NYY Pitch Prediction Analysis

John F. Adamek

2022-12-18

Predictive Analysis

Goal: Give the most likely pitch type for all of the pitches in the test dataset (year 3) using information from the training dataset (years 1-2)

The goal is to predict the type of pitch from the training set by only given a numerical value associated with the pitch type and not the actual name. This will be done through a series of steps:

Step 1: Check and visualize the data.

Step 2: Prepare the data to be fitted to each of the models.

Step 3: Evaluate model performance by examining its accuracy in predicting pitch type in the testing set

Step 4: Determine the model with the highest accuracy scores to predict pitch type in the *pitchclassificationtest* data

Step 5: Make final predictions

Step 6: Check and visualize the predicted results to the original data. To see if patterns match.

Methods

Step 1: The first step is to look at and visualize the data. What are the variables in the provided dataset? The basic descriptive means of the independent variables and observations for each pitcher were displayed. Findings show that the pitchers in this dataset are likely to be right handed pitchers due to their release point (initposx) being on the third base side of the pitching rubber (Tables 2 and 4).Additionally, we can see that pitch type 9 and 10 are most likely refer to fastballs due to greater initial speed with pitch type 9 associated with a 2-seam fastball/sinker and pitch type 10 associated with a 4-seam fastball based on greater horizontal movement towards a right-handed hitter (breakx) for type 9 and lesser vertical movements downward (breakz) for type 10. Furthermore, Pitcher 3 has only 12 observations (pitches) in the *pitchclassificationtrain* set which is not an efficient sample size to train and test a model for future predictions. Therefore, I will take this in consideration when determining the model to be used for final predictions. I will test separate models for individual pitchers and the total model performance for addressing Pitcher 3 and Pitcher 6. As expected, the correlation matrix show's significant ($p < .05$) correlations amongst independent variables ruling out regression based models such as logistic regression.

Based on the data and research question, I will fit and evaluate the performance of three machine learning classification algorithms: decision tree (DT), k-nearest neighbor (K-NN), and support vector machine (SVM).

Table 2: Basic Means of Variables

type	mph	spin	breakx	breakz	initx	initz	ext
2	76.985	2512.840	4.892	-6.841	-1.772	5.952	6.193

3	82.459	1867.164	1.059	-0.033	-1.383	5.754	6.207
4	83.821	1364.294	-5.607	3.453	-1.744	5.895	6.196
7	84.628	988.921	-2.907	2.479	-1.023	5.993	6.206
8	88.903	2346.131	1.233	4.711	-1.815	5.813	6.205
9	91.151	2065.167	-7.126	6.907	-1.828	5.856	6.204
10	92.141	2131.526	-3.185	9.447	-1.627	5.942	6.195

Table 3: Total Observations(pitches) for each Pitcher in Training Set

pitcherid	N
1	1049
2	2137
3	12
4	1840
5	5609

Note Pitcher 3 has n=12 observations

Table 4: Means of Variables for Individual Pitcher by Type

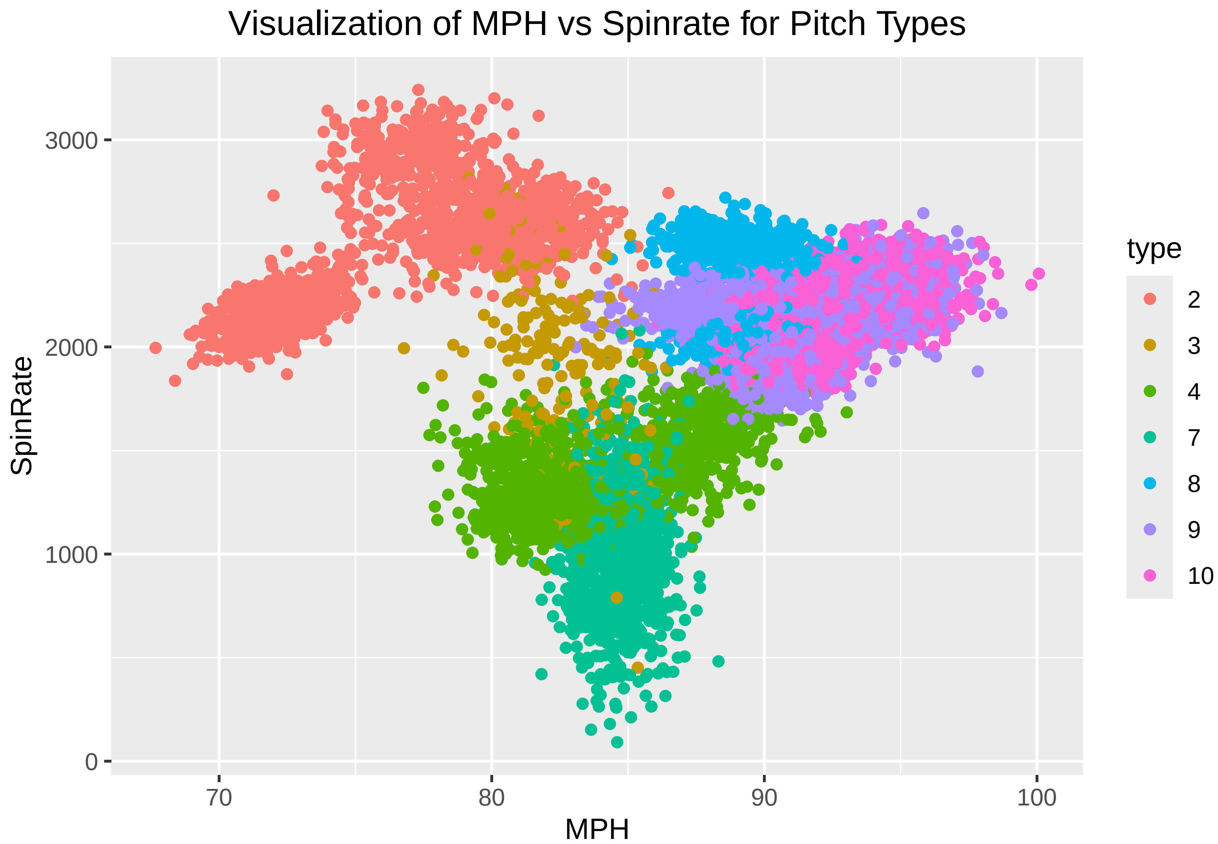
Pitcher	type	mph	spin	breakx	breakz	initx	initz	ext	pitches
Pitcher1	2	77.185	2951.308	5.829	-6.466	-1.854	6.397	6.183	257
	4	81.256	1432.336	-6.688	0.901	-1.838	6.413	6.198	255
	9	88.111	2206.983	-8.489	3.433	-2.062	6.275	6.186	421
	10	89.380	2232.940	-5.992	7.134	-1.886	6.405	6.194	116
Pitcher2	2	79.726	2574.145	5.550	-6.765	-2.184	5.860	6.199	505
	4	87.640	1619.410	-5.492	3.156	-2.352	5.743	6.185	257
	9	93.987	2208.271	-6.268	7.541	-2.265	5.758	6.214	614
	10	93.990	2241.333	-1.666	8.986	-2.268	5.856	6.196	761
Pitcher3	3	84.724	2044.592	-0.095	4.388	3.885	6.662	6.212	2
	9	86.873	2041.951	9.506	4.935	4.145	6.437	6.218	4
	10	87.670	2098.654	4.792	8.584	4.069	6.510	6.136	6
Pitcher4	2	81.272	2624.056	4.391	-2.727	-1.920	5.849	6.196	231
	4	87.025	1430.257	-7.692	3.305	-2.151	5.694	6.183	192
	8	88.950	2490.009	1.946	4.423	-1.990	5.846	6.210	444
	9	93.422	2309.789	-7.346	7.695	-2.076	5.860	6.192	303
	10	93.364	2336.793	-4.694	9.769	-1.968	5.923	6.196	670
Pitcher5	2	72.034	2167.257	3.957	-9.054	-1.235	5.861	6.190	490
	3	82.437	1865.416	1.070	-0.077	-1.435	5.746	6.207	203
	4	82.316	1211.610	-4.574	4.660	-1.331	5.807	6.203	626
	7	84.628	988.921	-2.907	2.479	-1.023	5.993	6.206	901
	8	88.813	2068.383	-0.143	5.268	-1.476	5.751	6.195	230
	9	90.447	1927.528	-7.096	7.429	-1.568	5.782	6.208	1610
	10	90.928	1981.326	-3.100	9.710	-1.167	5.956	6.194	1549

