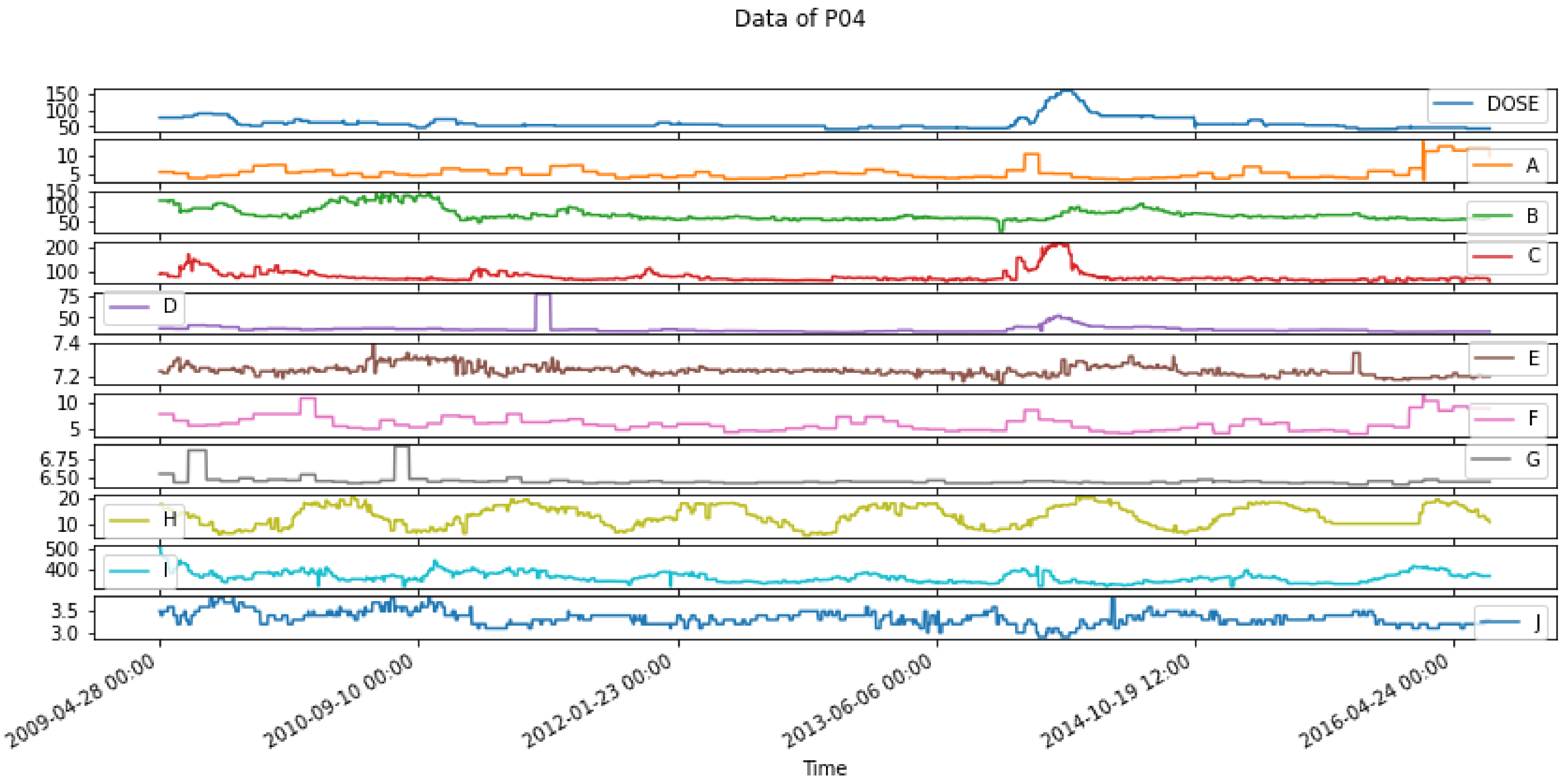
- We can see that 'A', 'D', 'E', 'F', 'G' and 'H' may have anomalies

