TLaP - Throw Like a Pro

Baseball Pitch Multi-Classification Using Expectation Maximization and Multi-Layer Perceptron

Jeremy Greenwood
WGU ID:
C964 - Computer Science Capstone
Western Governors University
August 2020

"The game has a cleanness. If you do a good job, the numbers say so."

- Sandy Koufax (Boswell 1979)

LETTER OF TRANSMITTAL

Subject: TLaP Product Proposal

135 Developer's Lane
Small Town, RI 54321
1 (321) 654 - 9876

August 3, 2020

XYZ Sports Corporation Headquarters
123 Fancy Building Drive
Big City, MA 12345

Dear Department of New Product Development at XYZ Sports Corporation:

The area of talent development within amateur baseball is sorely lacking a means of comparison with professional level players. Professional athletes have a finite timeline to build skills that can be competitive at that level. We propose using machine-learning to provide amateurs with information that will help them develop faster and more effectively.

The market for such a product is rife with opportunity. Colleges, universities, and secondary schools spend large sums of money annually on their baseball programs. Providing them with technology that can help measure and improve player performance on an individual basis would go a long way towards gaining an even larger portion of that market.

Total funding for project development is estimated at under $30,000. With over 10 years of experience in building sports analytics technologies, we feel this is a product that can provide great return on investment.

Thank you for your time. We look forward to hearing from you soon.

Sincerely,

Jeremy Greenwood

PROJECT PROPOSAL

Regarded as America's favorite pastime, baseball has a long and storied history in this country. One of the most thrilling aspects of the game is the high-pressure duel between pitcher and hitter. Countless fans of the sport have grown up with dreams of knocking the ball out of the park to win the big game. Often regarded as the single most difficult task in all of professional sports, the elation of hitting a baseball has enchanted many youngsters. Likewise, being the person responsible for hurling the ball towards the plate in meteoric fashion has enamored the hearts of fans for just as long.

According to the Aspen Institute, more than 25 million youth played baseball in 2018. Of those, 15 million played 13 or more games in a year. NCAA college and university programs had over 36,000 students on their baseball rosters for 2020. Of all major professional sports in the US, baseball has the highest percentage that make the transition from amateur to professional, with 9.9% of those who were eligible being drafted. Of all those players, the most common position is pitcher. While the rules necessitate that only one pitcher can pitch at a time, baseball-almanac.com lists the average roster of 25 players as having 12 pitchers: 5 starting pitchers and 7 relief pitchers. The position is a deeply deliberated aspect of the modern game.

For all of those pitchers, transitioning from amateur to professional is the result of years of training and development. This requires the aid of experienced and dedicated coaching and training staff. Even so, there is no way to directly compare oneself to the professional practitioner. Metrics like ERA, K, K/9, or WHIP can not provide a meaningful method of comparison from one level of play to the next. A high school senior with an ERA of 0.5 is very impressive. But, that is a poor indicator of how that same 18-year-old would fare against a collegiate-level senior batting with a .400 average, let alone against a hall-of-fame-calibre slugger like Michael Brantley of the Houston Astros.

But, what if there was an effective way to make that comparison? What if one could analyze an amateur players pitch and directly compare it to a library of professional pitches?

TLaP will be a product intended to do just that.

PROJECT SUMMARY

TLaP will use performance data to help accelerate player development. It will achieve this by allowing the direct comparison of one player to another. Through machine-learning, the product will produce analysis of every individual pitch and identify what type of pitch was thrown – Four-seam Fastball, Two-seam Fastball, Curveball, Slider, etc. By analyzing pitchers with similar physiology and throwing mechanics and comparing them to those working at the professional level, the developing player can better identify what aspects of their game to improve.

Currently, there is no other product on the market that allows for this sort of comparison. While the TrackMan system allows one to view and share data, it does not provide any comparative analysis or basis for doing such. That is where TLaP will set itself apart. TLaP's machine-learning algorithms will be trained using professional data, aggregated from MLB's own archives. This consists of measurements from 4 years of game play: over 2.8 million pitches.

TLaP AT A GLANCE

The principle use of the product is for analysis of pitching consistency. A mixed collection of pitch data is examined to identify how many of various pitch types were thrown. From that collection, individual pitchers are selected for further examination. Their respective pitches are grouped using a technique called clustering. These clusters provide a reference point for comparative analysis with the developing player. From that analysis, machine-learning is used to identify similarity with the professionals' pitches.

METHODOLOGY

The product will be developed using industry standard Agile practices. This will allow for incremental development and roll-out of new features and capabilities. It will also allow for minimal upfront investment. There are potential licensing issues with using the MLB data commercially. Should the licensing efforts fall through, this development path will allow the sunk costs to be minimized. Since the data is publicly available, there are no ethical or legal concerns for its use during development.

REQUIREMENTS

Pending a usage agreement with the local university, its baseball program, and facilities management, funding needs will be minimal. The university has agreed to allow the use of their facilities in exchange for licensed access to the technology for its baseball program for an as yet to be determined length of time. The facilities already have a TrackMan system integrated. The only remaining expense is that of the developer's labor and equipment. Initial estimation for the first leg of development is approximately 6 months to a working product. This is estimated to cost approximately $25,000 + $3000 for development computing hardware.

