Mass identification of Amaterasu particle using machine learning

OMU, Kohei Endo

TADAML Oct ,15 2025

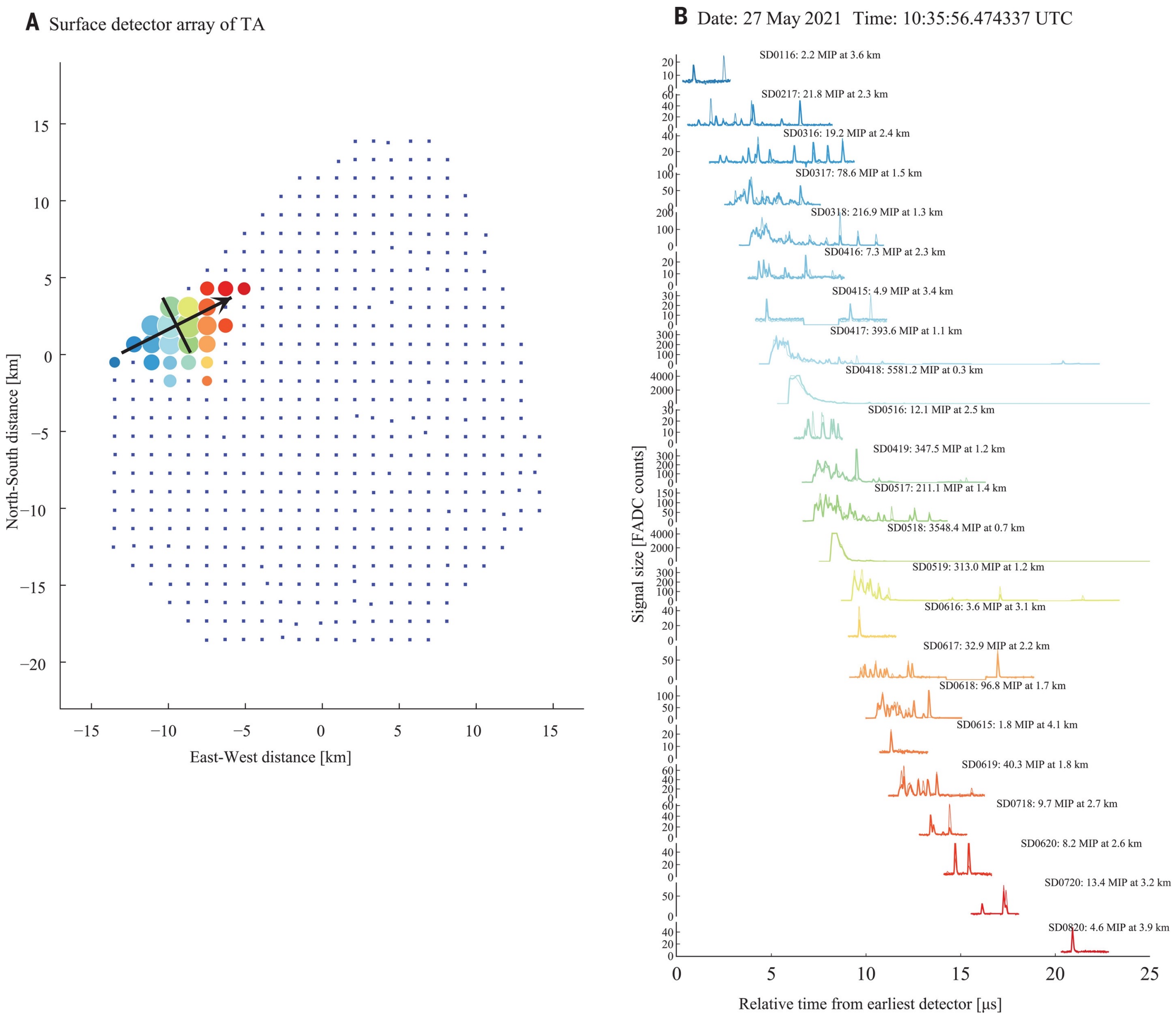
Amaterasu particle

- The morning of May 27, 2021

- The highest energy cosmic ray detected by TA

- Only SD data
→Difficult to identify mass

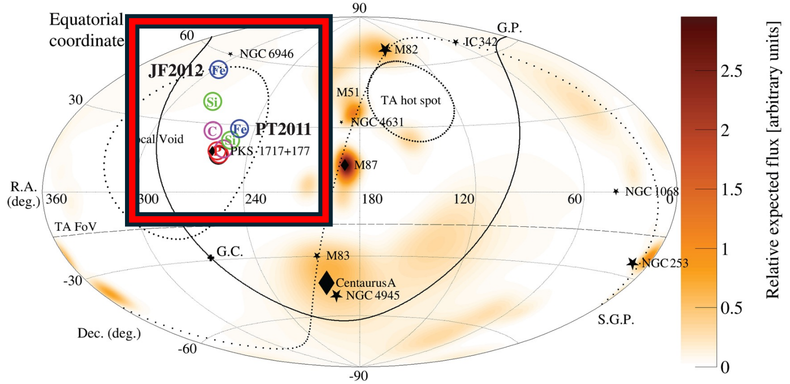
→**Try machine learning**

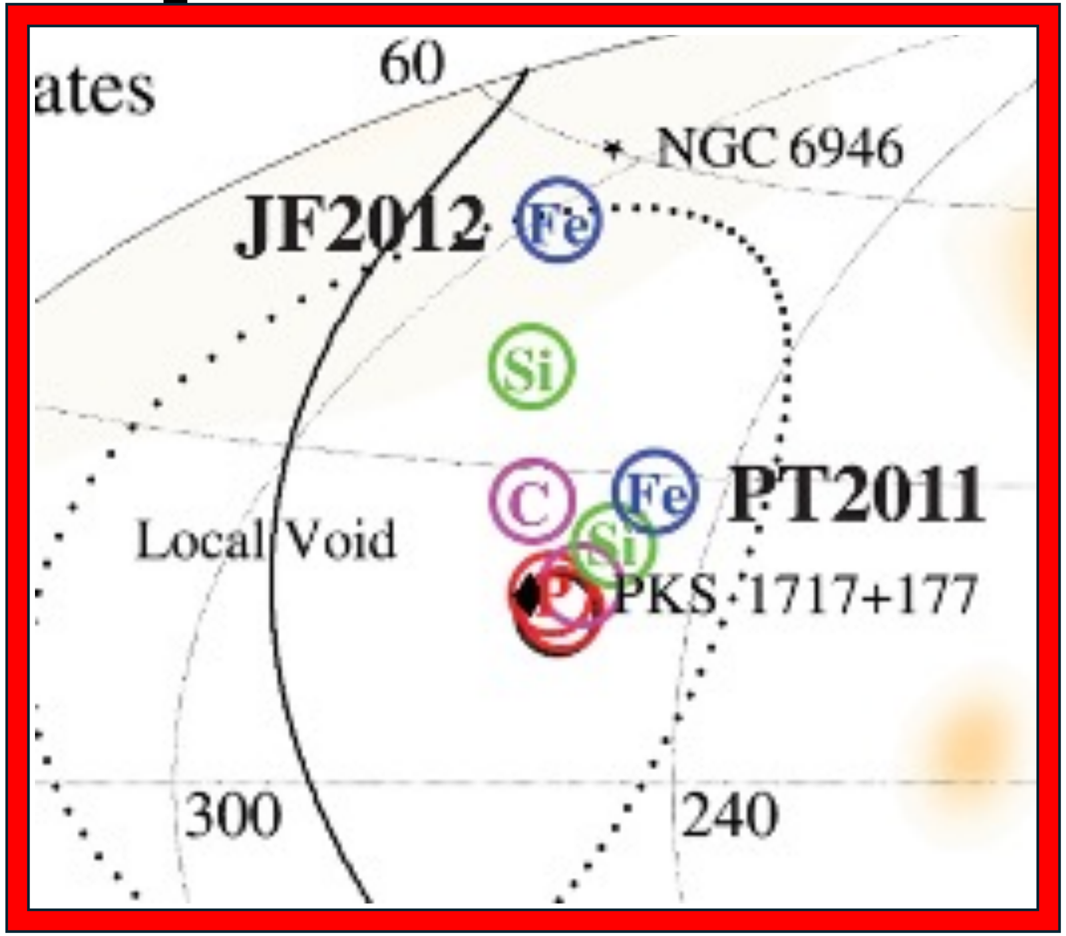


"An extremely energetic cosmic ray observed by a surface detector array"
https://doi.org/10.1126/science.abo5095

Motivation

- Amaterasu particle is also influenced by the magnetic field during the propagation process
- Light → almost straight
- Heavy → will be deflected more
- Identify mass of Amaterasu particle to help identify it’s origin





Research method

- Create MC events with reconstruction data of Amaterasu particle

- Extract features from SD's waveform data using TSFEL

