Effectiveness of the Infield Shift

Ruslan Davtian

4/15/2020

Data Description:

There are two data sets (infield_bip.csv and player_lkup.csv) that represent statcast batted ball characteristics such as exit velocity, launch angle, launch direction, landing distance, landing bearing, and hang time. Each row is a unique pitch from 4,601 different games between the 2018 and 2019 seasons. Also, there exists x, y positional coordinates of the fielders (except pitcher, catchers) for every pitch and the result of the play. The goal of this analysis is to build a model on some training data set to predict the probability of a batted ball being a hit. The data set set contains a total of 223,275 pitches and below are the first six rows and ten columns of the data set.

play_pk	game_pk	season	batter_id	pitcher_id	stands	throws	exit_velocity	launch_angle	launch_direction
11726380	530960	2018	592626	458924	L	L	90.409	-25.803	-0.791170
11726428	530960	2018	645277	477132	R	L	NA	NA	-149.664000
11726435	530960	2018	657077	592145	L	L	104.917	0.119	-2.556210
11726449	530960	2018	641355	592314	L	R	93.706	8.142	-0.485394
11726458	530960	2018	641355	592314	L	R	83.754	-28.945	47.196700
11726464	530960	2018	571970	592314	L	R	87.110	4.028	37.035600

Data Cleaning:

The first step is to filter the data to only the batted balls that I am interested in. Since the goal of the analysis is to eventually compare xBABIP between shifts and no shifts, I need to filter the data to meet the criteria of a batted ball in play. Below is the frequency and proportion table of game events 1.

Game Event 1	Freq	Proportion
foul	101822	0.456
hit_into_play	80531	0.361
hit_into_play_no_out	28821	0.129
hit_into_play_score	9020	0.040
foul_bunt	1127	0.005
hit_by_pitch	755	0.003
ball	632	0.003
swinging_strike	415	0.002
foul_tip	91	0.000
called_strike	22	0.000
swinging_strike_blocked	11	0.000
blocked_ball	8	0.000
bunt_foul_tip	1	0.000
missed_bunt	1	0.000

BABIP is defined as (H - HR) / (AB - HR - K + SF). Therefore, I only kept game events 1 that resulted in batted balls in play. After filtering to only have game events hit into play, hit into play no out, and hit into play score, the following game events 2 remain.

Game Event 2	Freq	Proportion
field_out	68035	0.575
single	29834	0.252
grounded_into_double_play	6233	0.053
force_out	6091	0.051
double	3892	0.033

Game Event 2	Freq	Proportion
field_error	2285	0.019
double_play	518	0.004
fielders_choice_out	504	0.004
fielders_choice	403	0.003
sac_bunt	327	0.003
triple	220	0.002
catcher_interf	8	0.000
sac_fly	7	0.000
triple_play	5	0.000
home_run	1	0.000
sac_bunt_double_play	1	0.000

I filtered out any game events 2 that are sacrifice bunts, catcher interference, home runs, and balks. Now, there exists 118,027 observations and only 632 observations have missing exit velocity and launch angle values. Also, 184 observations have at least one missing x, y coordinates for any position. I felt is was appropriate to drop these few missing values to have a complete data set.

Feature Engineering:

The next step is to use the x, y coordinates of each infielder excluding catchers and pitchers to define an indicator variable of whether the defense was shifted for each batted ball. A shift exists if there are three infielders on the pull side of the infield, and a non-shift otherwise. In other words, if a left handed hitter is batting, then there must be three infielders located to the right side of the infield (1B, 2B, SS OR 1B, 2B, 3B). If a right handed batter is batting, then there must be three infielders to the left side of the infield. Since the first baseman must always be to the right side of the infield, then only one alignment exists for a shift to a right handed batter (2B, 3B, SS). If the alignment of the fielders meet the criteria for a shift, then a shift exists. Otherwise, there is no shift. Positive x coordinates represent the right side of the infield while negative x coordinates identify the left side of the infield. Below are three tables that summarize average BABIP, shift percentage, and average BABIP across shift or no shift for the entire league, right handed batters, and left handed batters separately. Left handed batters have a slightly lower BABIP than right handed batters which can be justified by the higher shift percentage against left handed batters than right handed batters. Also, there seems to be a significant drop in average BABIP across all players when the fielders are shifted against fielders that are not shifted.

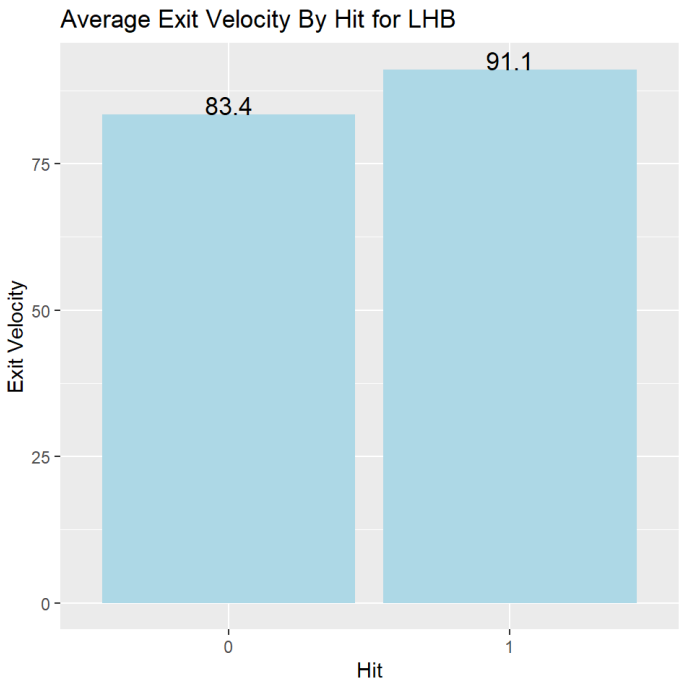
	BABIP
LEAGUE	0.286
LHB	0.282
RHB	0.289

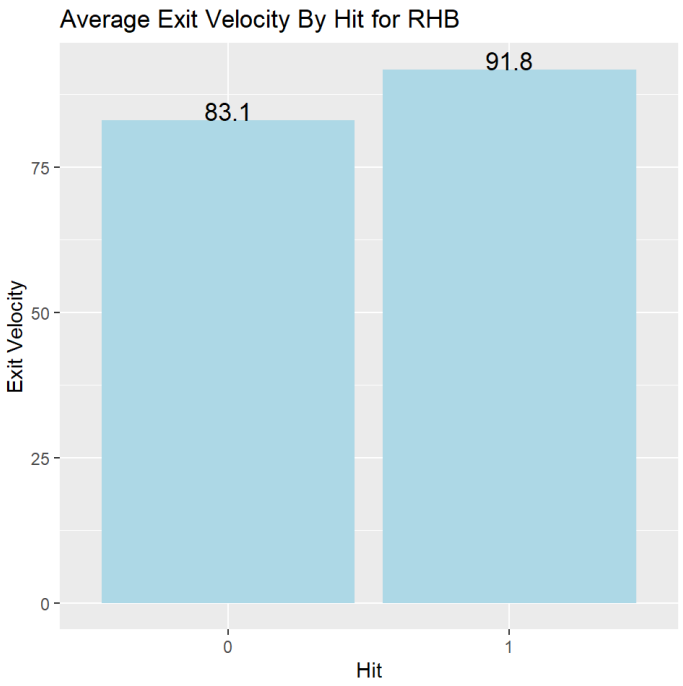
	Shift %
LEAGUE	0.192
LHB	0.316
RHB	0.109

