Pitch-Tipping Detection Via Human-Pose Analysis

Justin Feldman*
* Khoury College of Computer Science, Northeastern University, Boston, MA

***Abstract* – This project proposes a methodology for classifying baseball pitch-types using biomechanical data extracted from video samples via computer vision pose-estimation. Specifically, it identifies subtle, repetitive body movements to predict pitch types before release, a phenomenon known as *pitch-tipping*. By combining pose estimation, time-series modeling, and explainable ML techniques such as t-SNE and UMAP, this study provides both predictive models and interpretability tools to aid hitters in exploiting tells and helping pitchers eliminate them.**

***Index Terms* – Pose-Estimation, Action Recognition, Recurrent Neural Networks, Baseball, UMAP**

I. INTRODUCTION

The game of baseball has not fundamentally changed over the last century. America's pastime is still played on the same field dimensions, under the same general rule constraints, with the same equipment down to the very mud from a Delaware River tributary that the baseballs are rubbed with before entering circulation. Upon the adoption of baseball statistical analysis however, the game's strategies and teachings have revolutionized. Given baseball's discrete and controlled action space, the game is a prime environment for the benefit of machine learning insights upon which this project is based.

Every unique event that occurs during a game is catalyzed by a duel between the pitcher and the hitter. A pitcher's success is defined by their ability to withhold the hitter from reaching a base, and the hitter's is the reciprocal. The inner mechanizations of this duel are an intricate dance of timing and predictive strategies by both individuals. This generalized overview contextualizes pitch-tipping, the phenomenon when a pitcher's subtle repetitive movements prior to the ball's release 'tip-off' the hitter as to what type of pitch they are about to be thrown. Exposing pitch-tipping patterns are beneficial to the hitter and detrimental to the pitcher.

A. Problem Statement:

In pitch-tipping identification, the opponent carries out a methodology similar to that of a machine learning classifier with relative body part positions as an input and the pitch thrown as the output. This project serves as a study into the application of computer vision human-pose estimation models to accurately predict pitches with an ultimate objective to offer classification explainability and generate actionable insights from the predictions.

B. Results Overview:

While the neural network architectures explored were able to predict pitch-types with significant performance, there was no actionable difference between individual pitcher's models performances indicating one pitcher's tendencies to tip their pitches more than the others who were sampled in the study. Furthermore, and perhaps with correlation indicated by the lack of model differential, the explainability methods did not provide conclusive insights that would benefit a hitter or pitcher in real time. These outcomes may be indicative of the performance of the methods used in this project, the pitchers' abilities to maintain consistent movement patterns, or a combination of the two.

II. RELATED WORK

Upon the advent of accurate image classifier architectures such as ResNet and GoogLeNet, action detection through multi-frame analysis has been thoroughly examined. Karen Simonyan and Andrew Zisserman, the researchers behind the VGG convolutional neural network (CNN) architecture [6], proposed a two-stream architecture [5] that captures appearance information from video data frames and optical flow frames (displacements between frames) separately before fusing the streams together to make a final action prediction. This methodology, inspired by the human visual system, had one of the best performances on benchmarks such as UCF-101 and HMDB-51 at the time of its publishing in 2014. While this study paved the way for future work, the architecture incurs immense computational cost between the dataset size and quantity of operations.

Yu, Chen, et. al [1] proposed a very similar data structure as the structure used in this study with 3D dataframes of landmark columns and frame rows. However, they chose to focus on 2D CNN architectures as the main method of classification. Essentially, they maintained the image-based action classification method proposed by Simonyan and Zisserman, just with drastically reduced data requirements by using 25 landmarks instead of entire image pixel values. Using the UCF-101 behavioral recognition dataset, they were able to see conclusive classification results, especially in highlighting performance improvements as the density of frames used in a given frame range increased.

III. DATA PROCESSES

A. Dataset:

Major League Baseball's official database, MLB Film Room, served as the dataset for the project. MLB Film Room is an open-sourced video database of every baseball play in every Major League Baseball game as it was broadcast since 2017. All videos are intricately organized and labeled, specifically, down to the pitcher, the stadium, and the type of pitch the pitcher throws in the video. This level of supervision enabled me to gather video samples filmed in a consistent environment of a specific individual throwing specific pitches in bulk. Post-processing, approximately 800-900 video samples per pitcher with 100-300 samples per pitch-type were collected.

