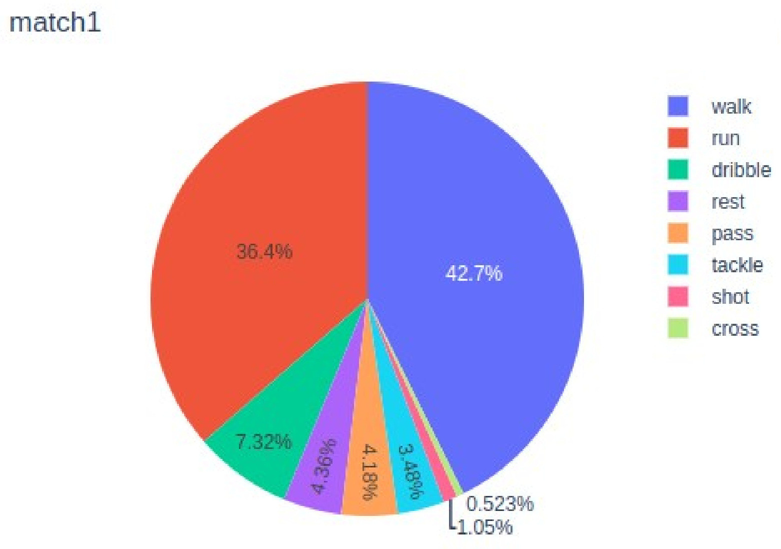
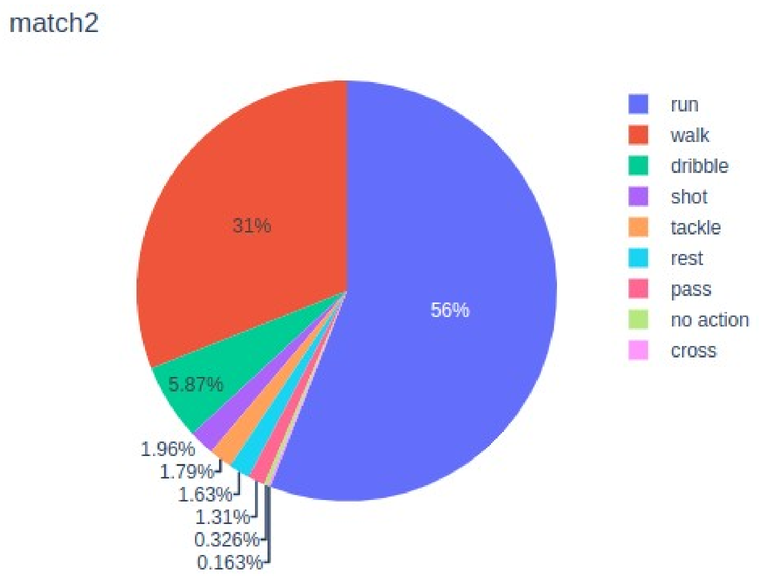
1. Data exploration:

analyze_dataset.py   save the figures comparsing in "figure/analyze"

alternative version is analyze_dataset.ipynb

From the different pictures, I guess match_1 is a player, match_2 is another players.