Feature distribution for each plant

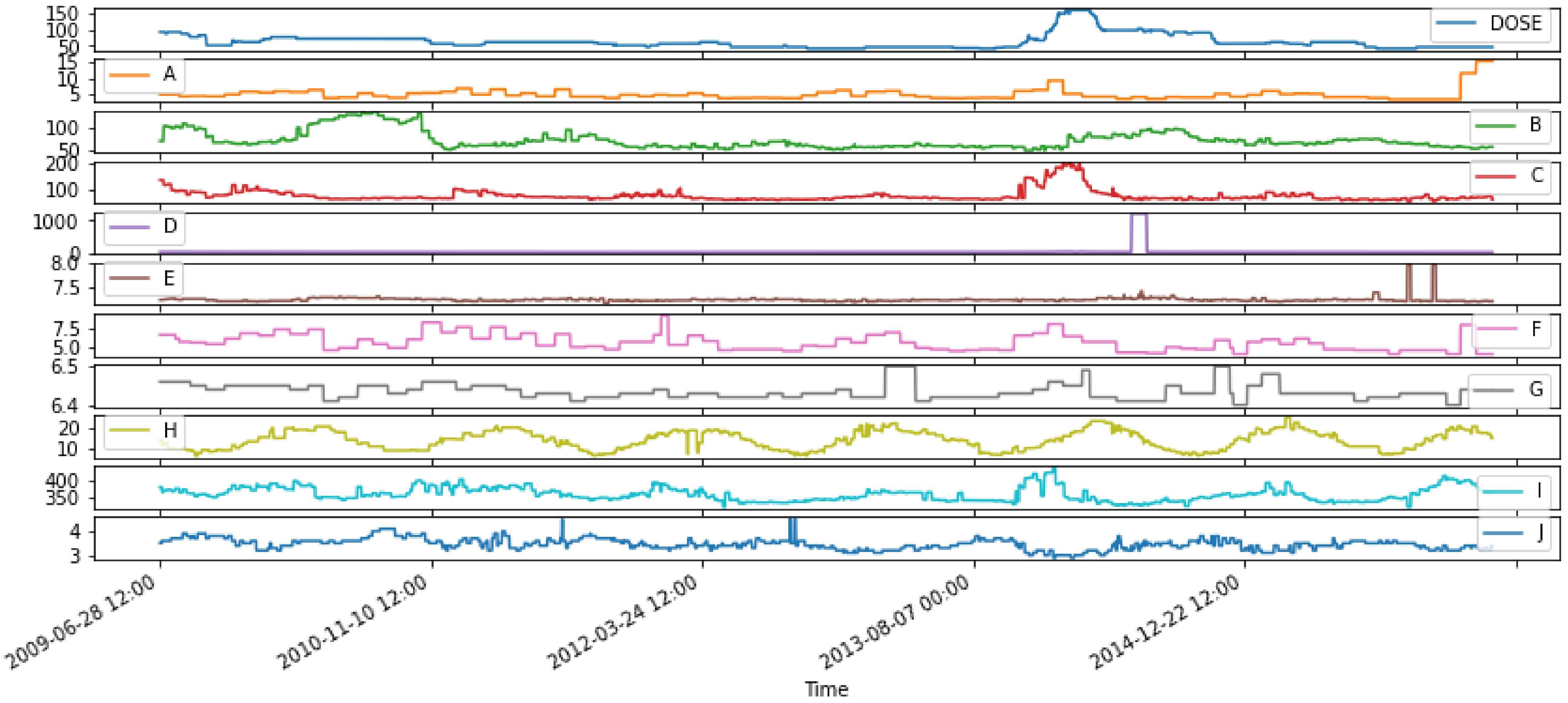



feature histogram for plant P09
DOSE
A
B
C
D
E
F
G
H
I
J

Time-series features for individual plant



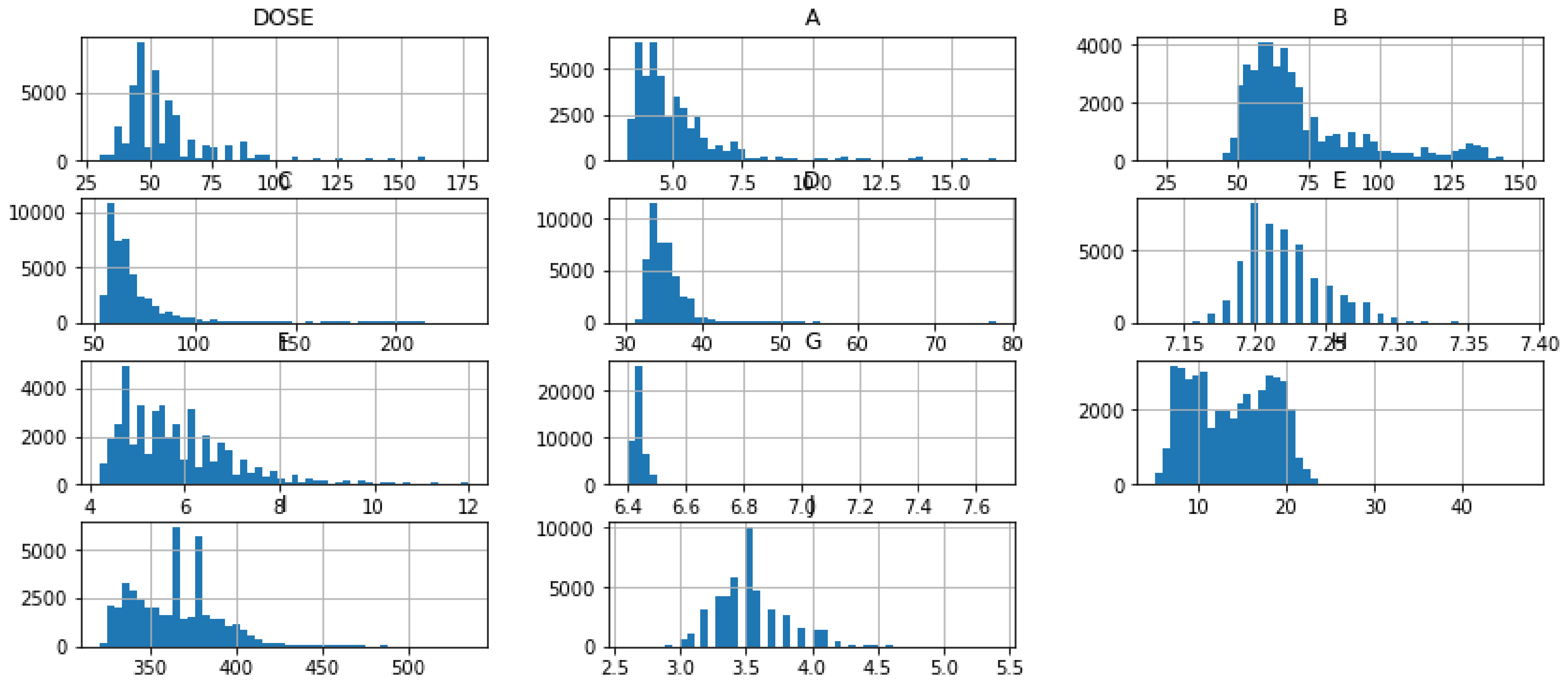
Data of P09

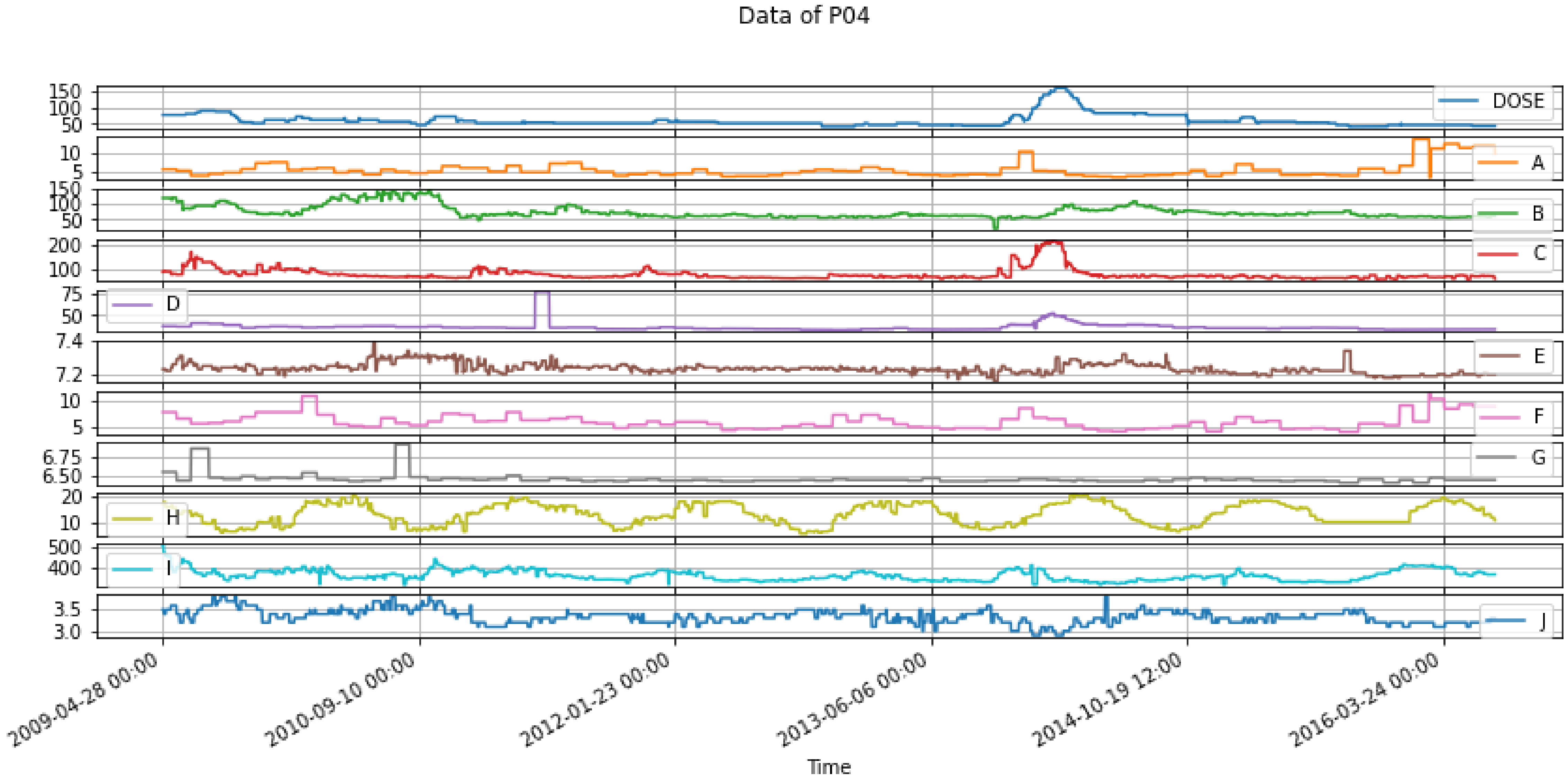

DOSE
A
B
C
D
E
F
G
H
I
J
150
100
50
15
10
5
100
50
200
100
1000
0
8.0
7.5
7.5
5.0
6.5
6.4
20
10
400
350
4
3
2009-06-28 12:00
2010-11-10 12:00
2012-03-24 12:00
2013-08-07 00:00
2014-12-22 12:00
Time

Dealing with missing values and anomalies

- Anomalies:
  - ‘A’ > 30
  - ‘D’ > 200
  - ‘E’ > 7.4
  - ‘F’ > 20
  - ‘H’ > 90
- Anomaly replacement:
  - A coarse processing: replacing anomalies with the mean of remaining data expect missing values
  - A fine processing: replacing anomalies at a plant with the mean of the data of that plant without missing values
  - I am using the coarse processing above.
- After handling anomalies, I replace a missing value with the mean of that column at that plant