B. Data Mining:

Landmark positions were extracted from each video sample using Google's MediaPipe [3] computer-vision human-pose estimation library for further modeling and analysis. MediaPipe uses a joint CNN architecture to first, detect the location of a subject in an image, and second, use a pretrained model to locate the pixel location of its available landmarks. Running MediaPipe over a video sample outputs 3D coordinate values and a confidence score of 33 landmarks per video frame. MediaPipe was run in parallel over all video samples using its highest quality setting to create a metadata set comprised of a landmark coordinate dataframe, video ID, video index, and pitch label.

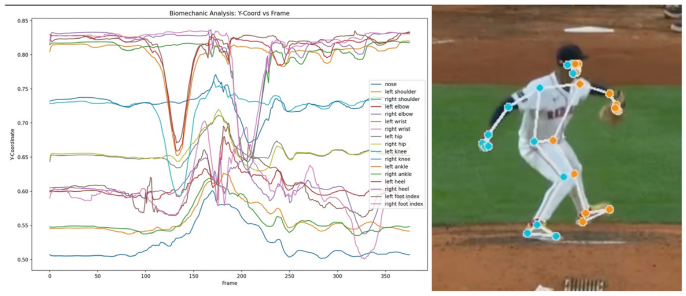


Figure 1: Landmark Plot with Overlaid Visualization

C. Preprocessing:

The following data preprocessing steps were taken to clean and prepare the data for further analysis:

- Unfit video samples were discarded based on a first-frame similarity score test with a user provided quality video sample.
- Z-coordinates were discarded due to my lack of confidence in 3D pose-estimation accuracy on the 2D video samples.
- Common landmarks were consolidated per frame for feature reduction and data smoothing purposes (i.e. left ankle, left foot index, left heel consolidated to left foot).
- Landmark coordinate values were scaled from pixel values to inches using the pitcher's height as reference.
- Landmark coordinate values were referenced to the pitcher's throwing-arm-side's foot's first frame coordinates as the origin.
- Data was cropped to a consistent range referencing from a consistent landmark value in every video, thus consistently capturing the window in which pitch-tipping could occur. The consistent landmark value chosen was the maximum y-value of the pitcher's throwing-arm-side's foot which rises at the same time relative to the throw release across every video. 100 frames were chosen prior to this reference point up to the arbitrary point where pitch-tipping was no longer applicable to the hitter.
- Every fourth frame of the 100 total was used to consolidate into 25 total frames and smooth the data.

IV. MODEL ARCHITECTURES

A. Logistic Regression:

Three logistic regression models were generated per pitcher, each using a different loss function:

Loss	Formula
L1/Lasso	$\frac{1}{2n} * \|y - Xw\|_2^2 + \alpha * \|w\|_1$
L2/Ridge	$\frac{1}{2n} * \|y - Xw\|_2^2 + \alpha * \|w\|_2^2$
Elastic-Net	$\frac{1}{2n} * \|y - Xw\|_2^2 + \alpha * \ell 1ratio * \|w\|_1 + 0.5 * \alpha * (1 - \ell 1ratio) * \|w\|_2^2$

* n = number of samples, y = target value, X = training data, w = coefficient vector, α = regularization term, $\ell 1ratio$ = mixing parameter of $\ell 1$ and $\ell 2$ losses

*Figure 2: Elastic-Net Loss, Hybrid L1 and L2 Loss**

B. Support Vector Machines (SVM):

Three SVM's were generated per pitcher, each using a different kernel function:

Kernel	Formula
Linear**	$K(x,z) = (x^T z)^n$
Polynomial***	$K(x,z) = (x^T z + c)^n$
Radial Basis***	$K(x,z) = e^{-\|x-z\|_2^2 / \sigma}$

* x = input data, z = transformation, σ = std. dev.
** Monomials of deg. n
*** Monomials of deg ≤ n

