MSBA Applied Project Report Template
Spring 2018
W.P. Carey, ASU

BASEBALL PITCH CLASSIFICATION

Presented by Team 3:
John M. Hilton
Mark Horowitz
Katherine Peterson
Krithika Ramachandran

1. Executive Summary

The era of big data is upon us, and the sports world has been leading the way, with Major League Baseball (MLB) collecting 17 petabytes of data annually about every single event that occurs on a baseball field. One particularly robust data set collected are the characteristics of every single pitch thrown in a Major League game. One thing teams and fans are both interested in, from a strategic and entertainment standpoint, is what type of pitch a pitcher is throwing. MLB currently has a method to attempt to assign intent to a particular pitch. If this process could be automated on an individual level through an unsupervised algorithm, it would have great benefit to those who follow and play the game. Not only would it reduce manual efforts that are currently undertaken during every game, but it could lead to valuable strategic insights for players and coaches. To solve this problem, we looked at several unsupervised algorithmic methods for clustering and classifying pitches based on the data collected by MLB. We utilized R to scrape data from the web, found open-source programs that could handle and manipulate high-volume data, and through much research and experimentation we arrived at a reasonably accurate model for clustering pitches. From this clustered data, we developed several insights that would be useful to those who play and coach the game at both the professional and amateur levels.

2. Background

Baseball analytics existed long before Moneyball, by author Michael Lewis, hit bookshelves in 2003, thus propelling baseball analytics into mainstream consciousness. The Society for American Baseball Research, or SABR, was founded in 1971 with the goal to focus on statistical analysis for baseball. The original definition of sabermetrics was “the search for objective knowledge about baseball" (Birnbaum). Richard Schell of SABR describes the first analytics tools used by the organization as an IBM mainframe and a database stored on punch cards kept in filing cabinets. The sabermetrics field has made many strides since its formal inception due to advanced computing and the World Wide Web. Now powerful cameras placed around the stadium capture and stream real-time data to online databases where amateur sabermetricians can develop new insights. Teams use this surplus of data to evaluate past performances, predict future outcomes, and continuously evaluate recruiting and player development strategies.

Analyzing pitches is just one field of research related to sabermetrics. We would not have public access to pitch data without two key players – a company called Sportvision and Major League Baseball Advanced Media, or MLBAM. Sportvision, recently acquired by SMT, is a broadcast company that provides real-time pictures and data for multiple professional sports, including golf, football, and baseball. For professional baseball, they place cameras around the stadium to capture player and ball movements. During a pitch, Sportvision takes roughly 20 pictures centered around the pitcher’s mound and home plate. Sportvision then sends this data to the MLBAM. MLBAM takes this data and places it on the online, public Gameday application (Fast). This application can then be used for graphics during broadcasting, real-time analytics, and reflective statistics. We accessed the data from this MLB Gameday site, specifically looking at datasets called PITCHf/x, which will be discussed in more detail in section 4: Methods.

3. Problem Statement

A MLB pitcher does not shout out the type of pitch they just threw from the pitcher's mound. Instead baseball analysts must take the known characteristics of the pitch and make a best guess as to the pitcher's intent and type of pitch. An individual pitcher's strategy and style can continuously change, but minute details of pitches are now captured and coupled with their success rate. To help develop insights on the pitches, pitches are classified into pitch types based on similarities with other pitches.

Successfully classified pitches can provide strategic insights. By joining the classification type and unique pitch characteristics to the pitch outcome, such as hit, swing, or miss, we can identify the most successful pitch types and analyze what characteristics make that pitch successful. In the hands of coaches, those success-linked pitch characteristics can potentially be enhanced in other pitchers. Baseball is a game of adjustments, and coaches can track pitch characteristics for a pitcher throughout each season to spot small, otherwise imperceptible trend changes in those characteristics, then develop batting strategies to counter those changes. Other potential applications of pitch classification include automating the many jobs devoted to scouting pitchers.

Currently MLBAM has its own algorithm and pitch classification method. The MLBAM classification is sufficient for understanding the general idea or type of pitch, but it is less useful for understanding the differences in pitches thrown by an individual pitcher. One pitcher could have a different curveball style than another pitcher, and it is important to know an individual's strengths, weaknesses, and strategy related to pitch types in his repertoire.

We will be building classification models that are focused on an individual and then league-wide level. We will then use those classification models to identify characteristics of successful pitches within that classification. An assumption we have is our classification models will be used for post-game evaluation and strategy rather than real-time analytics.

4. Methods

In this section, we will explore the data source, how to access the data, initial methods used to classify pitches, and final models built.

4.1 The Data - PITCHf/x

As mentioned in the Background section, MLB Gameday compiles pitch information captured by Sportvision cameras and and tracked in the PITCHf/x system. This PITCHf/x data has information on pitches going back to the 2006 season, with 95% of pitches captured by 2008 (Fast). The MLB Gameday website hosts this information in Extensible Markup Language (XML) format. Due to the size of the data, it is not feasible to download each file separately. To access the data, we used a package in the software R called pitchRx. PitchRx is an easy way to pull the data directly from the MLB Gameday source, and we scraped all PITCHf/x data going back to the 2008 season.

After scraping the data, R will return five data frames, or stored tables. Please see below table for name and summary of data stored in each data frame:

Table 1: PitchRx Dataframes

Name	**Example of Data Nodes**

action	event; pitch; player; inning; inning_side; score; event_num
atbat	pitcher; batter; score; home_team_runs; away_team_runs; event_num
pitch	type; start_speed; end_speed; break_angle; pitch_type; spin_rate; event_num
po	inning; inning_side; event_num
runner	id; start; end; score; inning; inning_side; event_num

Since it is difficult for R to store a large amount of data and run analysis in the same session, we needed to store the information on a database. We downloaded SQLite applications and then created SQLite databases from our scraped PITCHf/x data. We selected SQLite as our database application because it is a lightweight, disk-based database that does not require a separate server process when accessing the database. We then fetched data from this structured database for all R functions and analysis moving forward. See "Appendix 1 - Extracting Data and Creating Database" for details to extract data and store in SQLite.