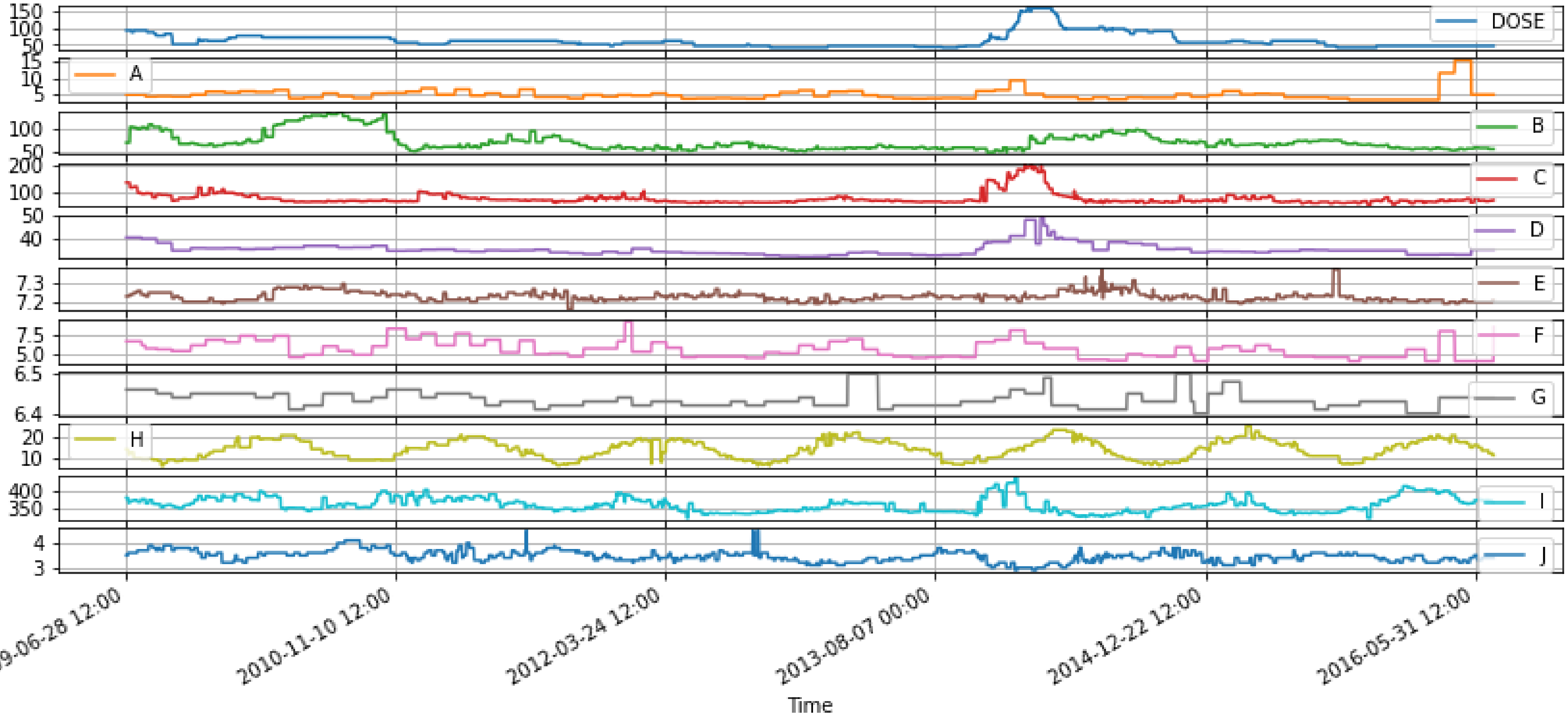
After anomaly and missing value replacement



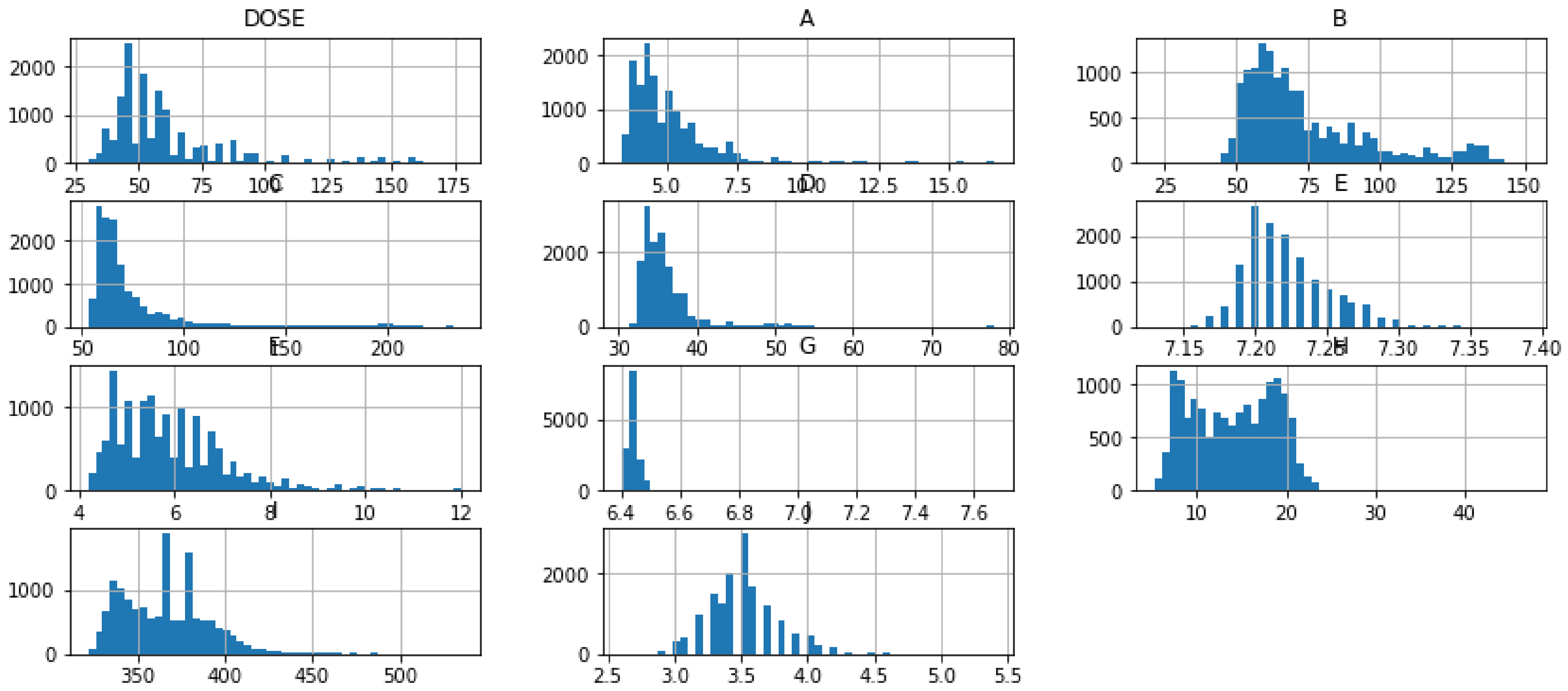


Data of P04
DOSE
A
B
C
D
E
F
G
H
I
J
2009-04-28 00:00
2010-09-10 00:00
2012-01-23 00:00
2013-06-06 00:00
2014-10-19 12:00
2016-03-24 00:00
Time