*Figure 3: SVM Model Kernels**

C. Deep Neural Networks:

Three deep neural network (DNN) architectures were explored per pitcher. A recurrent neural network (RNN), a long-short term memory neural network (LSTM), and a CNN. Due to the potentially unique model requirements of every pitcher in the study, I deployed a genetic algorithm [2] to train a variety of many-to-one RNN and LSTM models and output the objectively best fit model for that specific player's data. The objective fitness score function, as shown in *Equation 1*, incentivizes high test set accuracy while penalizing for an imbalance in train set accuracy to test set accuracy, an indicator of model overfitting.

Hyperparameter	Values
Learning Rate	0.0001, 0.0005, 0.001, 0.005, 0.01
Hidden Layer Depth	1, 2, 3, 4, 5
Hidden Layer Size	16, 32, 64, 128, 256
Training Epochs	5-50
Architecture Type	RNN or LSTM
Number Generations	5
Population Size	30
Mutation Rate	0.2
Crossover Rate	0.7
Test Set Ratio	0.2

Figure 4: Genetic Algorithm Encoding

$$\text{Fitness} = \text{test \%} - 0.25 * \max(0, \text{train \%} - \text{test \%})$$

Equation 1: Fitness Score Formula

The CNN architecture, as shown in *Figure 5* below, consists of three layers comprised of a convolution, a batch normalization, and a max pooling. Between the final two fully connected encoded layers is a dropout layer to mitigate model overfitting.

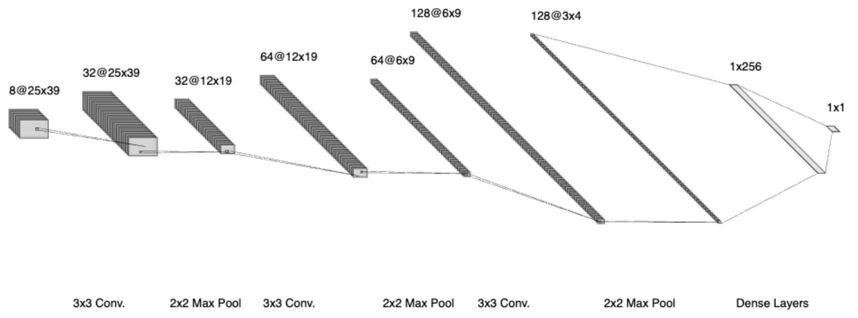


Figure 5: CNN Architecture

V. EXPLAINABILITY METHODS

A. Landmark Trajectory Plots:

Plots detail the landmark trajectories per video sample and averaged over all samples per pitch-type per pitcher. Looking for differences in the average trajectories per landmark per pitch type may uncover landmarks and frame ranges of interest. Further investigation across specific video samples may provide further enlightenments.

B. t-SNE and UMAP:

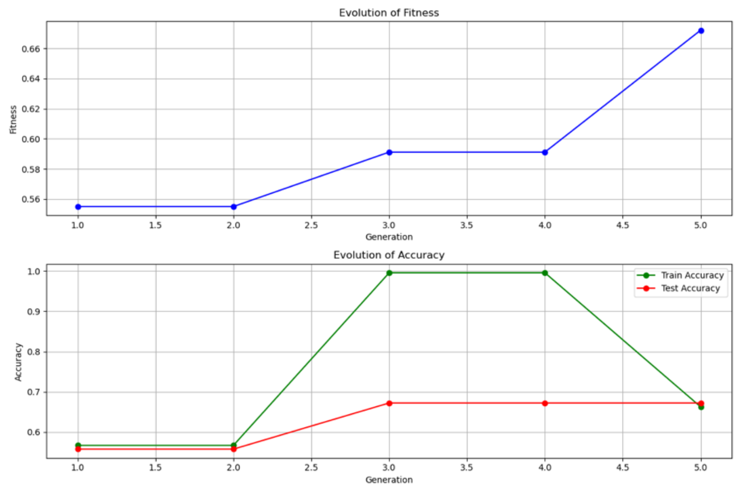
t-Distributed Stochastic Neighbor Embedding (t-SNE) [7] and Uniform Manifold Approximation and Projection (UMAP) [4] are dimensionality reduction methods that preserve the local relationships between data points in higher dimensions, and recreate those relationships in lower, visualizable dimensions. These methods are useful unsupervised machine learning techniques that enable you to visualize clusters and other patterns within your high-dimensional data. In a supervised environment, they are useful to see what features may separate the data's classifications.