4.2 Unsupervised Classification Models

A pitch's classification is considered unlabeled because there is no prior knowledge of what the classification should be and it cannot be measured or observed in space. Instead, we must take the characteristics of the pitch after it is thrown and classify it with pitches that have similar characteristics. In order to classify these unlabeled pitches, we used unsupervised classification models. The models are unsupervised, as opposed to supervised, because the classification is chosen without any knowledge of what type the pitch should be.

We focused on three types of unsupervised models: k-means, hierarchical, and model-based. These models are considered clustering models. Clustering is a technique for finding similarities between groups of observations.The groups of similar observations are then called "clusters" and the objective is to separate the observations from other clusters.

Each of the chosen unsupervised models have strengths and weaknesses. K-means and model-based are both fast to run and adjust requirements, but the analyst must specify the number of clusters for k-means and model-based only provides a probability of a pitch belonging to a cluster. Hierarchical is accurate, but is time and space consuming. While these models will be discussed in more detail in subsequent sections, see below table for a summary of the three models:

Table 2: Summary of Unsupervised Classification Models

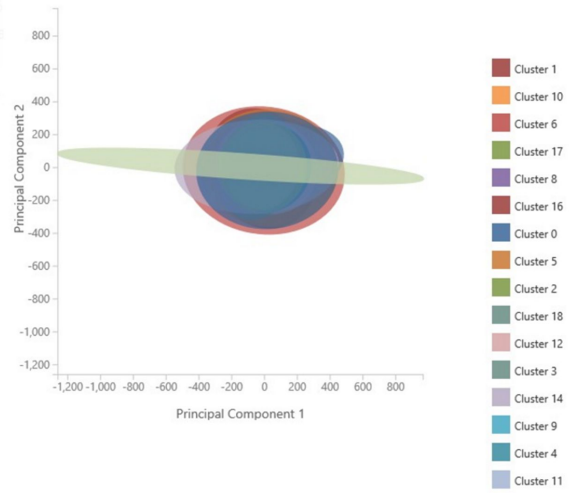
Clustering Model	K-Means	Hierarchical	Model-Based
Summary	• Continually adjusts cluster centroid	• Pairs of clusters • Clusters merged until all become one big cluster	• Normal distribution • Assumes that data comes from different sources
Strengths	• Fast • Uses hard assignment	• Accurate • Uses hard assignment	• Do not need to specify number of clusters • Fast
Weaknesses	• User specified clusters • May result in less than optimal number of clusters	• Time and space complexity	• Uses soft assignment • Only probability of belonging to a cluster is determined

4.4 Original Approach

Initial data was brought into Tableau to better understand the relationships between variables and their impact on pitch classification. MLBs pitch assignments were used as a guide in this initial evaluation phase. Through this visualization process in Tableau we determined that the variables that appear to correlate to pitch class were start speed, horizontal movement, vertical movement, pitch break, break angle, break length, and spin rate (Appendix 2). We used these identified dimensions for models moving forward.

We first focused on k-means clustering because it is an unsupervised learning method, with an iterative algorithm that puts data into clusters and continually adjusts the cluster centroid based on the average of the values in the cluster. We utilized Microsoft Azure ML, due to our familiarity with the tool from a prior class, to create the k-means clustering model. We input all data from 2008 through 2016 in to Azure ML and picked the number of clusters based on the number of MLB pitch classification. The Azure ML results were less than ideal, with no clear clusters, limited insights gained, and too many clusters.

Graph 1: Azure ML Clusters



4.5 Fresh Start - Cleaning the Data

Due to the unclear results from using all data, for all years, and for all pitchers, we decided to refine the data the model considered. We needed to focus on a smaller set of data to control for variation. The focus shifted from all pitchers to one pitcher at a time to prevent run errors due to data size and because different pitchers have different styles and techniques. Since we were developing different models, we wanted to make sure we used the same data and could compare model results side by side. We needed to select an initial pitcher that had a high volume of pitches and had multiple pitch types in their repertoire. We pulled all pitchers going back to 2008 and looked for a pitcher that had 10% of their total pitches in most of the MLB classification types. We selected Ian Kennedy as our first pitcher to test the validity of our models.

In addition to identifying a pitcher, we needed to clear up missing values before running models. We only chose regular season games due to high proportion of missing values in non-regular season games. For missing values, we replaced missing values with the variable's median. We chose the median so that

outliers would have less impact during the unsupervised classification models. The data was further limited to the home stadium of the identified pitcher to account for variability in the data collected from the Sportvision cameras because different stadiums have the cameras located as different heights and angles. “Appendix 3 - Data Clean-Up” provides more details in the data clean-up process. We used the now cleaned-up Ian Kennedy, regular season, and home stadium data for building and comparing our models.

4.6 K-Means Clustering

K-means clustering is an unsupervised learning method that clusters observations based on similarities across the variables. The analyst must specify the number of clusters, and then the model will randomly assigns the observations to the clusters. The model continues to adjust the observations by reassigning the observations to the cluster with the closest centroid until the within cluster variation is no longer declining (Kodali). The goal of k-means clustering is to minimize the within-cluster sum of square (WSS) (STHDA 2).