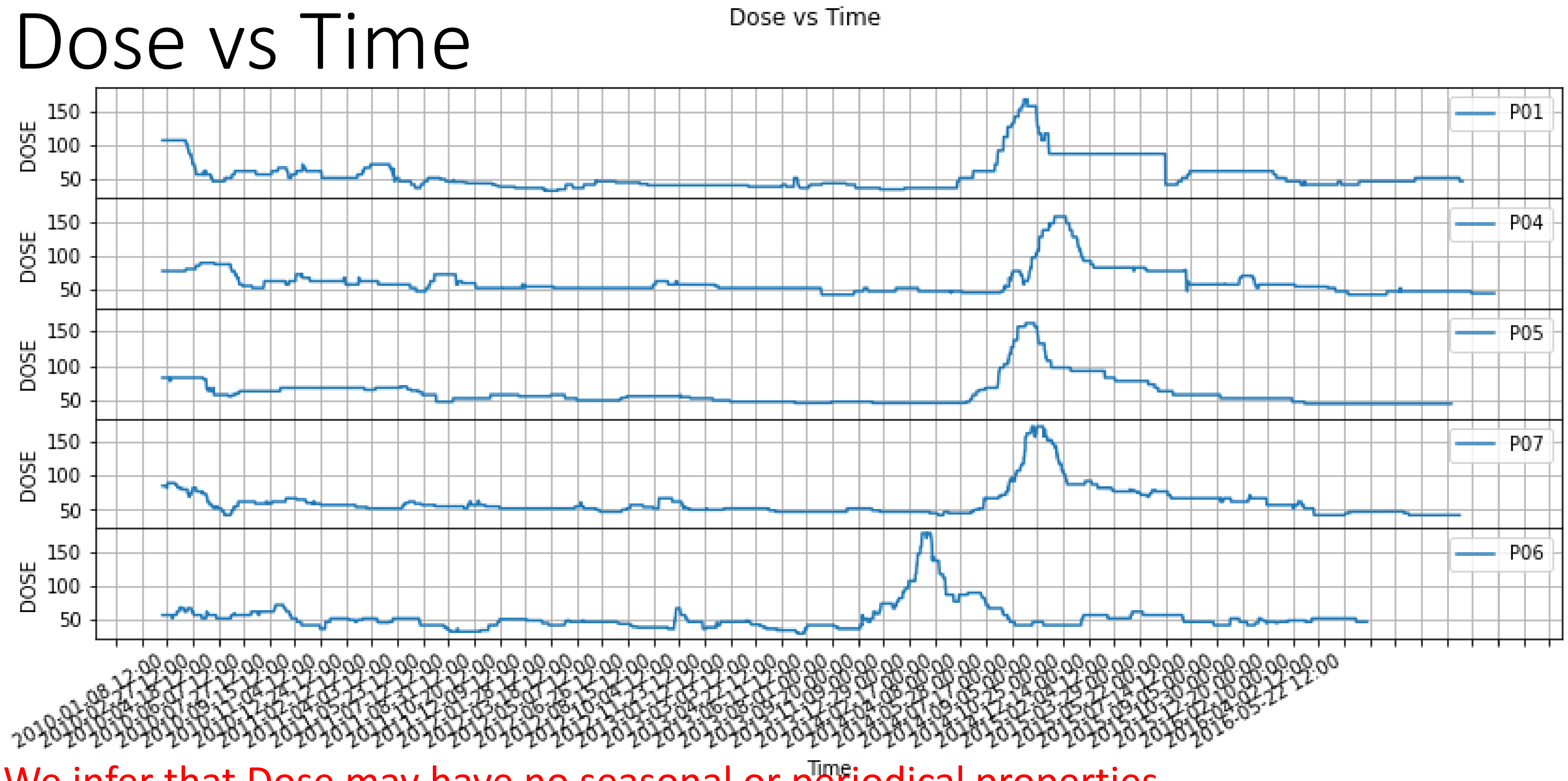
Data of P09


DOSE
A
B
C
D
E
F
G
H
I
J
150
100
50
15
10
5
100
50
200
100
50
40
7.3
7.2
7.5
5.0
6.5
6.4
20
10
400
350
4
3
2009-06-28 12:00
2010-11-10 12:00
2012-03-24 12:00
2013-08-07 00:00
2014-12-22 12:00
2016-05-31 12:00
Time

After deleting duplicates



Dose vs Time



- We infer that Dose may have no seasonal or periodical properties.
- Dose may depend on the quality parameters of water
- The response of 2014 might be different from other years

Model selection (cross validation)

- I used data after deleting duplicates as the training data
- Training years: [2009, 2010, 2011, 2012, 2013, 2014]. Test years: [2015, 2016]
- 5-fold CV
  - training years: [2009]   test years: [2010]
  - training years: [2009, 2010]   test years: [2011]
  - training years: [2009, 2010, 2011]   test years: [2012]
  - training years: [2009, 2010, 2011, 2012]   test years: [2013]
  - training years: [2009, 2010, 2011, 2012, 2013]   test years: [2014]

Model selection

MAE	P01	P02	P03	P04	P05	P06	P07	P08	P09	P10	ALL
RF	12.49		10.41	12.18	8.97	9.79	10.37	10.15	11.69	10.21	10.95
MLP	14.62		39.19	16.46	14.24	12.70	14.07	14.50	13.57	40.50	14.58
BLR	18.62		10.65	18.34	9.30	9.74	10.57	7.95	10.31	10.22	12.62
SVR	19.81		17.14	16.86	17.22	15.75	16.11	17.15	20.02	15.88	16.12
Gboosting	12.55		10.14	12.91	8.45	9.53	10.57	10.28	12.89	10.38	10.54

MAPE	P01	P02	P03	P04	P05	P06	P07	P08	P09	P10	ALL
RF	20.03		16.84	17.07	11.48	15.15	12.56	14.99	14.97	15.37	16.27
MLP	24.71		64.32	26.61	19.67	21.25	21.24	25.68	19.19	74.36	21.60
BLR	30.31		18.27	27.81	11.92	14.41	14.81	12.08	12.71	14.95	18.07
SVR	29.65		27.07	22.49	23.58	23.90	20.24	25.33	27.10	23.50	23.15
Gboosting	19.87		16.13	17.79	10.87	14.88	13.04	15.10	15.12	15.22	15.64

Some points

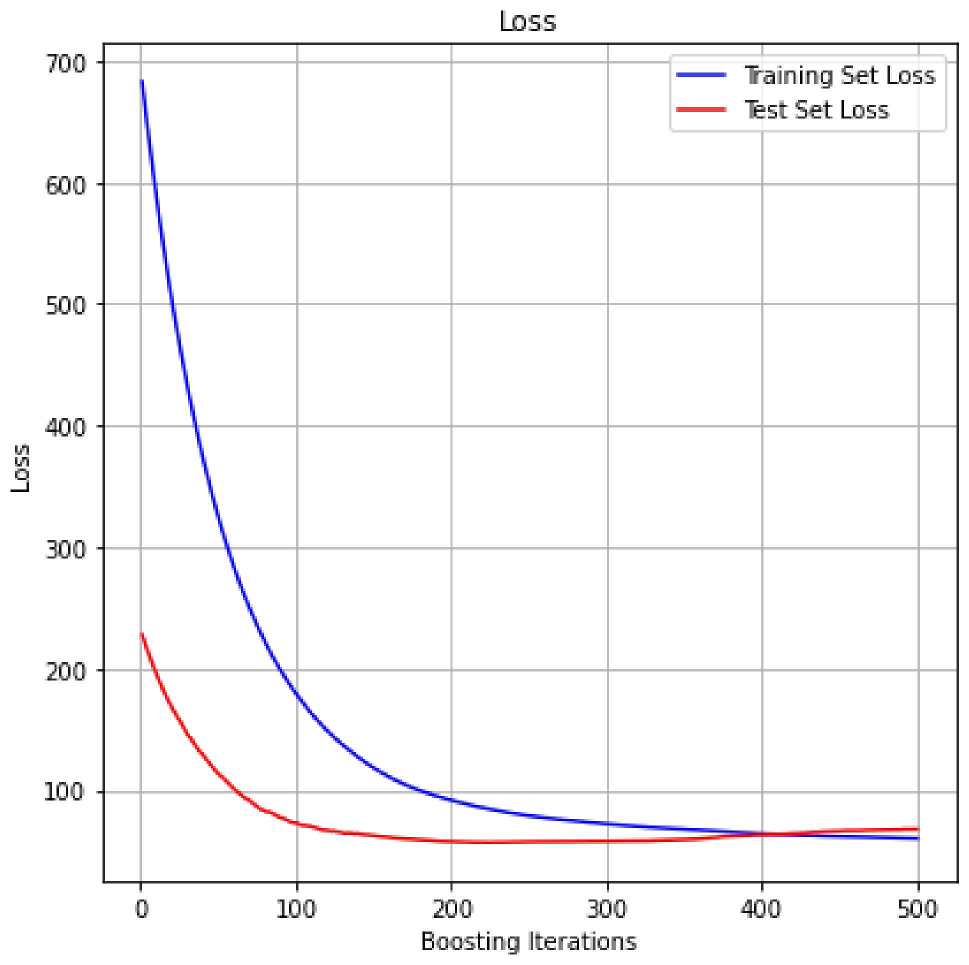
- Random forest and Gradientboosting performs similar.
- I will use Gradientboosting as our model in the regression task.
- When the year 2014 is the validation data, the metric value is usually high. It implies that 2014 data may be different from other years.