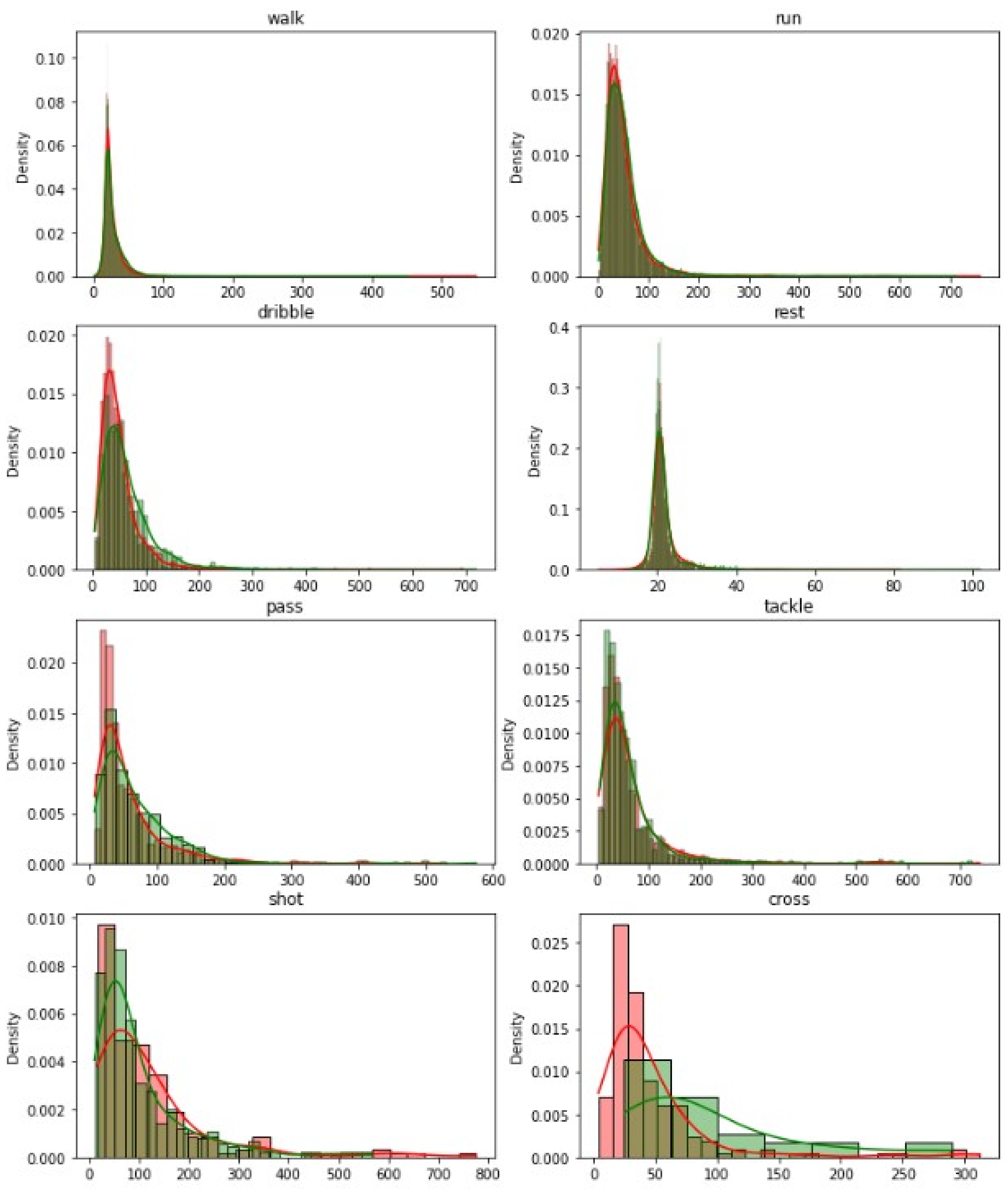