4.6.1 Determining Optimal Number of Clusters

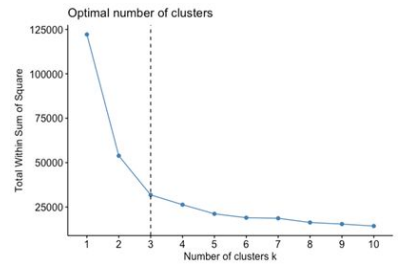
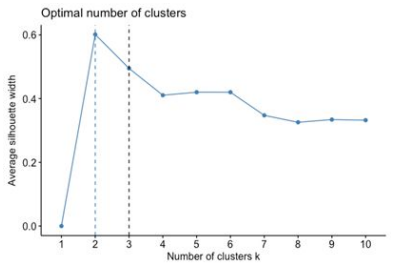
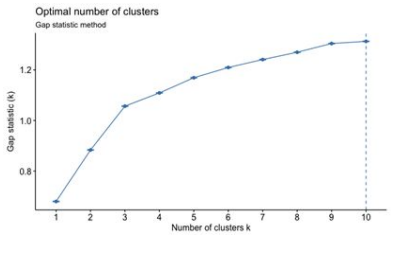
Since the analyst must specify the number of clusters the model should assign observations to, we first needed to decide the optimal number of clusters for the data. We used three methods to find the optimal numbers of clusters before running our k-means model: elbow, average silhouette, and gap statistic.

Elbow method calculates total WSS as a function of number of clusters.The location of a bend (knee) in the plot is generally considered as an indicator of the appropriate number of clusters. For the Ian Kennedy data, the elbow method recommended three clusters.

Average silhouette determines how well each object lies within its cluster by computing the average silhouette of observations for different numbers of clusters. The goal is to maximize the average silhouette. For the Ian Kennedy data, the average silhouette method recommended two or three clusters.

Gap statistic compared the total intra-cluster variable for different numbers of clusters, with a distribution that has no obvious clustering. The bend in the graph, or elbow, recommends the optimal number of clusters. For the Ian Kennedy data, the gap statistic method recommended three or ten.

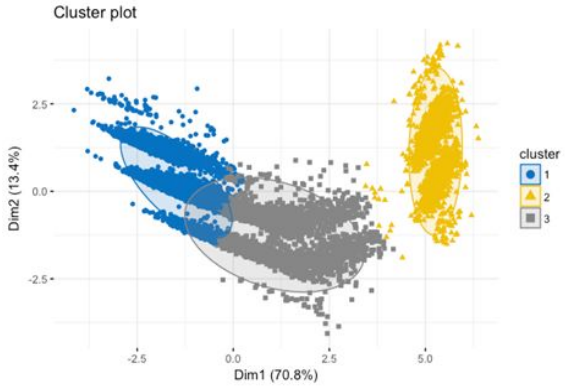
Table 4: Optimal Number of Cluster Results

Elbow Method	Silhouette Method	Gap Statistic
		

4.6.2 Running the K-Means Model

We used eclust() method in R, with three clusters, to perform cluster analysis. The function eclust() computes automatically the gap statistic for estimating the right number of clusters. The model output the following graph, with clusters appearing fairly separated:

Graph 2: Eclust() - K-Means Clustering Model Results



4.6.3 K-Means Cluster Validity

To test the model validity of our k-means model, we created a cross-table between the three clusters and the MLB identified pitch types. The table indicates that Cluster 1 has the majority of fastballs, bost four-seam and two-seam, Cluster 2 has a large number of knuckle curves and regular curve ball, nad Cluster 3 has a large number of change-ups. The k-means clusters seem to fairly separate and define the MLB classifications of the pitches.

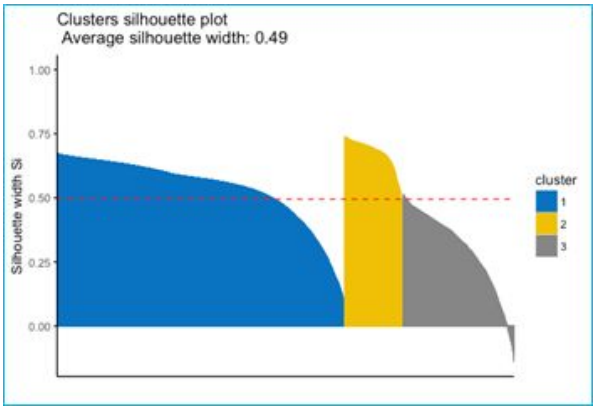
Table 5: Cross-Table on K-Means Clusters and MLB Pitch Type

```
> table(count_Ian_top$cluster, count_Ian_top$pitch_type)

     CH   CU   FC   FF   FT   IN   KC   PO   SL
  1 350    0   23 7503 1720    3    0    3    4
  2   0  167    0    0    0    0 1697    0   90
  3 2158    0  254  332   52   40    5    0  851
```

To continue to validate the cluster, silhouette width of the formed clusters was plotted on the silhouette plot. On a silhouette width close to one indicates that the observations are well clustered, whereas width close to negative one indicates that the observation is poorly clustered (STHDA 1). In the below silhouette plot, we see that Cluster 3 has some observations with negative silhouette width, indicating that these observations are poorly clustered:

Graph 3: Silhouette Plot for K-Means



Overall, when evaluating the k-means clusters, they appear to be fairly separated, have high between cluster variation, and, with the exception of some observations in Cluster 3, have silhouette widths higher than zero. See "Appendix 4 - K-Means Eclust() Model" for instructions on how to replicate the optimal number of clusters models, the eclust() model, and k-means validation results.

4.7 Hierarchical Clustering

Hierarchical clustering was evaluated as a possible alternative to k-means clustering for identifying "natural" clusters in existing data. A Hierarchical clustering process builds a tree called a dendrogram that represents clusters and the observations they contain. Unlike k-means clustering, Hierarchical clustering can be performed without specifying the number of clusters in advance (STHDA 5).

There are two main hierarchical clustering packages in R, hclust() and eclust(). These packages have a couple of important parameters: distance measure and linkage measure. A distance measure determines how distances among observations and clusters are calculated (ie. Euclidean, Manhattan). The linkage measure specifies the method that the clustering process uses to calculate pairwise distances between observations (for example, ward.D2 or complete). The clustering method builds the tree structure by combining observations based on distance. (University of Cincinnati).