Gradient boosting

- Training data:2009~2014, test data: 2015, 2016

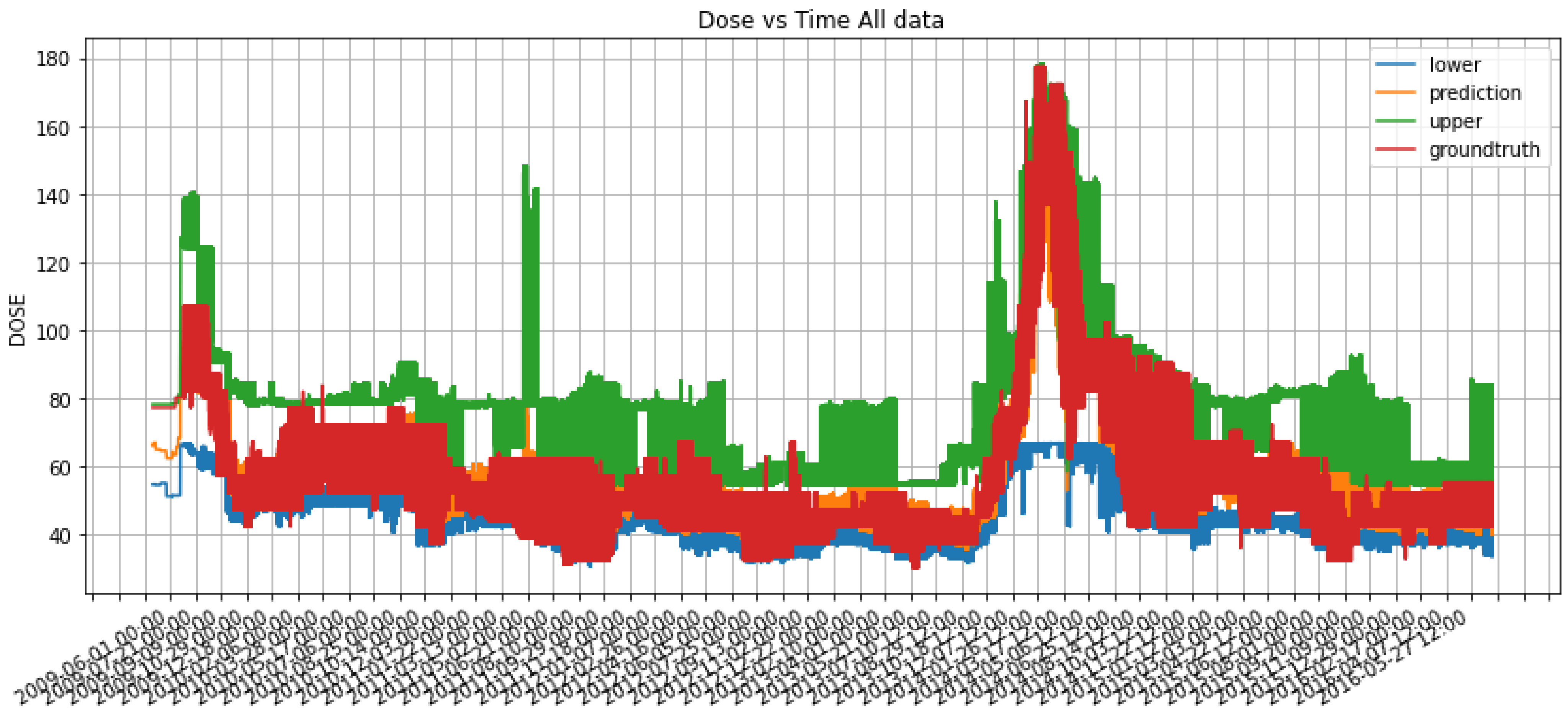
2012	6588
2011	6570
2010	6570
2013	6561
2014	6531
2015	6191
2016	3336
2009	2365

