I developed a model (boosted tree) that is able to predict strike accuracies at 85.9% which is 22.2 percentage points higher than the naïve model predicting everything as strikes.

Cleaning

- Summarize all variables in the test and train data sets provided
- Tilt raised problems but since it is redundant with spin_axis, remove it from model building
- Created is_strike variable in train data set using for loop through pitch_call variable
- Train and test have NA values in numeric fields; need to merge data frames together
- Replace NAs in joined data with median imputation
- Correct pitcher/batter levels to be "Left" and "Right"
- Start with separating back into original train and test samples
- Remove identifying, redundant, and zero-variance variables

Model Development

- Split into training (5,000 randomly selected rows) and holdout (577,205 rows) samples
- Using caret package, tested multiple models with base parameters; Figure 1 shows model name and accuracy on the holdout sample

Model	Naïve	Vanilla Logistic Regression	Regularized Logistic Regression	Vanilla Partition	Random Forest	Boosted Tree
Accuracy on Holdout	63.7%	59.8%	66.0%	84.0%	85.2%	85.4%

Figure 1: Model Success as Percent Correct on Holdout Sample (577,205 rows)

- Boosted tree is best model
  - Best model for Random Forest was with mtry=12, didn't change with other iterations
  - After multiple iterations, my Boosted Tree performed better because parameters were tuned to trees=10,150, interaction depth=5, shrinkage=0.00095, & nodes @ 5
- Tune Boosted Tree parameters and run final model on test data to make strike predictions
- Adjust final predictions from "Yes" and "No" to 1s and 0s and save into test data

Further Recommendations (Steps to Improve)

- Tune with more variables explaining specific catcher abilities
  - Framing
  - Overall Receiving
  - Passed Balls
- Build model again with a larger sample of training data to learn data tendencies better, but this requires higher computing power than I have available
- See if more advanced machine learning models like neural networks/deep learning are even better for this type of data