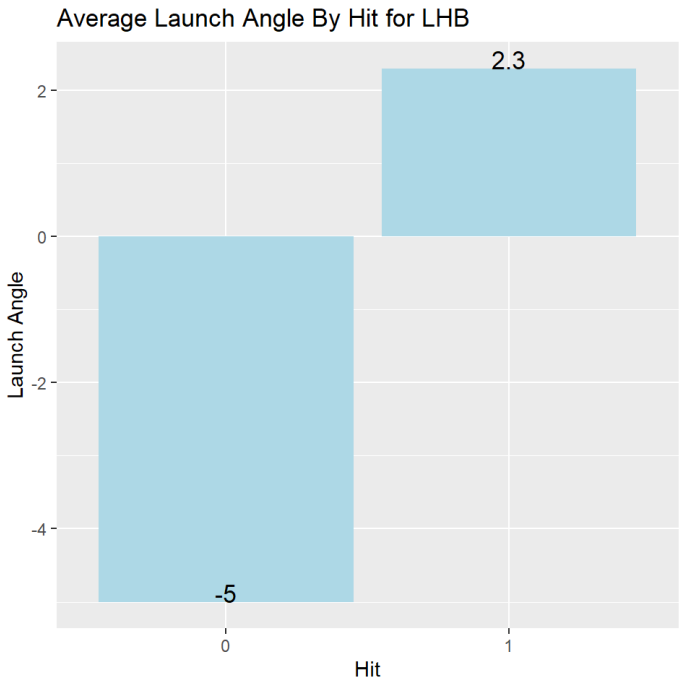
	Shift	No Shift
LEAGUE BABIP	0.257	0.293
LHB BABIP	0.251	0.297
RHB BABIP	0.270	0.291

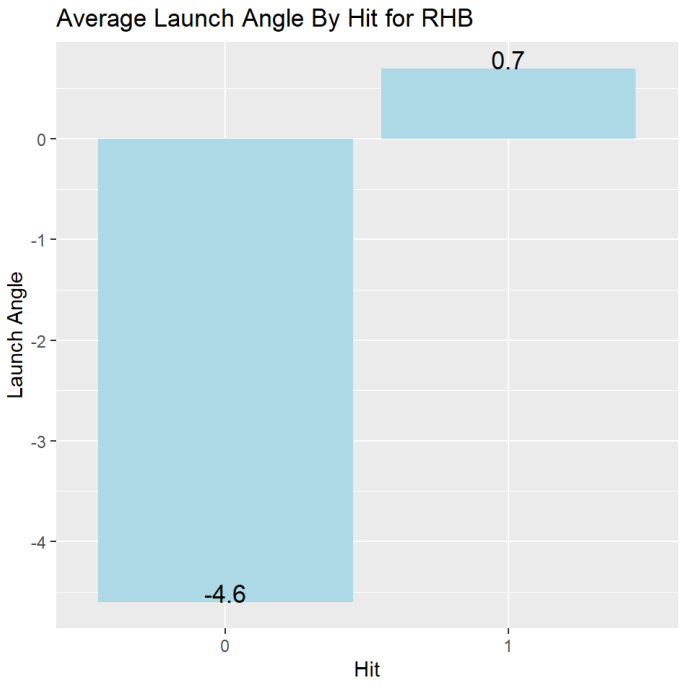
Data Visualizations:

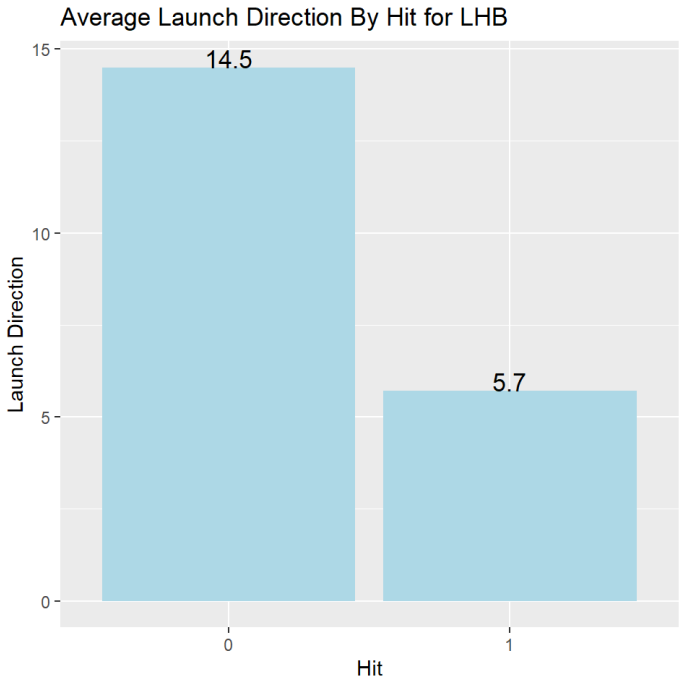
The summary tables above provide evidence that the shift indicator is worth to include in the model to predict hit vs no hit on a batted ball. The bar plots below investigate average batted ball characteristics (exit velocity, launch angle, launch direction, landing distance, landing bearing, and hang time) separated by batting side against hits versus no hits. There seems to be large differences in these variables among hit versus no hit which indicates that these variables may be useful to include in the model along with the shift indicator.