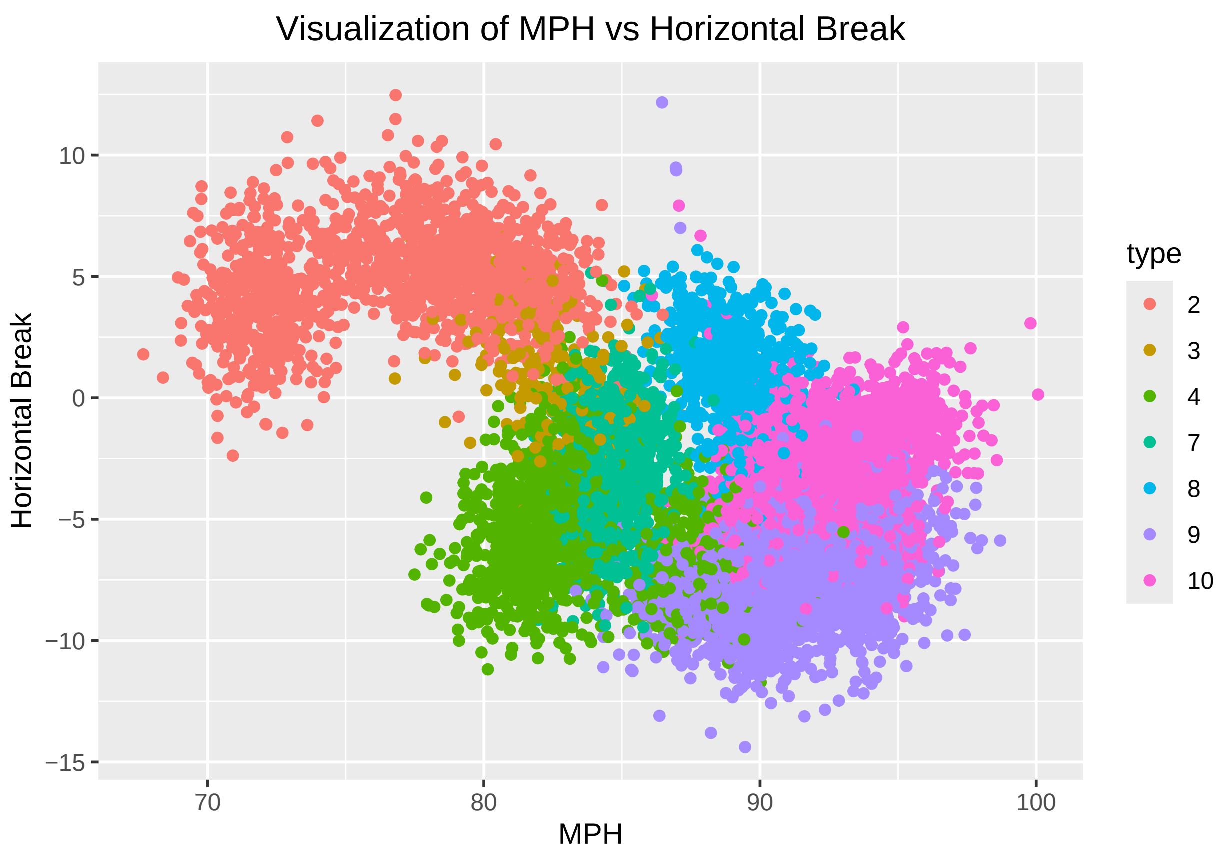
Table 1: Pitch Classification Dataset

Variables	Description
pitchid	a unique identifier for each pitch
pitcherid	identity of the pitcher (1-6)
yearid	year in which the pitch occurred (1-3)
height (in)	height in inches of the pitcher
initspeed (MPH)	initial speed of the pitch as it leaves the pitcher's hand
breakx (in)	horizontal distance where a pitch crossed the plate in relation to a hypothetical spinless pitch
breakz (in)	vertical distance where a pitch crossed the plate in relation to a hypothetical spinless pitch
initposx (ft)	horizontal position of the release point of the pitch
initposz (ft)	vertical position of the release point of the pitch
extension (ft)	distance in front of the pitching rubber the pitcher releases the ball
spinrate (RPM)	how fast the ball is spinning as it leaves the pitcher's hand
type	type of pitch that was thrown

Table 5: Correlation Matrix of Independent Variables

	initspeed	breakx	breakz	initposx	initposz	extension	spinrate
initspeed							
breakx	-0.54***						
breakz	0.87***	-0.60***					
initposx	-0.21***	0.07***	0.02*				
initposz	-0.13***	0.08***	-0.07***	0.21***			
extension	0.01	-0.01	0.01	0.01	-0.01		
spinrate	0.11***	0.37***	-0.10***	-0.45***	0.05***	-0.01	


Visualization of MPH vs Spinrate for Pitch Types
SpinRate
3000
2000
1000
0
70
80
90
100
MPH
type
2
3
4
7
8
9
10



Data Preparation

Step 2: The independent variables were first normalized to ensure the units were properly scaled. Prior to determining which algorithm to use for predicting the final pitch type, the *pitchclassificationtrain* dataset was split (75%/25%) into a training and testing set in order to evaluate model performance for the three different machine learning algorithms. The training set will be used to train each of the models which would then predict pitch type on the testing set. Model performance is evaluated based on the models ability to accurately predict the pitch type in the testing set. In addition, the training set was further separated for each of the five pitchers to run six separate models (five for each pitcher and one with data from all five pitchers) for the DT and K-NN. Models will be evaluated and compared based on their ability to accurately predict pitch type in the testing set. Because Pitcher 6 does not have any