- Expect mass of Amaterasu particle using machine learning (RandomForest)

MC simulation

- Interaction model : **EPOS-LHC, QGSJETII04**
- Mass : Proton, Iron
- Energy : 244 EeV
- Zenith : 38.6 degree
- Azimuth : 206.8 degree
- Core position : -9.471, 1.904 km
  (reuse 100 times in 30 m radius)
- Date : May 27, 2021, 10:35:55

Amaterasu reconstruction data is based on QGSJETII03
Energy is adjusted for each interaction model and mass

TSFEL

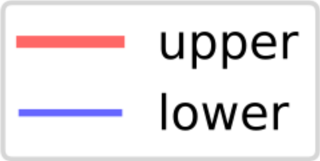
- TSFEL (Time Series Feature Extraction Library)
  - TSFEL is a Python package for efficient feature extraction from time series data.
    It offers a comprehensive set of feature extraction routines without requiring extensive programming effort.

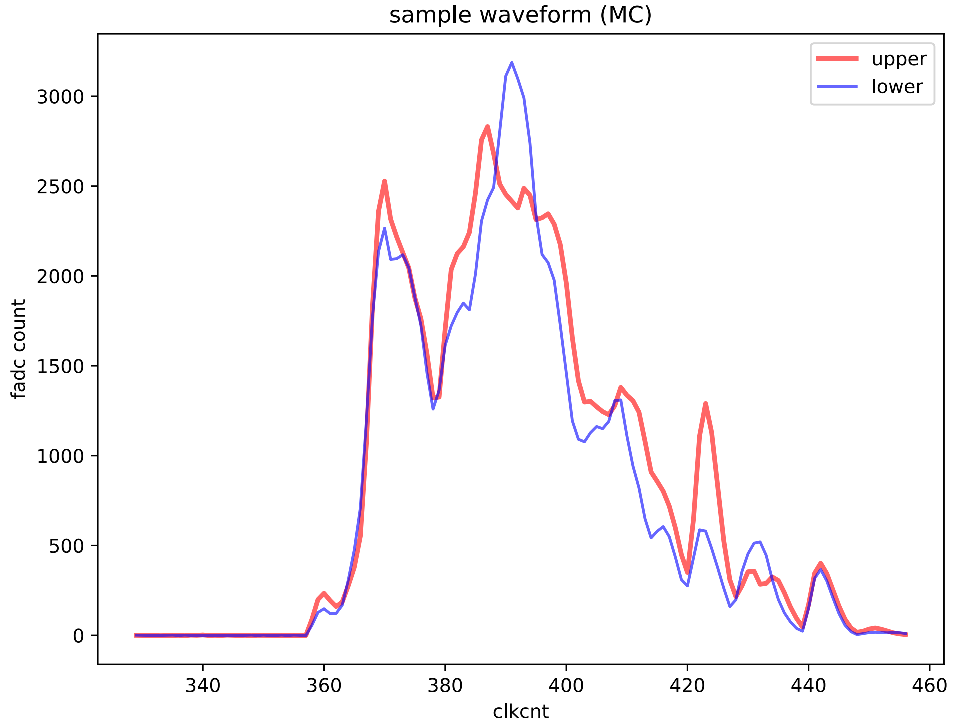
https://tsfel.readthedocs.io/en/latest/

- Extract **156 features** from SD's waveform for machine learning
  - 'Statistical', 'Temporal', and 'Spectral' domain