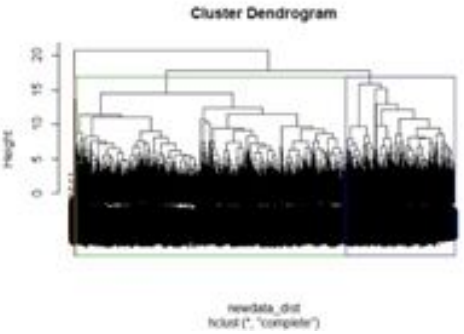
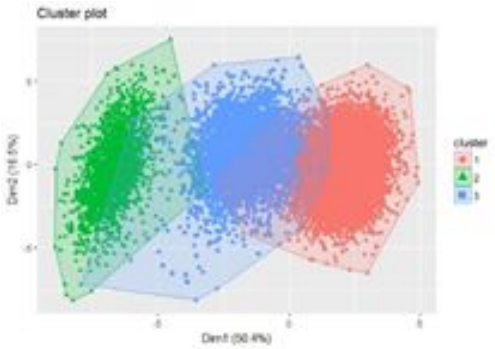
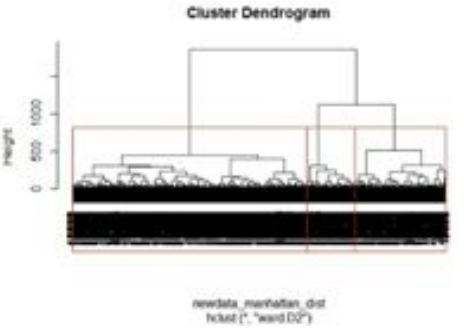
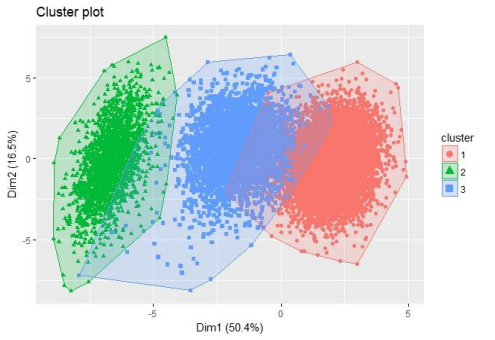
4.7.1 Running the Hierarchical Models

Four combinations of distance and linkage methods were evaluated: Euclidean-Ward.D2, Euclidean-Complete, Manhattan-Ward.D2, and Manhattan-Complete:

- Euclidean distance between two points (x1, y1) and (x2, y2) is the square root of (x1 - x2)^2 plus (y1 - y2) ^2 squared.
- Manhattan distance is the absolute value of (x1 - x2) plus the absolute value of (y1 - y2).
- Complete computes the distance for every observation in cluster A in cluster B and uses the largest distance as the difference between cluster A and cluster B, resulting in more compact clusters.
- Ward.D2 minimizes the total variance among observations in a cluster each time it combines observations into the same cluster. (STHDA 3)

We visualized the models using dendrograms and cluster plots. Scatterplots provide an easy way to evaluate the separation of clusters created by the clustering process. See the dendrogram and cluster plot for Euclidean-Complete and Manhattan-Ward.D2 below. The visualizations for the other two models can be found in "Appendix 5 - Hierarchical Models and Validation."

Table 5: Hierarchical Models with Dendrogram and Cluster Plot

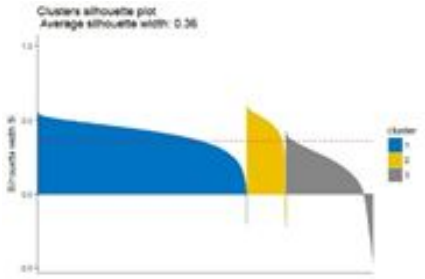
Euclidean-Complete		
Manhattan-Ward.2		

The Euclidean-Ward and Manhattan-Ward models created the simplest cluster structures based on tree height. The lower tree height for these models indicated smaller overall sum of squares errors and reduces the amount of processing time and memory needed to get access to clustering data. The Euclidean-Complete and Manhattan-Ward models exhibited the best cluster separation of the four models. However, the results indicate that none of the models did an especially good job of ensuring cluster separation. The cluster statistics for all tested models indicate that the initial data set "naturally" fits best into three clusters.

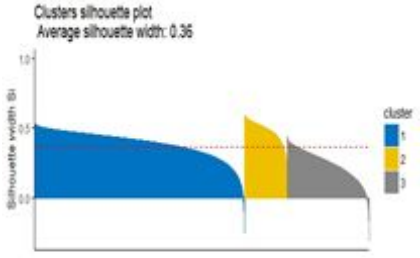
4.7.2 Hierarchical Silhouette Cluster Statistics

Similar to k-means, silhouette analysis provides a way to measure how well observations are clustered and estimate the average distance between clusters. This method can also indicate which observations have probably been placed in the wrong clusters and the clusters they should be placed in instead.

Table 6: Silhouette Analysis for Hierarchical Models



Manhattan-Complete - avg. width 0.36



Euclidean-Ward.D2 - avg. width 0.36

As mentioned in the k-means section, the silhouette width closer to one correlates with increase similarity between the observations in the cluster. When comparing the results of k-means versus every hierarchical silhouette plot, the k-means clusters have higher cohesion within their clusters with average silhouette width of 0.49. See "Appendix 5 - Hierarchical Models and Validation" for instructions on how to run the models and validate the models R.

