**Model specification**

Input:(Only Team\_Data is used in this project)

Upper section, expected performances:

(also named as ‘x1’ in codes, e.g. data\_dict[‘train’][‘x1’])

Home/Away

xG: Expected Goal

xGA: Expected Goal Against

xpts: Expected points

npxG: Expected Non-penalty Goals

npxGA: Expected Non-penalty Goals Allowed

npxGD

Lower section: expectation adjustment (Condition adj.) (t-1 to t-5):

(also named as ‘x2’ in codes, e.g. data\_dict[‘val’][‘x2’])

Home/Away

xGD = scored - xG: Actual - Expected Goal

xGAD = conceded - xGA: Actual - Expected conceded

xptsD = pts - xpts: Actual - Expected points diff

ppda\_att\_diff = ppda(att) - ppda\_allowed(att)

ppda\_def\_diff = ppda(def) - ppda\_allowed(def)

ppda\_coef

oppda\_coef

deepD = deep - deep\_allowed

npxGD

(if there is an equal sign after the column, it means it is manually created by the logic)

Output:

Win/Draw/Lose

(also named as ‘y’ in codes, e.g. data\_dict[‘test’][‘y’])

**Data Transformation**

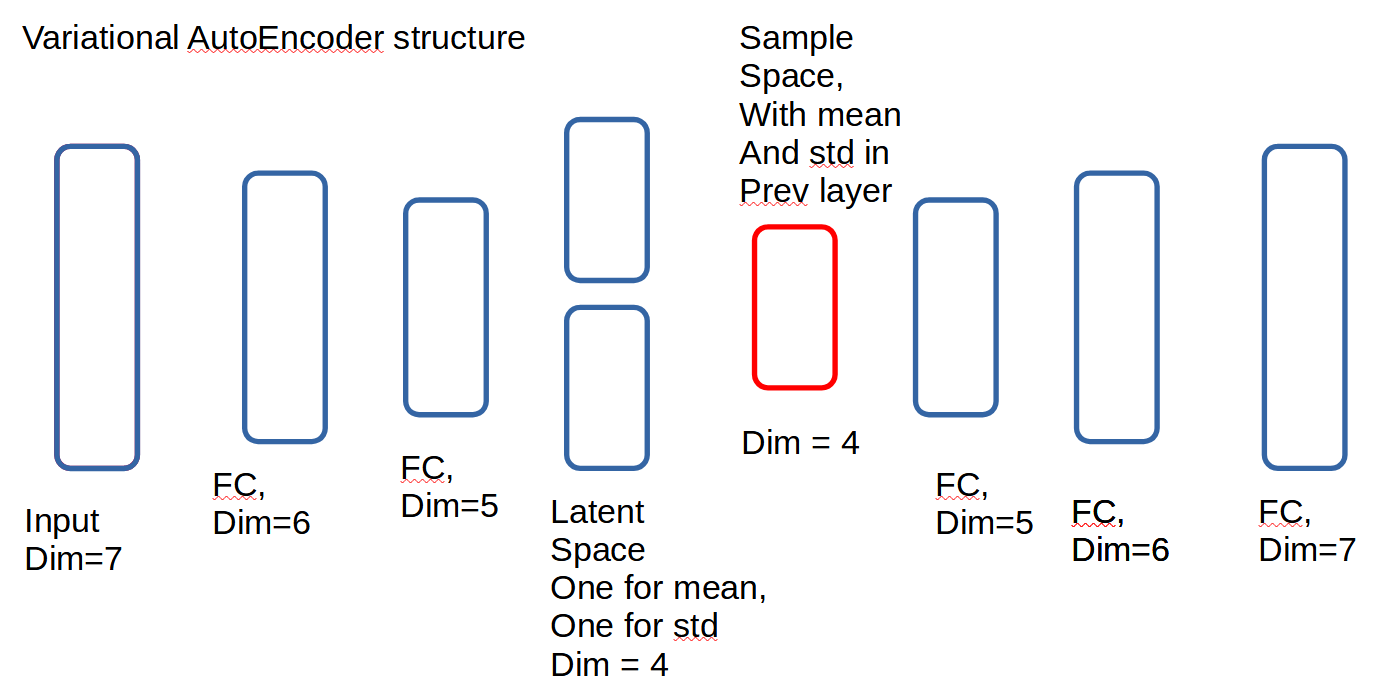
Only `Home/Away` is used as the categorical variables for model input. It is transformed to column “home\_game” which equals to 1 if its home game else 0.