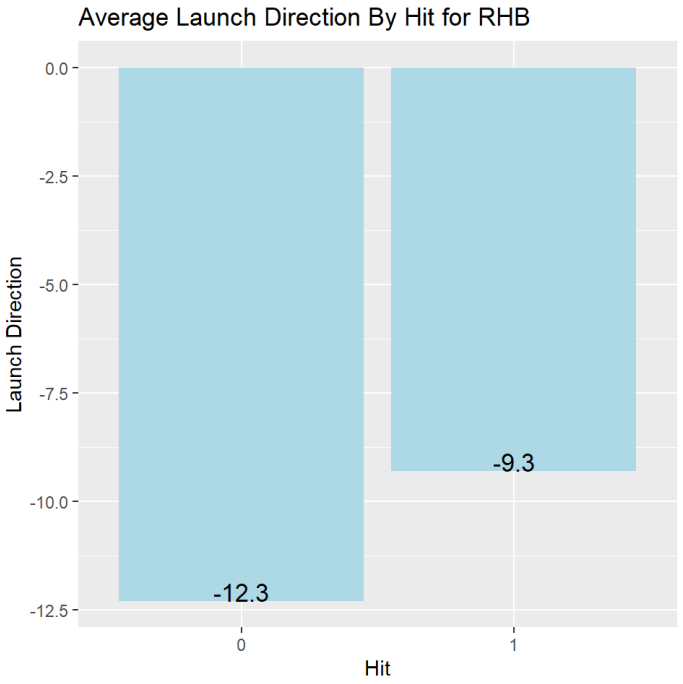

Average Exit Velocity By Hit for LHB
91.1
83.4
75
50
25
0
Exit Velocity
0
1
Hit

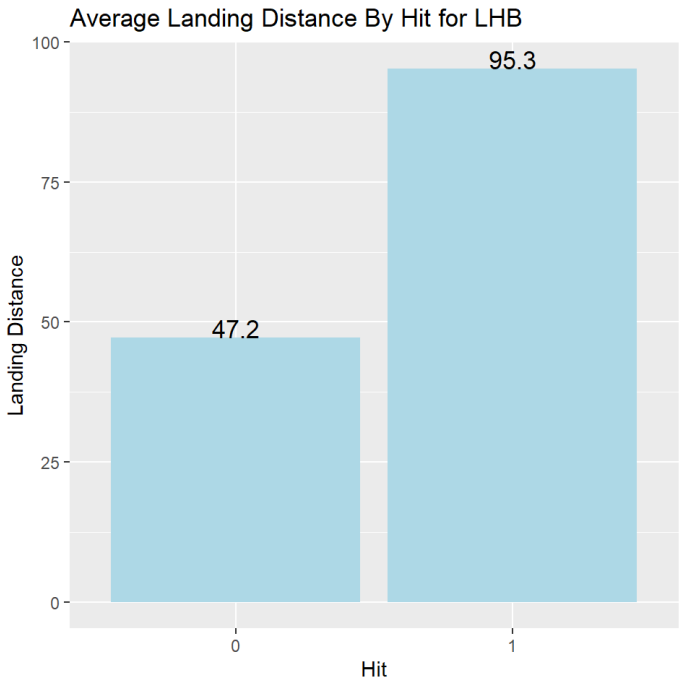

Average Exit Velocity By Hit for RHB
91.8
83.1
75
50
25
0
Exit Velocity
0
1
Hit

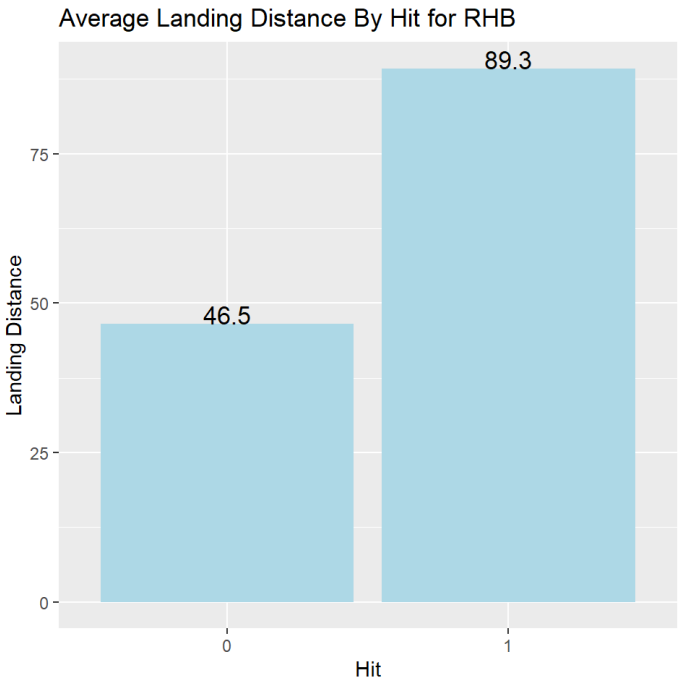

Average Launch Angle By Hit for LHB
2.3
2
0
-2
-4
-5
Launch Angle
0
1
Hit

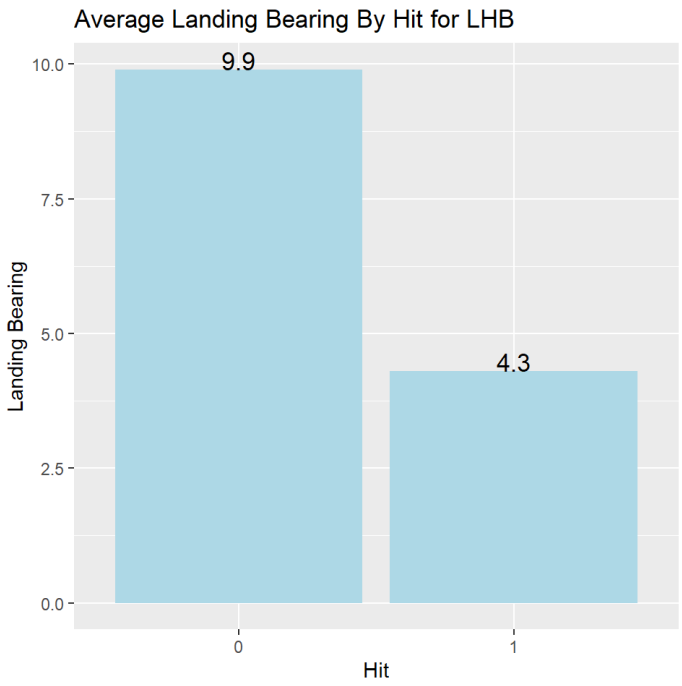

Average Launch Angle By Hit for RHB
0.7
0
-1
-2
-3
-4
-4.6
Launch Angle
0
1
Hit

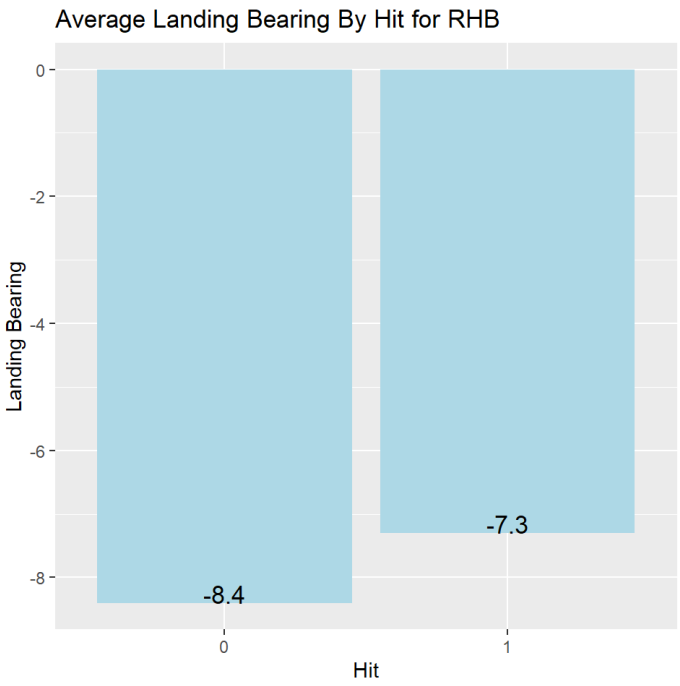

Average Launch Direction By Hit for LHB
Launch Direction
15
10
5
0
14.5
5.7
0
1
Hit

