CS7641-group-59.github.io

Predicting Tommy John Injuries Using Pitcher Statistics

CS 7641, Group 59, Fall 2024

Will Ferguson, Taeyun Lee, Paul Hutchison, Sterling Kalogeras, Chan Woo Moon

Contents

Introduction/Background

Summary

This project will attempt to accurately predict the risk of future UCL surgeries (“Tommy John”) in Major League Baseball pitchers using an ensemble of Machine Learning Methods.

Literature Review

Machine Learning has been applied to baseball injury prevention in a variety of ways in recent years. Notably, the American Journal of Sports Medecine recently conducted a case study to determine whether “pitchers who underwent UCLR would have a higher preinjury peak…pitch velocity” using a variety of ML techniques [1].

ML is also an effective indicator of non-pitching injuries. One such study indicated that “advanced ML models outperformed logistic regression in 13 of 14” sampled cases [2].

Similar studies have also been conducted on pre-professional and amateur athletes. One 2021 study focused on variables that influenced “elbow valgus torque and shoulder distraction force using a statistical model and a machine learning approach” [3].

Overall, machine learning is a well-supported and well-tested method of testing injury risk in baseball pitchers.

Dataset Description, with Links

This project will primarily use a compiliation of pitcher statistics from a Kaggle Dataset (linked below), as well as an open-source database of disabled list (DL) entrants from 2000-2016 (will act as our labels dataset). These can be accessed here:

Kagge Dataset (Primary)

Pitcher Stats, Baseball Prospectus

There’s No Crying in Baseball, by Robot Allie (GitHub) - Features

Definition and Motivation

Problem

UCL (Tommy John) surgery is one of baseball’s most common and severe injuries. However, it is historically nearly impossible to predict what pitchers are suspceptible to this injury, and when, as pitchers of all ages and abilities are susceptible [4].

Motivation

MLB franchises are highly motivated to identify risk factors for TJ surgery for two reasons. TJ surgery carries significant competitive implications, as patients are completely out of pitching for at minimum a year. Additionally, it is incredibly costly; the average surgery incurs a $1.9 million bill [5], while some injuries costs teams years of contracts that reach the hundreds of millions of dollars.

Data Preprocessing

Standardizing the data points, since some features such as average pitch velocity and pitch count could be skewed significantly based off other features such as time playing.

Next we will do feature engineering to create some derivative features that may be more indicative than the original ones giving better insights than existing models.

Cleaning the data highly linearly dependence to avoid numerical instability in our model. We will either use SVD or LU decomposition to identify it.

Data Cleaning

The data that we had collected over many seasons that needed to be standardized and merged into a consistent dataset. This was done in `project.ipynb` where there were stats for percentage of pitch type thrown and average velocity.

Feature Engineering

The first method that we utilized was creating polynomial combinations of features. This creates complex features that can help to capture nonlinear trends in the data. It also provides higher dimensionality in the dataset which can help with identifying trends since the raw data does not have very many features. We use this data as the source when running the models.

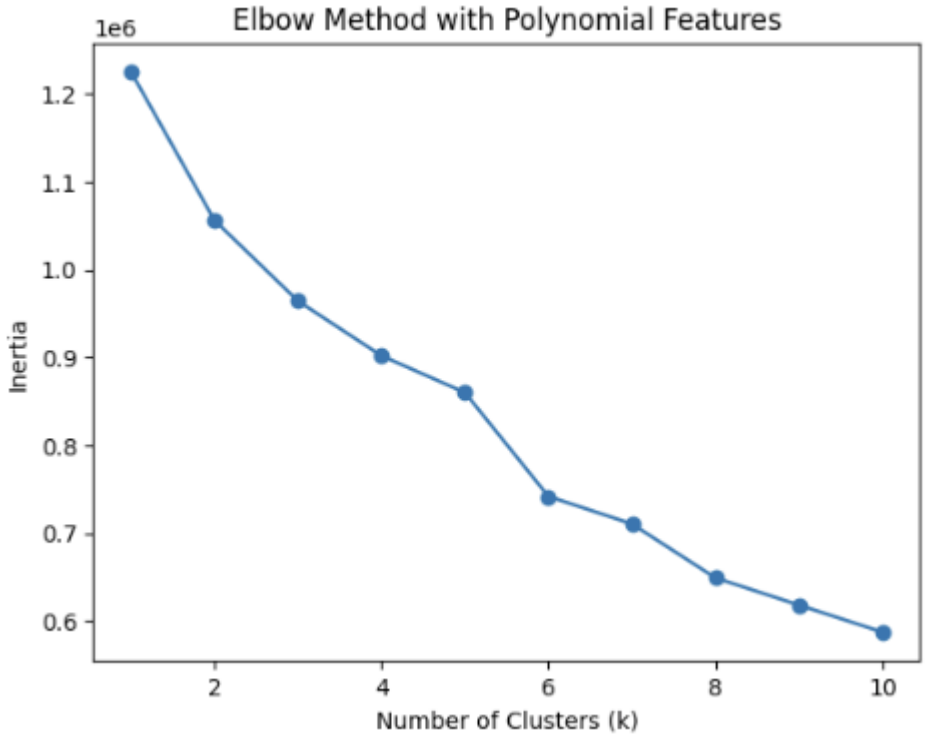
Unsupervised Models

K-Means

Clustering the pitchers using different narrower subsets of features could help to categorize the pitchers into groups that would give more insight into different pitcher groups.

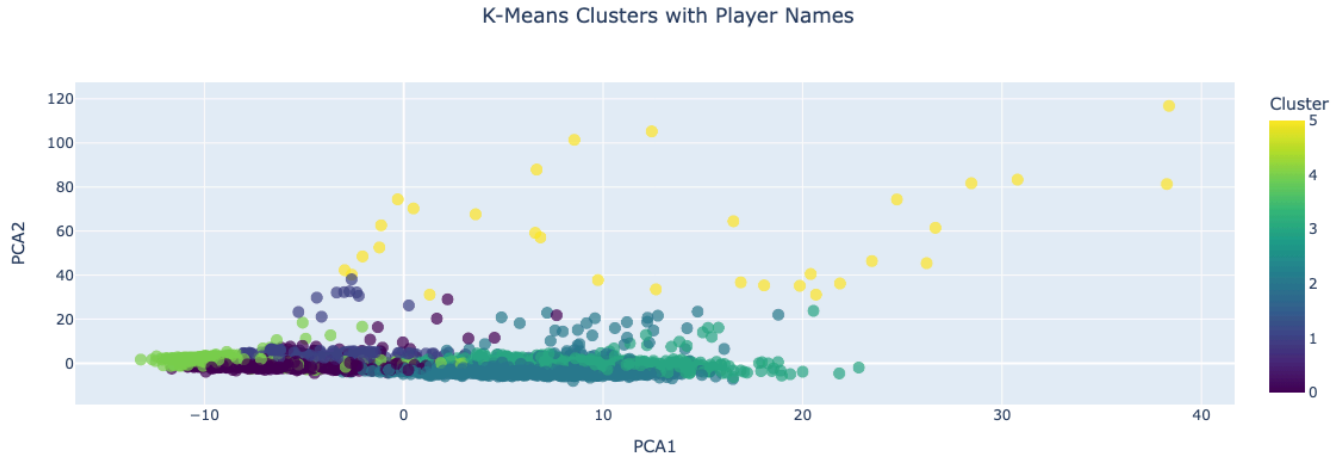
We have applied K-means clustering to categorize pitchers into distinct groupbs based on the stats that we have obtained. This would then identify patterns and subgroups among pitchers to reveal risk factors or commonalites amongst those susceptible to Tommy John injuries.

The implementation can be seen in (./K-means.ipynb), but the general steps were undergoing some additional data cleaninig such as averaging statistics with multiple entries and then some additional feature engineering.