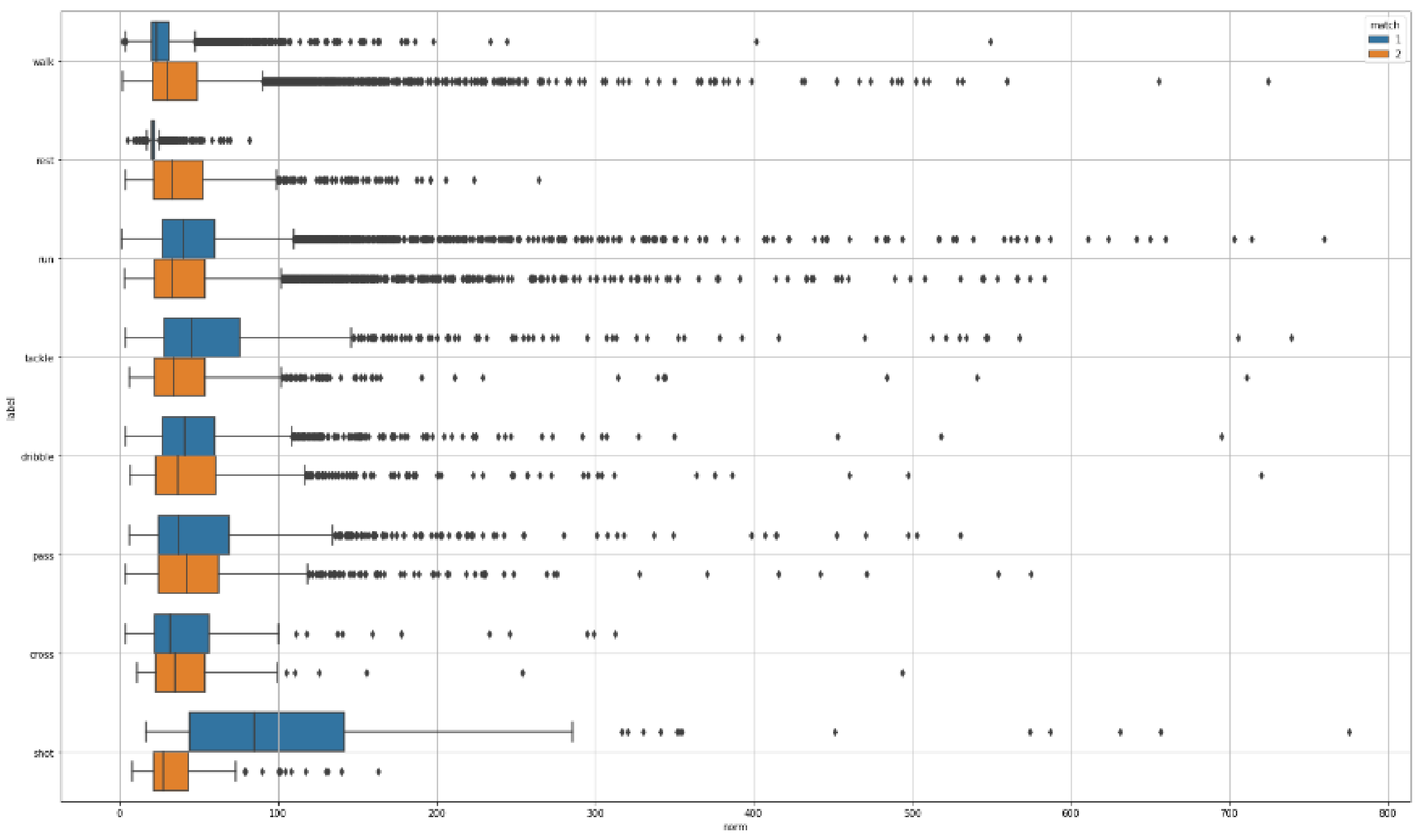
match_2 has one extra label "no action"