Dataset

- Use 64 SDs (8x8 array, DET0114 - DET0821)
- Each SD has 156 features
  -> 64 x 156 = **9984 features in 1 event**
- Each SD have 2 waveforms
  (upper and lower scintillator)
- Each waveform is analyzed independently
- 2 interaction models
  2 layers
  -> 4 Machine learning models
- Training : Validation = 9 : 1

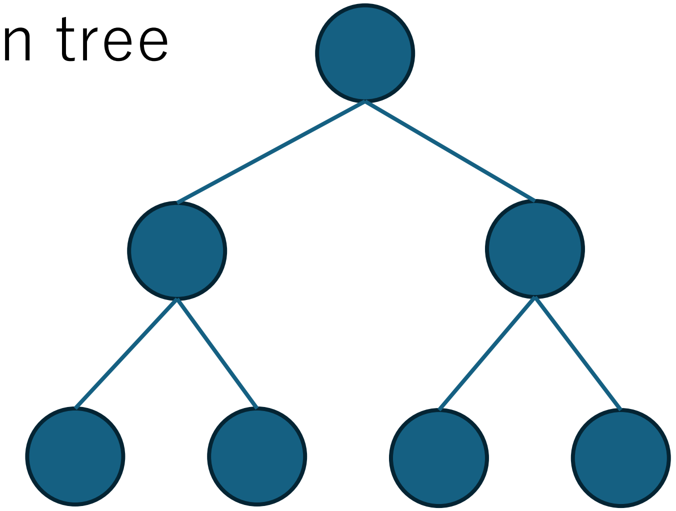




Model

- Model
  - Tuning hyperparameter with ‘Optuna’
  - Optimize to maximize ‘Accuracy’
    (Proton : 0, Iron : 1, separate at 0.5)
  - 10 Folds cross validation
    (All data is used for validation)

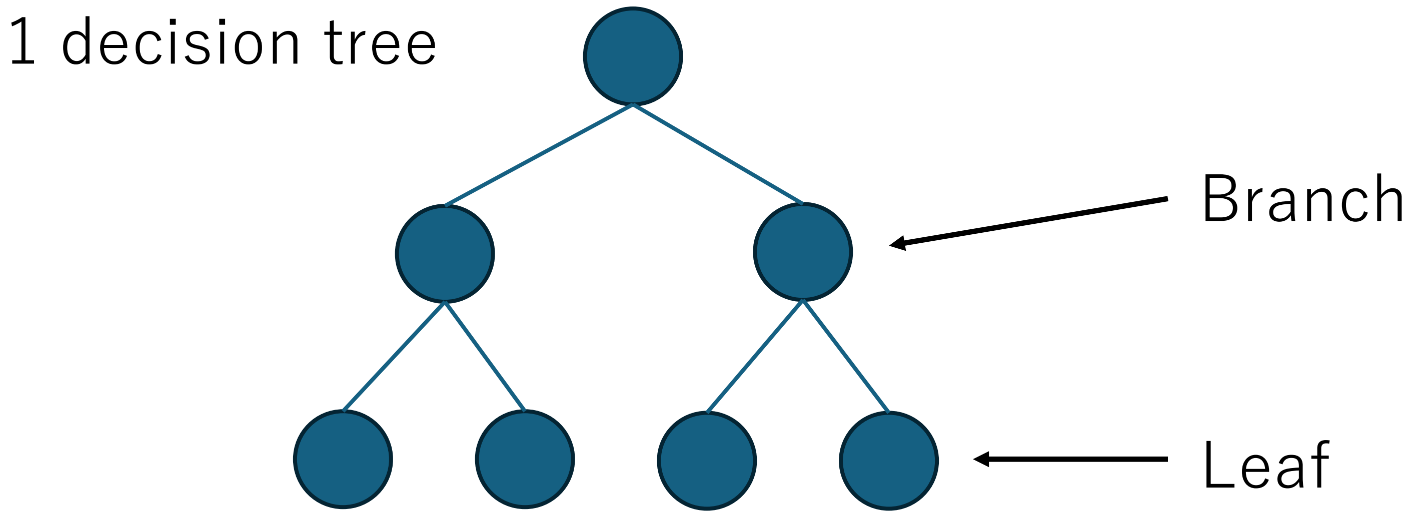
1 decision tree



- Method : **RandamForest**
  - Basic classification method
  - Use many ‘decision trees’
  - Search good branch question that can expect mass

Hyperparameter

- Optuna search best parameter
  - ‘n_estimators’ : 50-500 (the number of trees)
  - ‘max_depth’ : 10-50 (depth of branch)
  - ‘min_samples_split’ : 2-10 (minimum events in branch)
  - ‘min_samples_leaf’ : 1-10 (minimum events in leaf)

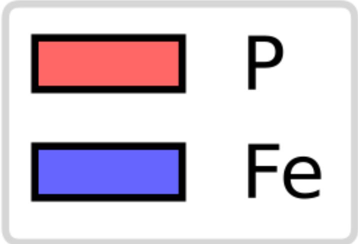


Hyperparameter

EPOS-LHC	Upper layer	Lower layer
n_estimators	356	426
max_depth	41	29
min_samples_split	10	3
min_samples_leaf	4	1

