MLB Game Score Prediction Model

My idea for my capstone project was a Machine Learning Model that could predict the score/score range for an MLB team's next game. To do this the model will be trained on a teams previous schedule and the resulting score/score range for each game. Later on player data could be added to the data as well. This model could be used by sports bettors to help aid in their bets or it could be used by odds makers who set the sports betting odds. Sports bettors could make more informed decisions and even get insight into teams that they do not follow that much. Odds makers could use the machine learning model as another resource on top of the statistical models and analysis they already employ.

The data I collect for my model comes from the pybaseball API, which is a python package that scrapes the Baseball Reference, Baseball Savant, and FanGraphs websites for MLB team and player data. I decided to collect data on all 30 teams over the last 10 seasons (2012-2021). For each team and every regular season game they played, I got their game number of the season, date, team, home or away, opponent, runs scored, runs against, wins and losses, rank, games back, day or night, and streak. These are all values we can know before a game that can be processed to predict the runs scored.

Since we know the runs scored for each game this will be a supervised learning problem. It will also be a regression problem because the runs are a continuous value. The model is trying to predict runs scored given the team, game number of the season, date, team, home or away, opponent, runs scored, runs against, wins and losses, rank, games back, day or night, and streak. At first we will perform a grid search using different parameters and three traditional machine learning models (linear regression, xgboost, random forest). Then if those do not perform well we may explore deep learning models such as a linear neural network or a convolutional neural network. Some other ideas for training the model were training 30 individual team models instead of one big model. This would only be considered if the original approach performs poorly with all kinds of models.

The final product will be a simple website that people can access online or on their phone. The application will have a selection with all the games for the day and when a user selects one it will process both teams and predict their score/score range. These predictions would be displayed next to each other on some sort of number line so you could evaluate the run ranges against each other. Later on I could add the sports betting odds for the game, so users could evaluate if they want to make a bet. In the background when a user selects a game, the same database we get our training data from will be queried to get both teams incoming statistics. Then for each team, this info is sent into the machine learning model API I built and this would return the team’s predicted score/score range. I will also save the team's prediction so my API would only get at max 30 calls a day which will reduce the cost.

Depending on the type of model I decide on, my project will need a CPU and maybe even a GPU. For this project I will do all my model training and testing on my kaggle account which has 30 hours of free GPU use per week so this will be plenty of processing power. When I deploy I will use my paper space account and it has 300$ which should be sufficient for the amount of calls my production application will receive. As for memory, my data is only a little over 45,000 points or 2.67 Mb in size so it should not be a problem for storage during any phase.