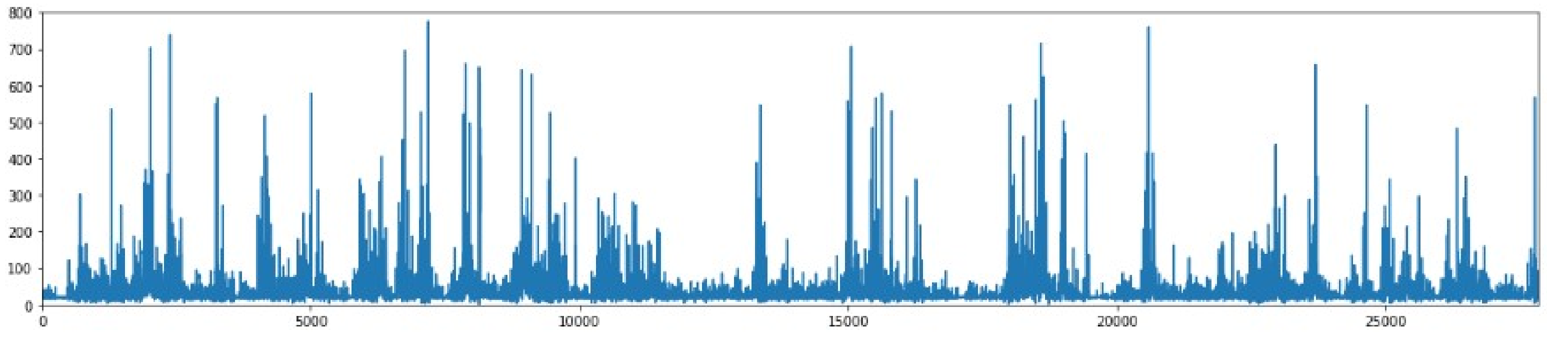
'run' and 'walk' plays the majority roles.

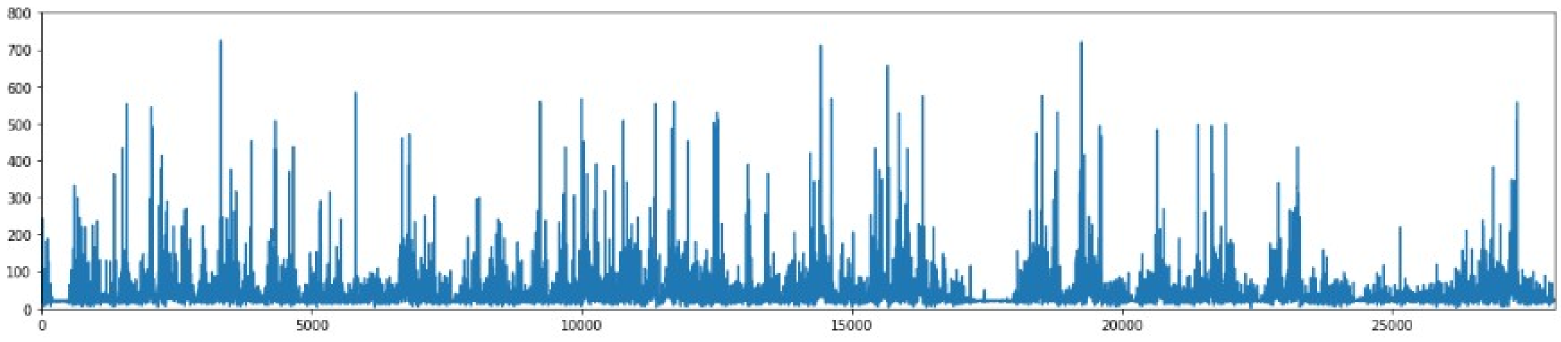
In match_1, "walk" takes more proportion than "run", while "run" takes more in match_2



The hist figure shows us: match1 and match2 has the similar distribution except cross







the acceleration movement in 2 matchs are both irregular.

To recreate the game:

At first, we should make the model to predict the next action and norm.

We can use 2 types of models:

1) tree based model: such as randomforest, xgboost, lightgbm
2) time series based model: arima, sarima, rnn, lstm, transformer

3. Explain the model:

Due to the calculation power, I only take match_1 to training the model.