t-SNE assesses the high dimensional relative location of datapoints to one another by grouping data points that would most likely be generated together under the same gaussian probability distribution function (PDF). UMAP assesses the high dimensional relative location of datapoints to one another by constructing a high dimensional graphical representation of the data. Both relationship structures are recreated in the visualizable dimension.

VI. RESULTS AND DISCUSSION

A. Modeling Results and Discussion:

Figure 4, below, shows the performances of the top RNN or LSTM models from the genetic algorithm output which was consistently the best performing model out of all the models explored. The player model accuracies do not reveal one individual pitch-tipping significantly more than another. However, most models show relative success in classifying pitches with the pose-estimation data. The other figures in this section show Nestor Cortes's model performances. *Figure 5* shows the evolution of the model outputs in a sample genetic algorithm process.



Generation	Model	Epochs	Learn Rate	Hidden Size	Depth
1	LSTM	18	0.001	16	5
2	LSTM	18	0.001	16	5
3	LSTM	27	0.01	64	2
4	LSTM	27	0.01	64	2
5	RNN	18	0.0005	32	2

Figure 5: Genetic Algorithm Progression

Figure 6, below, shows logistic regression and SVM model performances. These models either indicate poorer classification performance than the RNN/LSTM models or overfitting as indicated by the high train set accuracy relative to the test set accuracy.

Figure 7, below, shows Nestor Cortes's CNN model's training loss and train/test set accuracies per epoch. Most models show a similar pattern of rapid divergence in the train and test set losses and performances.

Overall, the models performed as expected. Given the complex and temporally dependent nature of the biomechanical datasets, the expectation was that RNN's and LSTM's would greatly outperform all of the logistic regression and SVM models. The majority of the models output by the genetic algorithm were RNN's. This is reasonable since LSTM's effectiveness is in relatively long-term memory recall. The benefits LSTM architectures bring in mitigating exploding or vanishing gradients during backpropagation may not outweigh the shorter-term effectiveness of the RNN architecture over the 25 frames of landmark data used per pitch.

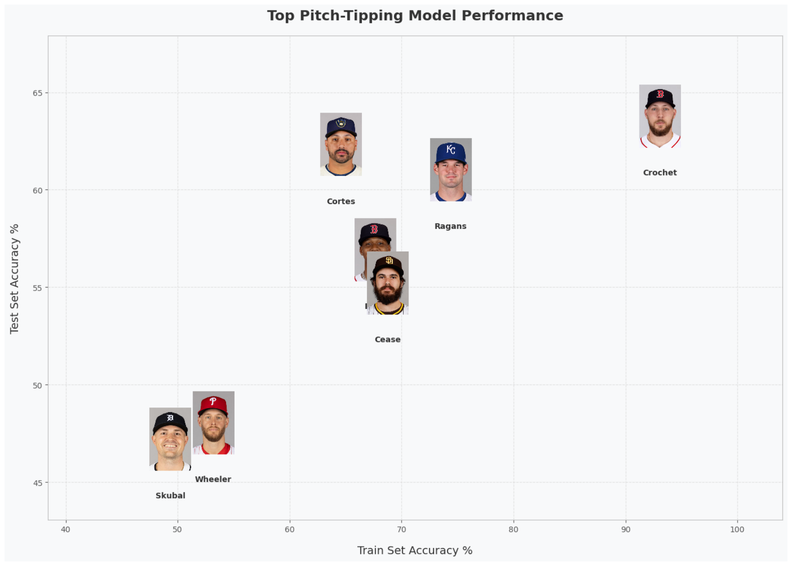
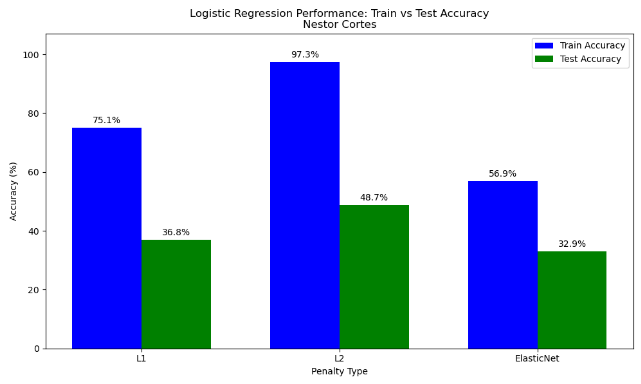
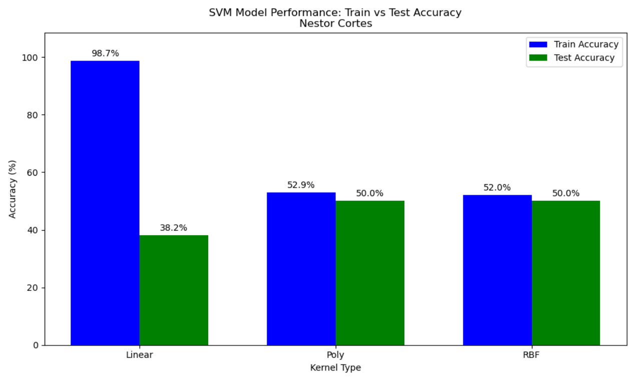


