Nicola LAMBO, Jackson YU, Naji EL-KHOURI MAIS 202

DELIVERABLE 3: REPORT

We have decided to train an RNN model to predict the statistics for the next match a soccer player from the Premier League or La Liga will play.

Regarding how we got to this iteration of our model, we initially thought, after some research, that an LSTM model would be optimal in predicting next season's stats, with the help of a database covering all players' final season stats in the top 5 European football leagues from 2015-2016 all the way to 2023-2024.

For the data preprocessing, we took each players statistics from each of their previous seasons with the oldest being 2015-2016 season and turned their stats into Pytorch tensors that have their position on the pitch (one-hot-encoded), their current club (one-hot-encoded), their league (one-hot-encoded), and then a sequences of numerical values such as goals, assist, matches played, matches started, minutes played, etc. After speaking to our TPM, we noticed that we could not possibly have enough seasons to build a functioning LSTM, and so our TPM suggested that we focus on matches rather than seasons, because we would have much more data to be used in a sequence and we could potentially get a working model. The immediate context within a timeframe of matches matters more for the player's current form, rather than the overall season, and so we settled on using an RNN model, because we don't need the long-term memory of an LSTM for our particular context.

There is much work to be done for our RNN model to be completed, since we spent a good deal of time trying to get the LSTM to work, and changed our approach quite drastically after realizing it would probably not work the way we intended. We have a simple frontend built, ready for when the model is finished to be implemented in the backend. By the expo date, we will make sure to have our working RNN model serving as the backend to the frontend we have built. We will measure the performance of our RNN once it is working as intended with a mean squared error function, and we will continue to train the RNN with more data until we see little to no improvement in terms of the mean squared error.

Regardless, in the spirit of handing in what we have worked on so far, we will upload our previous attempts at our LSTM model, along with the code used to handle the data, and the data itself.