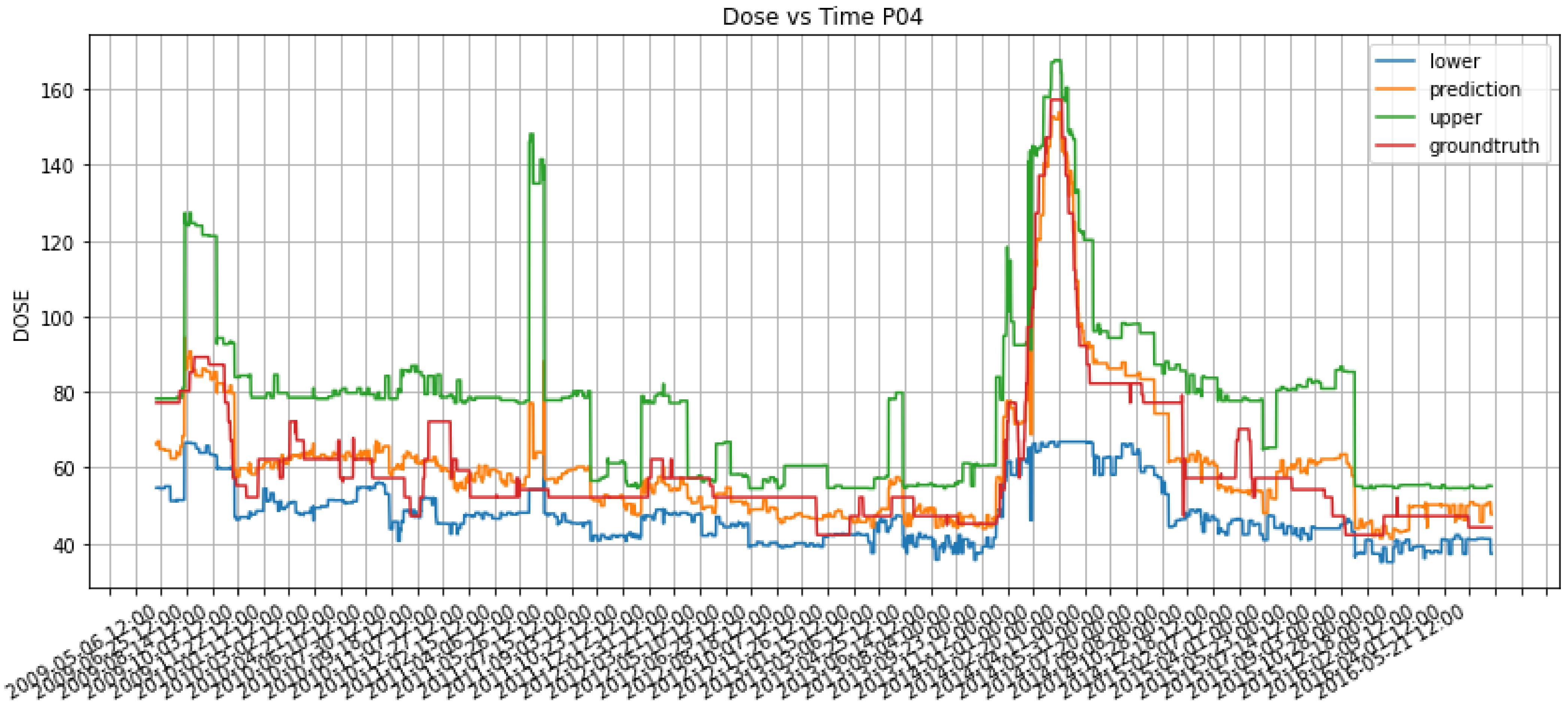
Training

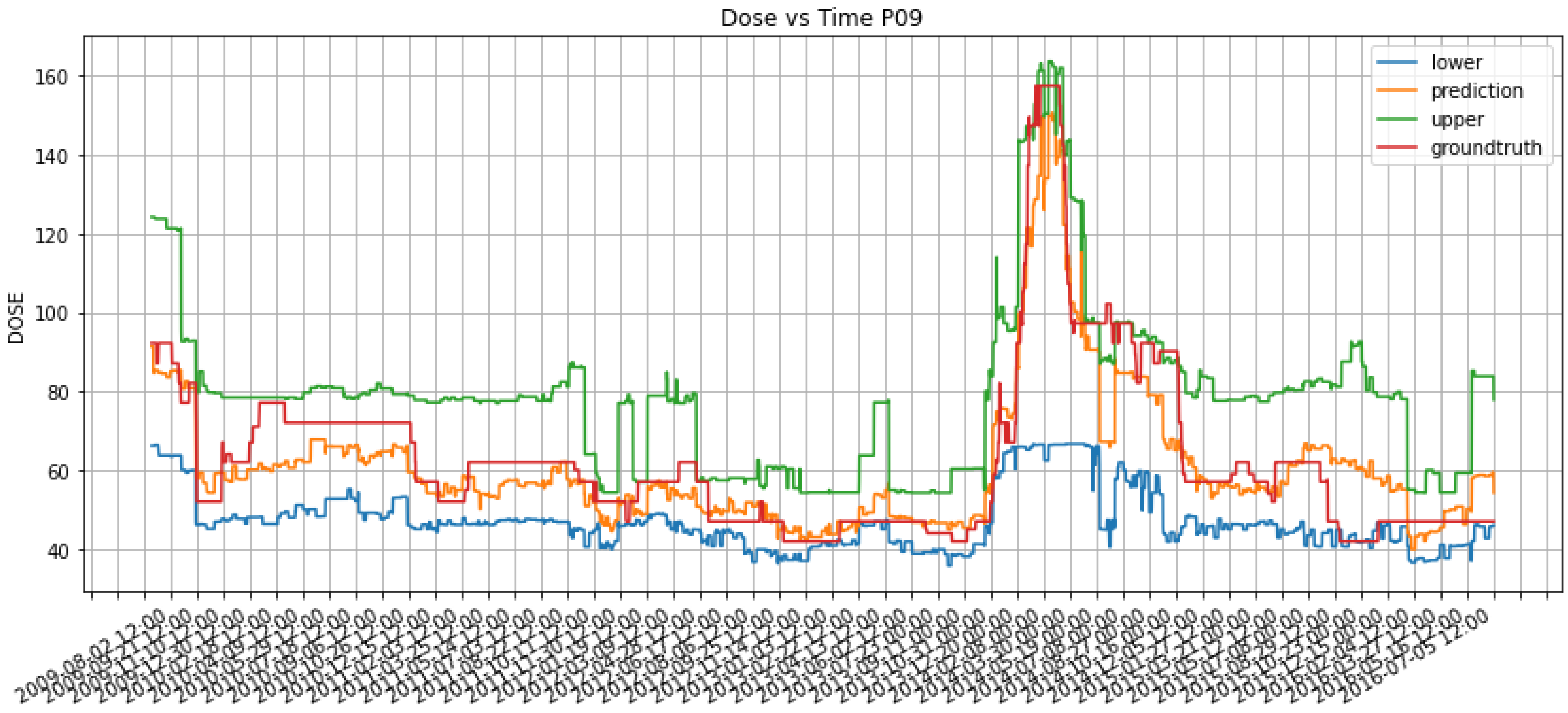


To avoid overfitting, I chose the number of stages performing gradient boosting **equal to 400**

Prediction and prediction intervals (95%)



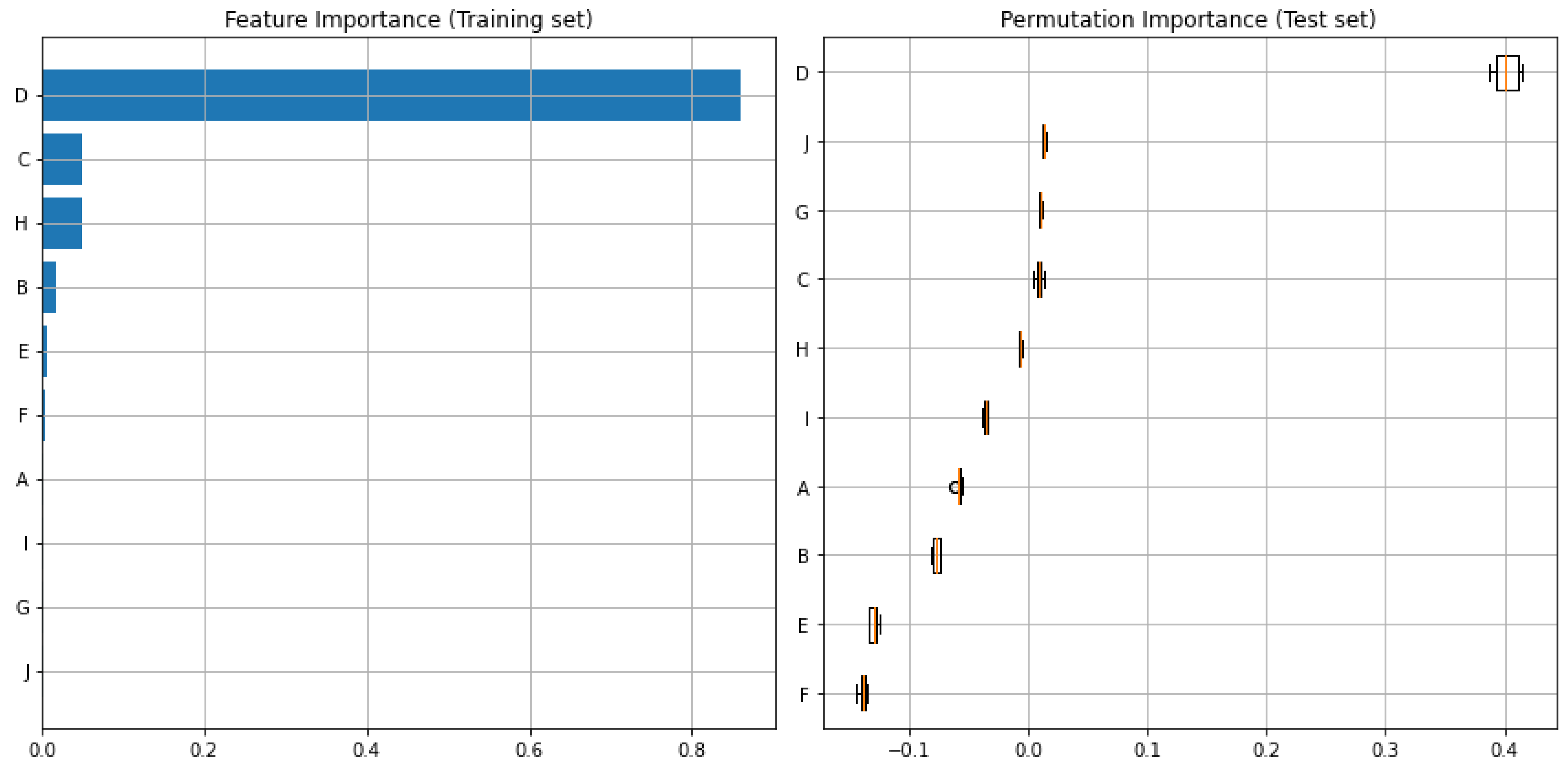

Dose vs Time P04
DOSE
lower
prediction
upper
groundtruth
160
140
120
100
80
60
40
2009-05-06 12:00
2016-05-21 12:00


Dose vs Time P09
DOSE
160
140
120
100
80
60
40
lower
prediction
upper
groundtruth
2009-08-02 12:00
2016-07-05 12:00

Metrics

- Training MAE: 5.78
- Test MAPE: 9.43%
- Test MAE: 6.15
- Test MAPE: 12.89%
- The number of data located in prediction intervals: 92.9%

Feature selection



- The feature importance from the training set and the test set identifies a common **strong feature 'D'**
- The other important features are **'C' and 'H'**

How to utilize this model

- For users who knows Python a bit:
  - Once we have finalized the model, we can save the model using pickle and load as well if we need it to predict an input.
- For non-tech users:
  - We may deploy the model to the cloud or AWS etc.

Once new data are collected, we can update or re-train the model.

Future work

- I roughly tuned the hyperaprameters of Gradientboosting. A fine hyperparameter tunning may improve the model performance.
- Xgboost has demonstrated a powerful prediction capability and a fast running. We can try it.
- Currently I merged all plants into one model. In future, mixed effects (fixed and random effects) can be considered into a suitable model because the all plants are along the same river.
- We can treat the dose prediction of individual plant as a task and then using multi-task learning may boost the dose prediction for all plants.
- Currently I used all historical data. We can consider discarding the long-ago data and check the model performance again.