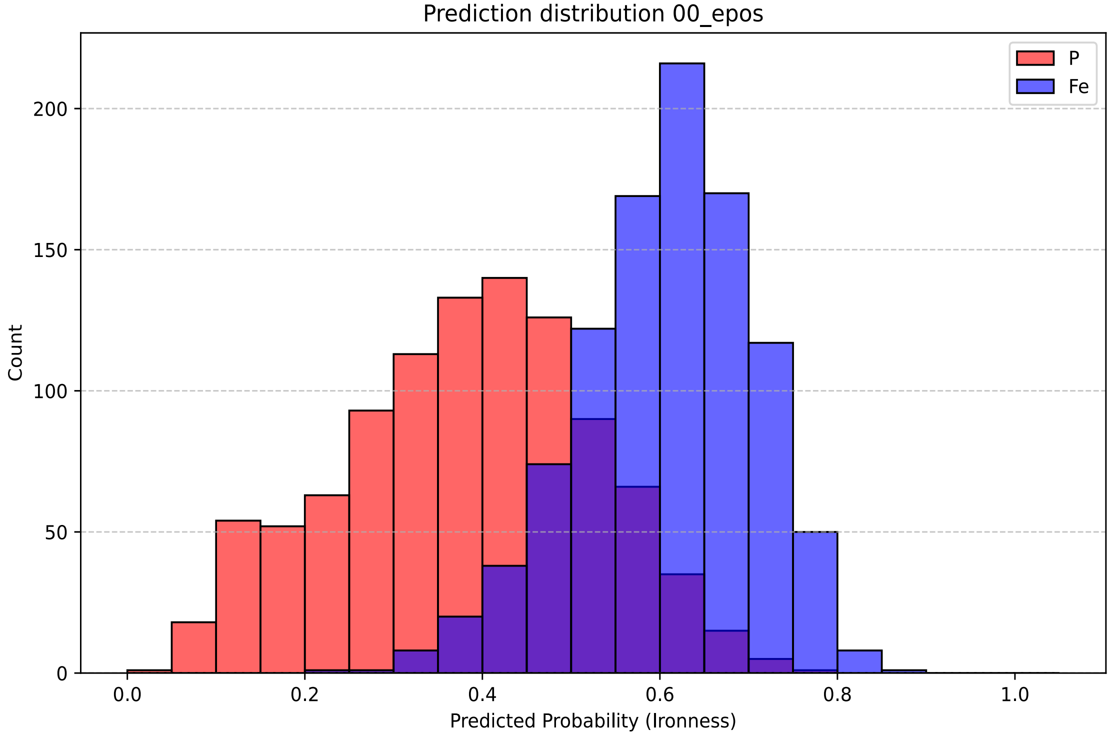
QGSJETII04	Upper layer	Lower layer
n_estimators	461	469
max_depth	33	27
min_samples_split	5	10
min_samples_leaf	1	2

Results (EPOS-LHC)