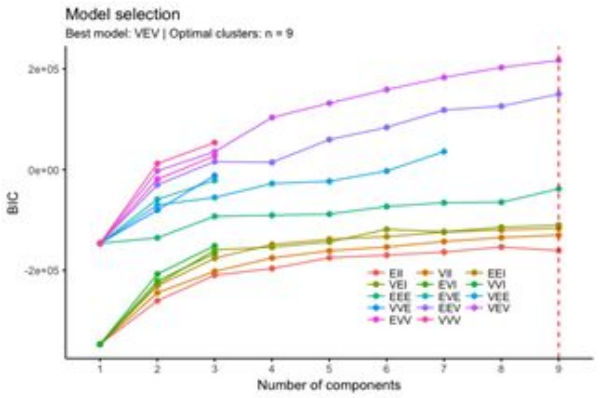
4.8 Model-Based Clustering

The final model we tried to classify pitches was model-based clustering. Model-based clustering uses Gaussian, or normal, distribution. Model-based clustering assumes that the data has more than two distributions and not just one distribution. . Unlike K-means where the user must specify the number of clusters and the model assigns the observations to those clusters, model-based models assume each observation has a probability of belonging in each cluster. Using the R package mclust, the model parameters will be estimated using an algorithm that centers the mean of each cluster and increases the density of the observations near the mean (STHDA 4).

4.8.1 Mclust Package

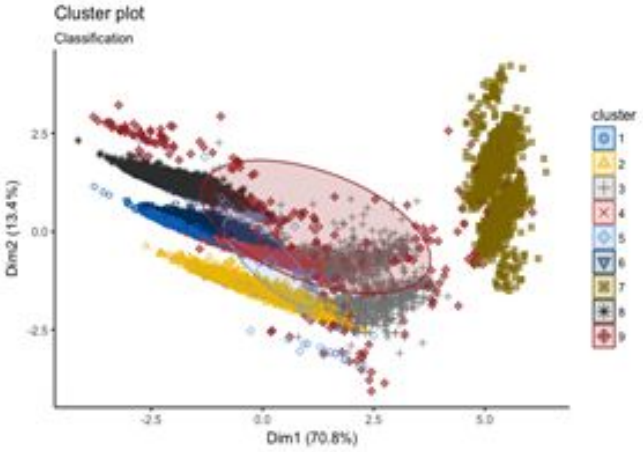
Mclust package is used to build the model-based clustering model for pitch classifications. The available model options are a combination of 3 identifiers: volume, shape, and orientation. The identifier combination consist of "E" for "equal", "V" for variable, and "I" for "coordinate axes" (STHDA 4). Running the mclust package against the Ian Kennedy data results in the following:

Graph 4: Model-Based Select



Mclust uses the Bayesian Information Criteria (BIC) to determine the best model, with a higher value correlating with a stronger model. The VEV (Variable, Equal, Variable) model with nine clusters best clustered the data. To better illustrated the VEV model, we plotted the clusters:

Graph 5: VEV Cluster Plot



The cluster plot for model-based appears more cluttered and clumsy compared to K-means and hierarchical, in part because there are nine clusters instead of three. See "Appendix 6 - Model-Based" for scripts used to create the model-based clusters.

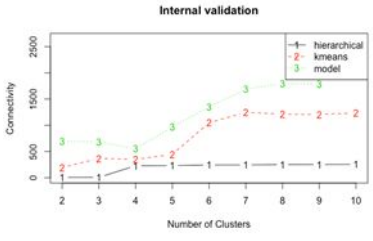
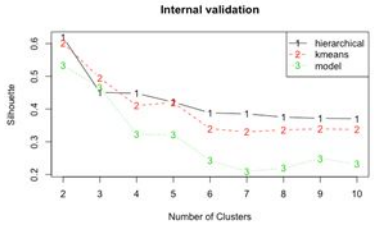
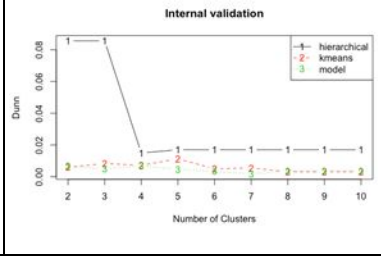
4.9 Model Validation and Selection: Choosing the Best Model

We now have three models created to cluster and evaluate Ian Kennedy's pitches: k-means, hierarchical, and model-based. In order to choose the best clustering model, the internal measures and stability of the models need to be compared to each other.

4.9.1 Internal Measures Comparison

Internal measures help compare the quality of the cluster models. Three common internal measures are connectivity, average silhouette width, and Dunn index.

Table 7: Internal Measures Comparison with Connectivity, Silhouette, and Dunn Index

Connectivity	Silhouette	Dunn Index
		

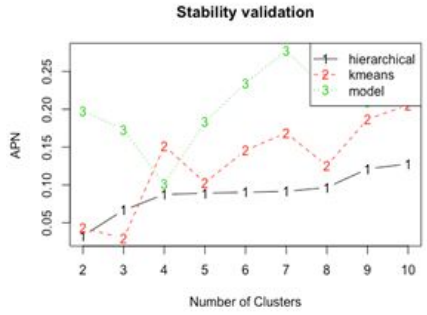
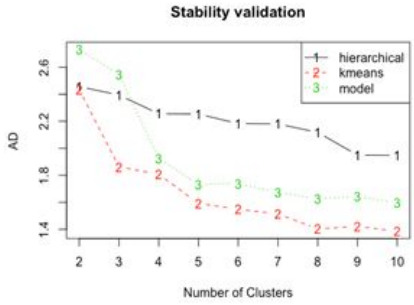
For connectivity validation, the nearest neighbors of each observation are found and sorted from nearest to farthest. The goal is to minimize the connectivity value (rdrr.io). From the connectivity internal validation plot, we see that hierarchical model has the lowest connectivity value. As mentioned before,

silhouette width should be closer to one to representent well clustered observations. Hierarchical has a high silhouette value with two clusters. Dunn index looks at the minimized distance between observations not in the same cluster, and then compares this distance to the largest within cluster distance. The Dunn index should be maximized (RDocumentation). Again, the hierarchical model shows strong internal measures when looking at the Dunn index.

4.9.2 Stability Comparison

Stability measures look at how the model changes when columns of observations are removed one at a time.Two of the stability measures are average proportion of non-overlap (APN) and average distance (AD). R package clValid was used to compare the k-means, hierarchical, and model-based models.