Deep learning proposal

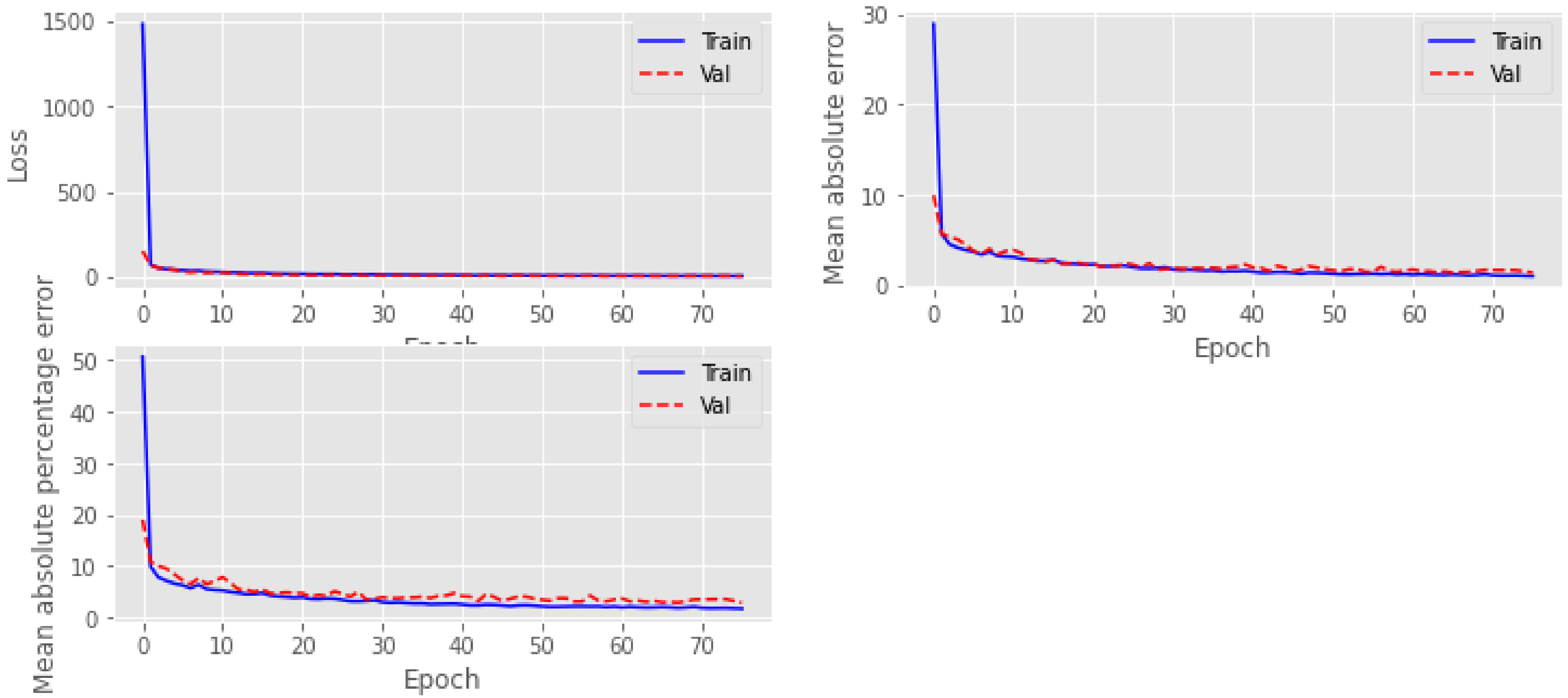
- Here I only focus on the LSTM framework. The pre-processing is similar with Gradientboosting.
- Since running deep learning framework takes amount of time, I ignored model selection and directly compared the LSTM framework to gradientboosting on all data
- LSTM has several settings. I was using the historical data to predict the next output without knowing the water parameter.
- The training data [2009, 2010, 2011, 2012, 2013, 2014]
- The validation data [2015]
- The test data[2016]

Model performance

MAE	Training	Validation	Test
LSTM	1.32	1.42	1.48
Gradboosting	5.78	7.31	3.98

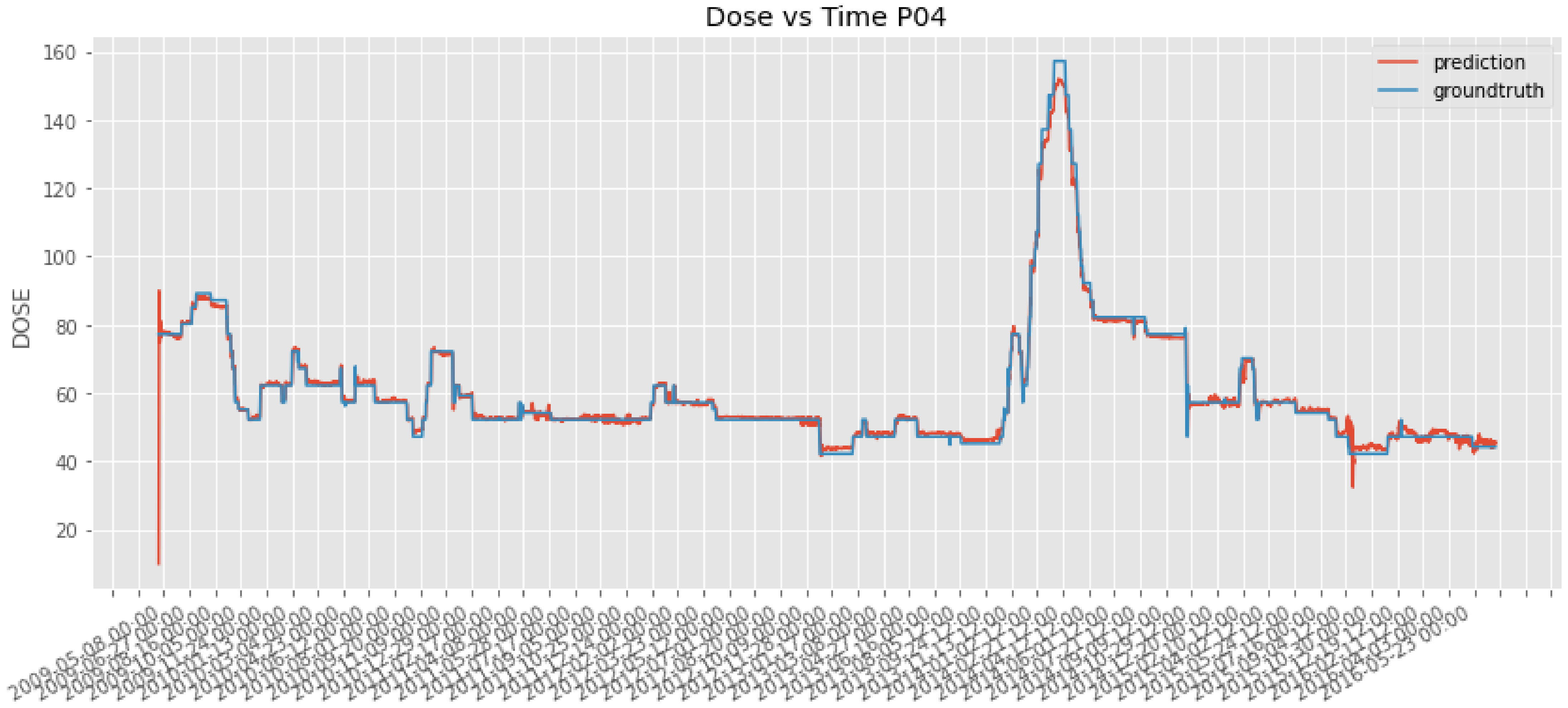
MAPE	Training	Validation	Test
LSTM	2.29	2.96	3.30
Gradboosting	9.43	15.21	8.59