data to train on, total model performance will be used to predict pitch type for Pitcher 6 in the *pitchclassificationtest* set. Additionally, due to the limited amount of data available for Pitcher 3, I expect to use the total model performance to predict pitch type for Pitcher 3 in the *pitchclassificationtest* set as well. If accuracy for the total model is greater then accuracy for the separate models, the total model will be used to predict performance for all pitchers. Otherwise, the individual pitcher data will be used to predict that pitchers pitch type in the *pitchclassificationtest* set. For instance, if the K-NN model had a greater predicted pitch type accuracy for Pitcher 2 compared to the total K-NN model then the model for Pitcher 2 will be used to predict pitch type for Pitcher 2 in the *pitchclassificationtest* data set.

Models

Step 3: For each of the three algorithms, separate models were trained on the training set and then predictions were made on the testing set (with the dependent variable, pitch type, removed). The results from the models predictions were compared to the actual results with performance being represented by an accuracy percentage.

Decision Tree

Six separate decision tree's were created, five for each pitcher and a total model using data from all five pitchers. After training the data for each model and making predictions on the testing set, the total model performance was 84% accurate in predicting pitch type. Greater performance was found for the separate models for Pitcher 1 (93%), Pitcher 2 (94%), Pitcher 4 (87%), and Pitcher 5 (88%) with an expected low accuracy of 66.67% for Pitcher 3 (Table 5).

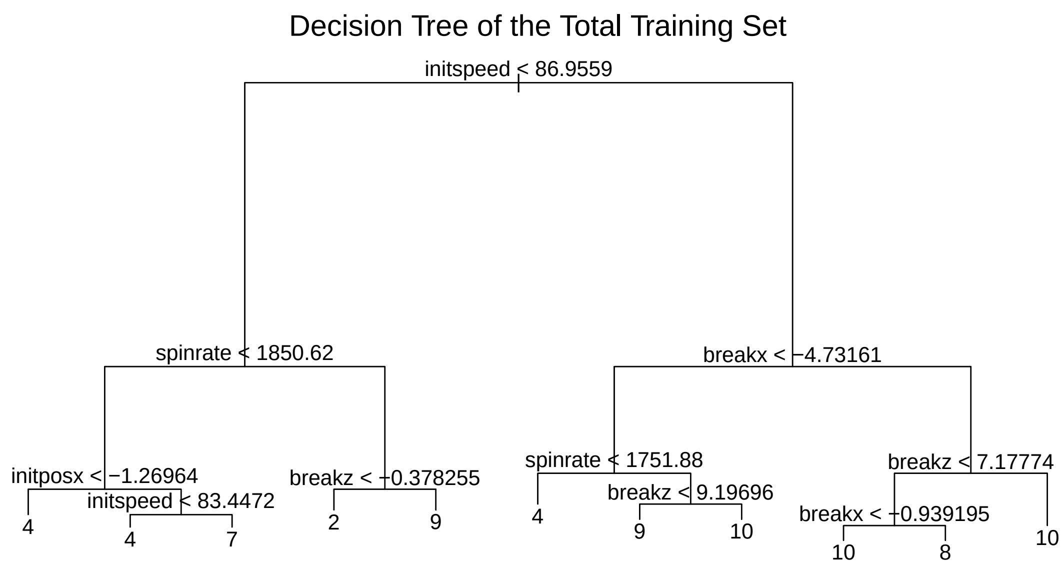


Table 6: Decision Tree Model Performance

Model	Accuracy
Total Model	0.84
Pitcher1	0.93
Pitcher2	0.94
Pitcher3	0.67
Pitcher4	0.87
Pitcher5	0.88

K-Nearest Neighbor

The same six separate model approach was used to train and test the data using K-NN. The K-NN algorithm greatly improved the predictive performance for the total model and each of the separate pitcher models (other than Pitcher 3). Total model accurately predicted 91% of the pitch type in the testing set with Pitcher 1 (96%), Pitcher 2 (96%), Pitcher 4 (93%), and Pitcher 5 (90%) all having greater accuracy then the decision tree model performance.