Table 8: Stability Comparison with APN and AD

APN	AD
	

APN represents the proportion of observations changing clusters each time a column of data is removed, with the goal to minimize the APN value to show a highly consistent cluster. The k-means model with three clusters has the most consistent cluster when viewing the APN value. AD measures the average distance between observations within a same cluster and how that distance changes when a columns of data is removed. Again, the goal is to minimize the AD within clusters (Brock, et al.). While the AD value continues to decrease at the number of clusters increases, there is a bend, or elbow, at three clusters for k-means, indicating an appropriate model with strong stability. See "Appendix 7 - Model Validation and Selection" for script used for internal and stability measures.

4.9.3 Recommended Clustering Model

Based on the stability and internal measures validation, the k-means model with three clusters is a strong model to classify the pitches for Ian Kennedy. Hierarchical fares well as far as internal measures are concerned, but it is only usable for small datasets because hierarchical clustering suffers from time and space complexity.

While this particular model was used to cluster Ian Kennedy pitches, the same processes could be used to classify the pitches of other players. First, clean the data by selecting only regular season and home stadium games. Then evaluate the appropriate number of clusters using elbow, silhouette, and gap statistic methods. With the identified optimal number of clusters, use the eclust package to create a k-means model. Finally, evaluate the model's internal and stability measures. These k-means models customized to

individual pitchers isolate the type of pitches a pitcher has in their repertoire, where additional analysis can then be done to understand what makes a certain pitch type successful for that pitcher.

5. Recommendation and Conclusions

With pitches effectively clustered, we can begin to turn our attention to some of the many practical applications of the clustered data. By joining the cluster and unique pitch variables to the pitch outcome, we can gain tremendous insights into the specific variables that correlate most closely to a successful pitch. Since the variables being clustering around are real, measurable values such as the amount of spin and movement on the ball, these insights can be put in the hands of coaches at all levels so they can develop methods for developing the characteristics linked to pitch success. The potential for insights from this data is limited only by the number of questions that can potentially be asked of it, but we'll focus here on three questions posed of the data whose answers would be extremely valuable to professional players and coaches.

5.1 What pitch characteristics make a fastball effective?

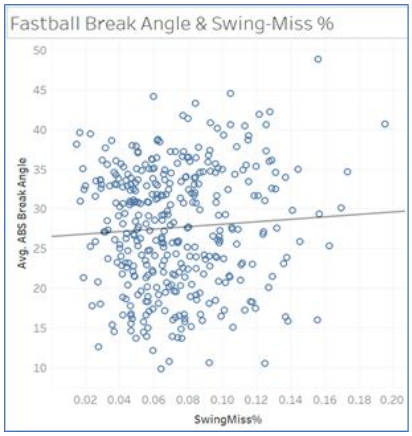
To get at this question, we first need to define how we can tell whether any pitch is effective. There are many possible definitions for this used by sabermetricians, but for our purposes here we chose a fairly straightforward measure - swing-and-miss percentage. This is essentially the number of times a batter swung at a particular pitch and did not make contact – the higher this percentage, the more effective we can assume the pitch to be.

To help derive insights from the data, all clustered pitch data using model-based function from 2016 was loaded to Tableau Public, and a workbook with various sheets was created (Appendix 10 - Insights from Baseball Clusters). The first views of the data looked at the same variables useful for clustering and check to see if those variables correlate to a higher swing-and-miss percentage. These initial views were limited to pitches in the clusters that most closely align to MLB's fastball designation from our model-based cluster (clusters 6 and 8). The example on the left looks at correlation between break angle and swing-and-miss percentage and does not find significant correlation. The example on the right does show correlation between pitch speed and swing-and-miss percentage.

Table 9: Correlation of Pitch Features to Swing-and-Miss %

Measure	Break Angle	Start Speed	Break Length	Spin Rate
R-Squared	0.004	0.204	0.12	0.12
P-Value	0.225	<.001	<.001	0.007

Graph 6: Visualize Fastball Break Angle and Start Speed Correlation with Swing-Miss



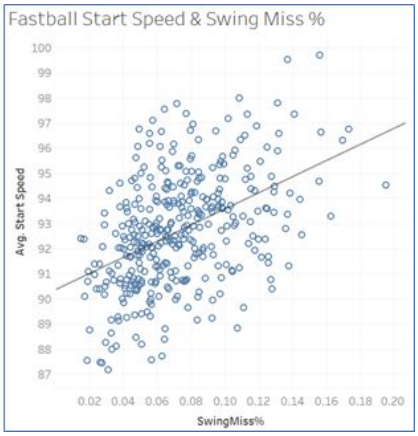
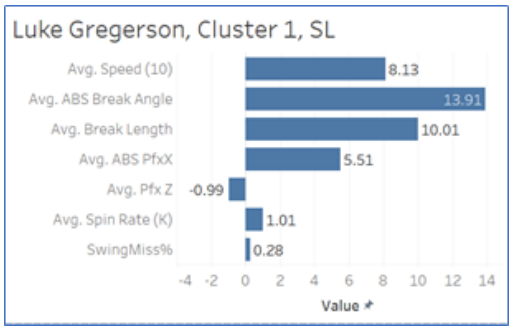


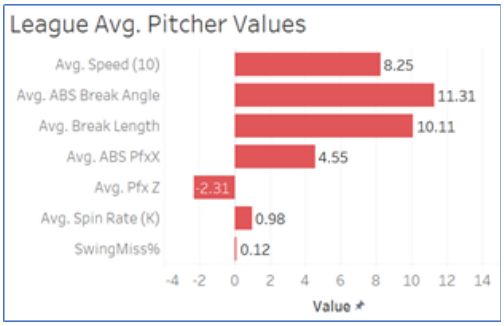
Table 9 summarizes a variable used for clustering and its correlation to swing-and-miss percentage. The r-squared and p-values that start speed is by far the best predictor of swing-and-miss percentage, with break length and spin rate showing some possible correlation, and virtually no correlation to break angle. The insight garnered from this data is that velocity is the biggest contributor to the success of a fastball, and players and those responsible for their development would be well-served to undertake training programs that enhance velocity. A particularly interesting finding is that, in addition to having some correlation to swing-and-miss percentage, spin rate is also positively correlated to speed. This means that if pitchers want to enhance their velocity they should look for ways to increase the amount of spin they put on the baseball (presumably backspin). Coaches could potentially work with players on methods to enhance spin, through either grip changes or arm/wrist/finger action during the delivery.