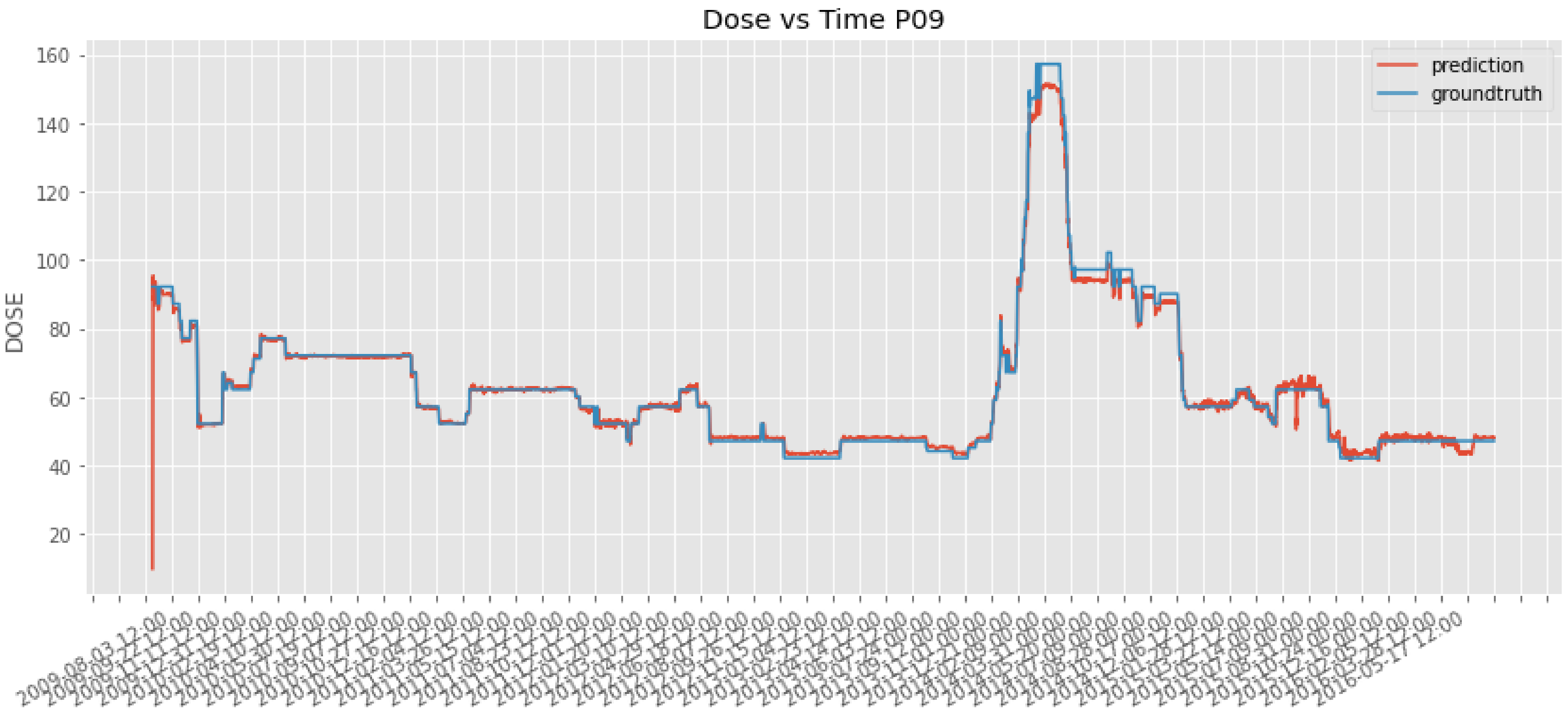
The loss curve



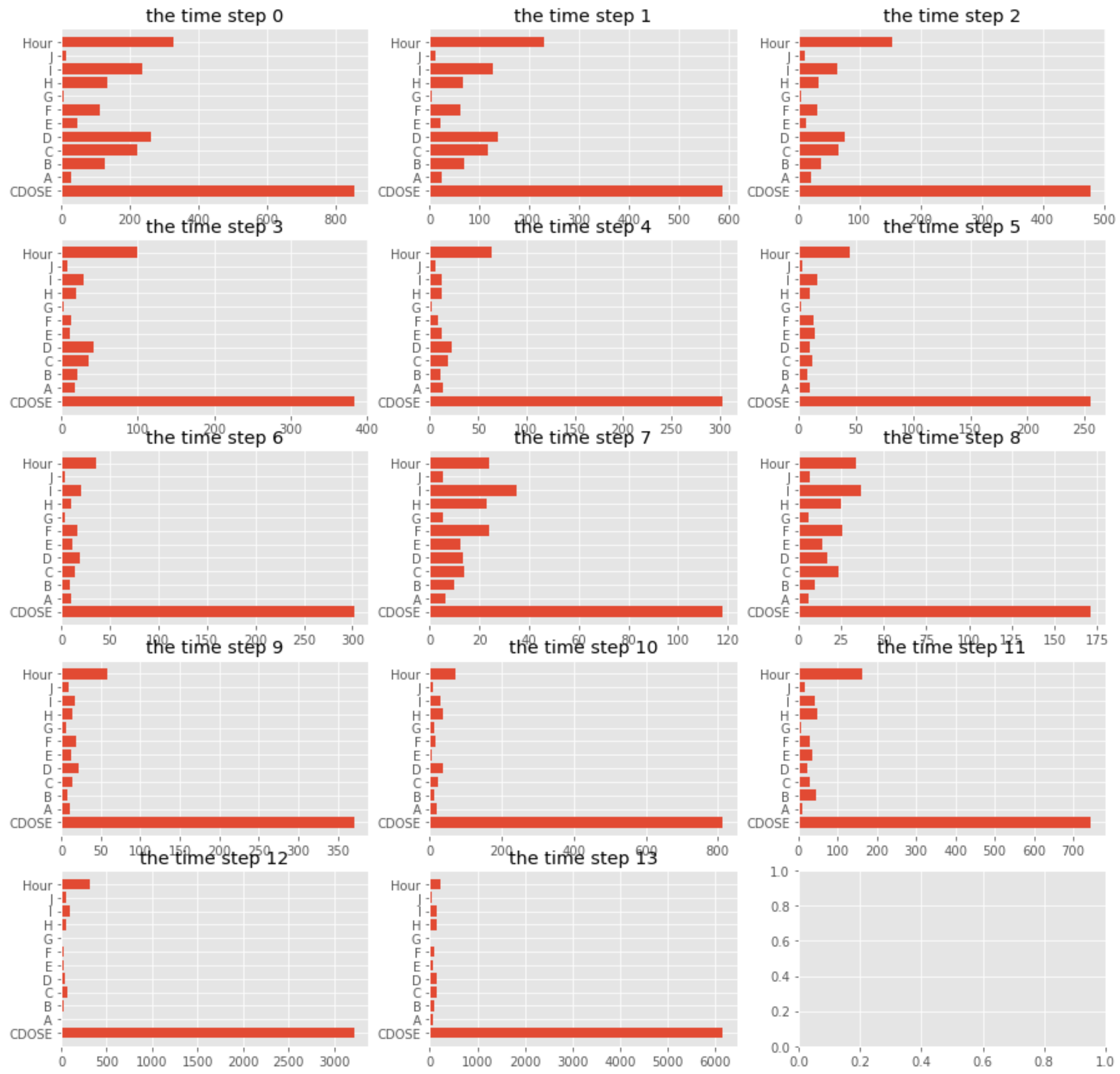
I was using the weights of the epoch 66

Prediction


Dose vs Time P04
DOSE
160
140
120
100
80
60
40
20
prediction
groundtruth
2009-05-08 00:00
2016-05-23 00:00


Dose vs Time P09
DOSE
160
140
120
100
80
60
40
20
prediction
groundtruth
2009-08-03 12:00
2016-05-17 12:00

Interpretability (Shapley)



- Since there are 14 time steps in one data, each plot shows the importance of variables to the final output at the given timestep.
- **The DOSE at the previous timestep effects most on the current prediction.**

Potentials

- Hyperparameter tunning of deep learning may improve model performance, such as tuning layers, learning rates and batch size tec. NAS or Bayesian optimization approaches can be used to solve this problem.
- The number of time steps I set is 14 (one week). I did not try other values.
- I am currently using all data from all plants to train a common model. Training a single model for individual plant ? Or adding the column ‘PLANT’ (a categorical variable) to feature matrix?
- Feature importance can be obtained by other approaches, such as premutation, LIME and varGrad etc.
- Uncertainty can be approximated by MC dropout.
- Residual LSTM is alternative.
- For more examples about LSTM, please check my github https://github.com/licheng0794