Cincinnati Reds

Pitch Mix Report

Summary

This report will consist of 2 parts:

1. **Discussion of the chosen model and the data input and output of the model, as well as what the results can tell us about the model**
2. **Discussion of trade offs and what could be improved upon given further data or time**

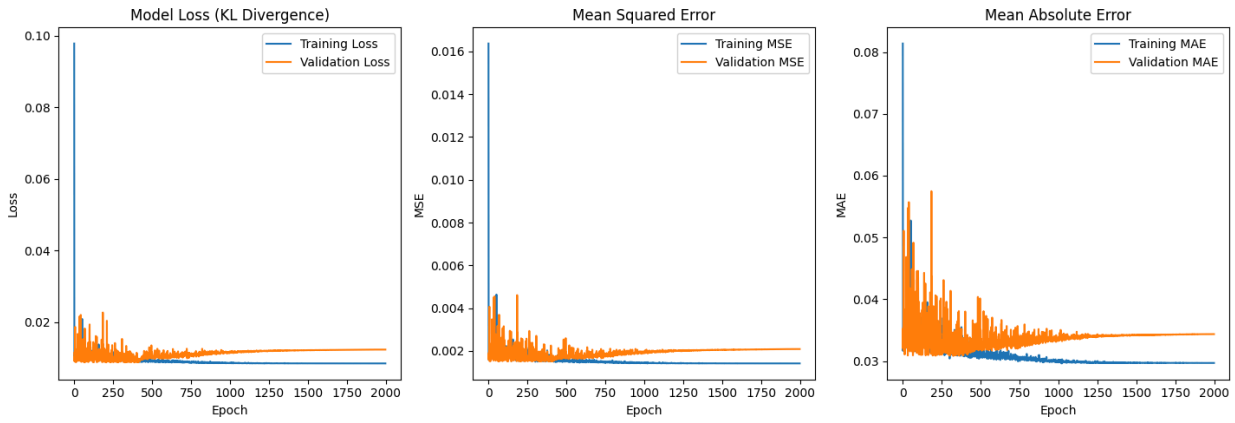
The Model

The model that I decided to use for this project was a sequential neural network, used for a multi-class regression style of problem. For the loss function I used Kullback Leibler Divergence as this is best for problems where the output is a probability distribution instead of a strict one-hot encoding classification style problem. The output of my model was a 3 class probability distribution representing each type of pitch in the pitch mix for the next year (Breaking Balls, Fast Balls, and Off-Speed pitches). Softmax was used as a final activation function as it makes sure that these probability distributions add up to 1. The inputs of my model where as follows:

- The current year pitch mix as percentages for the batter
- The current year Strike, Balls, and In-Play balls for the batter
- The current years % of in-play balls by pitch type for the batter

If I had more time to work on this project I would definitely include a greater amount of features for the model, as the choice to use an aggregated data approach over a time series based approach was to take advantage of the width of data given over the depth.

Finally for the model, below I've included a graph of the loss functions of my model, for what I determined to be the best learning rate, batch_size, epochs, and validation_split for the model (These are included in the pitch_model.py file).



We can see from these graphs that we have used 2000 epochs and that as the model goes on validation loss actually grows worse. I was able to produce better validation losses when decreasing my epochs or validation split, or by increasing my batch size. The issue with these approaches was that the model would over-generalize, predicting the same prediction no matter the input, instead of taking into account features and their effects on the output. Despite the decision not to go with those over-generalizing models, they still show us that the chosen inputs may not be the strongest for predicting pitch-mix.

Reflection and Struggles

Looking back, if I had more time it would make more sense to approach the problem using a time series model, as this would allow the model to benefit from the great depth of data, instead of aggregating it down to only ~450 training data points. The time constraint of this project definitely presented a challenge for using a time series model, as it may have taken many more hours to get my abandoned time series model to be more accurate then that of a model with aggregated data. Another thing I would've liked to implement given more time would be some form of embedding layer that could help the model understand specific players as separate entities, and influence its predictions on a batter to batter basis.

Another constraint presented was the lack of data. For example if the data included 2024 years data on which pitchers a batter was expected to face, then I could have made much better use of the pitchers data which may have greater weight on the pitches thrown over the batter. I also noticed that the data had 4 players who hadn't received pitches in 2023. This means relying on their 2022 data to make predictions which may have resulted in much less reliable results for those players. Overall I believe the model tells a lot about pitch mixes in general as well as some players with largely differing predicted pitch mixes, but with more time I believe I could definitely improve on many features of the model.