IMPACT

As TrackMan is installed in every MLB park, every affiliated minor league park, and thousands of collegiate and recreational level parks throughout the country, the potential usage rates are very favorable. On top of that, TrackMan offers an affordable portable unit which many high school and community programs have expressed interest in purchasing.

While working exclusively with TrackMan systems may appear to be limiting, that is only a result of its ubiquity throughout the marketplace. TLaP will not be beholden to TrackMan's system. In fact it will work with whatever method can provide the requisite data points.

TLaP is one of the most exciting concepts we have had opportunity to explore. We look forward to seeing this product's effect on the development of young players.

BUSINESS VISION

PURPOSE

We believe that the locus for sports in modern society is the continued pursuit of personal improvement.

TLaP exists to help pitchers elevate their game to the next level. Through close analysis and diligent study, we believe that every player can find what it takes to improve. Our goal is to reduce the amount of time required to make those improvements.

BUSINESS OPPORTUNITY

The pipeline from amateur to professional athlete is rife with opportunity for business growth. We feel that TLaP is just the beginning of tapping into that market. The days of improvement through purchasing equipment are limited, if not gone. Improvements in golf club, bowling ball, baseball batt, etc. technology are severely curtailed through rules and regulation. However, sports improvement will never be curtailed by improvement in skill.

IT EXECUTIVE DETAIL

BACKGROUND

In baseball, a Pitch (Wikipedia 2019) is defined as the act of throwing a baseball towards home plate to start a play. MLB.com currently lists 13 different types of pitches thrown in the sport. For fans of watching the game on television and online, MLB supplements broadcasts with information to help the viewer identify which type of pitch was delivered. In 2006, Sportsvision's **PITCHf/x** system (Dimeo 2007) was installed in all MLB stadiums with this purpose.

The **PITCHf/x** system provided telemetry and a visual depiction of every pitch thrown. The data was used to provide umpires with feedback as part of MLB's Zone Evaluation System in addition to webcast content within MLB's **Gameday** broadcast. The system utilized three permanently affixed stadium cameras to determine speed and location of the baseball as it traveled from the pitcher's mound towards home plate.

During the 2015 season, MLB introduced **Statcast** to all 30 MLB stadiums. (Casella 2015) **PITCHf/x** became part of the **Statcast** package. **Statcast** added the ability to continuously track player movements throughout the game.

In 2017, **PITCHf/x** was replaced by **TrackMan.** (Diemert 2017) In addition to cameras, **TrackMan** used Doppler radar to monitor player and ball movements during gameplay. Due to differences in the location of **start_speed** measurement (50' from home plate using **PITCHf/x** versus the release point with **TrackMan** - approximately 54.5' on average, varying with pitcher physique), pitchers throughout the league appeared to be throwing faster than in prior seasons. (Tangotiger blog 2017) Because of this and other difficulties in rollout (Arthur 2017) (nderson 2017), the inter-season data is somewhat inconsistent.

There exist other issues in the underlying technology. Schifman (2018) points out that accuracy in tracking vertical ball movement and location is diminished under the **TrackMan** system. However, accuracy in detecting horizontal ball movement and location remained largely consistent. Schifman also asserts that significant "park bias" exists within the telemetry data. Marchi (2011) suggests this is a result of calibration error.

PROBLEM DEFINITION

The goal of this project is to develop a machine-learning model capable of classifying a given baseball pitch as one of the following pitch types: Changeup, Curveball, Cutter, 4-seam Fastball, 2-seam Fastball, Sinker or Slider.

CUSTOMER EXPECTATION

The product will allow amateurs and semi-professional athletes to compare their own pitch performance to that of professional level athletes.

EXISTING SYSTEMS

Currently, there are no commonly available products that allow for this type of analysis. The TrackMan system - in place at many locations throughout the country - provides real-time telemetry of pitches. It also allows sharing of this data. However, it does not accommodate the comparative analysis between different users. It also fails to classify the type of pitch delivered.

DATA OVERVIEW

The data set can be found at Kaggle.com. It was aggregated from MLB's **Gameday** website by Paul Schale and consists of **Statcast** pitch data for the 2015-2018 regular seasons (*pitches.csv*). In addition, the data set also contains information related to other aspects of the game (*atbats.csv, games.csv, player_names.csv, & ejections.csv*).

The data was aggregated from the 5 files into an SQL database using MySQL. The database is constructed around the Entity Relationship Diagram found in APPENDIX I. Some file names and data types were modified to align with the ERD. Specifically, within *pitches.csv*, **code** and **type** were renamed to **pitch_code** and **pitch_class** to avoid potential keyword conflicts.

The following attribute data types were altered to more accurately reflect their underlying structure.