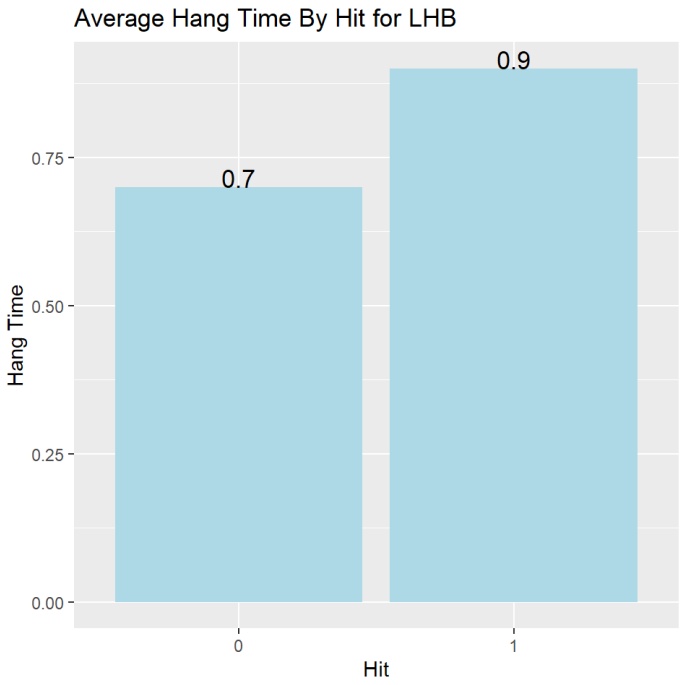

Average Launch Direction By Hit for RHB
Launch Direction
0.0
-2.5
-5.0
-7.5
-10.0
-12.5
-12.3
-9.3
0
1
Hit

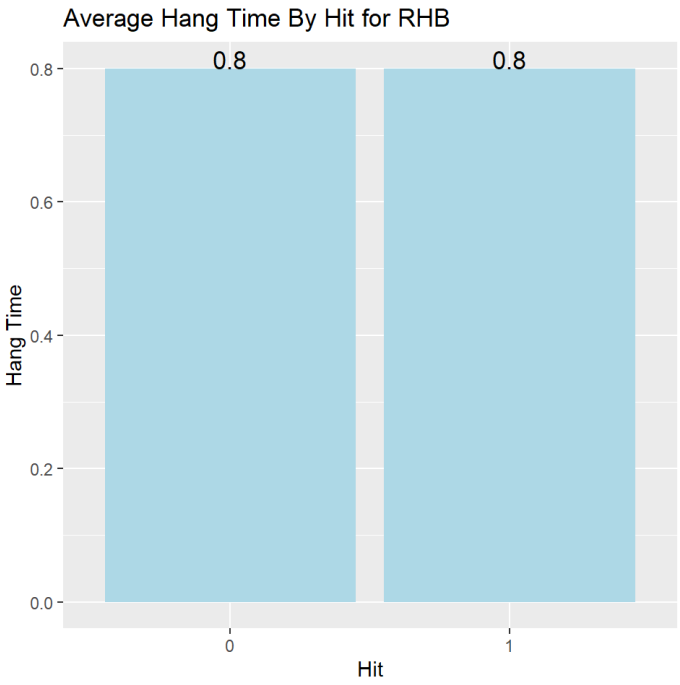

Average Landing Distance By Hit for LHB
Landing Distance
100
75
50
25
0
47.2
95.3
0
1
Hit


Average Landing Distance By Hit for RHB
Landing Distance
75
50
25
0
46.5
89.3
0
1
Hit









Multicollinearity Check:

Before I perform any modeling, I want to check if any of the variables selected as candidates for the models explain similar variation in batted balls resulting in hits or no hits. Therefore, I checked Variance Inflation Factors (VIFs) among the variables. Any two variables over 9 reveal severe multicollinearity among them and one of them should be removed. The VIFs are located below.

exit_velocity	launch_angle	launch_direction	landing_distance	landing_bearing	hang_time	shift
1.2	4.88	8.57	2.15	8.53	4.97	1.02

There seems to be an issue with multicollinearity with launch direction and landing bearing. I should not use both of these variables since their vif values are around 8 and their correlation coefficient is 0.92. There are no other issues with multicollinearity.

Pre-Processing and Machine Learning:

First, I split the infield_bip.csv data set into a train, test split. Since I have data on 2018 and 2019, I decided to use 2018 as the train set and 2019 as the test set. About 53.5% of the data is 2018 and 46.6% of the data is 2019. Although this is not a traditional 70/30 split, I believe splitting on year makes sense and there is still over 60,000 rows to train models. For consistent and improved results, I standardized all predictor variables in the training and testing sets and used 5-fold cross validation across each algorithm. I used cross validated accuracy and kappa performance metrics to rank the algorithms.

Logistic Regression:

I decided to build a logistic regression model first to use as a base model to compare against. The output below is from the final logistic

model chosen. The final model incorporates shift indicator, landing distance, landing bearing, exit velocity, hang time, and launch angle. Each of these variables are statistically significant and the coefficient for shift is negative as expected. Adjusting for all other variables, the log odds of a batted ball being a hit with a shift is -0.3248137. Therefore, the odds of a batted ball being a hit with a shift is e^(-0.3248137) or 0.722662 which is equivalent to a decrease of odds by about 28% than for the same batted ball against no shift.

Below, are the performance metrics for the test set. I looked at accuracy by result (hit/out) as well as overall accuracy and kappa metrics that will be used to compare against other algorithms. As the base model, the logistic regression model's overall accuracy on the test set is about 76.6%.

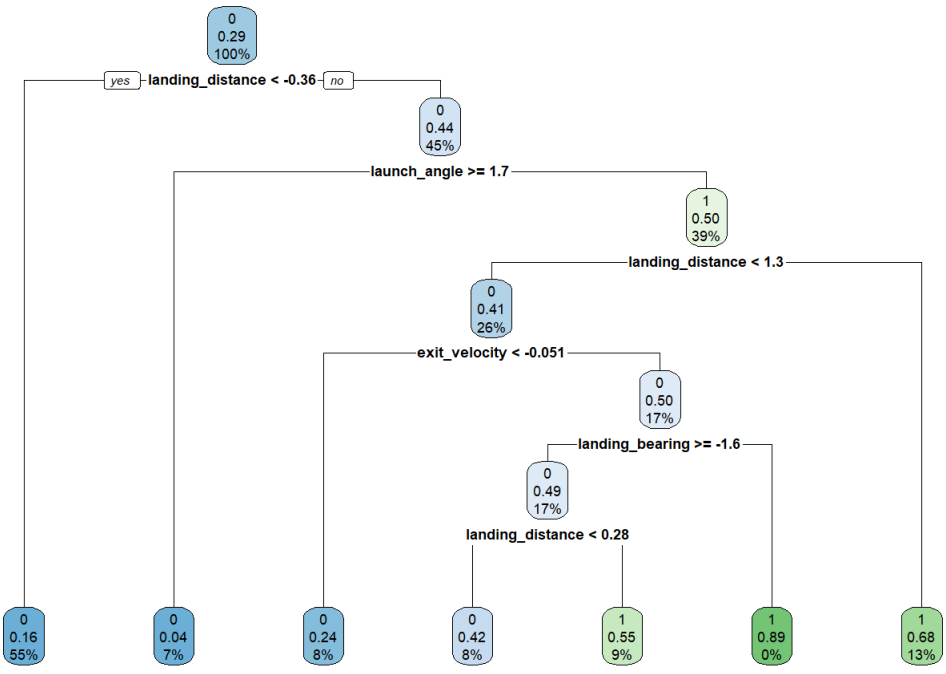
	Estimate	Std. Error	z value	p value
(Intercept)	-1.1075436	0.0117741	-94.065873	0.0e+00
shift1	-0.3248137	0.0289667	-11.213332	0.0e+00
landing_distance	1.4660016	0.0192261	76.250661	0.0e+00
landing_bearing	-0.0693015	0.0105969	-6.539823	0.0e+00
exit_velocity	0.4037401	0.0126982	31.795012	0.0e+00
hang_time	-1.2552342	0.0342514	-36.647681	0.0e+00
launch_angle	0.1350688	0.0287424	4.699281	2.6e-06