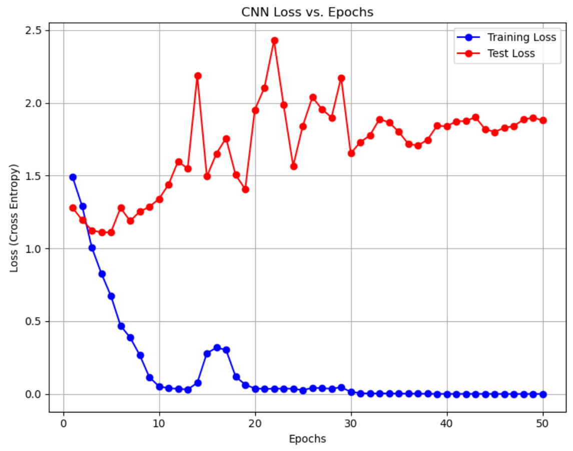
Figure 4: RNN/LSTM Model Performance per Player

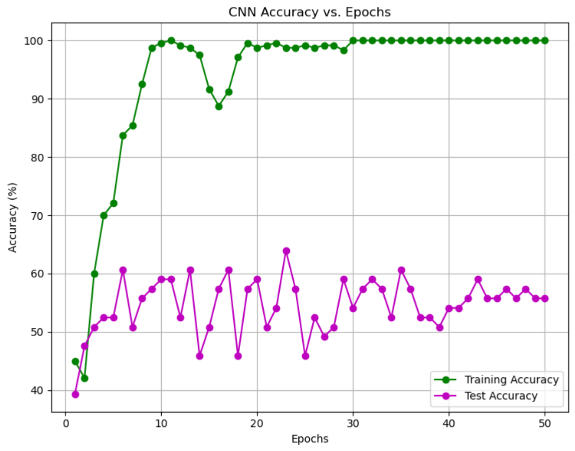




a. Logistic Regression *b. SVM*

Figure 6: Nestor Cortes Logistic Regression/SVM Model Performances





a. Loss vs Epoch *b. Accuracy vs Epoch*

Figure 7: Nestor Cortes's CNN Loss and Accuracy per Epoch

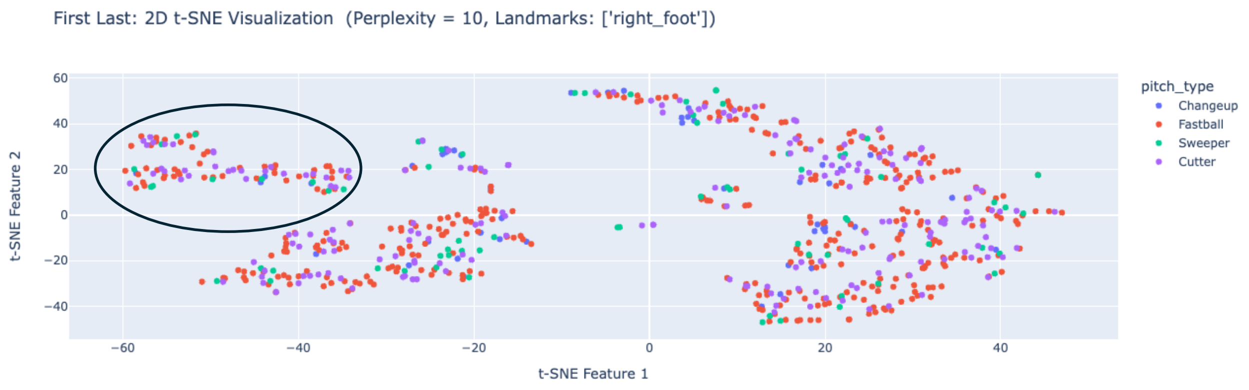


Figure 8: Garrett Crochet's Right Foot t-SNE Representations

Logistic regression classifiers do not handle outlier or misclassified data well due to the rigidity of the loss functions. Baseball pitchers usually have multiple, consistent forms of pre-throw techniques called windups. These different windups would result in multiple clusters of all pitch types which would not be compatible with logistic regression models of any kernel or loss function. Refer to *Figure 8* where the t-SNE plot clearly displays this phenomenon.

SVMs, serving as a single-neuron, shallow neural network, had higher expectations than logistic regression especially with kernels applied. As a single-neuron, shallow neural network, the linear kernel would not have enough depth to create significantly separable data as the overfit train/test set ratios signify. The polynomial and RBF kernels do show improvement in both test accuracy and the overfitting issue. Ultimately, given the temporal complexity of the data and potential outliers due to data quality issues from MediaPipe's limitations, a robust DNN would better suit the data.