Attribute Name	Previous Data Type	Current Data Type
ab_id	float	integer
pitch_num	float	integer
s_count	float	integer
b_count	float	integer
outs	float	integer
on_1b	float	Boolean
on_2b	float	Boolean
on_3b	float	Boolean
b_score	float	integer

Table 1 – Type Modifications in Database Construction

Further detail of the individual attributes and characteristics of the data is found in APPENDIX II.

Ultimately, only the **pitch** and **atbats** tables (derived from *pitches.csv* and *atbats.csv*) were used. The two were joined, removing all of the preceding categorical attributes except **ab_id** & **p_throws**, while adding **player_id** & **inning**. This removed unnecessary noise and allowed filtering of pitches by player, handedness, and starting pitching.

PITCH CLASSIFICATION

In a blog post at ProBaseballInsider.com (Bernier, n.d.), Doug Bernier describes 9 pitch types that commonly occur in major league play. At his website, Lokesh Dhakar provides a visual description of the flight path for 12 similar pitch types. Based on these descriptions and those found at MLB.com, it is possible to group pitches into 3 larger classes:

- Fastball – exhibit high velocities which limit reaction time available for the batter
- Offspeed – exhibit a decrease in velocity, but similar appearance of motion to Fastballs
- Breakingball – exhibit deceptive horizontal and/or vertical movement during the flight path

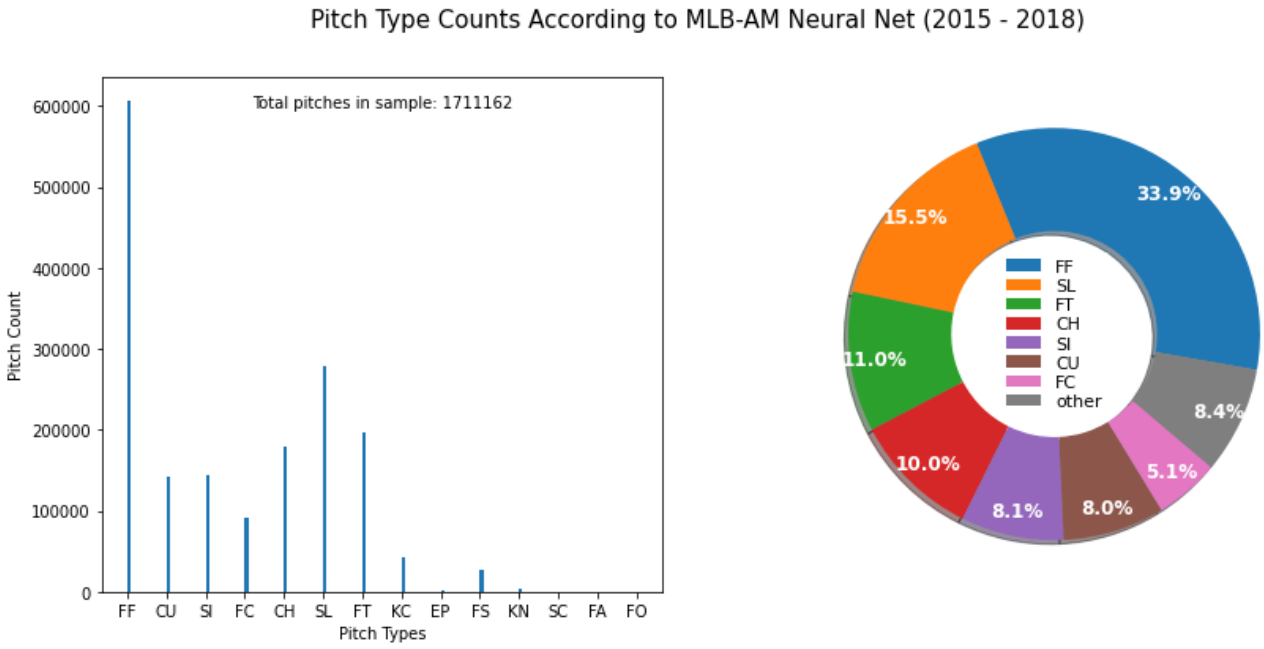
	MLB	Bernier	Dhakar	MLB-AM Abbr
Offspeed	Changeup	Changeup	Changeup	CH
	Knuckleball	Knuckleball	-	KN
	Eephus	-	-	EP •
	-	-	Palmball	
	-	-	Circle Changeup	
Breaking Ball	Knuckle-curve	see Curveball	-	KC
	Curveball	Curveball	Curveball	CU
	-	Slurve	Slurve	
	Slider	Slider	Slider	SL
	Screwball	-	Screwball	SC •
Fastball	4-seam Fastball	4-seam Fastball	4-seam Fastball	FF (FA)
	2-seam Fastball	2-seam Fastball: (runs)	-	FT
	Sinker	2-seam Fastball: (sinker)	2-seam Fastball	SI
	Splitter	Split-finger	Splitter	FS
	Cutter	Cut Fastball	Cutter	FC
	Forkball	-	Forkball	FO •