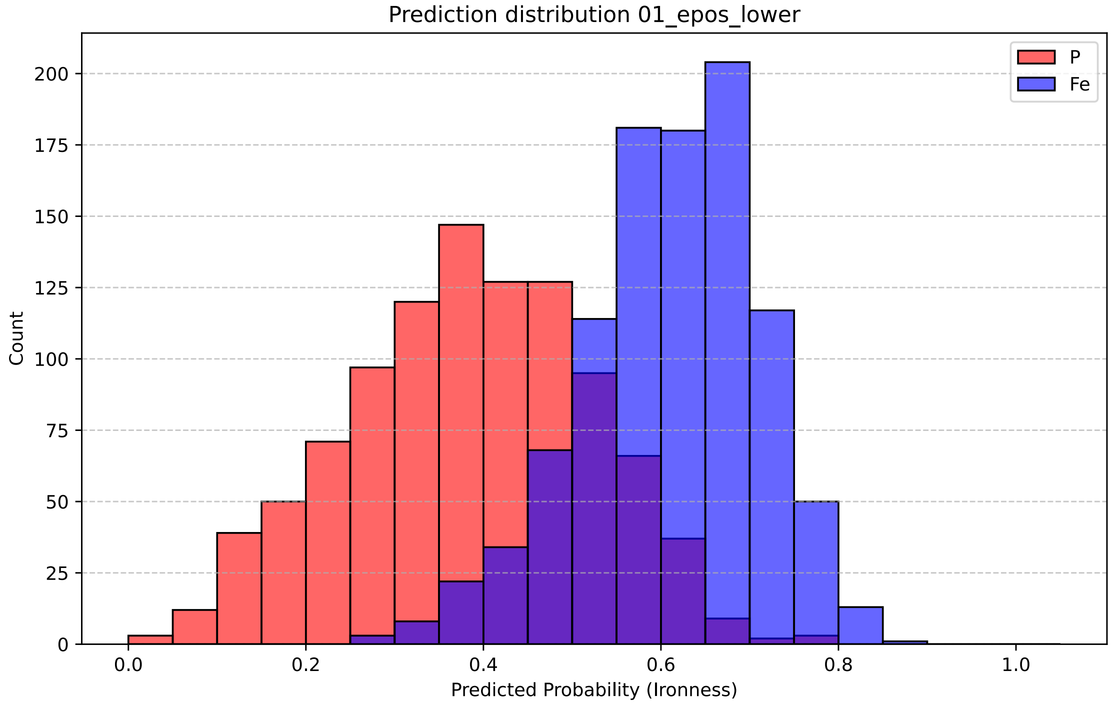


- Validation events prediction

Upper

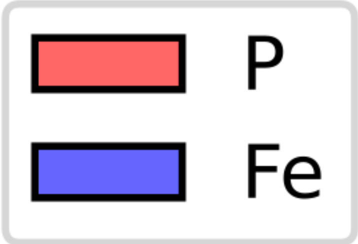


Lower

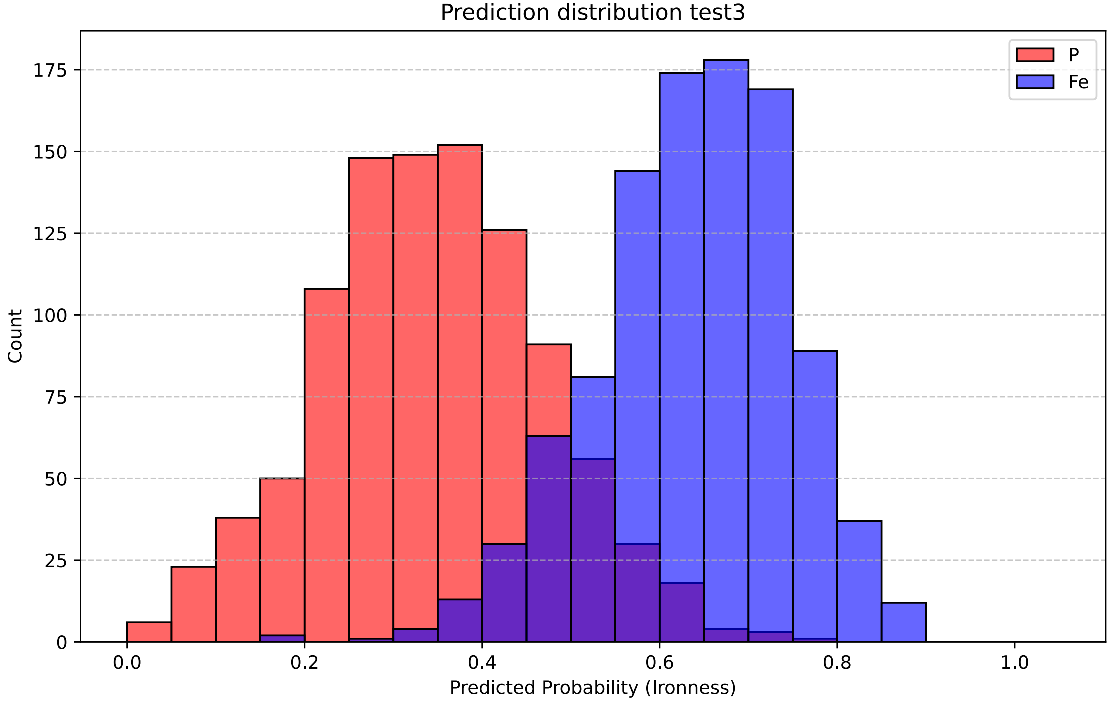


Results (QGSJETII04)