Finally we tested different values of k and decided that 6 was optimal using the elbow method judging from the photo above.

Analysis, Visualization, Metrics:



From the plot we are able to identify the groups of players into 5 different group and locate each player on which group that they fit it. In addition those who have a bigger group number are those who are more succeptible to the injury. The visualization is coherent as players who have the injuries have some correlation yet they are scattered around as we cannot fully identify the reason behind the injury.

The problem involves applying a machine learning model to predict player injuries in a dataset where the majority of players are not injured. Due to this imbalance, the model overfits to the dominant “Not Injured” class. | Class | Precision | Recall | F1-Score | Support | |————–|————–|————–|————–|————–| | **Not Injured** | 0.88 | 1.00 | 0.93 | 2,956 | | **Injured** | 0.00 | 0.00 | 0.00 | 419 | | **Accuracy** | | | 0.88 | 3,375 |

Here, the model achieved a high accuracy score of 88%, but this metric is incorrect because the model failed to identify any injured players. The metrics for the “Injured” class—precision, recall, and F1-score—are all zero. This happens because the model assigns almost all data points to the “Not Injured” cluster, which is a reflection of its bias towards the majority class.

Addressing the Imbalance with SMOTE To address this issue, the Synthetic Minority Oversampling Technique (SMOTE) was applied. SMOTE creates synthetic samples for the minority class (“Injured”) to balance the dataset. Additionally, overlapping rows between the training and test sets were removed to prevent data leakage. When overlapping rows weren’t removed the data had 1.00 in all categories.

Below is the classification report summarizing the model’s performance on the cleaned test set:

Class	Precision	Recall	F1-Score	Support
Not Injured	0.90	0.85	0.88	589
Injured	0.25	0.35	0.29	83
Accuracy			0.79	672

While the initial model struggled due to the imbalance in the dataset, applying SMOTE and cleaning the data allowed the K-Means clustering model to better capture the characteristics of injured players. Although the overall accuracy decreased, the balanced approach ensures that the model is no longer biased towards the dominant “Not Injured” class and can provide meaningful predictions for the minority class.

Next Steps

We suggest a number of future actions to alleviate the difficulties in categorizing minority situations (damaged pitchers). In order to guarantee that the generated synthetic samples accurately represent the minority class without adding noise, we will first optimize the SMOTE settings, namely the number of nearest neighbors (k_neighbors). To improve sample creation and eliminate noisy data points, more sophisticated methods as Borderline-SMOTE or SMOTE-ENN will also be investigated.

Second, in order to improve predictions, we want to integrate SMOTE with ensemble techniques like Random Forest, taking use of the variety in synthetic examples. To lessen the effect of imbalance during model training.

Supervised Models

Random Forest

Overview:

At the beginning of this project, we hoped to use this algorithm to identifying the features that are most important to our analysis, and hoped its robust ensemble method would also show promise as a predictor.

Our preliminary implementation focused on testing the random forest's capability only on our features which were complete across our entire dataset. This led us to create a random forest that classified injury risk based on the following indicators:

- IP: Innings pitched by the pitcher in the sampled season
  - Pit: Pitches thrown by the pitcher in the sampled season
  - Release Angle Statistics:
    - release_extension_std
    - release_pos_x_std
    - release_pos_z_std
  - Max Velocity: Max pitch speed, in MPH

The random forest was made up of 100 10-layer trees with a balanced class weighting strategy. Balanced class weighting is a technique that overemphasizes the weights of the minority class at training time, which can potentially allow for better identification of a minority class that occurs very rarely (useful in our case as only about 10% of sampled pitchers in our initial dataset had Tommy John surgery). The model was trained on a 70-30 train-test split ratio and implemented using `sklearn`. The source code for the implementation can be viewed here.

Analysis, Visualization, Metrics:

Our initial training run performed poorly, as it was generally unable to identify cases in the minority class (pitchers who suffered a Tommy John injury). As seen in the confusion matrix below and the score report table, the model can claim high accuracy metrics, but is nonetheless generally useless in its current configuration. Even its most important feature rankings (also shown below), carry some immediate reliability concerns.

	precision	recall	f1-score	support
0 (No injury)	0.93	0.99	0.96	248
1 (Injury)	0.00	0.00	0.00	18
accuracy			0.92	266
macro avg	0.47	0.50	0.48	266
weighted avg	0.87	0.92	0.90	266