Table 2 – Different Types of Pitches

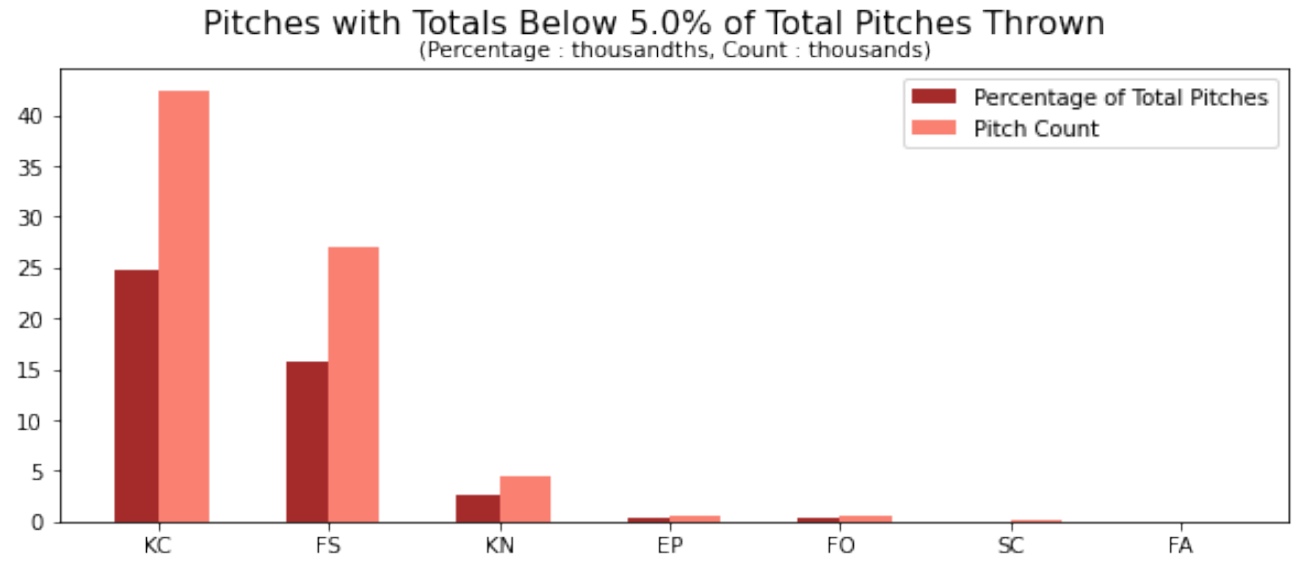
Pitches marked (•) *rarely occur in actual gameplay.* (MLB.com) *Not included: Intentional Ball* (IN) *or Pitchout* (PO)

Currently, (July, 2020) MLB.com *identifies* 4-*seam Fastballs as FA. Previously they were indicated as FF* (June, 2020). *Within the data set, FA occurs in a small portion of the 2015 data. All other 4-seam Fastballs are labeled FF.*

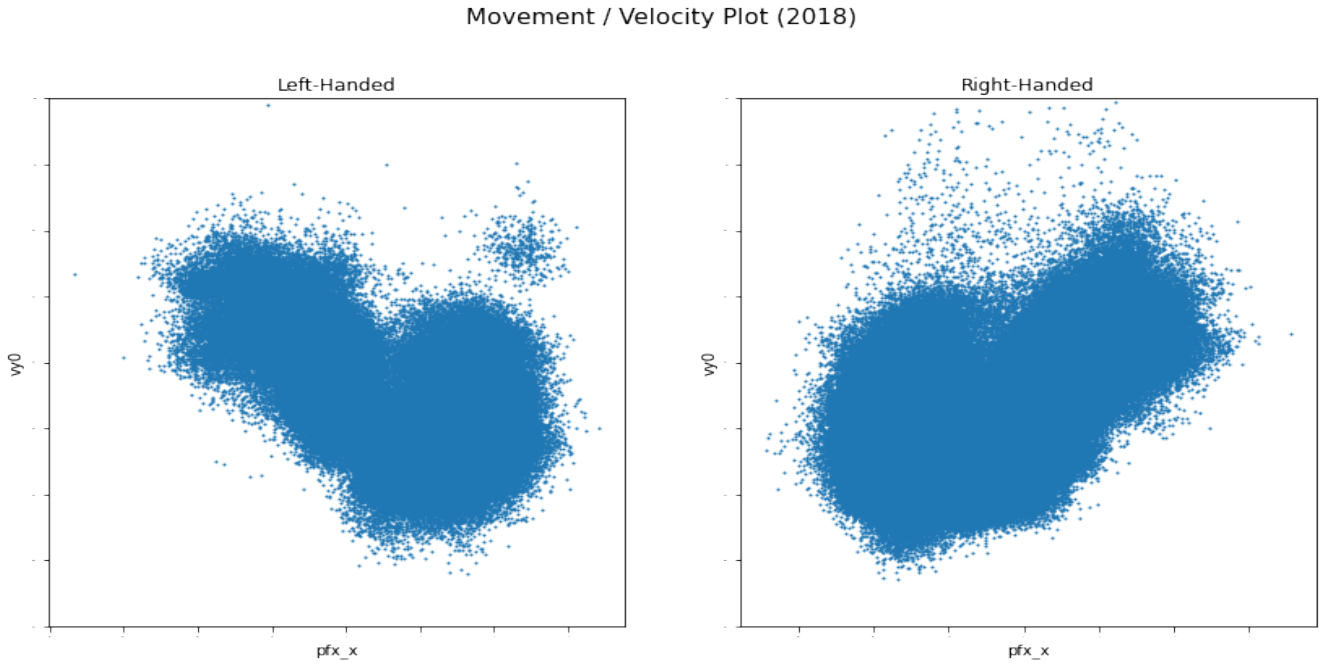
Examining the data set in toto reveals the following pitch distribution:



Of the 16 pitch types identified in Table 2, 14 are indicated within the data set. Of those 14, 7 pitch types make up over 90% of the data. Within those 9 pitch types, 7 occur > 5% frequency.



Krebs (2013) examined the use of cluster analysis in softball pitch classification. She suggested using rates of horizontal movement (**pfx_x**) and velocity (**vy0**). When examining the entire data set, this plot is too dense to clearly identify specific pitch types. When examining individual seasons, the same issue persists.



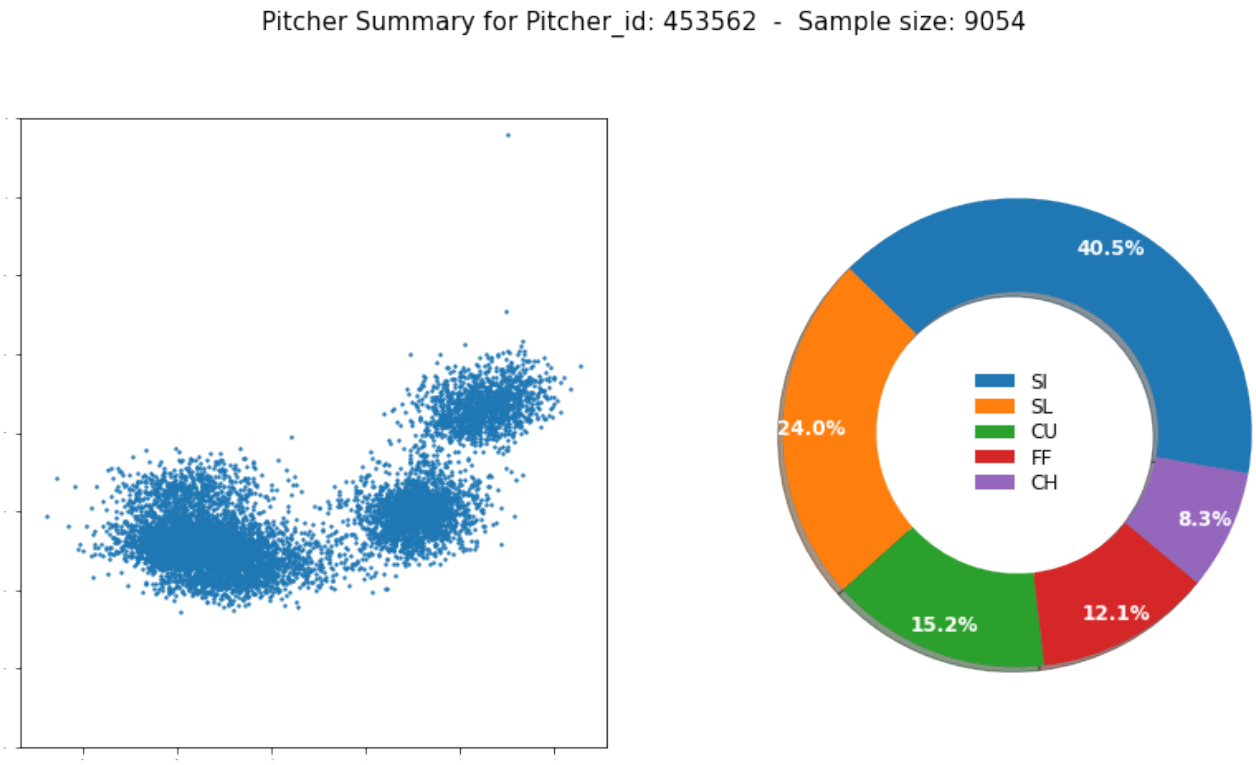
To effectively examine pitch classification, the information needs to be pared down and examined at a more granular level.

DATA IN DETAIL

The data set contains 1332 distinct **pitcher_id**'s. Within that number are 57 pitchers who received enough votes to place in the top-10 of Cy Young voting for 2015-18. Of those, 7 different winners were chosen by the Baseball Writer's Association of America. It is a reasonable assumption that their respective performances will be consistent enough to discern observable patterns.