The CNN architecture did perform better than the logistic regression and SVM models overall. In Nestor Cortes's example shown, the train-set and test-set accuracies reach roughly 85% and 62% respectively after 7 training epochs. This level of accuracy imbalance does not necessarily indicate overfitting but does indicate the model may not be as representative as a model with a smaller differential and similar test-set accuracies like the RNN and LSTM models. While a more tailored CNN architecture may have offered better results, the effectiveness of the simple RNN/LSTM models redirected my focuses towards them.

B. Explainability Results and Discussion:

Ultimately, the explainability methods used were less explanatory as they were exploratory. Of the pitchers observed in the study, there was not an actionable case of pitch-tipping exposed. *Figure 8* shows the t-SNE representation of Garrett Crochet. He had the highest test accuracy among the samples in *Figure 4* at 64%. Using all his landmarks, there is not a clear distinction between the pitches thrown. Recall, the general clusters are likely resulting from different types of windups. However, focusing on landmark 'right_foot', small clustering distinctions can be made between the color-coded pitches. For instance, the cluster circled appears to have significantly more Cutter and Fastball datapoints than Sweeper and Changeup datapoints. This observation is a bit of a reach, but this is the theoretical process that can be taken.

With this insight, we can evaluate the 'right_foot' xy-coordinate motion over the course of the 100 frames selected just before the pitch is released using the average motion per pitch type plots shown in *Figure 9*. Around frame 80, you can see a discernable difference in the y-coordinates of the fastball and cutter. This specific landmark positioning at that explicit range of frames can potentially be translated to actionable insights for a hitter to look out for or for a pitcher to fix.