Table 1: Initial Metrics for Random Forest

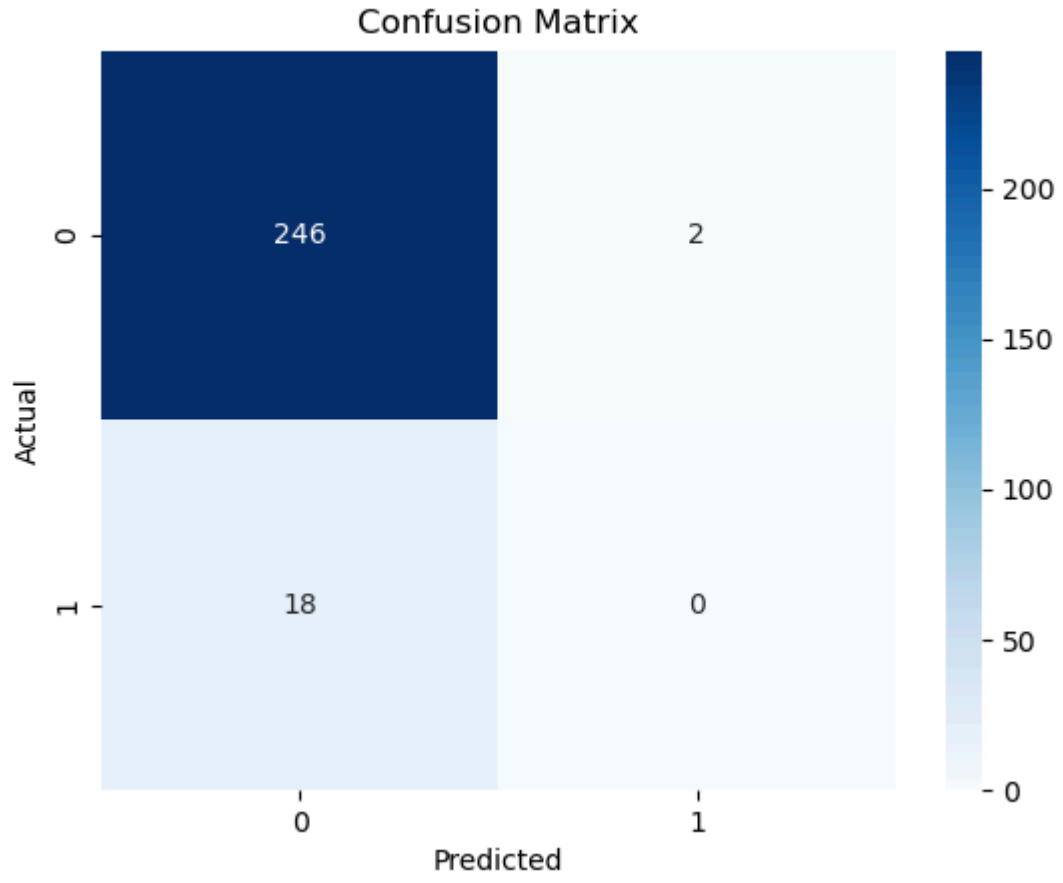


Figure 1: Confusion Matrix for Initial Random Forest Implementation

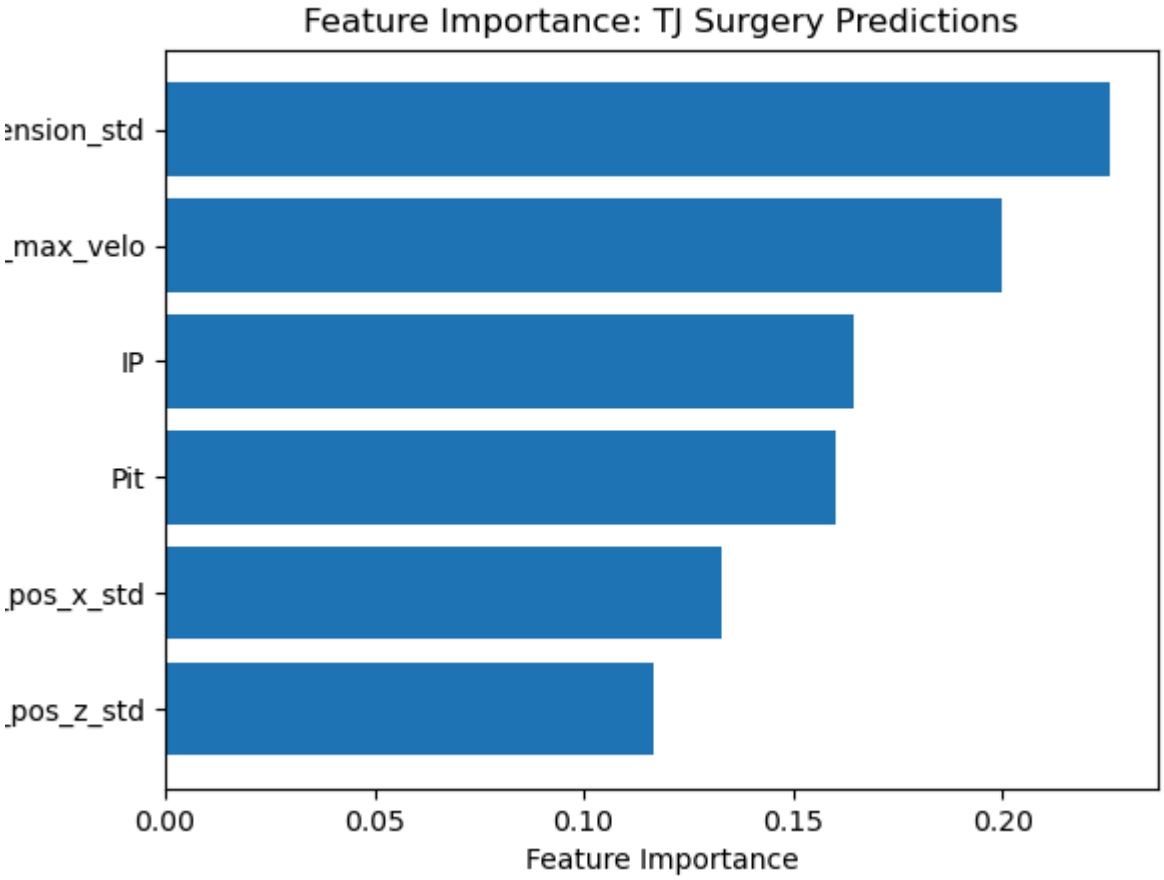


Figure 2: Most Important Features

Analysis, Cont'd: An Improvement using SMOTE:

The clear problem with this early implementation was that, given the prohibitive size of the minority class, the model can reach very high accuracy numbers by predicting the majority class with near-perfect accuracy, but failing on the much more important minority case. Therefore, a model that is better at identifying true positives in the minority case would be far more useful, even at a small accuracy tradeoff.

Therefore, the first step to improving these results is clearly synthetic data generating. This preprocessing method will give us a reliable way to control the ratio of majority-minority cases, hopefully making our ensemble more informed. To do this, we leveraged the `imbalanced-learn` library (a subset of `sklearn`) to employ a Synthetic Minority Over-sampling Technique (SMOTE) to restructure our dataset.

This produced some immediate improvements, as shown in the figures and table below. Our second training/testing pass was done with the same random forest hyperparameters, but this time with a 70-30 majority-minority class ratio, and a 5-nearest-neighbors interpolation method. While this resulted in a small accuracy tradeoff, we immediately see improvements in our ability to correctly identify Tommy John injuries. This data is shown below:

	precision	recall	f1-score	support
0	0.94	0.90	0.92	194
1	0.63	0.73	0.67	44
accuracy			0.87	238
macro avg	0.78	0.81	0.80	238
weighted avg	0.88	0.87	0.87	238

Table 2: SMOTE Metrics

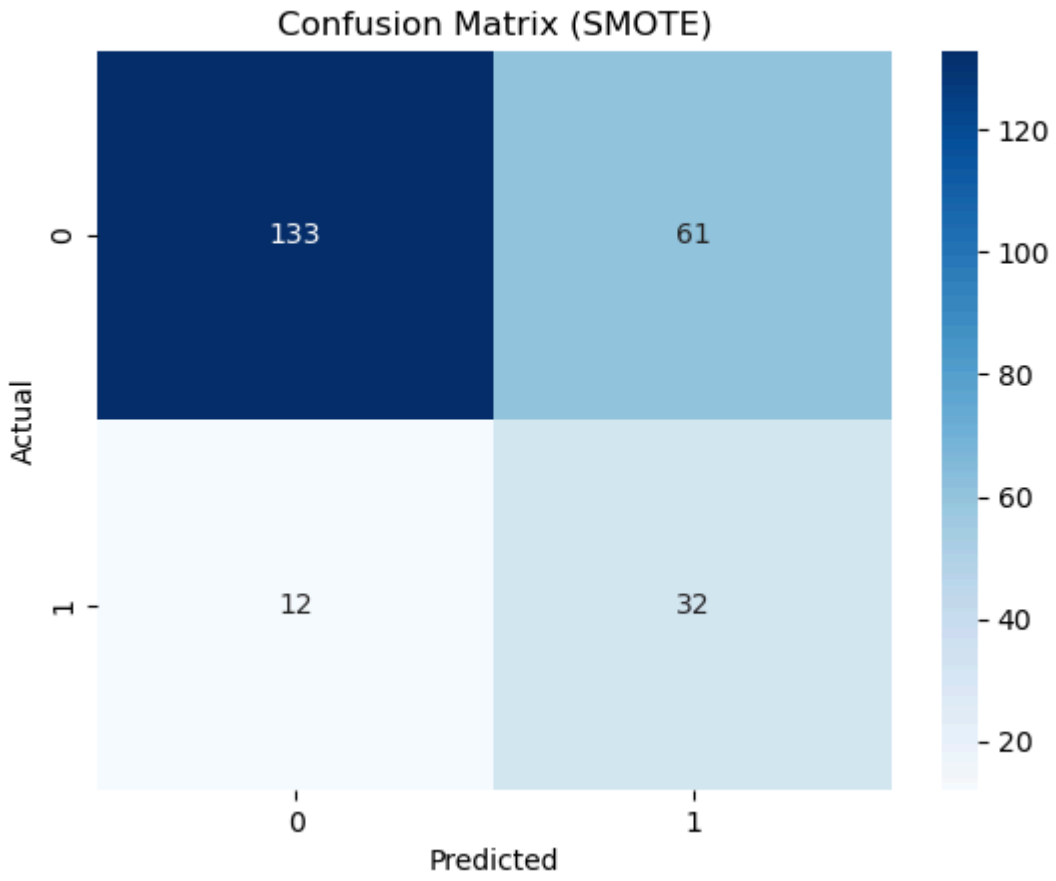


Figure 3: Confusion Matrix, SMOTE

Another benefit of this approach is we can now inspect the most important features, with slightly more confidence in their reliability.