	0	1
Accuracy by No Hit/Hit	0.794	0.645

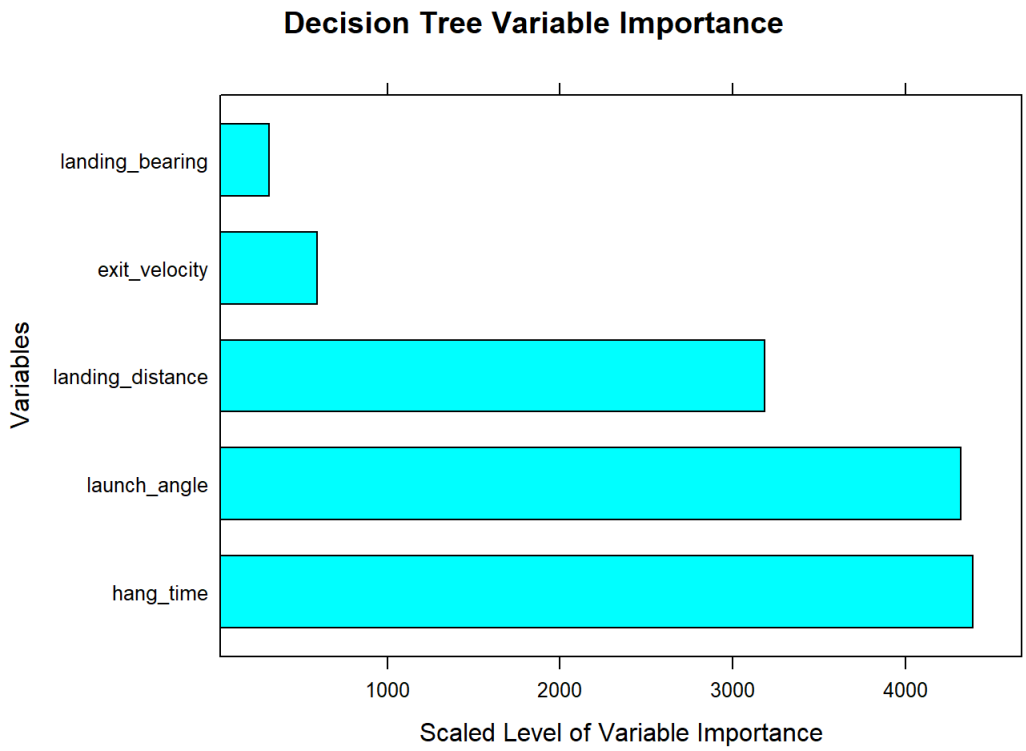
	Overall Test Accuracy
Accuracy	0.766
Kappa	0.361

Decision Tree:

I decided to build a decision tree first to look at the first few splits in order to determine the most important variables. I fixed the complexity parameter to not over fit or produce too long of a tree and chose 10 as minimum number of observations that must exist in a node in order for a split to be attempted. A plot of the decision tree can be found below.



At each split, the tree shows which variables the split was made but the values shown do not make sense because they are standardized. The final tree has depth 5 with 7 terminal nodes. The variables used in this tree as nodes are landing distance, launch angle, exit velocity, and landing bearing. It is important to note that a deeper tree many use all of the variables but there is a risk of over-fitting the training data set and performing worse on the testing set.



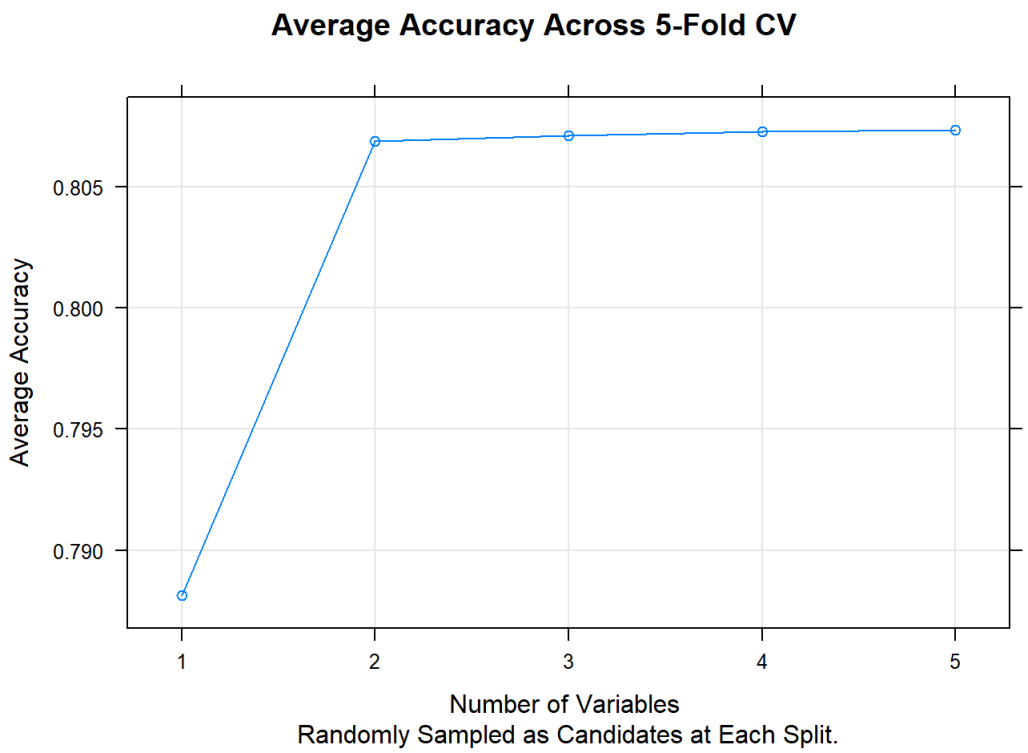
Above, is a bar chart showing the scaled levels of variable importance. The top 5 most important variables in order of significance for the decision tree is hang time, launch angle, landing distance, exit velocity, and landing bearing. Notice that this decision tree did not use the shift indicator variable which means the tree does not produce different batted ball hit probabilities for shift and no shift alignments adjusting for all other variables. Below, is the accuracy by result as well as overall test accuracy and kappa values. The overall accuracy for the decision tree is no different than that of the logistic regression at about 76.6%.