Table 7: K-NN Model Performance

Model	Accuracy
Total Model	0.91
Pitcher1	0.96
Pitcher2	0.95
Pitcher3	0.67
Pitcher4	0.93
Pitcher5	0.90

Support Vector Machine (SVM)

As a result of K-NN resulting in an accuracy score above 90% for each separate pitcher model, a multiclass support vector algorithm was ran on the total model to improve the models performance for predicting Pitcher 3 and Pitcher 6 in the *pitchclassificationtest* set. The SVM resulted in a slight improvement in overall model performance (92%) compared to the K-NN total model.

Final Model Results and Predictions

Step 4: The separate K-NN models for Pitcher 1, Pitcher 2, Pitcher 4, and Pitcher 5 reported accuracy scores above 90% (Table 7). Therefore, it was decided to use the total *pitchclassificationtrain* data for each of the four pitchers to train K-NN models and make final pitch type predictions for these four pitchers in the *pitchclassificationtest* data set.

SVM reported the highest predictive accuracy for the total model (92%). It was therefore decided to train SVM on the total *pitchclassificationtrain* data to make final pitch type prediction for Pitcher 6 as well as Pitcher 3 (due to low observation of training data) in the *pitchclassificationtest* data set.

Table 8: Comparing Model Performance

	Total.Model	Pitcher.1	Pitcher.2	Pitcher.3	Pitcher.4	Pitcher.5
Decision Tree Model	0.84	0.93	0.94	0.67	0.87	0.88
K-NN Model	0.91	0.96	0.95	0.67	0.93	0.90
SVM Model	0.92	NA	NA	NA	NA	NA

Table 9: Predicted Model Decision

Variables	Description
Pitcher 1	K-NN: Pitcher specific model
Pitcher 2	K-NN: Pitcher specific model
Pitcher 3	SVM: Total model
Pitcher 4	K-NN: Pitcher specific model
Pitcher 5	K-NN: Pitcher specific model
Pitcher 6	SVM: Total model

Step 5: After training K-NN on the total *pitchclassificationtrain* data for each pitcher. Final predictions were made using each of the four pitchers separate K-NN models. SVM was trained on the total *pitchclassificationtrain* data and final predictions were made for Pitcher 3 and Pitcher 6. When final predictions were made for each pitcher, the data was merged together to produce a final data set of all pitcher's with their predicted pitcher type.

Step 6: The predicted results were displayed along with the actual (i.e., *pitchclassificationtrain*) data by pitch type and pitcher to visualize if patterns match. Although it appears that velocity had decreased from years 1-2 to year 3 (91, 92 mph vs 87, 89mpg) overall patterns appears similar (e.g., pitch 7 had the overall lowest spin rate, pitch 2 the largest vertical break). Interestingly, it appears that Pitcher 3 and Pitcher 6 are both left-handed pitchers due to both having an initial release point on the first base side of the rubber. This may reduce accuracy rating due to the fact that the data was essentially training on right-handed pitchers to predict pitch type for a left-handed pitcher.

Table 10: Years 1-2

type	mph	spin	breakx	breakz	initx	initz	ext
2	76.99	2512.84	4.89	-6.84	-1.77	5.95	6.19
3	82.46	1867.16	1.06	-0.03	-1.38	5.75	6.21
4	83.82	1364.29	-5.61	3.45	-1.74	5.89	6.20
7	84.63	988.92	-2.91	2.48	-1.02	5.99	6.21
8	88.90	2346.13	1.23	4.71	-1.81	5.81	6.21
9	91.15	2065.17	-7.13	6.91	-1.83	5.86	6.20
10	92.14	2131.53	-3.19	9.45	-1.63	5.94	6.19

Table 11: Final Predictions

PredictedPitchType	mph	spin	breakx	breakz	initx	initz	ext
2	75.67	2686.30	5.32	-5.93	-1.94	6.04	6.20
3	83.63	2008.46	4.93	4.61	3.68	6.41	6.22
4	82.13	1476.81	-6.11	2.43	-1.93	6.03	6.21
7	84.31	1050.90	-1.44	2.47	-0.60	6.03	6.17
8	86.48	2274.63	3.97	6.01	1.63	6.22	6.21
9	87.52	2125.88	-3.19	5.37	-0.43	6.03	6.20
10	89.14	2223.90	-2.05	8.91	-1.06	6.01	6.20

Table 12: Actual Individual Pitcher by Pitch Type: Years 1-2

Pitcher	type	mph	spin	breakx	breakz	initx	initz	ext
Pitcher1	2	77.18	2951.31	5.83	-6.47	-1.85	6.40	6.18
	4	81.26	1432.34	-6.69	0.90	-1.84	6.41	6.20
	9	88.11	2206.98	-8.49	3.43	-2.06	6.28	6.19
	10	89.38	2232.94	-5.99	7.13	-1.89	6.41	6.19
Pitcher2	2	79.73	2574.14	5.55	-6.77	-2.18	5.86	6.20
	4	87.64	1619.41	-5.49	3.16	-2.35	5.74	6.19
	9	93.99	2208.27	-6.27	7.54	-2.26	5.76	6.21
	10	93.99	2241.33	-1.67	8.99	-2.27	5.86	6.20
Pitcher3	3	84.72	2044.59	-0.10	4.39	3.89	6.66	6.21
	9	86.87	2041.95	9.51	4.94	4.14	6.44	6.22
	10	87.67	2098.65	4.79	8.58	4.07	6.51	6.14
Pitcher4	2	81.27	2624.06	4.39	-2.73	-1.92	5.85	6.20
	4	87.02	1430.26	-7.69	3.31	-2.15	5.69	6.18
	8	88.95	2490.01	1.95	4.42	-1.99	5.85	6.21
	9	93.42	2309.79	-7.35	7.70	-2.08	5.86	6.19
	10	93.36	2336.79	-4.69	9.77	-1.97	5.92	6.20
Pitcher5	2	72.03	2167.26	3.96	-9.05	-1.24	5.86	6.19