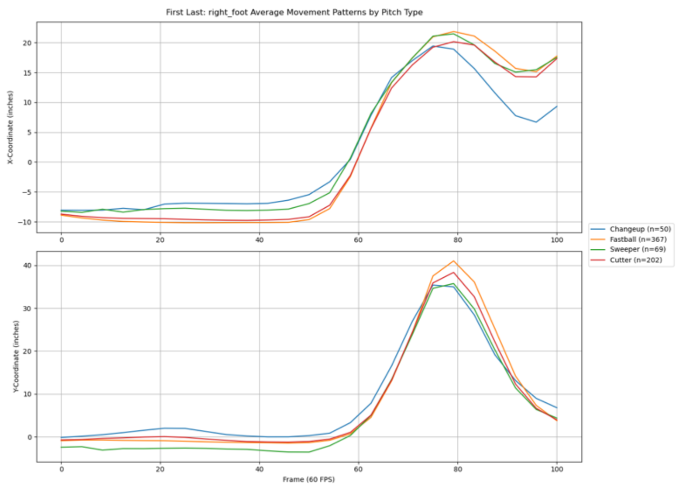


Figure 9: Garrett Crochet's Right Foot Representations

VII. CONCLUSION AND FUTURE WORK

Given the relative success of the models to classify pitch types based solely on 25 frames of information acquired from an “over-the-shelf” human pose-estimator using video samples derived from a largely uncontrolled environment, I am led to believe that the methods used in this study have very practical applications after further investigation and refinement.

I would have liked to process more player’s data. I am confident that there would have been more conclusive results if I were able to extend these approaches to the 416 pitchers on active MLB rosters. Professional baseball pitchers, naturally, are typically pretty good at not pitch-tipping. The small sample size I chose, especially being comprised of the more elite pitchers in the league during the 2024 season, is not representative.

Regarding the modeling process, I would like to train all the models with a binomial classifier architecture and observe if the actionability of the results improve. Instead of a multiclass classifier to predict each individual pitch, a binomial classifier classifying ‘fast’ and ‘off-speed’ pitches may be more successful and provide similarly actionable insights. Timing is the most significant insight a hitter can have.

As previously stated, the explainability methods were closer to exploratory methods since they were not directly tied to the models’ training processes. I was not able to successfully integrate a SHAP analysis to my models like I intended to. The SHAP library provides insights as to how heavily different features were weighted and ultimately contributed to the model’s predictions. In theory, a SHAP analysis could highlight the exact features and the exact frames under which those features most contributed to the model’s output predictions.

References

[1] Action Recognition Algorithm Based on 2D Human Pose Estimation Method. (n.d.). *International Journal of Computer Applications*, 182(24), 1–6. https://doi.org/10.5120/ijca2018918214

[2] Jiwani, A., Feldman, J., Garapati, V. K. R., Vennelakanti, A., Hogale, V., & Zou, J. (n.d.). *Music reconstruction using genetic algorithms*. Khoury College of Computer Sciences, Northeastern University.

[3] Lugaresi, C., Tang, J., Nash, H., McGuire, M., Chang, C., Yong, M. G., … & Pfister, T. (2019). *MediaPipe: A framework for building perception pipelines*. arXiv preprint arXiv:1906.08172. https://doi.org/10.48550/arXiv.1906.08172

[4] McInnes, L., Healy, J., & Melville, J. (2018). *UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction*. arXiv preprint arXiv:1802.03426. https://doi.org/10.48550/arXiv.1802.03426

[5] Simonyan, K., & Zisserman, A. (2014). *Two-stream convolutional networks for action recognition in videos*. In *Advances in Neural Information Processing Systems* (pp. 568–576). https://arxiv.org/abs/1406.2199

[6] Simonyan, K., & Zisserman, A. (2015). *Very deep convolutional networks for large-scale image recognition*. In *International Conference on Learning Representations (ICLR)*. https://arxiv.org/abs/1409.1556

[7] van der Maaten, L., & Hinton, G. (2008). *Visualizing data using t-SNE*. *Journal of Machine Learning Research*, 9(Nov), 2579–2605.

[8] Yadav, S., & Vishwakarma, D. K. (2020). *Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network*. In *A Deep Learning Approach for Disease Prediction and Prognosis* (pp. 85–104). Springer. https://doi.org/10.1007/978-3-030-53394-9_5

VIII. ADDITIONAL DELIVERABLES

GitHub Repository:
https://github.com/jfeldm02/Pitch-Tipping-Detection