	0	1
Accuracy by No Hit/Hit	0.809	0.617

	Overall Test Accuracy
Accuracy	0.766
Kappa	0.389

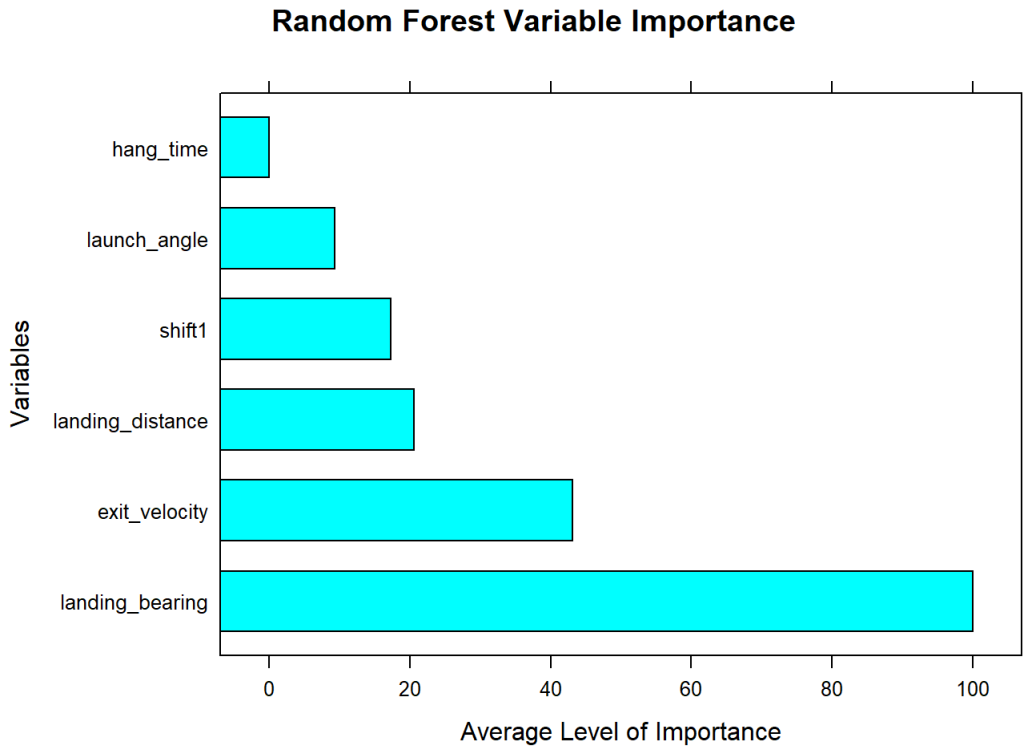
Random Forest:

Decision Trees might perform well on trained data but there could be possible issues with generalizing to new, unseen data. Therefore, I will try random forest as the next modeling technique. They are more difficult to interpret than decision trees but usually have higher predictive power. I used the same variables as the previous model and I chose the number of trees parameter to be 250. Also, I determined the number of variables (1 to 5) to be randomly sampled as candidates at each split through cross validation. A plot of this can be found below.



At each of the 250 trees, a random sample of variables to be elected as candidates for each split is important to not allow the same splits to happen over and over for each tree. It is determined that the best number of variables to have for random selection on each split is 5 from cross validated accuracy.

Below, is the average, scaled variable importance chart for the random forest algorithm. The most important variables in order of significance for the random forest model is landing bearing, exit velocity, landing distance, shift, launch angle, and hang time. Interestingly, the random forest has hang time as the least important variable while the decision tree has hang time as the most important. I would trust the random forest more than the decision tree since the random forest is built from averaging results of 250 different trees using different sub samples of the training data that lowers variation in the prediction.



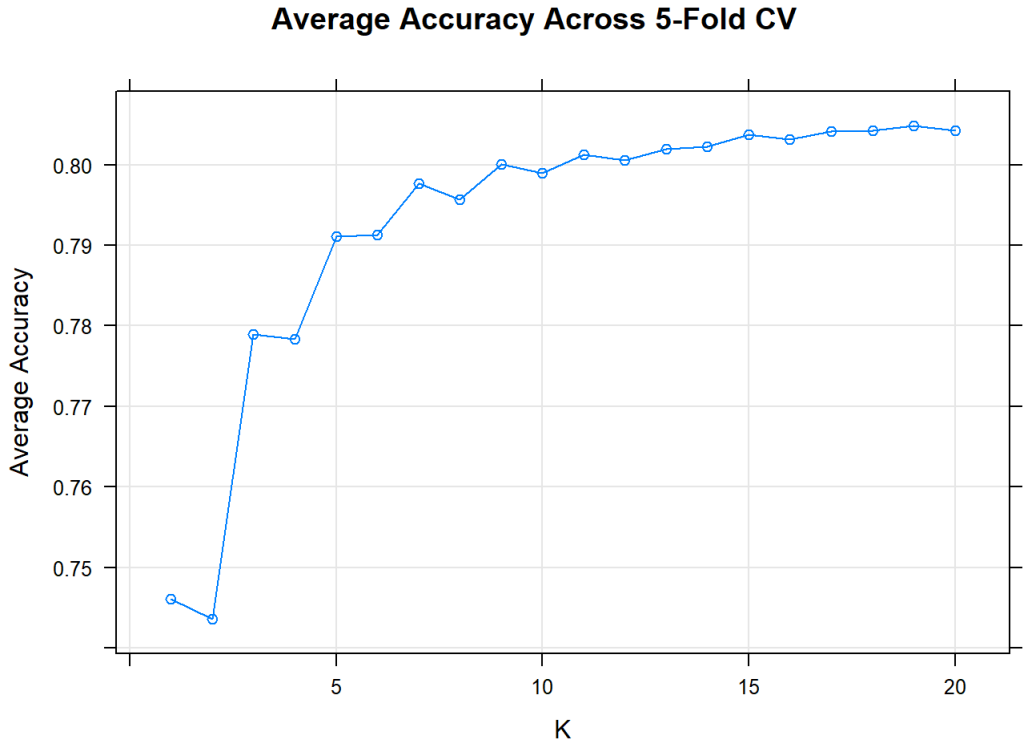
Below, is the accuracy by hit/no hit and overall accuracy and kappa values for the random forest algorithm. The random forests algorithm performed better than the decision tree with about 80.2% accuracy.

	0	1
Accuracy by No Hit/Hit	0.832	0.701

Overall Test Accuracy	
Accuracy	0.803
Kappa	0.485

K-th Nearest Neighbors:

KNN is one of the simplest algorithms as it calculates the Euclidean distances between each of the predictors and a class for the testing observation is determined through the majority class of its k closest neighbors. K is chosen through cross validation that maximizes accuracy and kappa values. A plot of cross validated accuracy versus K is shown below.