	National League		American League	
Year	Name	Pitcher-ID	Name	Pitcher-ID
2015	Jake Arrieta	453562	Dallas Keuchel	572971
2016	Max Scherzer	453286	Rick Porcello	519144
2017	Max Scherzer	453286	Corey Kluber	446372
2018	Jacob deGrom	594798	Blake Snell	605483

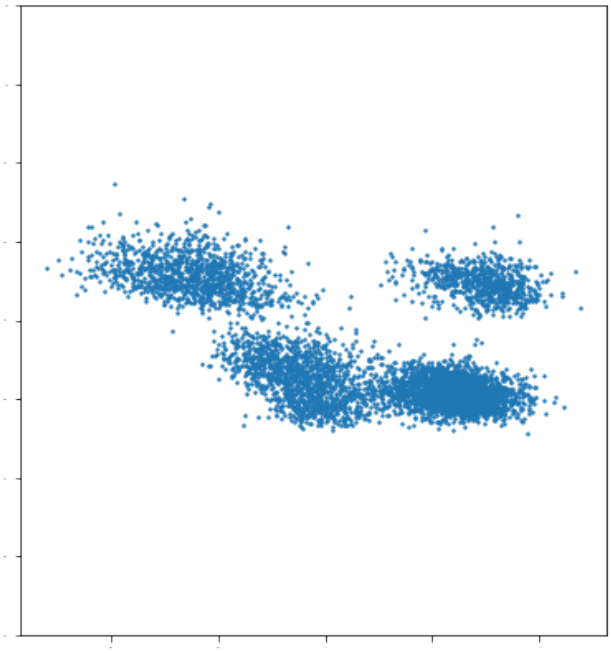
Table 3 – Cy Young Winners for 2015-18 Seasons

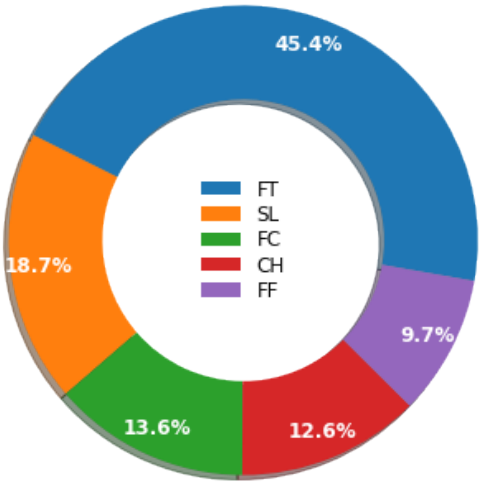


Jake Arrieta 2015-18

The above scatter plot of Jake Arrieta's data appears to indicate 3 or 4 clusters. The pitch distribution indicates that there should be 5. Examination of others in the list of winners reveals similar undercounting.

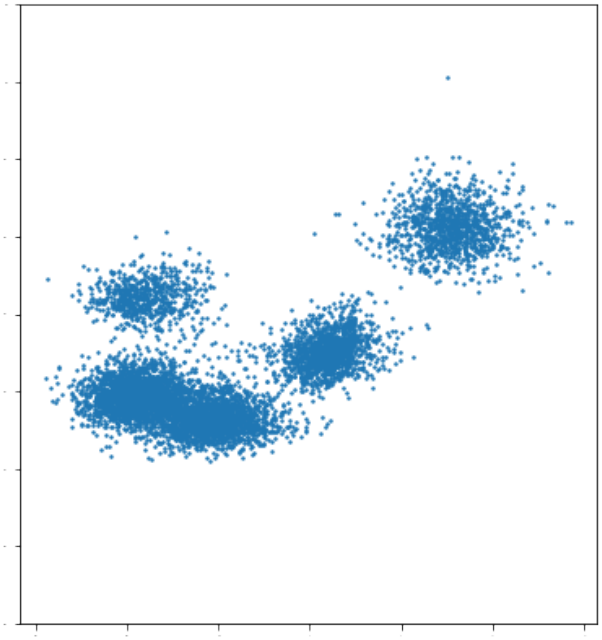
Pitcher Summary for Pitcher_id: 572971 - Sample size: 5915

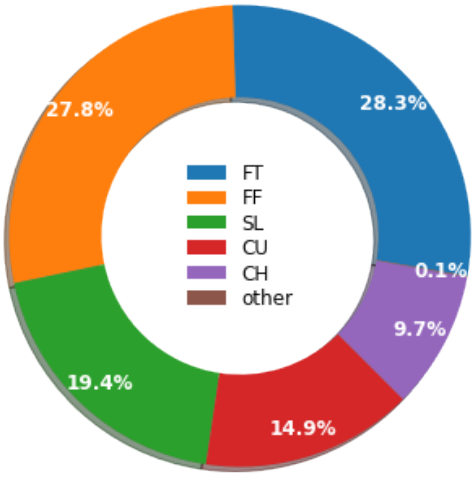




Dallas Keuchel 2015-18

Pitcher Summary for Pitcher_id: 519144 - Sample size: 7302





Rick Porcello 2015-18

METHODOLOGY

Development will be divided into the following phases:

Phase 1: Initial Data Analysis

- Examine available data from seasons 2015-2018
- Identify any rules or implementation changes that may affect homogeneity of the data set
- Verify quality of the data within the data set
- Verify quality and reliability of measurements
- Handle any necessary imputation or transformation
- Assemble data into one aggregated data set for analysis