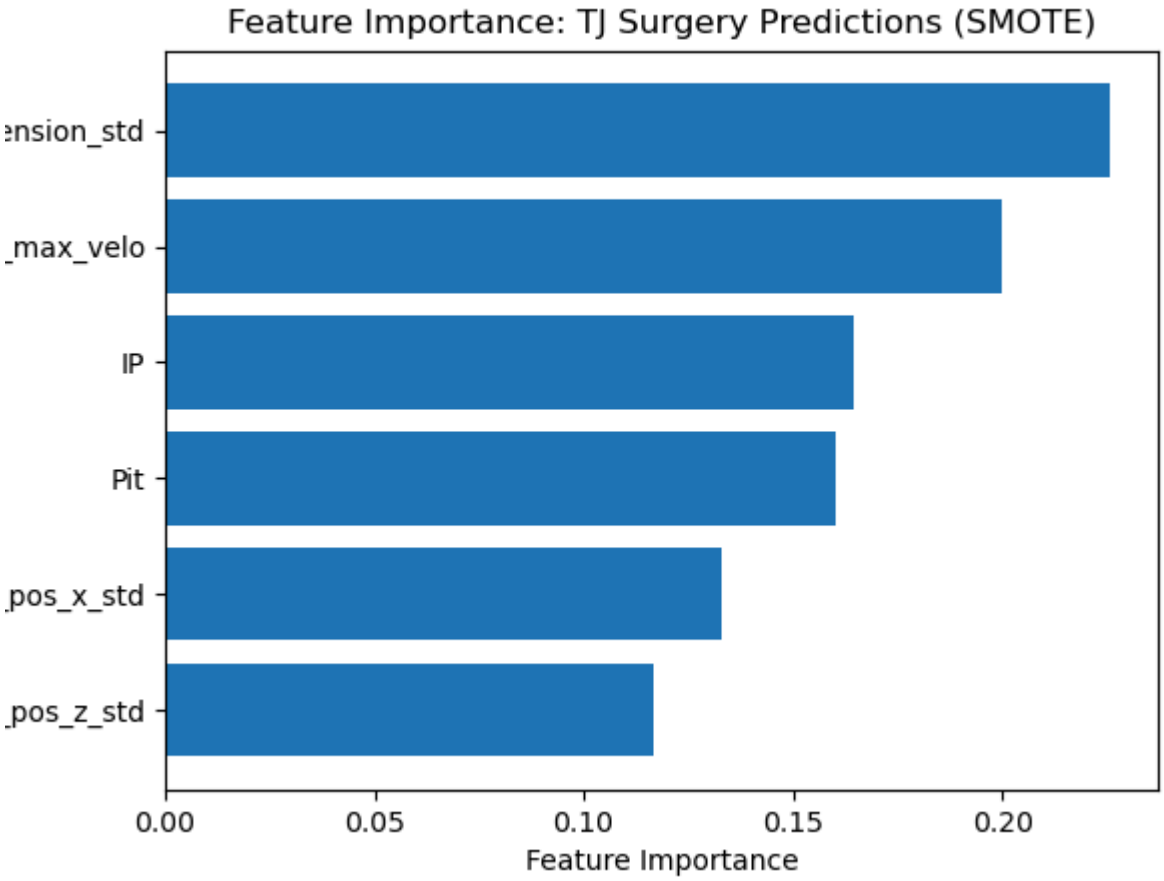


Figure 4: Most Important Features, SMOTE

Next Steps

Our initial progress on random forests introduces several areas of potential improvement and research. Obviously, the forest hyperparameters (as well as the synthetic data hyperparameters) need to be tuned, with the ultimate end goal of minimizing the overall misclassification rate, especially in the injury case.

These results also point to a continual data need. The next steps of our project will require the use of several more data collection and preprocessing techniques beyond SMOTE. Our main dataset currently contains just under 900 entires, and there is certainly more historical data available for us to continue to augment our training data. Ultimately, we hope to raise the ratio of non-synthetic injury cases above the 10% it currently sits at. This will improve the integrity of our original dataset, and potentially increase the ability for SMOTE to optimize our results.

SVM

Overview:

We chose to use SVM due to its simplicity in being able to separate data via a hyperplane for the purpose of classification and for its ability to make use of different kernels to be able to identify the data.

Our preliminary implementation focused on testing the random forest's capability only on our features which were complete across our entire dataset. This led us to create a SVM model trying to find the decision line between injured and non-injured players:

- IP: Innings pitched by the pitcher in the sampled season
  - Pit: Pitches thrown by the pitcher in the sampled season
  - Release Angle Statistics:
    - release_extension_std
    - release_pos_x_std
    - release_pos_z_std
  - Max Velocity: Max pitch speed, in MPH

The SVM implementation was implemented with a polynomial kernel that introduces non-linearity since this injury appears to effect pitchers at varying points in their careers. Balanced class weighting is a technique that overemphasizes the weights of the minority class at training time, which can potentially allow for better identification of a minority class that occurs very rarely (useful in our case as only about 10% of sampled pitchers in our initial dataset had Tommy John surgery). The model was trained on a 70-30 train-test split ratio and implemented using `sklearn`. The source code for the implementation can be viewed here.

Analysis, Visualization, Metrics:

Our initial run yielded almost identical results as the first random forest implementation, as it was generally unable to identify cases in the minority class (pitchers who suffered a Tommy John injury). As seen in the confusion matrix below and the score report table, the model has high accuracy metrics, but the that is because the training data set is dominated by pitchers without the injury as is the case in the real world. This makes the model in its current state effectively useless making it as if you were to always guess that a player would not get or have had the injury.

	precision	recall	f1-score	support
0 (No injury)	0.92	1.00	0.96	183
1 (Injury)	0.00	0.00	0.00	17
accuracy			0.92	200
macro avg	0.46	0.50	0.48	200
weighted avg	0.84	0.92	0.87	200

Table 1: Initial Metrics for SVM with polynomial kernel

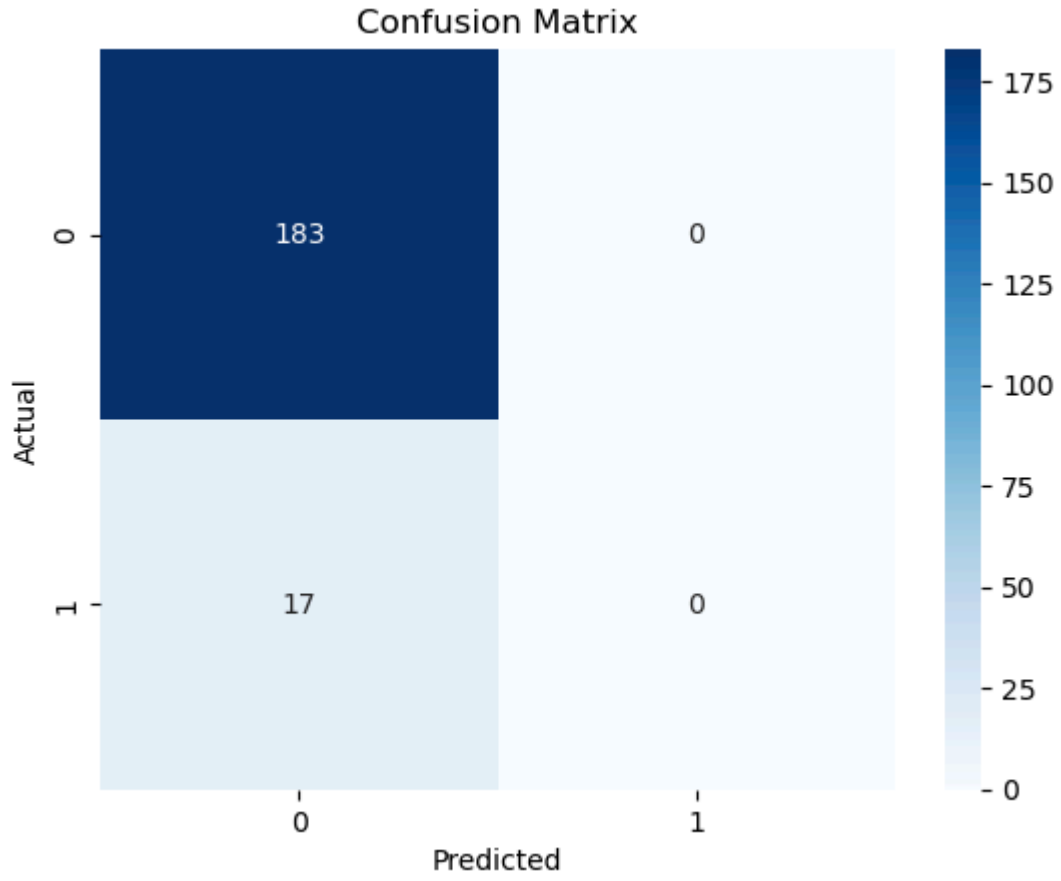


Figure 1: Confusion Matrix for Initial SVM Implementation with polynomial kernel

Other Kernels:

We see that with our chosen polynomial kernel that the amount of false negatives is quite high. We will implement some other kernel types to see if we can improve the identification of pitchers who have undergone the surgery. We will see that while this gives us some ability to predict an injury it is damaging to our predictions of when there is not one.

	precision	recall	f1-score	support
0 (No injury)	0.99	0.58	0.73	183
1 (Injury)	0.17	0.94	0.29	17
accuracy			0.61	200
macro avg	0.58	0.76	0.51	200
weighted avg	0.92	0.61	0.69	200

Table 2: Initial Metrics for SVM with linear kernel

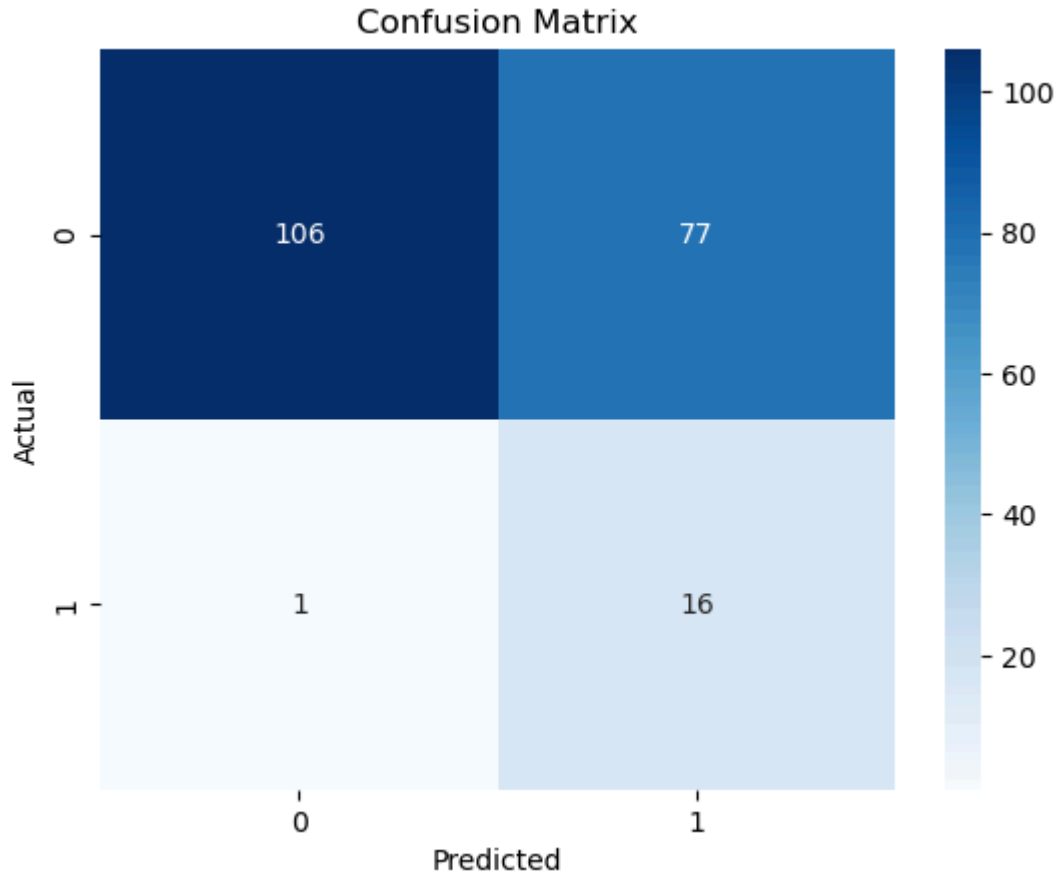


Figure 2: Confusion Matrix for SVM Implementation with linear kernel

| | precision | recall | f1-score | support | |————–|———–|———-|———–|———| | **0 (No injury)** | 0.97 | 0.38 | 0.55 | 183 | | **1 (Injury)** | 0.12 | 0.88 | 0.21 | 17 | | **accuracy** | | | 0.42 | 200 | | **macro avg**| 0.54 | 0.63 | 0.38 | 200 | | **weighted avg** | 0.90 | 0.42 | 0.52 | 200 | Table 3: Initial Metrics for SVM with rounded basis function kernel

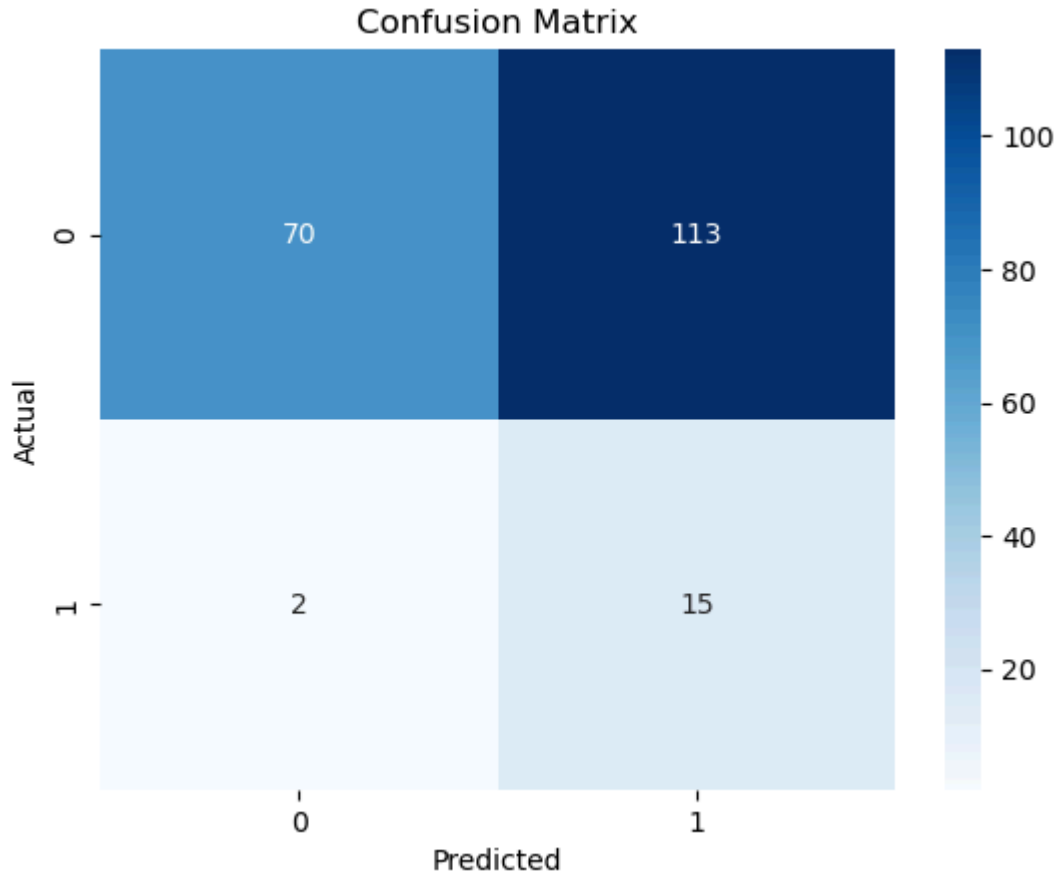


Figure 3: Confusion Matrix for SVM Implementation with rounded basis function kernel

From these two other kernels, we see the surprising fact that the linear kernel actually gives the highest accuracy in predicting players who have had the injury and improves the accuracy of the prediction of those that are not injured at the loss of recall. Going forward, we will only use the linear kernel for our improvements with SMOTE.