3	82.44	1865.42	1.07	-0.08	-1.43	5.75	6.21
4	82.32	1211.61	-4.57	4.66	-1.33	5.81	6.20
7	84.63	988.92	-2.91	2.48	-1.02	5.99	6.21
8	88.81	2068.38	-0.14	5.27	-1.48	5.75	6.20
9	90.45	1927.53	-7.10	7.43	-1.57	5.78	6.21
10	90.93	1981.33	-3.10	9.71	-1.17	5.96	6.19

Table 13: Predicted Individual Pitcher by Pitch Type: Year 3

Pitcher	PredictedPitchType	mph	spin	breakx	breakz	initx	initz	ext
Pitcher1	2	76.29	2940.96	5.74	-6.35	-1.85	6.40	6.20
	4	80.56	1432.89	-6.44	0.71	-1.83	6.40	6.23
	9	86.66	2209.15	-7.17	3.24	-2.05	6.29	6.20
	10	87.47	2271.93	-4.56	6.26	-1.89	6.39	6.20
Pitcher2	2	73.48	2573.31	5.55	-6.74	-2.18	5.85	6.19
	4	81.68	1635.80	-5.51	3.40	-2.35	5.74	6.20
	9	87.44	2186.73	-6.68	7.35	-2.25	5.76	6.20
	10	87.67	2237.46	-2.35	8.76	-2.26	5.84	6.21
Pitcher3	2	84.13	2269.58	5.47	4.57	4.21	6.41	6.12
	3	83.79	2024.08	5.33	5.03	4.10	6.46	6.22
	4	85.06	1694.66	10.66	5.92	4.00	6.53	6.21
	7	84.75	1623.93	10.24	5.84	4.01	6.45	6.24
	8	85.39	2135.16	5.38	7.17	4.11	6.49	6.20
	9	83.87	2156.94	2.97	5.60	4.14	6.46	6.17
	10	85.94	2083.01	5.63	8.41	4.06	6.51	6.20
Pitcher4	2	80.47	2620.86	4.38	-2.71	-1.91	5.85	6.21
	4	86.10	1442.49	-7.76	3.63	-2.15	5.70	6.21
	8	88.03	2499.23	2.07	4.32	-1.99	5.85	6.21
	9	92.55	2298.29	-7.29	7.27	-2.04	5.86	6.19
	10	92.57	2340.87	-4.82	9.68	-1.99	5.91	6.19
Pitcher5	2	71.85	2195.99	3.96	-8.94	-1.24	5.87	6.20
	3	81.76	1825.41	0.33	-0.33	-1.31	5.78	6.20
	4	82.04	1225.32	-4.70	4.70	-1.33	5.79	6.17
	7	84.27	1002.53	-2.42	2.19	-0.98	6.00	6.17
	8	88.30	2055.71	-0.50	4.75	-1.50	5.70	6.22
	9	90.07	1943.43	-7.33	7.49	-1.60	5.76	6.22
	10	90.53	1967.73	-3.23	9.61	-1.16	5.96	6.20
Pitcher6	9	87.03	2021.60	3.21	5.51	2.09	5.97	6.20
	10	91.60	2048.80	3.13	10.87	2.17	6.10	6.10