5.2 What was the most effective pitch thrown by a pitcher in 2016, and what made it effective?

To start investigating this question, we begin with the same measure used to evaluate fastballs, swing-and-miss percentage. The data showed that the most effective pitch in baseball in 2016, with a swing-and-miss percentage of 28%, were pitches from Luke Gregerson in cluster #1. A comparison of clusters to MLB pitch types (below) shows that cluster #1 is likely a fairly pure slider cluster. Luke Gregerson's slider was the most effective pitch in baseball, but we needed to determine what features made the pitch successful. Comparing these features to those of the league average for similar clustered pitches allows us to do so. The graphs below compare Luke Gregerson's cluster 1 slider values to those of the league average for speed, break angle, break length, horizontal movement (PfxX), vertical movement (PfxZ), and spin rate. We can see that Luke Gregerson's slider has a significantly higher break angle and horizontal movement (PfxX) than sliders in the rest of the league. Speed, break length, and spin rate are similar to the league average, and Luke Gregerson's slider actually has less vertical downward movement (drop) than the average slider in the league.

Graph 7: Comparing Luke Gregerson's Slider





The takeaway here is that horizontal movement could be the key component to Luke Gregerson's slider success, and pitchers and coaches can look for grip techniques and practice methods that enhance horizontal movement in developing this pitch.

5.3 Why would a similar pitch get grouped into two different clusters, and what can we learn from this?

When manual pitch type is lined up against clustering, many pitchers' pitches group into very clean clusters. Some, however, do not. Is this an error in clustering, or can it give us valuable insight into a pitchers' in-game strategy? As an example, we'll next look at Diamondbacks pitcher Zack Greinke's slider (last row of the chart below). Rather than grouping cleanly into one cluster, Greinke's slider groups into multiple clusters, primarily clusters 1 and 4.

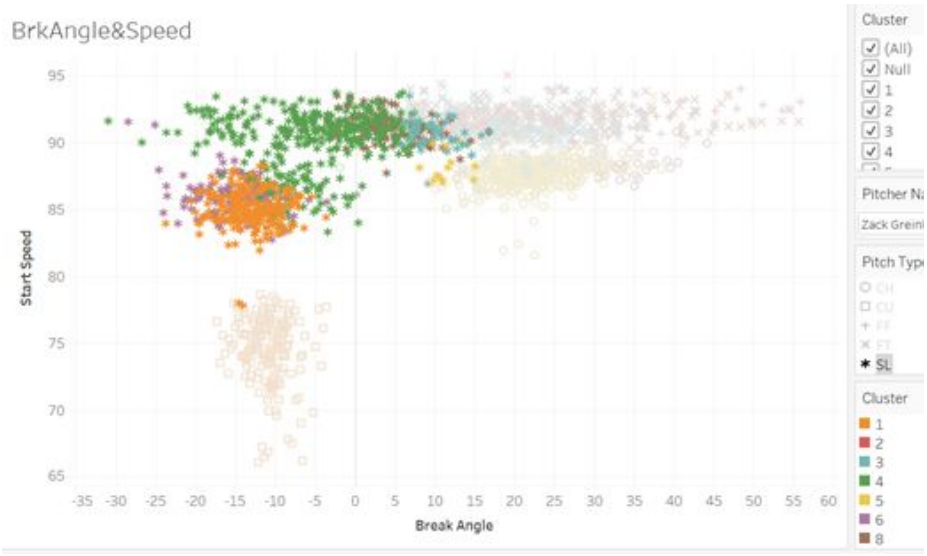
Table 10: Cluster & PitchType

Cluster&PitchType-Indv

	Cluster							
Pitch T..	1	2	3	4	5	6	8	9
CH			1	1	263		41	2
CU	171							2
FF		12	1	3			62	1
FT		37	139	6	109		307	2
SL	219		44	348	12	136	116	5

The scatter plot below shows all Greinke's pitches in 2016, arrayed by their start speed and break angle. Highlighted are those identified by MLB's manual classification process as sliders. The orange plots are sliders grouping to cluster 1 and the green plots are sliders grouping to cluster 4. What we seem to be seeing here are two variants of the same pitch. The more commonly thrown slider (green, cluster 4) generally seems to be thrown at a higher velocity and with less break. The slightly less common slider (orange, cluster 1) is thrown at a slower velocity with more break.

Graph 8: Break Angle and Speed for Greinke



This example highlights one of the great benefits of clustering, which is that it allows us to see past a simple view of how a pitcher holds and releases a particular pitch and gives insight into the true intent of that pitch. By breaking these pitches into clusters, one can get insight into the mind and strategy of a pitcher, and how he attacks hitters, all in a matter of seconds after pitch data is clustered. Further analysis can even show the specific situations where each pitch type is thrown, gaining further insight into strategy with very little manual effort. Such information is invaluable to MLB hitters and hitting coaches, as they prepare their players to counter the strategies of each pitcher they face.

5.4 Conclusion

While there is likely no perfect algorithm for capturing a human being's intent in throwing a pitch, we have developed a model that does a reasonably accurate job clustering pitches in a way that allows pitch type to be inferred. This clustered data allows for the development of insights into pitch characteristics that can be used by players and coaches to enhance their performance from a training perspective. It also allows for insight into in-game strategies employed by pitchers, which can then be used by players and coaches to develop counter-strategies. Once data is effectively clustered as we have done, there is almost no limit to the amount of insights that can be developed, as each stone overturned contains meaningful data. Major League teams would be well served to analyze this data consistently throughout each season to develop their own insights on every pitcher on their own team and every pitcher they face.

