For this assignment, I chose to use the PyBaseball library to explore how advanced statistics can inform pitch predictions for a baseball player. My initial goal was to create a model that could predict the type of pitch thrown a single player, in this case I chose Zack Wheeler, based on various game and pitch context features. I chose this dataset since it is what I was interested in using for the project in order to use more features to predict single game matchups as wins or losses.

Initially, I encountered issues importing PyBaseball into VSCode, for some reason initial pip installs into terminal did not work as it did for NumPy or Pandas. To address this, I tested the code in Google Colab using pip install pybaseball, where it worked smoothly. Eventually, I set up a virtual environment in Python within VSCode, which allowed me to use PyBaseball locally without any issues.

I began by importing pitch-by-pitch data using PyBaseball's statcast\_pitcher function, selecting Wheeler's data from last regular season, March 28 to September 30, 2024. Originally, my feature set was limited to previous pitch type, count and numbers of outs, but after reviewing PyBaseball's comprehensive statcast dataset, I expanded it to include variables like release\_speed, spin\_rate, release\_pos\_x, and release\_pos\_z, which are for analyzing pitch mechanics. For simplicity, I restricted the data to a single pitcher, but this code could easily adapt to any player by updating the player ID in the get\_player\_id function.

Once I had the data, I faced some challenges due to missing values in rows when using NumPy arrays, so I opted to work with Pandas DataFrames instead, as this allowed me to easily drop rows with incomplete data. Next, I transformed the target variable, pitch\_type, into numerical categories using Pandas as well making it easier for the classifier compatibility.

I chose Logistic Regression as my primary algorithm due to its efficiency in multiclass classification tasks and a good fit to predict the next pitch as data came in. To evaluate the model's performance across different configurations, I implemented an n-fold cross-validation from scratch with five folds. I trained the model on 80% of the data and reserved 20% for testing as instructed by the assignment. I then conducted a custom grid search to tune two key hyperparameters: C for regularization strength and penalty, L1 or L2 regularization. The best configuration was C at 1000 with L1 regularization.

The final model achieved a high accuracy of 99.84% and an F1 score of 0.998 on the test set, indicating excellent predictive power. These metrics suggest that the features selected from the statcast dataset, combined with effective parameter tuning, provided the model with the ability to distinguish pitch types accurately. This assignment demonstrated that PyBaseball data, coupled with effective features and model tuning, can significantly predict pitch types.