The train, val, test set ratio is 0.7, 0.20, 0.1

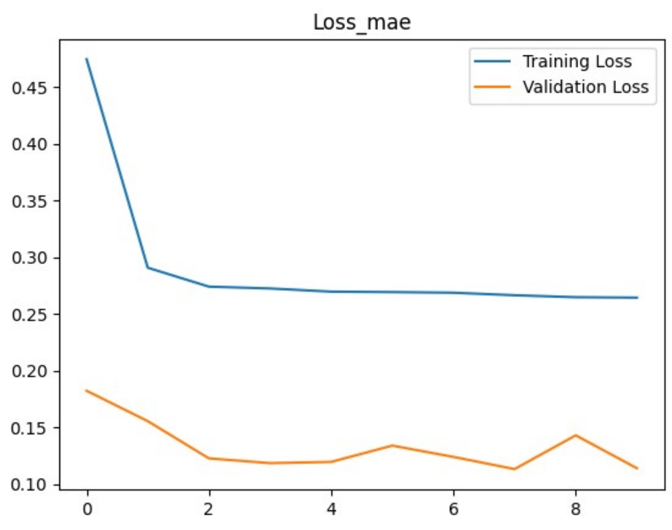
Taking the past 10 timesteps to predict the next step

1) For LSTM: I´ve taken

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_2 (LSTM)               (None, 11, 50)            10400
 lstm_3 (LSTM)               (None, 25)                7600
 dense_1 (Dense)             (None, 8)                 208
=================================================================
Total params: 18,208
Trainable params: 18,208
Non-trainable params: 0
_________________________________________________________________
```

The norm is normalized with log.

With epochs=10, loss=´mae´, I chosed the model in the epoch which has the smallest value in validation set.



Evaluation:

Train Score: 0.17 MSE

Val Score: 0.13 MSE

Test Score: 0.15 MSE

Train Score: 0.26 MAE

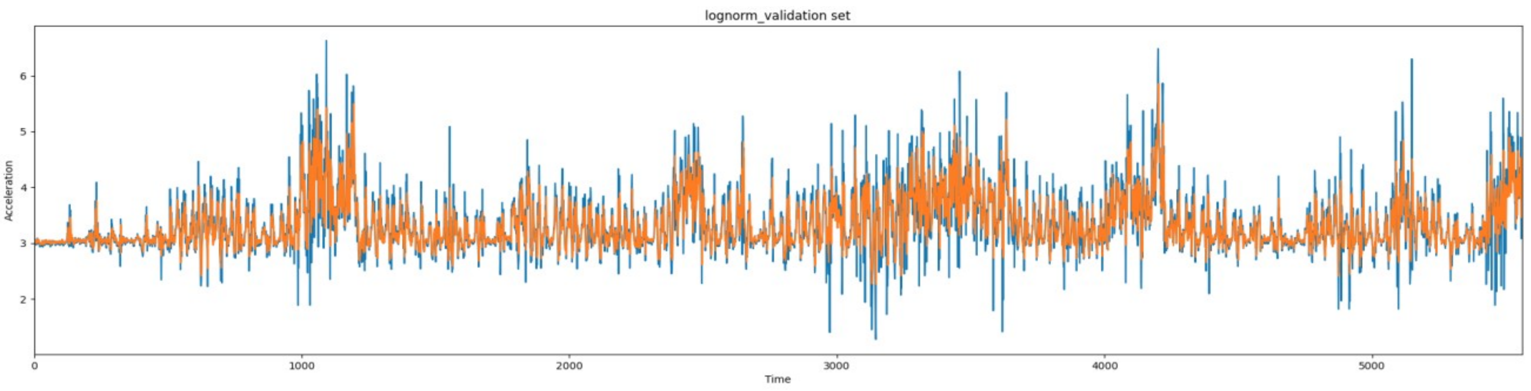
Val Score: 0.22 MAE