Appendix (R code)

```
########################
#
# Input and Check Data
#
########################
```

```r
# Input Data
train <- read.csv("C:/Users/jadam/Box/Job Applications & Content/Data_Projects/pitchclassificationtrain
test <- read.csv("C:/Users/jadam/Box/Job Applications & Content/Data_Projects/pitchclassificationtest.c
train$type <- factor(train$type)
train$pitcherid <- factor(train$pitcherid) #added from KNN
# Means of current data
tablemean <- train %>%
  group_by(type) %>%
  summarise(mph = mean(initspeed),
            spin = mean(spinrate),
            breakx = mean(breakx),
            breakz = mean(breakz),
            initx = mean(initposx),
            initz = mean(initposz),
            ext = mean(extension))

# Total observations
totalobs <- train %>%
  group_by(pitcherid) %>%
  summarise(N = n())
# Descriptives of Individual SP Type
SP_type <- train %>%
  group_by(pitcherid, type) %>%
  summarise(mph = mean(initspeed),
            spin = mean(spinrate),
            breakx = mean(breakx),
            breakz = mean(breakz),
            initx = mean(initposx),
            initz = mean(initposz),
            ext = mean(extension),
            pitches = n())
SP_type$Pitcher <- c("Pitcher1", "", "", "", "Pitcher2", "", "", "", "Pitcher3", "", "", "Pitcher4", ""
SP_type <- SP_type %>%
  ungroup() %>%
  select(Pitcher,type:pitches)

# Correlations among variables
source("C:/Users/HP/Box/R_codes/correlation_matrix.R")
# Run correlation matrix
cor_matrixFR <- as.data.frame(correlation_matrix(train[, -c(1:4,12)], digits = 2, use = 'lower', replac

# Table of total means
knitr::kable(tablemean, align = "c", caption = 'Basic Means of Variables', digits = 3) %>%
  kableExtra::kable_styling(latex_options = "HOLD_position")

# Table of Total observations
knitr::kable(totalobs, align = "c", caption = 'Total Observations(pitches) for each Pitcher in Training
  kableExtra::kable_styling(latex_options = "HOLD_position") %>%
  footnote(general = 'Pitcher 3 has n=12 observations', general_title = "Note", footnote_as_chunk = T)

# Table of Each SP's descriptives
knitr::kable(SP_type, align = "c", caption = 'Means of Variables for Individual Pitcher by Type', digit
  kableExtra::kable_styling(latex_options = "HOLD_position")
```

```
# Table of Each SP's descriptives
knitr::kable(cor_matrixFR, align = "c", caption = 'Correlation Matrix of Independent Variables', digits
  kableExtra::kable_styling(latex_options = "HOLD_position")

########################
#
# VISUALIZATION GRAPHS
#
########################

#ggplot(train, aes(x=type, y=breakx, fill=type))+
#  geom_boxplot()+
#  ggtitle("Pitch Type vs Horizontal Break")+
#  xlab("Pitch Type")+
#  ylab("Horizontal Break")
# Visualize MPH vs spin rate
ggplot(train) +
  geom_point(mapping = aes(x=initspeed, y= spinrate, color = type)) +
  xlab("MPH") +
  ylab("SpinRate") +
  ggtitle("Visualization of MPH vs Spinrate for Pitch Types") +
  theme(plot.title = element_text(hjust = 0.5))

ggplot(train, aes(initspeed, breakx, color = type))+
  geom_point()+
  ggtitle("Visualization of MPH vs Horizontal Break")+
  xlab("MPH")+
  ylab("Horizontal Break") +
  theme(plot.title = element_text(hjust = 0.5))

###################
#
# DECISION TREE
#
##################

# Set up dataset for 6 different models
trainM <- train %>%
  select(initspeed:type)
SP1 <- train %>%
  filter(pitcherid == 1) %>%
  select(initspeed:type)
SP2 <- train %>%
  filter(pitcherid == 2) %>%
  select(initspeed:type)
SP3 <- train %>%
  filter(pitcherid == 3) %>%
  select(initspeed:type)
SP4 <- train %>%
  filter(pitcherid == 4) %>%
  select(initspeed:type)
SP5 <- train %>%
  filter(pitcherid == 5) %>%
```

```r
  select(initspeed:type)
# Decision Tree
treeM <- tree(type ~ ., data = trainM)
sp1D <- tree(type ~ ., data = SP1)
sp2D <- tree(type ~ ., data = SP2)
sp3D <- tree(type ~ ., data = SP3)
sp4D <- tree(type ~ ., data = SP4)
sp5D <- tree(type ~ ., data = SP5)
# Misclassifications
missclassM <- summary(treeM)[[7]][1]/summary(treeM)[[7]][2]
missclass1 <- summary(sp1D)[[7]][1]/summary(sp1D)[[7]][2]
missclass2 <- summary(sp2D)[[7]][1]/summary(sp2D)[[7]][2]
missclass3 <- summary(sp3D)[[7]][1]/summary(sp3D)[[7]][2]
missclass4 <- summary(sp4D)[[7]][1]/summary(sp4D)[[7]][2]
missclass5 <- summary(sp5D)[[7]][1]/summary(sp5D)[[7]][2]

# Model Accuracy
##Split training data into training and testing set
set.seed(27)
splitM = sample.split(trainM$type, SplitRatio = 0.75)
split1 = sample.split(SP1$type, SplitRatio = 0.75)
split2 = sample.split(SP2$type, SplitRatio = 0.75)
split3 = sample.split(SP3$type, SplitRatio = 0.75)
split4 = sample.split(SP4$type, SplitRatio = 0.75)
split5 = sample.split(SP5$type, SplitRatio = 0.75)
##Training & Test set
training_set = subset(trainM, splitM == TRUE)
test_set = subset(trainM, splitM == FALSE)
training_set1 = subset(SP1, split1 == TRUE)
test_set1 = subset(SP1, split1 == FALSE)
training_set2 = subset(SP2, split2 == TRUE)
test_set2 = subset(SP2, split2 == FALSE)
training_set3 = subset(SP3, split3 == TRUE)
test_set3 = subset(SP3, split3 == FALSE)
training_set4 = subset(SP4, split4 == TRUE)
test_set4 = subset(SP4, split4 == FALSE)
training_set5 = subset(SP5, split5 == TRUE)
test_set5 = subset(SP5, split5== FALSE)
## Training Tree
treeD_training <- tree(type ~ ., training_set)
sp1D_training <- tree(type ~ ., training_set1)
sp2D_training <- tree(type ~ ., training_set2)
sp3D_training <- tree(type ~ ., training_set3)