The plot converges quickly after K = 10 but in this case, the K which produced the best overall accuracy and kappa values is 19.

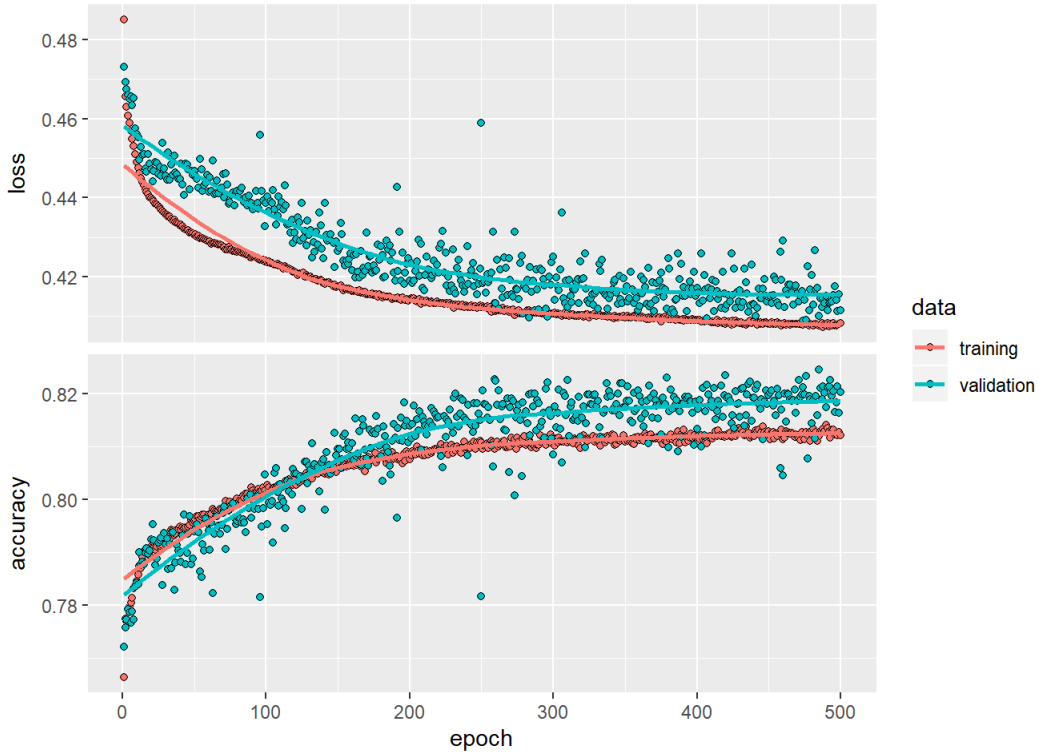
Below, is the accuracy by hit/no hit and overall accuracy and kappa values for the KNN algorithm. The algorithm performs similarly to the random forest (79.9% accuracy) and better than the decision tree or logistic regression.

	0	1
Accuracy by No Hit/Hit	0.827	0.698

Overall Test Accuracy	
Accuracy	0.799
Kappa	0.472

Neural Network:

For the last algorithm, I created a multi-hidden layer neural network to classify the result of batted balls (hit/no hit). Neural networks are very difficult to explain or interpret the results but when hyperparameterized correctly, can produce strong predictive results. I chose to have only two hidden layers where the first layer has 64 nodes and the second layer has 32 nodes. Both layers use relu as the activation function and the output layer has 2 units for the 2 result types. Also, the output layer uses the softmax activation function to transform the outputs into probabilities between 0 and 1 for binary classification. This process of feeding observations into the input layers and calculating the output layer is called forward propagation. In order to improve the accuracy of the predictions, the back propagation process needs to be optimized to minimize some loss function (categorical cross-entropy). In this process, I chose to use 200 iterations or epochs with batch sizes of 128 for faster training. Also, I used 0.2 as the validation split during the training. The plot of the accuracy and loss for both training and validation sets throughout the 200 iterations can be found below.



The neural network converged after about 100 iterations and the accuracy from both training and validation sets seem to be around 80% which is similar to random forest and KNN algorithms. The overall accuracy for the neural network is significantly better than the decision tree and logistic regression model and only slightly worse among random forest and KNN.

Train Accuracy
0.815

	0	1
Accuracy by No Hit/Hit	0.829	0.721

Overall Test Accuracy	
Accuracy	0.806
Kappa	0.487

Ranking Algorithms

In order of overall accuracy, the algorithms from highest to smallest are as follows, (random forest, knn, neural network, decision tree, logistic regression). Therefore, I chose to use the random forest model to make predictions on the testing set instead of the other models. To make sure the testing predictions make sense, it is important to compare the distribution of results from the predictions against the distribution of actual results from the testing set.

Below is the distribution of hit/no hit for the test data.

Hit	Out
0.713	0.287

Below is the distribution of hit/no hit for the predictions made using the random forest model.

Hit	Out
0.776	0.224

Below is the two way table Confusion Matrix comparing the actual testing hit/no hit (columns) versus the predicted hit/no hit (rows).

	0	1
0	35217	7101
1	3647	8566

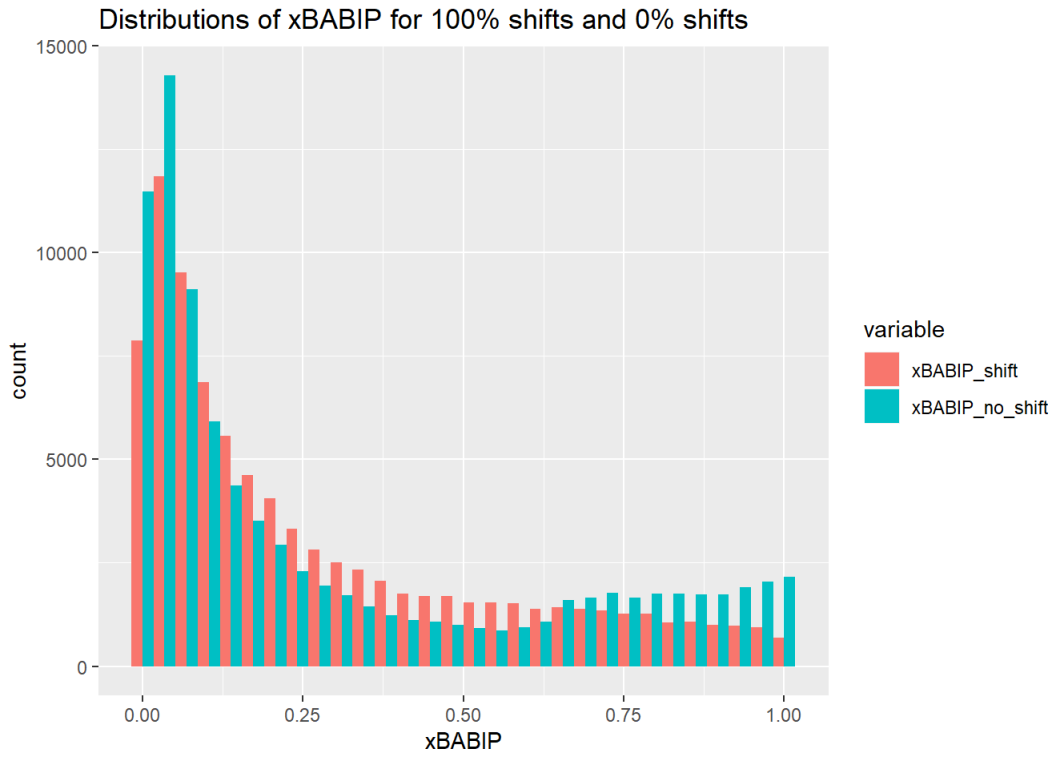
The proportion of hits for both the testing set and the prediction set are very similar so the predictions from the random forest model make sense.