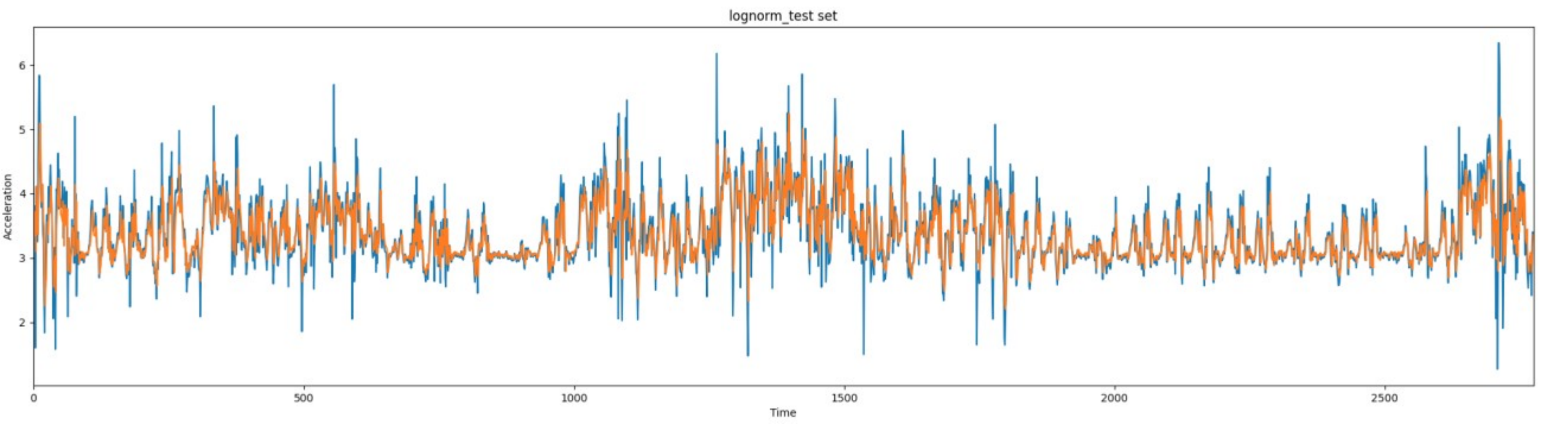
Test Score: 0.24 MAE

Train Score: 0.99 Acccuracy

Val Score: 0.98 Accuracy

Test Score: 0.99 Accuracy





2) For random forest,

Train Score: 0.02 MSE

Val Score: 0.13 MSE

Test Score: 0.14 MSE

Train Score: 0.10 MAE

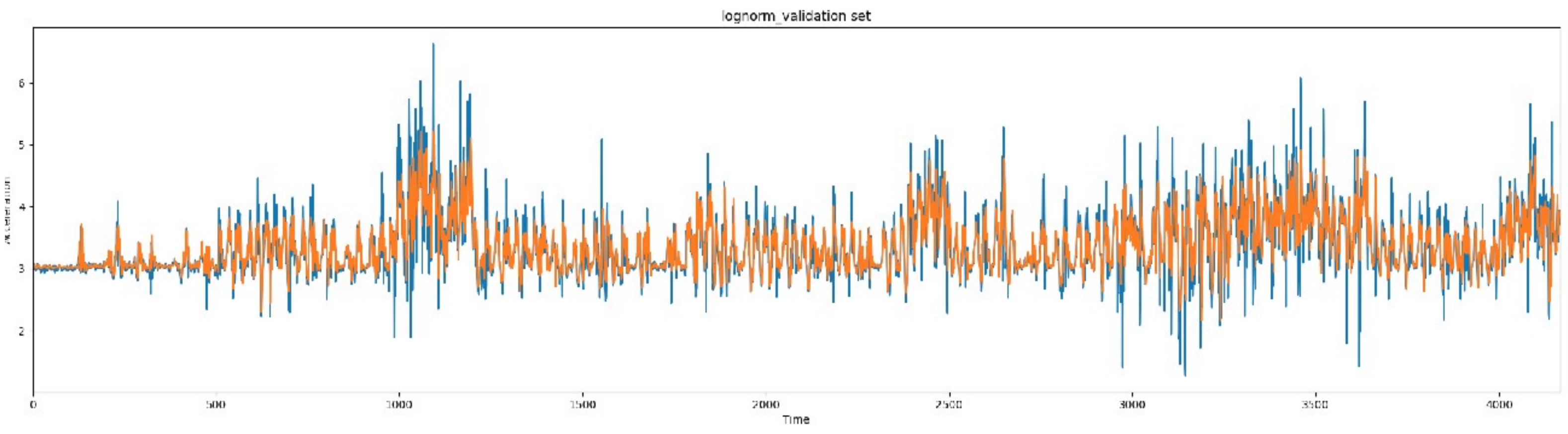
Val Score: 0.22 MAE

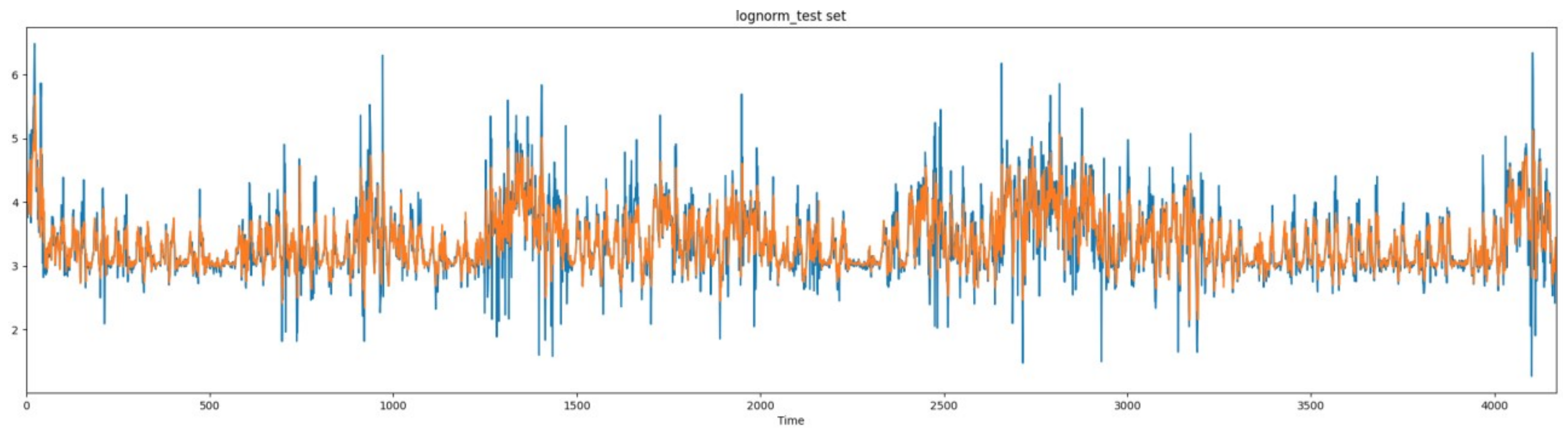
Test Score: 0.23 MAE

Train Score: 1.00 accuracy

Val Score: 0.98 accuracy

Test Score: 0.99 accuracy





4.At first, we should make the model to predict the next action and norm.

We can use 2 types of models:

1) tree based model: such as randomforest, xgboost, lightgbm

2) time series based model: arima, sarima, rnn, lstm, transformer

I chosed randomforest and lstm to predict the next timestep.

Then we can at first use mplsoccer to create the pitch, and take cpgames to create the game. (please see the small game).

Then combine with my predict model.

At this moment, since the deadline is only 1 week, I don´t have time to combine the two.

I would like illustrate my idea:

- Calculate the distance L2 from each players to the ball , choose the player which has the smallest distance to get the ball (player_get)
- Considering the distance as the following to predict action
  1) distance between player_get and the rival teams´ players
  2) distance between player_get and the own teams´ players
  3) distance between player_get and the goal
- Considering the action according to the predict model

5.

a. Yes it is possible to generate as many games as we want.

b. yes,It is possible to generate the game of any length.

We can put the prediction value to the next line as the input

c. to generate the game with more attacking, we can change:

-give a higher probability of one action to another actions

- give a larger norm´s range to random select values