sp4D_training <- tree(type ~ ., training_set4)
sp5D_training <- tree(type ~ ., training_set5)
## Make predictions on the test set
tree.predM = predict(treeD_training, test_set[,-8], type="class")
tree.pred1 = predict(sp1D_training, test_set1[,-8], type="class")
tree.pred2 = predict(sp2D_training, test_set2[,-8], type="class")
tree.pred3 = predict(sp3D_training, test_set3[,-8], type="class")
tree.pred4 = predict(sp4D_training, test_set4[,-8], type="class")
tree.pred5 = predict(sp5D_training, test_set5[,-8], type="class")
##Accuracy
```

```
m <- confusionMatrix(table(tree.predM, test_set$type))$overall[1]
m1 <- confusionMatrix(table(tree.pred1, test_set1$type))$overall[1]
m2 <- confusionMatrix(table(tree.pred2, test_set2$type))$overall[1]
m3 <- confusionMatrix(table(tree.pred3, test_set3$type))$overall[1]
m4 <- confusionMatrix(table(tree.pred4, test_set4$type))$overall[1]
m5 <- confusionMatrix(table(tree.pred5, test_set5$type))$overall[1]

# Table of DT
dtmodel <- data.frame(Model = c('Total Model', 'Pitcher1', 'Pitcher2', 'Pitcher3', 'Pitcher4', 'Pitcher5
                 Accuracy = c(m,m1,m2,m3,m4,m5))

# Plot the decison Tree of the total Model
plot(treeD_training)
text(treeD_training, cex= 1.1)
mtext("Decision Tree of the Total Training Set", line = 1, cex = 1.5)

# Kable of DT
knitr::kable(dtmodel, align = "c", caption = 'Decision Tree Model Performance', digits = 2) %>%
  kableExtra::kable_styling(latex_options = "HOLD_position")

######################
#
# K-NEAREST NEIGHBOR
#
######################

# Functions
# Normalize function
normfun <- function(x){
  return((x - min(x)) / (max(x) - min(x)))
}
# PreProcess function
preprocess <- function(x){

  train.n <- as.data.frame(lapply(x[, -c(1,9)], normfun))
  train.n$type <- x$type
  # Split Train Data to test model
  set.seed(27)
  split = sample.split(train.n$type, SplitRatio = 0.75)
  training_set = subset(train.n, split == TRUE)
  test_set = subset(train.n, split == FALSE)
  return(list(training_set, test_set))
}

# Subset data for the separate models
trainM_knn <- train %>%
  select(pitcherid,initspeed:type)
SP1_knn <- train %>%
  filter(pitcherid == 1) %>%
  select(pitcherid,initspeed:type)
SP2_knn <- train %>%
  filter(pitcherid == 2) %>%
  select(pitcherid,initspeed:type)
```

```r
SP3_knn <- train %>%
  filter(pitcherid == 3) %>%
  select(pitcherid,initspeed:type)
SP4_knn <- train %>%
  filter(pitcherid == 4) %>%
  select(pitcherid,initspeed:type)
SP5_knn <- train %>%
  filter(pitcherid == 5) %>%
  select(pitcherid,initspeed:type)

##Split training data into training and testing set
# Total model
dfL <- preprocess(trainM_knn)
training_setk <-dfL[[1]]
test_setk <- dfL[[2]]
# SP1
dfL <- preprocess(SP1_knn)
training_setk1 <-dfL[[1]]
test_setk1 <- dfL[[2]]
# SP 2
dfL <- preprocess(SP2_knn)
training_setk2 <-dfL[[1]]
test_setk2 <- dfL[[2]]
# SP 3
dfL <- preprocess(SP3_knn)
training_setk3 <-dfL[[1]]
test_setk3 <- dfL[[2]]
# SP 4
dfL <- preprocess(SP4_knn)
training_setk4 <-dfL[[1]]
test_setk4 <- dfL[[2]]
# SP 5
dfL <- preprocess(SP5_knn)
training_setk5 <-dfL[[1]]
test_setk5 <- dfL[[2]]

# Build KNN Model
knn.M = knn(train = training_setk[, -8],
              test = test_setk[, -8],
              cl = training_setk[, 8],
              k = 3,
              prob = TRUE)
# SP1
knn.1 = knn(train = training_setk1[, -8],
             test = test_setk1[, -8],
             cl = training_setk1[, 8],
             k = 9,
             prob = TRUE)
# SP2
knn.2 = knn(train = training_setk2[, -8],
             test = test_setk2[, -8],
             cl = training_setk2[, 8],
             k = 5,
```

```
             prob = TRUE)
# SP3
knn.3 = knn(train = training_setk3[, -8],
             test = test_setk3[, -8],
             cl = training_setk3[, 8],
             k = 3,
             prob = TRUE)
# SP4
knn.4 = knn(train = training_setk4[, -8],
             test = test_setk4[, -8],
             cl = training_setk4[, 8],
             k = 5,
             prob = TRUE)
# SP5
knn.5 = knn(train = training_setk5[, -8],
             test = test_setk5[, -8],
             cl = training_setk5[, 8],
             k = 5,
             prob = TRUE)

# Model Evaluation
am <- confusionMatrix(table(knn.M,test_setk[, 8]))$overall[1]
m1 <- confusionMatrix(table(knn.1,test_setk1[, 8]))$overall[1]
m2 <- confusionMatrix(table(knn.2,test_setk2[, 8]))$overall[1]
m3 <- confusionMatrix(table(knn.3,test_setk3[, 8]))$overall[1]
m4 <- confusionMatrix(table(knn.4,test_setk4[, 8]))$overall[1]
m5 <- confusionMatrix(table(knn.5,test_setk5[, 8]))$overall[1]

# Table of KNN
knnmodel <- data.frame(Model = c('Total Model', 'Pitcher1', 'Pitcher2', 'Pitcher3', 'Pitcher4', 'Pitche
                 Accuracy = c(am,m1,m2,m3,m4,m5))


# Kable of KNN
knitr::kable(knnmodel, align = "c", caption = 'K-NN Model Performance', digits = 2) %>%
  kableExtra::kable_styling(latex_options = "HOLD_position")

# Testing
SP1testkn <- test %>%
  filter(pitcherid == 1)
SP2testkn <- test %>%
  filter(pitcherid == 2)
SP4testkn <- test %>%
  filter(pitcherid == 4)
SP5testkn <- test %>%
  filter(pitcherid == 5)

# Total Model (to be used for SP3 and SP6)
final_pred = knn(train = train[, -c(1:4,12)],
                 test = test[, -c(1:4,12)],
                 cl = train[, 12],
                 k = 3,
                 prob = TRUE)
```

```
testkn <- test
testkn$PitchPredKNN <- final_pred

# Pitcher 1
final_pred1 = knn(train = SP1_knn[, -c(1,9)],
             test = SP1testkn[, -c(1:4, 12)],