Part 2)

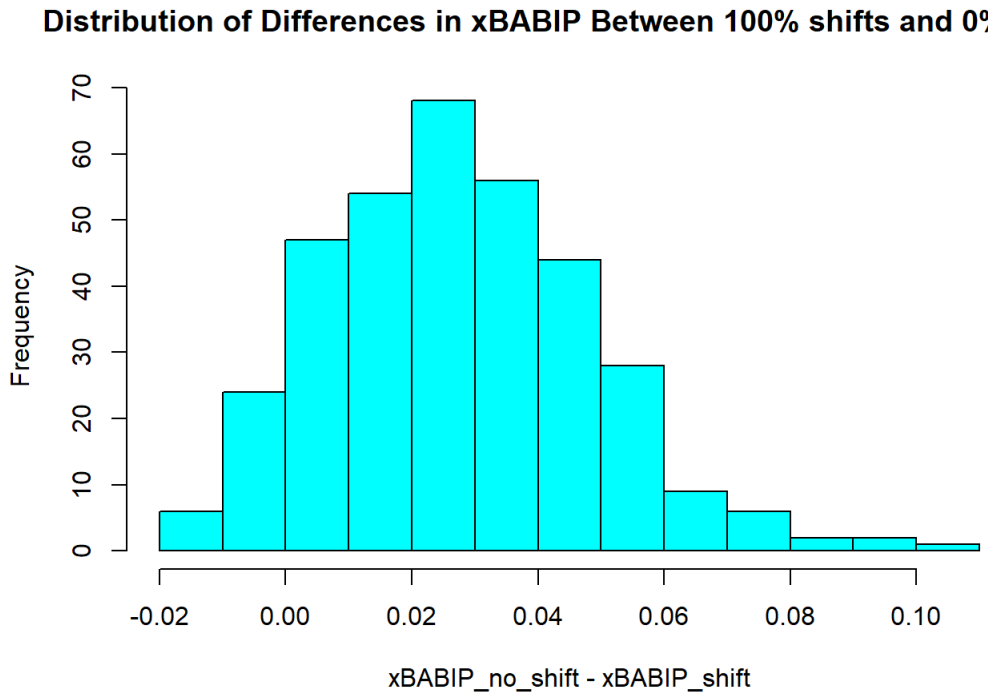
First, I filtered the overall data set that consists of both 2018 and 2019 seasons to just batters who have at least 100 balls in play. Using the random forest model, I will compute each batter's xBABIP by predicting each batted ball's hit probability twice where each hitter faces the shift 100% of all batted balls in play and where each hitter faces the shift 0% of all batted balls in play. Lastly, I will average hit probability by batter for both shift/no shift to find the xBABIP for each batter by shift.

a)

Below is an overlaying distribution plot for both xBABIP predictions with 100% shifts on batted balls and 0% shifts on batted balls for all batters. The difference is slight but the mean and median of xBABIP on shifts are lower than the mean and median of xBABIP on non-shifts which is expected. The difference is slight but noticeable in the plot as well. The distribution of xBABIP for 100% shifts is slightly shifted left compared to the distribution of xBABIP for 0% shifts.



Below, is another plot that shows the distribution of xBABIP differences for all batters (xBABIP 0% shifts minus xBABIP 100% shifts). The distribution is centered around 0 which means on average, xBABIP drops 0 when batters hypothetically face shifts every time compared to batters who hypothetically face no shifts every time.



b)

Here is list of the top 15 batters where the shift is most effective. Notice how these batters are almost exclusively left handed batters.

batter_name	stands	xBABIP_no_shift	xBABIP_shift	xBABIP_diff
Gallo, Joey	L	0.3605	0.2604	0.1001
Bruce, Jay	L	0.3332	0.2332	0.1000
Carpenter, Matt	L	0.3330	0.2428	0.0902
Granderson, Curtis	L	0.3707	0.2834	0.0873
Seager, Kyle	L	0.3200	0.2392	0.0809
Moreland, Mitch	L	0.3036	0.2284	0.0753
Moran, Colin	L	0.3330	0.2581	0.0750
Muncy, Max	L	0.3533	0.2813	0.0721
Beaty, Matt	L	0.3241	0.2530	0.0711
Odor, Rougned	L	0.3373	0.2664	0.0709
Vogelbach, Daniel	L	0.3388	0.2687	0.0701
Seager, Corey	L	0.3039	0.2340	0.0699
Ruiz, Rio	L	0.3218	0.2547	0.0670
Moncada, Yoan	R	0.3442	0.2773	0.0668
Nimmo, Brandon	L	0.3109	0.2445	0.0664

Here is list of the top 15 batters where the shift is most ineffective. Notice how these batters are exclusively right handed batters.

batter_name	stands	xBABIP_no_shift	xBABIP_shift	xBABIP_diff
Pina, Manny	R	0.2472	0.2657	-0.0186
Phegley, Josh	R	0.2925	0.3097	-0.0172
Healy, Ryon	R	0.2512	0.2649	-0.0137
Diaz, Aledmys	R	0.2525	0.2661	-0.0136
Severino, Pedro	R	0.2919	0.3037	-0.0118

batter_name	stands	xBABIP_no_shift	xBABIP_shift	xBABIP_diff
Sanchez, Gary	R	0.2063	0.2172	-0.0108
Duvall, Adam	R	0.2438	0.2523	-0.0084
Harrison, Josh	R	0.2682	0.2762	-0.0080
Brinson, Lewis	R	0.2472	0.2549	-0.0076
Robles, Victor	R	0.2532	0.2597	-0.0065
Suzuki, Kurt	R	0.2954	0.3006	-0.0051
Hernandez, Gorkys	R	0.2344	0.2391	-0.0048
Gattis, Evan	R	0.2663	0.2710	-0.0047
Nunez, Renato	R	0.2926	0.2965	-0.0039
Schoop, Jonathan	R	0.2459	0.2498	-0.0039

c)

Below is a table summarizing the average xBABIP for 100% shifts and 0% shifts across batter stands (R vs L). Clearly, the shift is more effective for left handed batters than right handed batters.

stands	xBABIP_no_shift	xBABIP_shift	xBABIP_diff
L	0.304	0.259	0.045
R	0.300	0.285	0.015