Phase 2: Exploratory Data Analysis

- Search for features exhibiting strong correlation with pitch type
- Identify features within the data set potentially useful to pitch type classification
- Prototype machine learning models capable of pitch type classification

Phase 3: Product Development

- Refine machine learning models
- Develop minimally viable product
  - End-User Interface
  - Analyze/Classify User data

Phase 4: Documentation

- Prepare data product for presentation
- Integrate output from graphical EDA techniques

DELIVERABLES

Jupyter notebook file with dashboard for viewing the data

Python utility functions for

- importing pitch data to a MySQL database
- extracting database data to CSV files for examination
- generating:
  - display summaries of the data
  - cluster plots of data segments
- creating, training, & tuning multi-classifiers from the data set
- exporting the trained classifier

Minimal working demonstration model

Trained classifier

Markdown file of instructions for installation & use

IMPLEMENTATION

The trained classifier will be implemented within the project dashboard. This model will demonstrate the product's utility. It will be a generalized model, capable of operating on the entire data set. It will classify the pitch type with accuracy above what would occur through random classification.

TESTING / VALIDATION / VERIFICATION

Unit testing will be used on all subcomponents of the preprocessing sequence. All other aspects of the product will utilize libraries which are maintained and tested by third-parties.

The classification model will be validated using stratified *k*-fold cross-validation on two subsets of the data consisting of two seasons and one season respectively.

Final classification results will be verified by using the holdout method on a single season subset of the data.

SECURITY

Few security considerations exist for the product. The primary point of security failure will be the use of *.pkl* files for storing trained classifiers. This could result in the unintended introduction of malicious code. This concern will be mitigated by the use of strict file permissions and ensuring protected access to the data.

MONITOR AND MAINTENANCE

In the production version, the user will have the option of using a model generated for each selected pitcher, in lieu of a generalized model. In order to ensure continued accuracy and effectiveness, maintenance will be necessary to update the available models with data from the most recent seasons.

The usage of individual pitcher models will be monitored in a log file. This will allow the developers to limit the number of models that need to be made available, reducing program footprint, and increasing the quality of service by removing unnecessary maintenance tasks.

SOFTWARE REQUIREMENTS

The product will be developed in a manner that is platform agnostic. While the final version will be run using MacOS High Sierra 10.13.6, it will port to other operating systems with minimal effort. The following technologies will be used to develop the product. Versions indicated are minimum:

- MySQL 8.0.19
  - mysql-connector-python 8.0.19
- Python 3.6
- Python libraries:
  - Jupyter Notebook 6.0.3
  - Numpy 1.18.2
  - Pandas 1.0.3
  - SciKit-Learn 0.22.2post1
  - Matplotlib 3.2.1
  - Seaborn 0.10.0

Development will be completed by a single developer. Development time will last approximately 6 months.

PROJECT TIMELINE

Goal / Milestone	Artifacts	Start Date	End Date	Responsible	Accountable	Complete Date
Project Planning	Proposal Submission	04/13/20	04/15/20	Developer	Project Manager, Project Sponsor	04/15/20
Proposal Acceptance	Approval Form, Release Form, IRB Form	04/15/20	04/15/20	Developer	Project Manager, Project Sponsor	04/15/20
Planning – Development	ERD	04/16/20	04/24/20	Developer	Project Manager	04/24/20
Database Build	Database	04/24/20	04/27/20	Developer	Developer	04/27/20
Data Aggregation	*pitches.csv*	04/27/20	04/28/20	Developer	Developer	04/28/20
Meta Analysis	Data Dictionary	04/28/20	05/01/20	Developer	Project Manager	05/06/20
Planning – Data pre-processing	Preprocess Flowchart	05/01/20	05/08/20	Developer	Developer	05/06/20
Utility programming	Preprocessing Library	05/06/20	05/15/20	Developer	Developer	05/20/20
Descriptive Analysis	Clustering Library	05/18/20	06/12/20	Developer	Project Manager	06/26/20
Exploratory Data Analysis	Clustering Plot	06/12/20	06/26/20	Developer	Developer	06/26/20
Predictive Analysis	Modeling Library	06/15/20	07/06/20	Developer	Project Manager	07/01/20
Model justification & selection	Trained Model	07/06/20	07/10/20	Developer	Developer	07/10/20
Planning – Prototype Build	-			Developer	Project Manager	
Principle programming	Dashboard	07/13/20	07/24/20	Developer	Developer	07/30/20
Principle programming	Demo script	07/27/20	08/07/20	Developer	Developer	08/06/20
Documentation	*README.md*	08/10/20	08/14/20	Developer	Project Manager	08/10/20
Submission	*Project.zip*	08/17/20	08/28/20	Developer	Project Manager, Project Sponsor	08/26/20
Acceptance Testing	-	08/31/20	09/04/20	Project Manager	Project Sponsor	09/04/20

DESIGN & DEVELOPMENT

DESCRIPTIVE METHODS

An unsupervised approach similar to Krebs' cluster analysis was used to divide **pitch_type**s into subsets of similar type. Mills (2015) suggests Expectation Maximization (EM) as a superior alternative to KMeans for this task. SciKit-Learn offers an EM implementation called the Gaussian Mixture Model (GMM). According to the documentation (Pedregosa et al, 2011), this is a probabilistic model that "assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters." Like KMeans, EM is a centroid-based clustering technique.