             cl = SP1_knn[, 9],
             k = 3,
             prob = TRUE)
SP1testkn$PitchPredKNN <- final_pred1
# Pitcher 2
final_pred2 = knn(train = SP2_knn[, -c(1,9)],
             test = SP2testkn[, -c(1:4, 12)],
             cl = SP2_knn[, 9],
             k = 3,
             prob = TRUE)
SP2testkn$PitchPredKNN <- final_pred2
# Pitcher 4
final_pred4 = knn(train = SP4_knn[, -c(1,9)],
             test = SP4testkn[, -c(1:4, 12)],
             cl = SP4_knn[, 9],
             k = 3,
             prob = TRUE)
SP4testkn$PitchPredKNN <- final_pred4
# Pitcher 5
final_pred5 = knn(train = SP5_knn[, -c(1,9)],
             test = SP5testkn[, -c(1:4, 12)],
             cl = SP5_knn[, 9],
             k = 3,
             prob = TRUE)
SP5testkn$PitchPredKNN <- final_pred5
# Pitcher 3 and 6
SP3testkn <- testkn %>%
  filter(pitcherid == 3)
# Pitcher 6
SP6testkn <- testkn %>%
  filter(pitcherid == 6)

# Merge together
final_KN <- rbind(SP1testkn, SP2testkn, SP3testkn, SP4testkn, SP5testkn, SP6testkn)

##########################
#
# SUPPORT VECTOR MACHINE
#
##########################

# Test and Train Model
## Split training data into training and testing set
splitsvm = sample.split(train$type, SplitRatio = 0.75)
training_setsvm = subset(train, splitsvm == TRUE)
test_setsvm = subset(train, splitsvm == FALSE)
## Fit the model
```

```r
svm1 = svm(formula = type ~ .,
         data = training_setsvm[, -c(1:4)],
         type = 'C-classification',
         kernel = 'radial')
## Model Evaluation: Predict on test set
y_predsvm <- predict(svm1, test_setsvm[, -c(1:4, 12)])
## Accuracy: Confusion Matrix
svm_acc <- confusionMatrix(table(y_predsvm, test_setsvm$type))$overall[1]

# Make Final Predictions
## Fit the model
svm_M = svm(formula = type ~ .,
         data = train[, -c(1:4)],
         type = 'C-classification',
         kernel = 'radial')
## Predict Pitch type on final test set
testsvm <- test
testsvm$PitchPred_svm <- predict(svm_M, test[, -c(1:4, 12)])

# Table of SVM
svmmodel <- data.frame(Model = c('Total Model'),
                 Accuracy = c(svm_acc))

###########################
#
# FINAL MODEL COMPARISON
#
##########################
# Table Comparing Models
comp <- data.frame("Total Model" = as.numeric(c(t(dtmodel)[2], t(knnmodel)[2], svmmodel[1,2])),
                 "Pitcher 1" = as.numeric(c(t(dtmodel)[4], t(knnmodel)[4], '')),
                 "Pitcher 2" = as.numeric(c(t(dtmodel)[6], t(knnmodel)[6], '')),
                 "Pitcher 3" = as.numeric(c(t(dtmodel)[8], t(knnmodel)[8], '')),
                 "Pitcher 4" = as.numeric(c(t(dtmodel)[10], t(knnmodel)[10], '')),
                 "Pitcher 5" = as.numeric(c(t(dtmodel)[12], t(knnmodel)[12], '')))

rownames(comp) <- c("Decision Tree Model", "K-NN Model", 'SVM Model')

# Kable Comparing Models
knitr::kable(comp, align = "c", caption = 'Comparing Model Performance', digits = 2) %>%
  kableExtra::kable_styling(latex_options = "HOLD_position")

#######################
#
# FINAL PREDICTIONS
#
######################
# Extract Model specific predictions
Finala <- final_KN %>%
  filter(pitcherid %in% c(1,2,3,4)) %>%
  rename('PredictedPitchType' = 'PitchPredKNN')
Finalb <- testsvm %>%
  filter(!pitcherid %in% c(1,2,3,4)) %>%
```

```r
  rename('PredictedPitchType' = 'PitchPred_svm')
# Merge together
FinalNYY <- rbind(Finala, Finalb)

#######################
#
# PREDICTION V ACTUAL
#
#######################3
# Predicted Data: Final
NYYPred_by_pitch <- FinalNYY %>%
  group_by(PredictedPitchType) %>%
  summarise(mph = mean(initspeed),
            spin = mean(spinrate),
            breakx = mean(breakx),
            breakz = mean(breakz),
            initx = mean(initposx),
            initz = mean(initposz),
            ext = mean(extension))
NYYPred_by_pitcher <- FinalNYY %>%
  group_by(pitcherid, PredictedPitchType) %>%
  summarise(mph = mean(initspeed),
            spin = mean(spinrate),
            breakx = mean(breakx),
            breakz = mean(breakz),
            initx = mean(initposx),
            initz = mean(initposz),
            ext = mean(extension))

# Format table
NYYPred_by_pitcher$Pitcher <- c("Pitcher1", "", "", "", "Pitcher2", "", "", "", "Pitcher3", "", "", "",
                                "Pitcher4", "", "", "", "", 'Pitcher5', "", "", "", "", "", "", 'Pitche
NYYPred_by_pitcher <- NYYPred_by_pitcher %>%
  ungroup() %>%
  select(Pitcher,PredictedPitchType:ext)

# Kable Model Comparisons - by pitch
knitr::kable(tablemean, align = "c", caption = 'Years 1-2', digits = 2) %>%
  kableExtra::kable_styling(latex_options = "HOLD_position")
knitr::kable(NYYPred_by_pitch, align = "c", caption = 'Final Predictions', digits = 2) %>%
  kableExtra::kable_styling(latex_options = "HOLD_position")
# Kable Model Comparisons - by pitcher
knitr::kable(SP_type[, -10], align = "c", caption = 'Actual Individual Pitcher by Pitch Type: Years 1-2
  kableExtra::kable_styling(latex_options = "HOLD_position")
knitr::kable(NYYPred_by_pitcher, align = "c", caption = 'Predicted Individual Pitcher by Pitch Type: Yea
  kableExtra::kable_styling(latex_options = "HOLD_position")

###########################
#
# Export and save FinalNYY
#
#############################
FinalNYY <- FinalNYY %>%
```

```
  select(pitchid, PredictedPitchType)
#write_csv(FinalNYY, "C:/Users/HP/Desktop/NYY/Adamek_NYYPredictions.csv")
#write_csv(FinalNYY, "C:/Users/jadam/Box/Job Applications/NYY/Adamek_NYYPredictions.csv")
```