Improvement via SMOTE:

	precision	recall	f1-score	support
0 (No injury)	0.92	0.69	0.78	194
1 (Injury)	0.34	0.73	0.47	44
accuracy			0.69	238
macro avg	0.63	0.71	0.63	238
weighted avg	0.81	0.69	0.73	238

Table 4: SMOTE Metrics for SVM with linear kernel

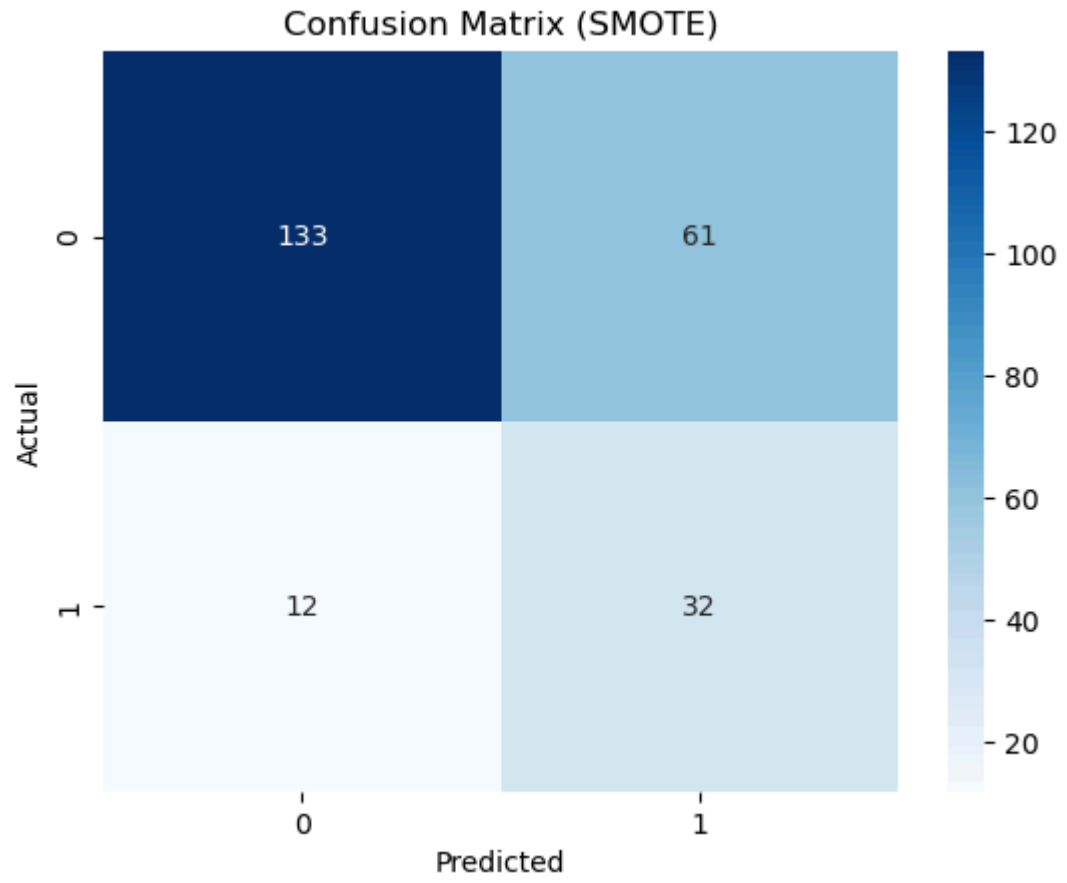


Figure 4: SMOTE Confusion Matrix for SVM Implementation with linear kernel

We made use of SMOTE the same way that we had with the previous algorithms in order to generate synthetic data to help with our prediction accuracy. When doing this with the linear kernel we see that there is some improvements over not using, but the impact is not as significant as when this process was done with the random forest algorithm.

Next Steps:

This algorithm did not yield an accuracy quite as high as we saw in the other algorithms implemented, but there is significant hyper parameter tuning that could be done in order to improve the accuracy of the model. For instance tuning these regularization constant or changing the kernel could lead to much better prediction accuracy. However, this model likely needs much more data than is currently collected in order for successful prediction.

Results and Discussion

Comparisons of Methods

Necessity of Data Augmentation (SMOTE):

None of the models had better prediction accuracy than majority vote guessing that the player did not have TJS, and before any hyperparameter tuning or use of SMOTE the statistics were essentially just doing majority vote. We saw that using K-means, random forest, and SVM were

no different than majority vote before any hyper parameter tuning of data processing with SMOTE. However, once we introduced SMOTE to preprocess the data points we were often able to guess that a player had been injured correctly. That is, given that a datapoint was an injured player then with probability close to 1 that the model would guess that the player was injured.

Post-Augmentation Comparisons:

Given the issues presented by the small minority class and the misleading nature of accuracy scores on this dataset, it's important to compare these models with two qualifications:

1) The models are only being compared on their post-augmentation performance 2) F1 score will be used as the primary comparison metric to score different models

Given these qualifications, it becomes clear that random forests were the clear best predictor of Tommy John injuries, as it displayed significantly better predictions for the injury cases. See the Charts below:

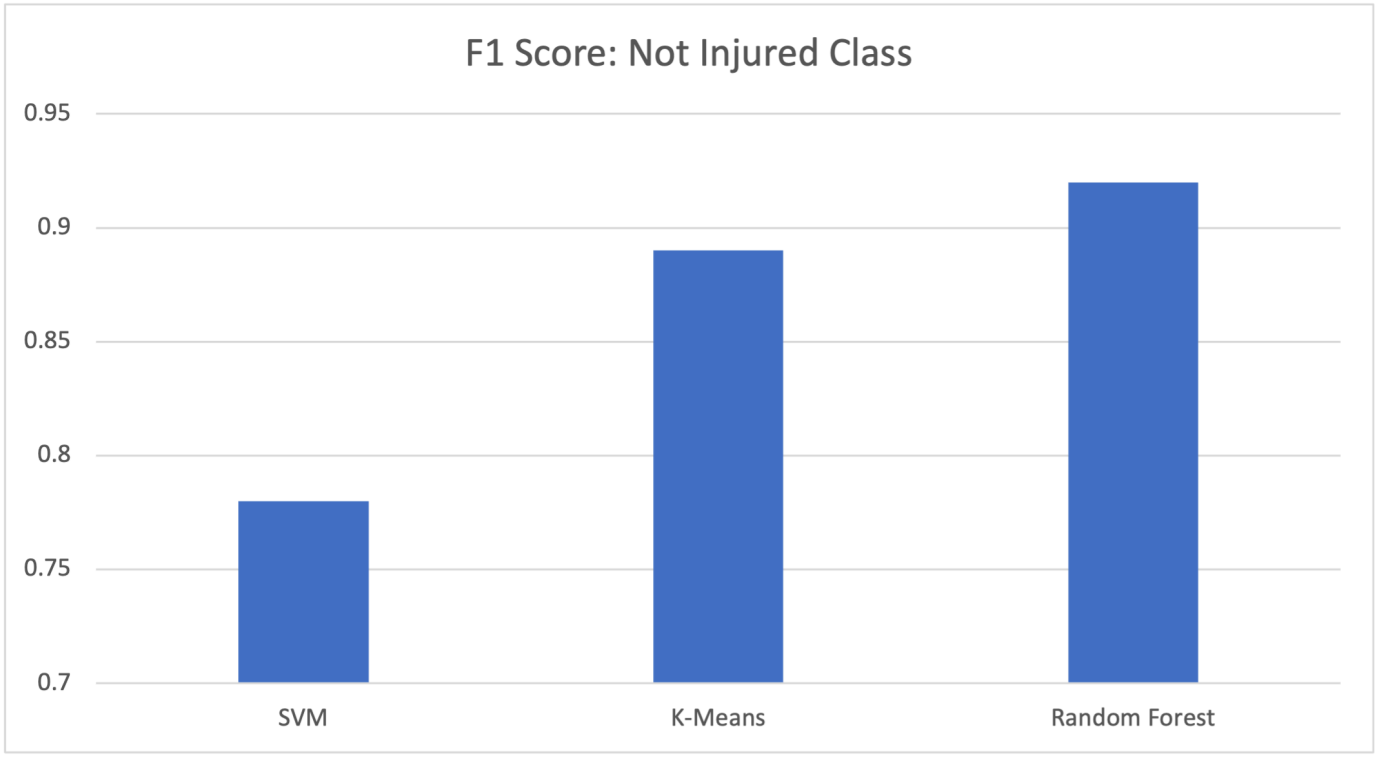


Figure 1: F1 Score: Not Injured Class

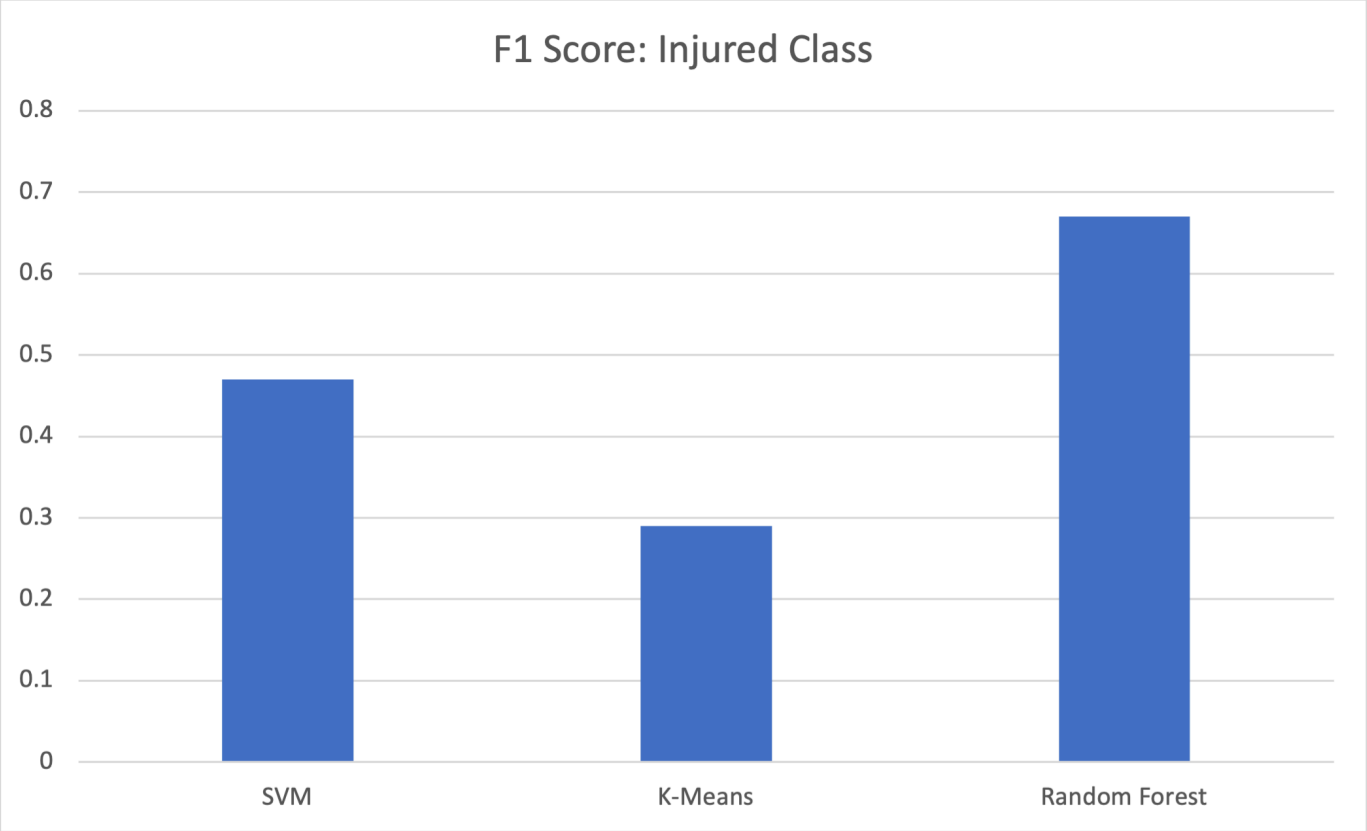


Figure 2: F1 Score: Injured Class

Since the random forest was best at making these predictions, it makes sense to rely on that predictor's insights when determining the most important features (see: Random Forests Section). Therefore, **we can tentatively conclude that the most important features when predicting a pitcher's injury risk are their release position and release velocity.**

Next Steps

Given the significant imbalance in our training data, the results of this study are generally promising. However, to make any more deterministic conclusions, those data weaknesses still need to be addressed over time.

Comprehensive pitching stats were only made available in the late 200s, so there is at most a few thousand data points that can contribute to the model, and within those very few have undergone TJS. Thus, there are several important next steps that could be taken to achieve higher accuracy than what we have attained. Future or more in depth experiments would likely find success by implementing some of the following suggestions:

1. Finding more data with features independent from those currently collect. This would allow for an investigation into if other features that were not captured in the utilized datasets were actually important.

2. Further Data Augmentation: Given the promise shown using techniques like SMOTE, synthetic data is likely a nontrivial benefit to injury risk prediction, and should be explored even as we wait for more current data.

3. Exploring Regression Models over Classification Models: Since the data set that we used only spans the past about 15 years, it is likely that some of the uninjured pitchers are still active which means that those that were falsely identified as injured by the models could actually be at high risk of injury. It would make more sense to use models that would identify a probability of injury at some point in their career since there statistics are still changing.

4. Hyperparameter tuning, especially for unsupervised models. K-means does not have many hyperparameters, but random forest and SVM do that could be calibrated to get a much better prediction accuracy.

Project Goals

We want to be able to utilize the given risk factors to predict how likely it is for a pitcher to have Tommy John surgery. With a successful implementation of this model, teams would be able mitigate risk of injury. We will evaluate the following metrics:

1. Accuracy: ratio of correctly predicted outcomes to the total number of cases. Starting point to help us determine how often our model is right, but it may not be sufficient if the data is imbalanced.

2. Precision: ratio of correct predictions of needing surgery to all positive predictions.

3. Recall/Sensitivity: proportion of true positives to the total number of actual positive cases.

4. F1 Score: Tommy John surgery is relatively rare, so accuracy alone might not be a good metric. Instead, we can use the harmonic mean of precision and recall.

References

[1] Chalmers, Peter et al. “Fastball Pitch Velocity Helps Predict Ulnar Collateral Ligament Reconstruction in Major League Baseball Pitchers”. *American Journal of Sports Medecine*, vol 44 no. 8. March 2016. [Abstract]. Available: https://pubmed.ncbi.nlm.nih.gov/26983459/. [Accessed October 2, 2024].