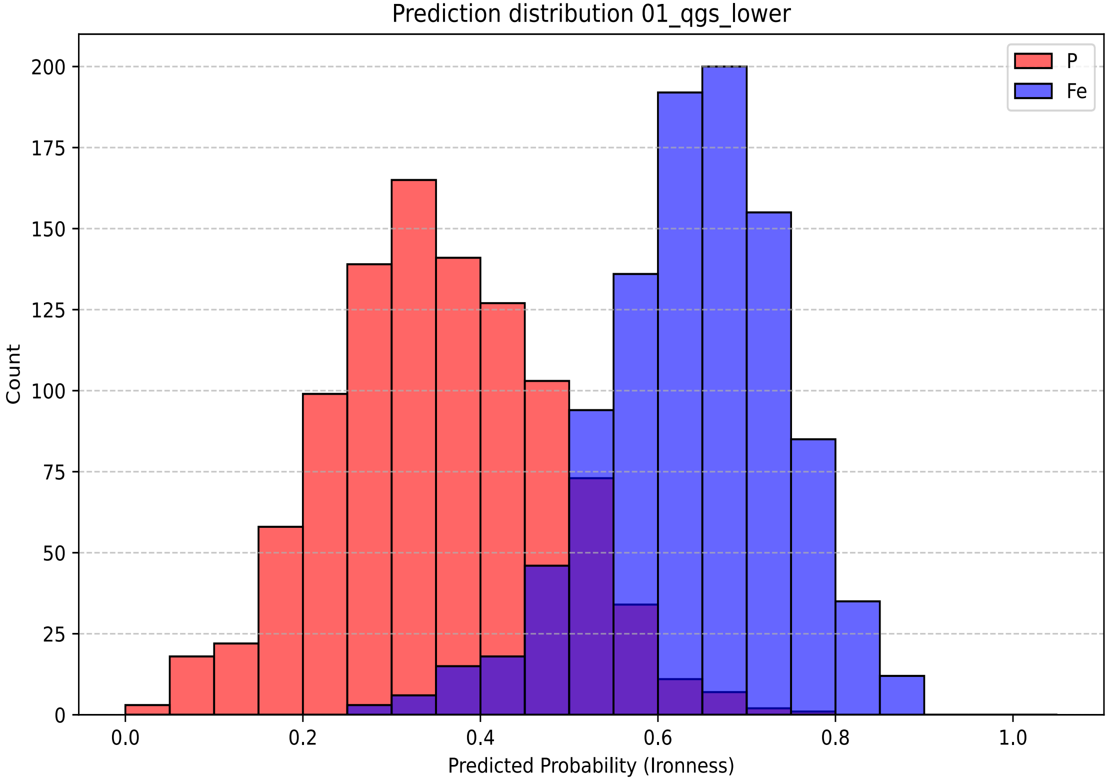
- Validation events prediction



Upper



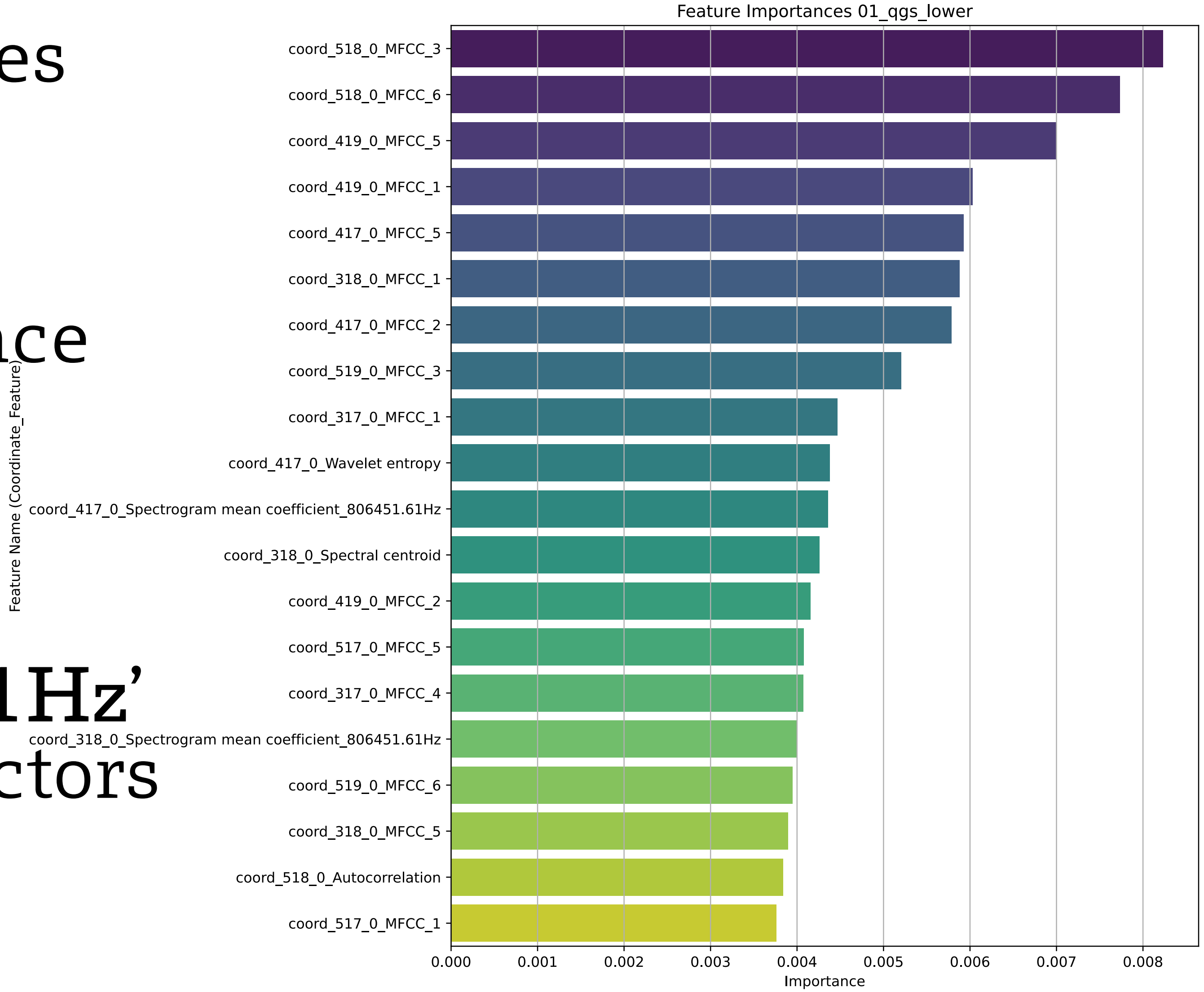
Lower



Feature importance

- RandomForest can provides **'Feature importance'**
- Even the highest importance is less than 1%
- **'MFCC'** 'Spectrogram…_**806451.61Hz**' is important in many detectors

QGSJETII04_lower



Feature importance

- MFCC : Mel-frequency cepstral coefficients
  - Analyzes high frequencies broadly and **low frequencies in detail**.
  - Originally used for audio analysis.
    （This is because humans perceive differences in low frequencies more clearly than in high frequencies.）
- 806451 Hz ≒ 1 MHz
  - SD’s sampling late
    - 50 MHz → 20 ns bins（A single particle scale）
  - Frequencies effective for classification
    - **1 MHz → 1 μs scale**（**An air shower scale**）

Summary

- Extract features from SD’s waveform using **TSFEL**
- Expect the mass of Amaterasu particle with **RandamForest**
- QGSJETII04 seems more efficient for learning
- The low frequencies in air shower waveform
  may be important for classification

spectrogram_mean_coeff (signal, fs[, bins])	Calculates the average power spectral density (PSD) for each frequency throughout the entire signal duration provided by the spectrogram.

• A waveform data

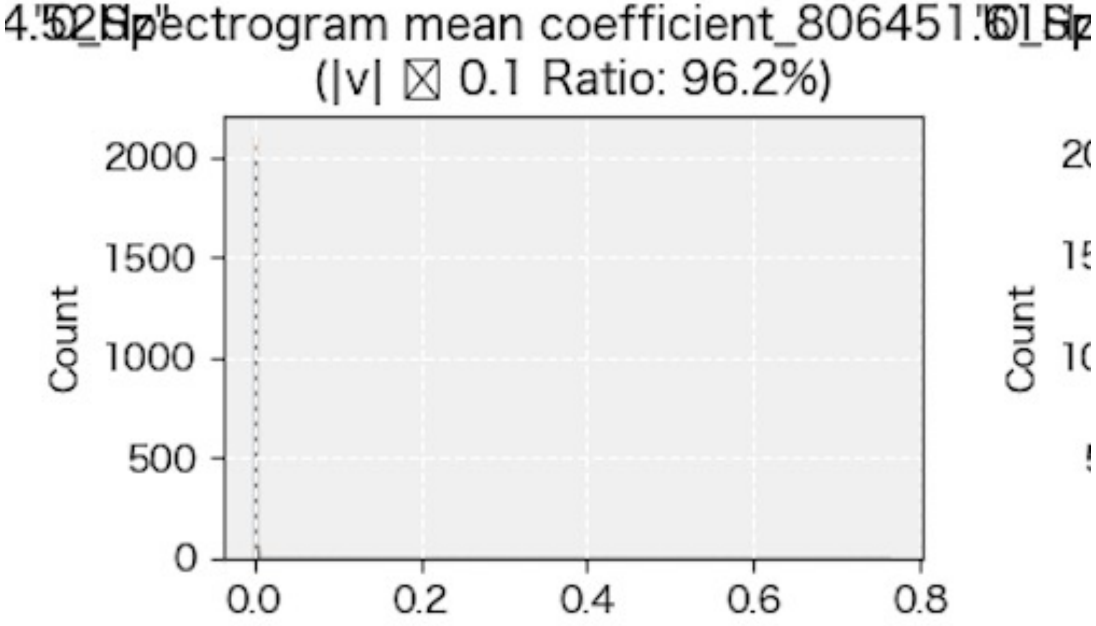
```
420,151,101
421,190,135
422,181,139
423,146,120
424,110,91
425,88,80
426,90,96
427,109,121
428,126,135
429,116,125
430,127,159
431,174,224
432,222,261
433,247,258
434,235,234
435,219,216
436,220,215
437,223,222
438,202,212
439,197,224
440,193,230
441,171,220
442,134,190
```