6. References

Birnbaum, Phil. "A Guide to Sabermetric Research." Society for American Baseball Research. <https://sabr.org/sabermetrics>

Brock, Guy, Datta, Somnath, Datta, Susmita, Pihur, Vasyl. "clValid: An R Package for Cluster Validation." Validation measures. Journal of Statistical Software. Mar. 2008. <https://www.jstatsoft.org/article/view/v025i04>

Fast, Mike. "What the Heck is PITCHf/x." The Hardball Times Baseball Annual 2010. <http://baseball.physics.illinois.edu/FastPFXGuide.pdf>

Kodali, Teja. “K Means Clustering in R.” R-bloggers. 28 Dec. 2015. <https://www.r-bloggers.com/k-means-clustering-in-r/>
QuickR. “Cluster Analysis.” <https://www.statmethods.net/advstats/cluster.html>
RDocumentation.”Dist.” <https://www.rdocumentation.org/packages/stats/versions/3.4.3/topics/dist>
RDocumentation. “Dunn.” <https://rdrr.io/cran/clv/man/connectivity.html>
rdrr.io. “Connectivity: Connectivity Index - Internal Measure.” <https://rdrr.io/cran/clv/man/connectivity.html>
Schell, Richard. “SABR, Baseball Statistics, and Computer: The Last Forty Years.” Society for American Baseball Research. <https://sabr.org/research/sabr-baseball-statistics-and-computing-last-forty-years>
Statistical Tools For High-Throughput Data Analysis (STHDA 1). “Cluster Validation Statistics: Must Know Methods.” <http://www.sthda.com/english/articles/29-cluster-validation-essentials/97-cluster-validation-statistics-must-know-methods/>
Statistical Tools For High-Throughput Data Analysis (STHDA 2). “Determining the optimal number of clusters: 3 must known methods - Unsupervised Machine Learning.” <http://www.sthda.com/english/wiki/print.php?id=239>
Statistical Tools for High-Throughput Data Analysis (STHDA 3), “Hierarchical Clustering Essentials - Unsupervised Machine Learning.” <http://www.sthda.com/english/wiki/print.php?id=237>
Statistical Tools for High-Throughput Data Analysis (STHDA 4). “Model Based Clustering Essentials.” <http://www.sthda.com/english/articles/30-advanced-clustering/104-model-based-clustering-essentials/>
Statistical Tools For High-Throughput Data Analysis (STHDA 5). “Types of Clustering Methods: Overview and Quick Start R Code.” <http://www.sthda.com/english/articles/25-cluster-analysis-in-r-practical-guide/111-types-of-clustering-methods-overview-and-quick-start-r-code/#hierarchical-clustering>
University of Cincinnati. “UC Business Analytics R Programming Guide.” Hierarchical Cluster Analysis. <https://uc-r.github.io/hc_clustering>

7. Appendix – Reproduction of Results

Additional details and script for most Appendices found in Team File Exchange on Blackboard.

Appendix 1: Extracting Data and Creating Database
Summary: Tools, instructions, and scripts used to extract and store data used to classify MLB pitches.
Tools Used: RStudio, SQLite
File Exchange: Appendix 1: Extracting Data and Creating Database

1. In R, use pitchRx to scrape PITCHf/x data
2. Create database in SQLite to store data frames as tables

Appendix 2: Visualizing Pitches
Summary: Used Tableau to identify variables that correspond with pitch type.
Tools Used: Tableau
Link: https://public.tableau.com/profile/matt.hilton#!/vizhome/2008PitchData/BrkLengthSpeed

Appendix 3: Data Clean-Up
Summary: R scripts used to clean up the PITCHf/x data, including only using home stadium, regular season, and filling missing data with medium
Tools Used: RStudio, SQLite
File Exchange: Appendix 3: Data Clean-Up

1. Collect pitcher, batter, and location data from SQLite
2. Identify Ian Kennedy data, only regular season
3. Count top venues to identify the home stadium, take only venues with more than 1000 pitches
4. Identify missing values
5. Fill in missing values with median for the variable

Appendix 4: K-Means Eclust() Model
Summary: Script used to determine number of clusters, run K-means model, and validate model.
Tools Used: RStudio
File Exchange: Appendix 4: K-Means EClust() Model

1. Determine optimal number of clusters using Elbow, Silhouette, and Gap Statistic scripts
2. Run eclust() to create the k-means model
3. Validate the model with Silhouette Width

Appendix 5: Hierarchical Models and Validation
Summary: Steps to perform and validate hierarchical models.
Tools Used: RStudio
File Exchange: Appendix 5: Hierarchical Models and Validation

1. Calculate distance between observations in with Euclidean function
2. Calculate the clusters and assignments
3. Calculate the optimal number of clusters
4. Create cluster statistics for eclust() model
5. Visualize clusters with dendrograms
6. Create cluster scatterplots to further visualize model
7. Validate model by using average silhouette width and visualize plot

Appendix 6: Model-Based
Summary: Scripts and instructions to create and validate mode-based clusters in R.
Tools Used: RStudio
File Exchange: Appendix 6 - Model-Based

1. Plot models with eclust() and BIC
2. Plot cluster and uncertainty

Appendix 7: Model Validation and Selection
Summary: Scripts to compare the k-means, hierarchical, and model-based models
Tools Used: RStudio
File Exchange: Appendix 7 - Model Validation and Selection

1. Run cValid() package for internal measures for all models
2. Run cValid() package for external measures for all models

Appendix 8: Insights from Baseball Clusters
Summary: Tableau models that show how characteristics correlate with successful pitches across different pitch types.
Tools Used: Tableau
Link: https://public.tableau.com/views/2016PitchData