The remaining columns that listed above are all numeric. Each column will be standardised using the same column but 1 year before to compute its mean and std used in standardisation. (e.g. `xG` for 2015 Manchester City is standardized with the mean from the xG of all teams in 2014 to prevent data leakage issues).

The data is then split to train, val and test using 2015-2017, 2018 and 2019 data respectively.

**Model Structure**

**VAE:**

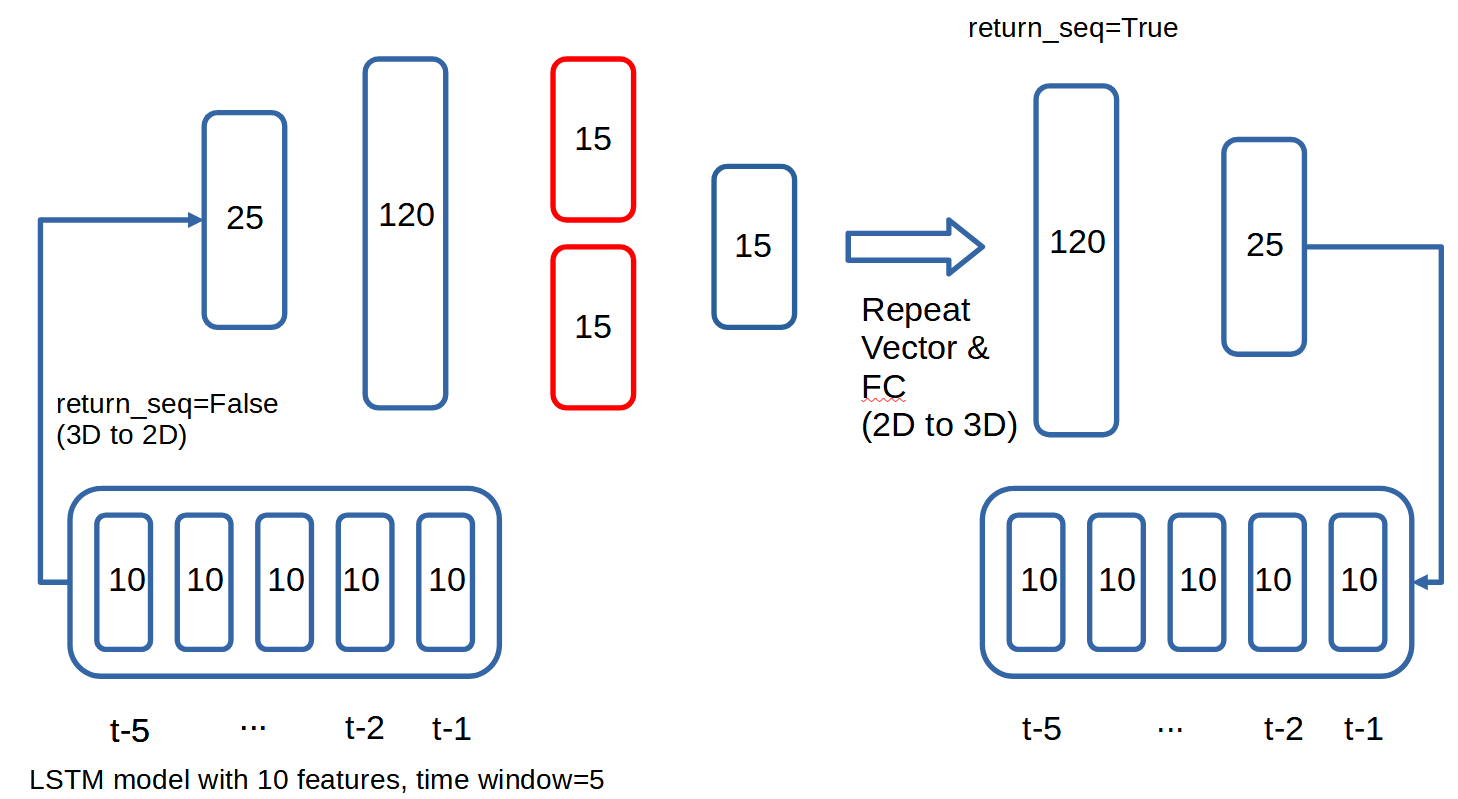
****

\* FC means Fully Connected

Red is rectangle is the sampled vector from the two blue rectangle (mean and std latent space) in previous layer

It is build with upper section data (aka ‘x1’ data only)

**LSTM VAE:**



The input of LSTM model is the 1 categorical and 9 numeric features stated in the above section, with the time window frame, t=5. So the input is dim is (5,10).