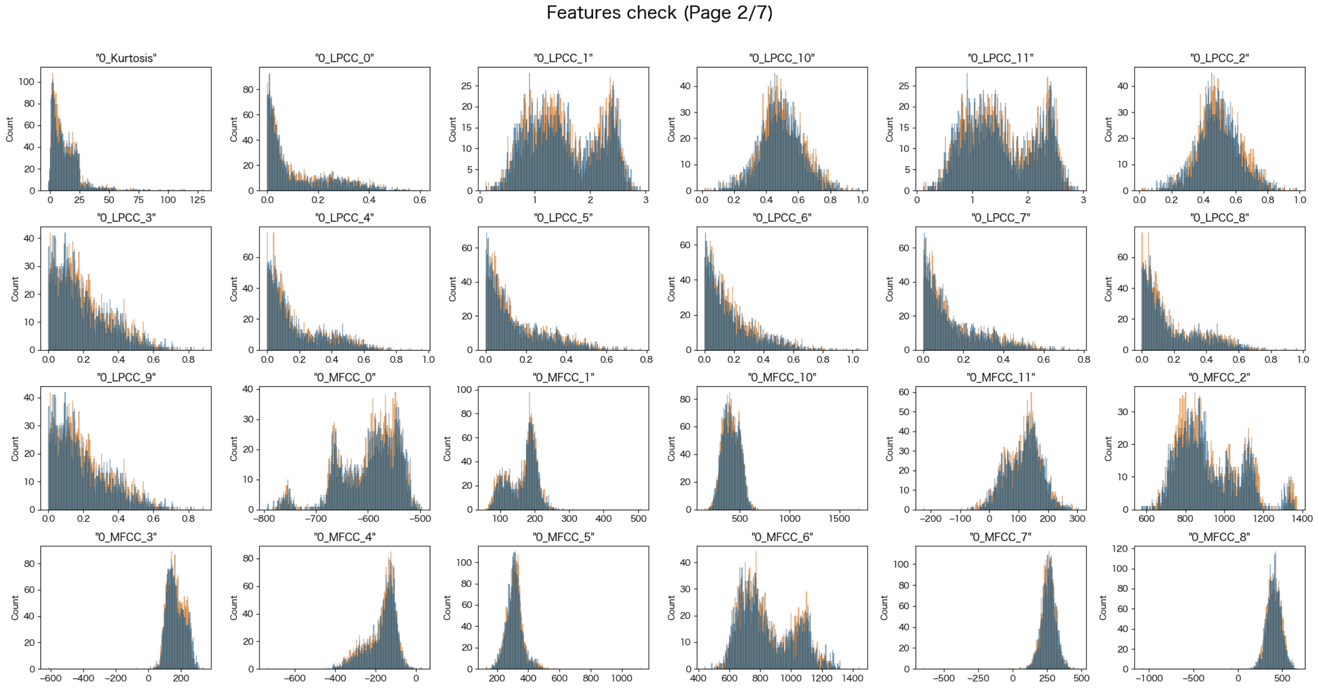
Optuna search good parameter

```python
#Optuna
def objective(trial, X, y):
    params = {
        'n_estimators': trial.suggest_int('n_estimators', 50, 500),
        'max_depth': trial.suggest_int('max_depth', 10, 50),
        'min_samples_split': trial.suggest_int('min_samples_split', 2, 10),
        'min_samples_leaf': trial.suggest_int('min_samples_leaf', 1, 10),
        'random_state': 42, 'n_jobs': USECORES
    }
    model = RandomForestClassifier(**params)
    skf = StratifiedKFold(n_splits=N_SPLITS, shuffle=True, random_state=42)
    scores = []
    for train_index, val_index in skf.split(X, y):
        X_train, X_val = X[train_index], X[val_index]
        y_train, y_val = y[train_index], y[val_index]
        model.fit(X_train, y_train)
        score = accuracy_score(y_val, model.predict(X_val))
        # score = log_loss(y_val, model.predict_proba(X_val))
        scores.append(score)
    return np.mean(scores)
```

Extracted features sample

- Left : Spectrogram mean coefficient 806451.61Hz
- Right : features which have various value
  MFCC is bottom