[2] Karnuta, Jaret et al. “Machine Learning Outperforms Regression Analysis to Predict Next-Season Major League Baseball Player Injuries: Epidemiology and Validation of 13,982 Player-Years From Performance and Injury Profile Trends, 2000-2017”. *Orthopedic Journal of Sports Medecine*, vol 8 no. 11. November 2020. [Abstract]. Available: https://pubmed.ncbi.nlm.nih.gov/33241060/. [Accessed October 2, 2024].

[3] Nicholson, Kristen et al. “Machine Learning and Statistical Prediction of Pitching Arm Kinetics”. *American Journal of Sports Medecine*, vol 50 no. 1. November 2021. [Abstract]. Available: https://journals.sagepub.com/doi/abs/10.1177/03635465211054506. [Accessed October 2, 2024].

[4] Miller, John. "'The Need for Speed': Tommy John Surgery has Become Most Prevalent in Youth Baseball Players". *nbc26.com*, para 1, May 24, 2024. [Online]. Available: https://www.nbc26.com/sports/youth-sports/the-need-for-speed-tommy-john-surgery-has-become-most-prevalent-in-youth-baseball-players. [Accessed October 2, 2024].

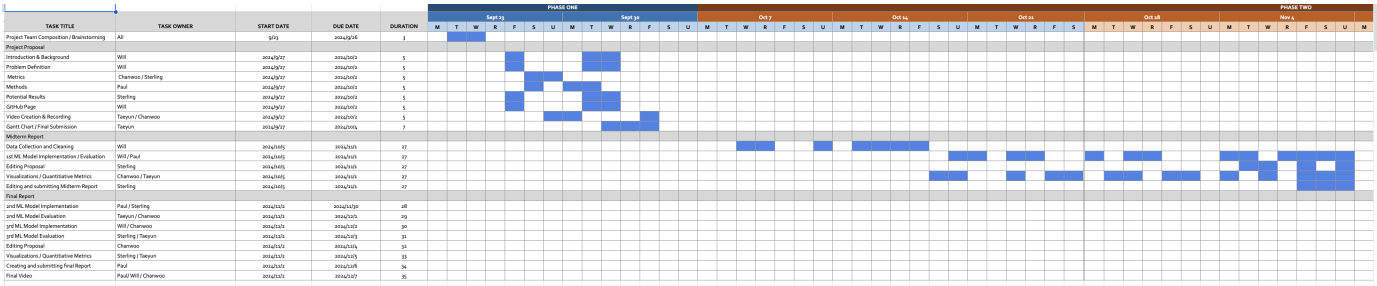
[5] Meldau, Jason et al. "Cost analysis of Tommy John Surgery for Major League Baseball Teams". *Journal of Shoulder and Elbow Surgery*, vol 29 no. 1. January, 2020. [Abstract]. Available: https://pubmed.ncbi.nlm.nih.gov/31668501/. [Accessed October 2, 2024].

Gantt Chart

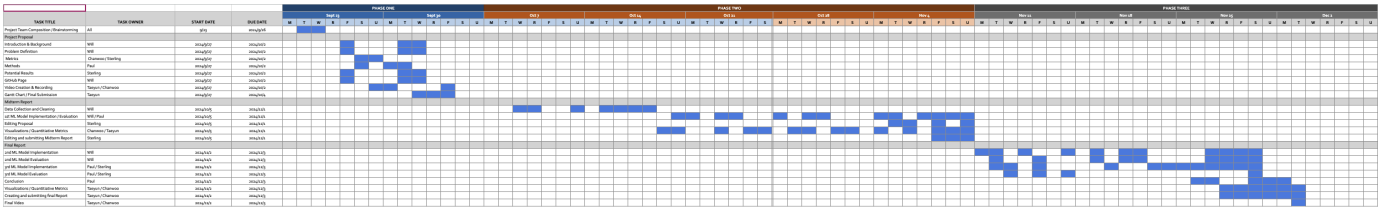
Proposal

					PHASE ONE												
					Sept 23							Sept 30					
TASK TITLE	TASK OWNER	START DATE	DUE DATE	DURATION	M	T	W	R	F	S	U	M	T	W	R	F	S	U
Project Team Composition / Brainstorming	All	9/23	2024/9/26	3														
Project Proposal																		
Introduction & Background	Will	2024/9/27	2024/10/2	5														
Problem Definition	Will	2024/9/27	2024/10/2	5														
Metrics	Chanwoo / Sterling	2024/9/27	2024/10/2	5														
Methods	Paul	2024/9/27	2024/10/2	5														
Potential Results	Sterling	2024/9/27	2024/10/2	5														
GitHub Page	Will	2024/9/27	2024/10/2	5														
Video Creation & Recording	Taeyun / Chanwoo	2024/9/27	2024/10/2	5														
Gantt Chart / Final Submission	Taeyun	2024/9/27	2024/10/4	7														
Midterm Report																		
1st ML Model Implementation	Taeyun / Chanwoo	2024/10/5	2024/11/1	27														
1st ML Model Evaluation	Paul / Sterling	2024/10/5	2024/11/1	27														
Editing Proposal	Will	2024/10/5	2024/11/1	27														
Visualizations / Quantitiative Metrics	Will	2024/10/5	2024/11/1	27														
Editing and submitting Midterm Report	Will	2024/10/5	2024/11/1	27														
Final Report																		
2nd ML Model Implementation	Paul / Sterling	2024/11/2	2024/11/30	28														
2nd ML Model Evaluation	Taeyun / Chanwoo	2024/11/2	2024/12/1	29														
3rd ML Model Implementation	Will / Chanwoo	2024/11/2	2024/12/2	30														
3rd ML Model Evaluation	Sterling / Taeyun	2024/11/2	2024/12/3	31														
Editing Proposal	Chanwoo	2024/11/2	2024/12/4	32														
Visualizations / Quantitiative Metrics	Sterling / Taeyun	2024/11/2	2024/12/5	33														
Creating and submitting final Report	Paul	2024/11/2	2024/12/6	34														
Final Video	Paul/ Will / Chanwoo	2024/11/2	2024/12/7	35														

Mid Term



Final



Contribution Table

Proposal

Name	Contribution
Will Ferguson	Wrote the motivation/description/datasets of the proposal and built the GitHub pages
Taeyun Lee	Worked on Presentation and Video and organized by creating charts and handling the final submission
Chanwoo Moon	Worked on Presentation and Video and metrics section of the proposal
Sterling Kalogeras	Wrote the metrics and expected results section
Paul Hutchison	Wrote the method section of the proposal

Mid Term

Name	Contribution
Will Ferguson	Motivation/description/datasets of proposal and built the GitHub pages; data collection and first model
Taeyun Lee	Worked on first unsupervised model implementation along with documentation
Chan woo Moon	Worked on first unsupervised model implementation and data visualization
Sterling Kalogeras	Wrote the metrics and expected results section; editing midterm report and submitting
Paul Hutchison	Wrote the method section of the proposal; worked on first supervised model implementation

Final

Name	Contribution
Will Ferguson	Worked on the overall results section of the documentaiton and final profread of the whole paper
Taeyun Lee	Worked on creating the presentation and the video along with logistic stuff such as Gantt Chart
Chan woo Moon	In charge of creating the presentation and the video along with submission of the project
Sterling Kalogeras	Worked on inital SVM implementation and the write up part corresponding to it
Paul Hutchison	Worked on SVM improvements and the SVM conclusion part of the documentation

The link to both the chart and table is below
https://docs.google.com/spreadsheets/d/133hKJR-TpGdRX6YxMOyqjWuLo6p4oV4eVbARgMnnMHE/edit?usp=sharing