It will then be connected to a 25-dim LSTM layer with return\_sequence=False, so it will be a latent vector. The vector will then be fully connected to 1 layer then to the 2\*dim=15 latent vector to be the representation of the latent space mean and standard deviation.

We will then draw samples from the latent space with dim=15, and will go through Repeat vector layer then the LSTM and Dense layer to try to reproduce the input.

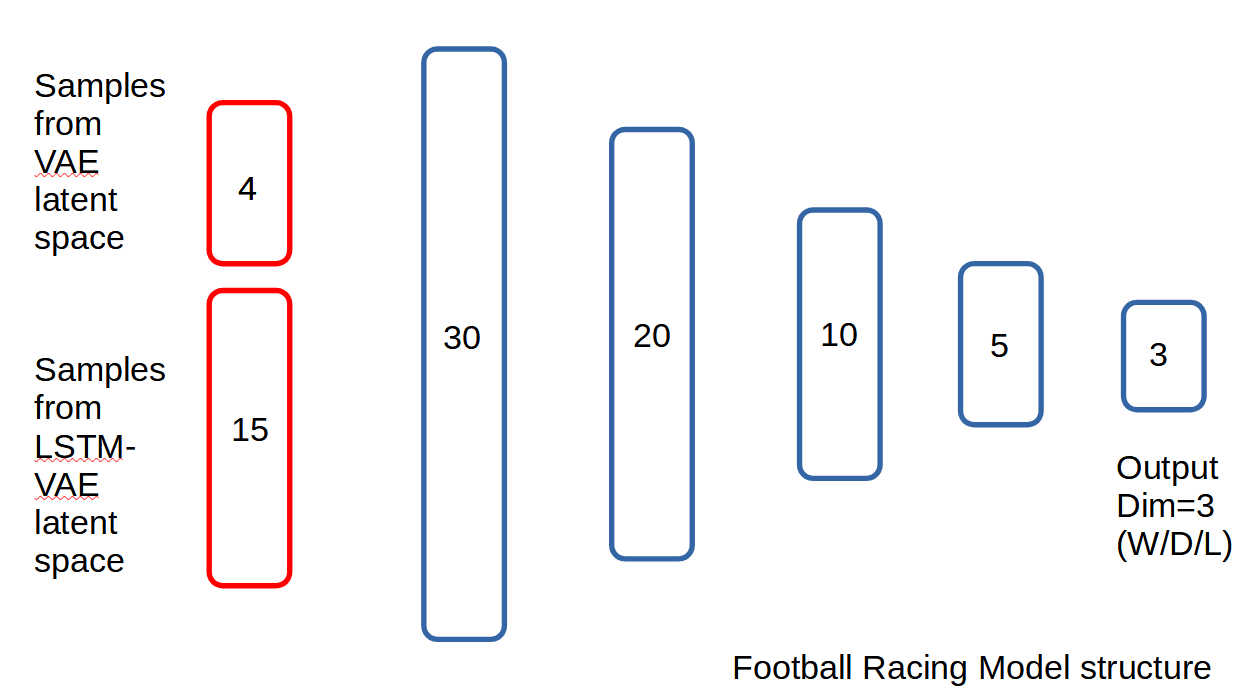
Same as the VAE model, we will draw samples from the latent space mean and std to use as input of the final Football Racing Model.

It is build with lower section data (aka ‘x2’ data only)

(Remarks: the input data of the model is a 3-d array with dimension: (# of train records, time-window, # features) = (1980, 5, 10) in training.

Note that the time window is chosen to be 5 as we would like to use the previous 5 games' expectation vs actual performance difference to feed into the model to learn such that we will not lose a lot of data. More can be discussed in the presentation if interested.

**Football Racing Model Structure:**



We sample from the VAE and LSTM-VAE encoder models’ latent space and pass through a fully connected NN with 3 outputs (Win/Draw/Lose).

Note that the model will output a diff result every time as the input is not static and samples are drawn from latent space. We will be able to leverage this property to estimate how certain our model prediction is for each incident. This will be further discussed in the report side.