How get waveform

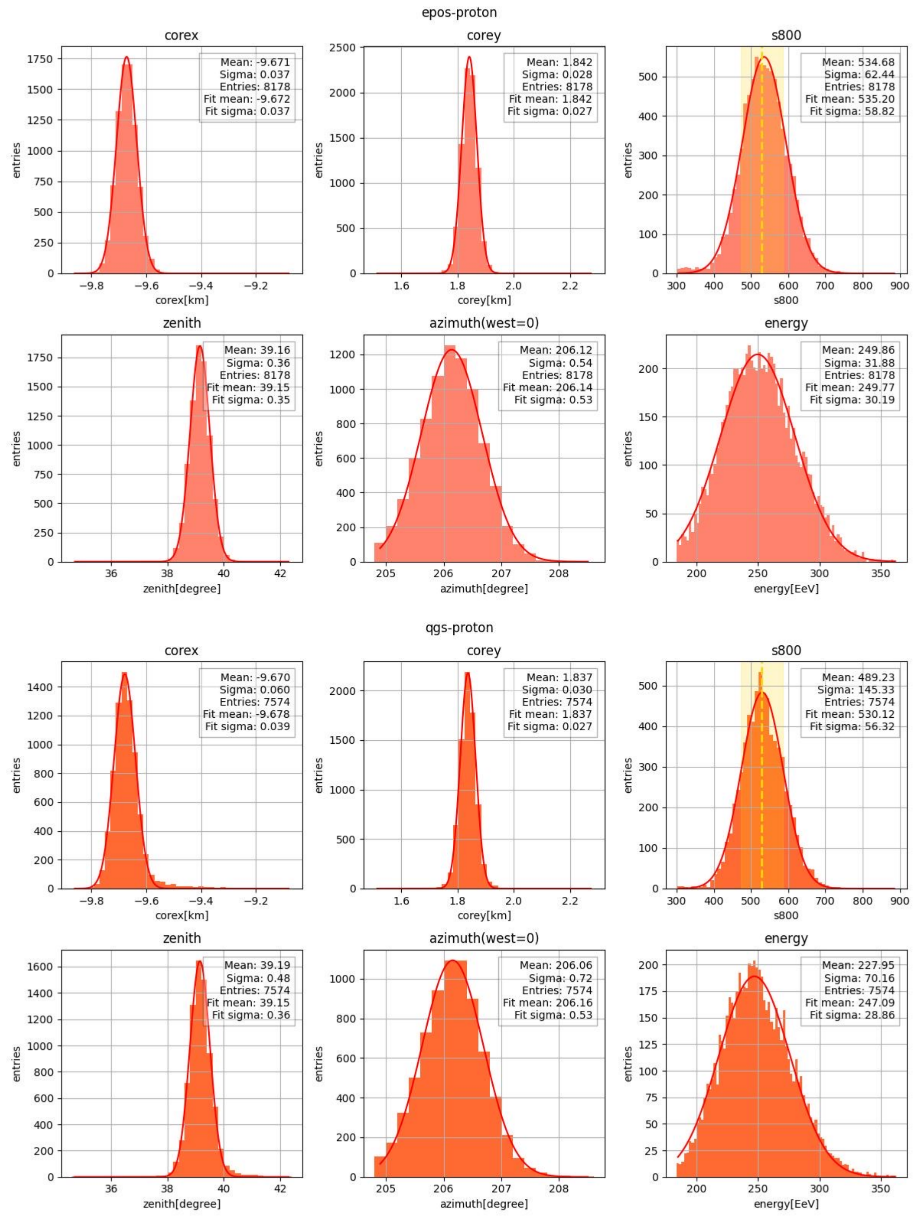
- I use dstio made by komae-san

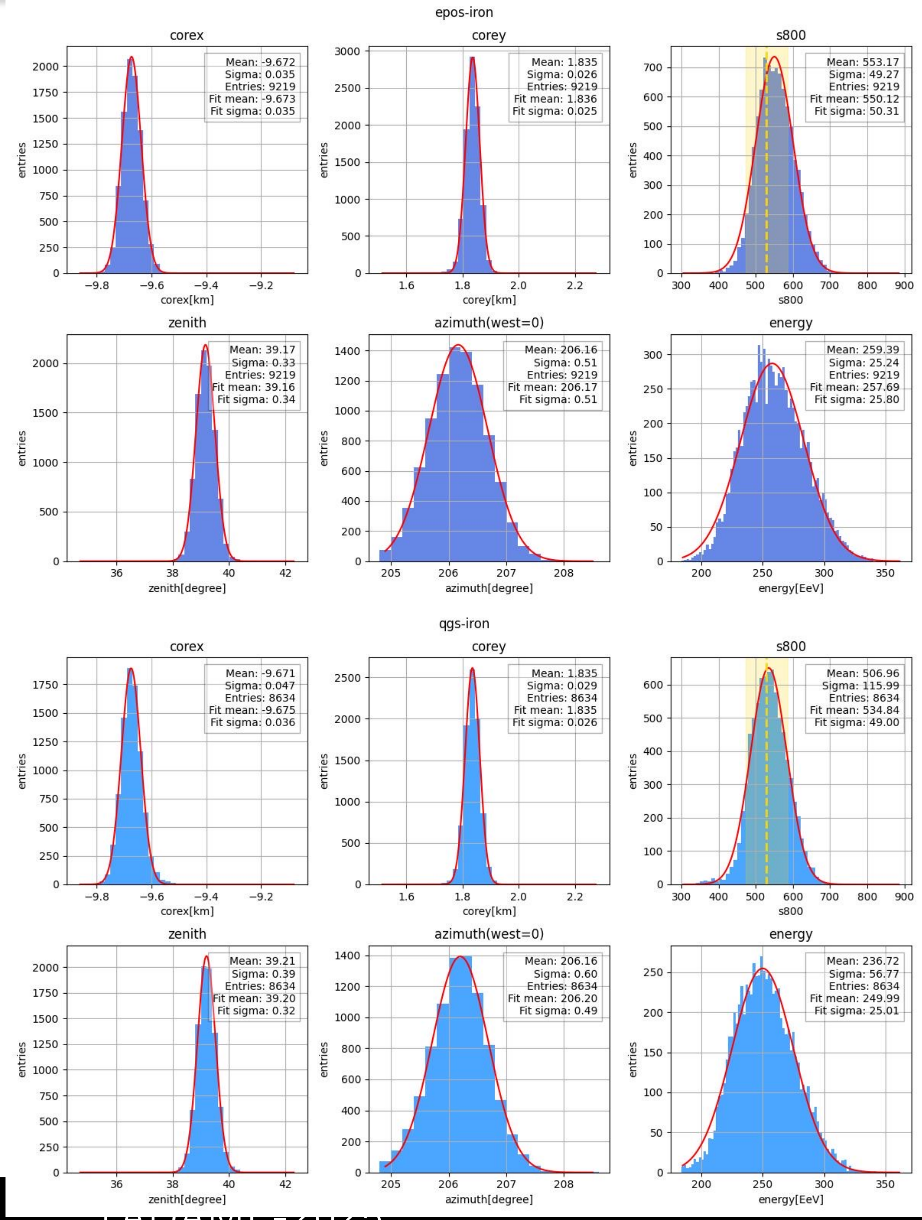
```
try:
    event = dstio.read_dst(dst_path)
except Exception:
    continue

dat_dir = os.path.join(save_dir, os.path.basename(dst_path))
for i, evt in enumerate(event[0:100]):
    try:

        integral_data_list = []
        event_dir = os.path.join(dat_dir, f"event{i}")
        os.makedirs(event_dir, exist_ok=True)
        wf_dir = os.path.join(event_dir, "wf")
        os.makedirs(wf_dir, exist_ok=True)


        sdids = evt['rusdraw']['xxyy']
        fadc = evt['rusdraw']['fadc']
        clkcnt = evt['rusdraw']['clkcnt']
        pchped = evt['rusdraw']['pchped']
        mip = evt['rusdraw']['mip']
        # 範囲内で最小のclkcntを取得
        min_clkcnts = [
            clkcnt[i] for i in range(len(sdids))
            if sdids[i] in sdid_set and any(val > 30 for val in fadc[i][0])
        ]
```

Energy

- EPOS, P : 261 EeV
- EPOS, Fe : 240 EeV
- QGS, P : 300 EeV
- QGS, Fe : 272 EeV

MC reconstruction data





Model

- EPOS-LHC
  - 'number of events' : P:10000, Fe:9900
  - 'n_estimators' : 50-500 (the number of trees)
  - 'max_depth' : 10-50 (depth of branch)
  - 'min_samples_split' : 2-10 (minimum events in branch)
  - 'min_samples_leaf' : 1-10 (minimum events in leaf)
- QGSJETII04
  - 'number of events' : P:9241, Fe:9190
  - 'n_estimators' : 50-500
  - 'max_depth' : 10-50
  - 'min_samples_split' : 2-10
  - 'min_samples_leaf' : 1-10

Optuna search best parameter in this range