DATA PREPARATION

Prior to examination, the data needed to be processed (see APPENDIX III). The following removals were made to ensure data quality:

- remove all entries with missing values
  Because of the large size of the data set, imputing missing values was not attempted.
- remove all entries with low **type_confidence** (< 1)
  type_confidence *is an indicator of label likelihood generated by the MLB-AM neural net. Entries with low* ***type_confidence*** *would reduce the quality of the data under examination.*
- if there are multiple **pitcher_id**'s in the data set:
  Because of rules regulating roster-size, some position players are forced to occasionally pitch. This is an attempt to remove those instances.
  - remove those with low numbers of batters faced (< 50)
  - remove those with low pitch counts (< 100)
- remove all pitches > 5-feet from the strike-zone center
  The distance from the center of home plate to the outside line of either batter's box is 56.5". Any pitch beyond 5 feet from the strike-zone center is likely an error in execution. Any estimation of the ***pitch_type*** *of such a pitch is likely to be flawed.*

The next set of alterations were made to enable processing by the machine-learning library:

- remove unused, undefined, and/or derived features:
  - **pitcher_id**, **ab_id**, **inning**, **nasty**, **zone**
- encode **p_throws** (categorical feature)
- remove **pitch_type**s not being examined:
  - Rarely used pitches (as indicated by MLB)
    - Eephus, Forkball, Screwball
  - Intentional Balls
    - Intentional Ball (2015-16), Pitchout
  - Uncategorized pitches
    - UN, AB, FA
  - Pitches occurring in less than 5% of the data set
    - Knuckleball, Knuckle-curve, Splitter
- encode **pitch_type** (categorical feature)
- separate target values from the rest of the data

After the preprocessing sequence was finalized, the data was organized into training, tuning, and testing sets. The training set consisted of all data from the 2015-16 seasons. The tuning and testing sets were sourced from the 2017 and 2018 seasons, respectively.

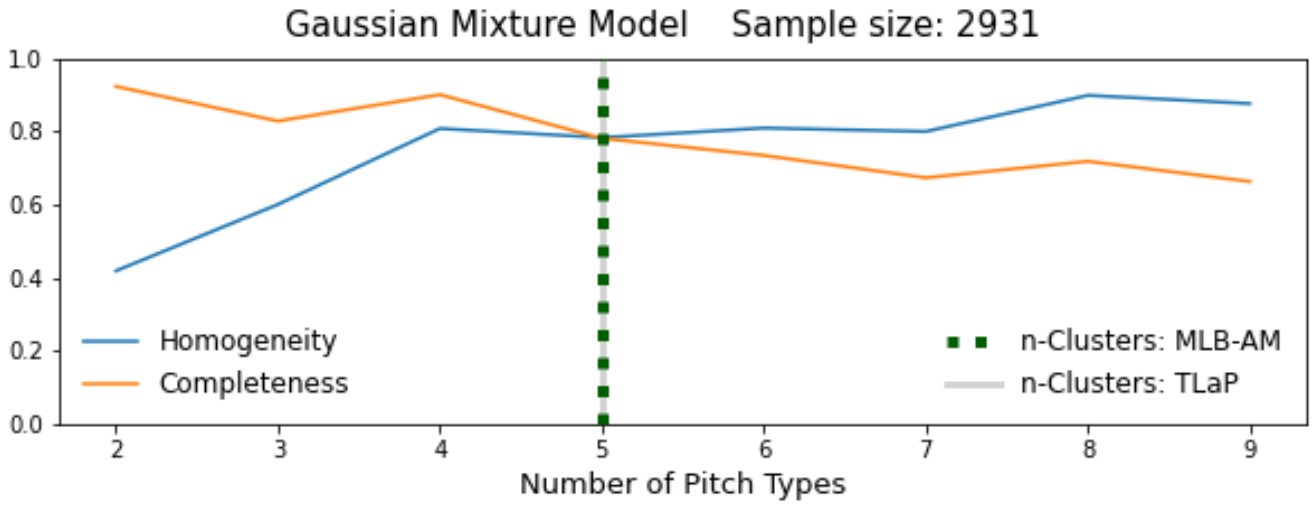
DETERMINING *n*-CLUSTERS

Prior to using EM, the number of clusters (*n*-) must be determined. This was estimated by applying GMM to the data *k*- times. The absolute difference between homogeneity (H) and completeness (C) scores for each *k*- was calculated:

$$f(\mathrm{k}) = | \mathrm{H} - \mathrm{C} |$$

n- is the minimum value for $f(k)$, where *k*- is between 2 (the minimum number of possible clusters) and 9 (a practical limit on the number of different **pitch_type**s found in a sample of one pitcher).

Samples were selected from Cy Young winners during the 2015-18 seasons as well as arbitrarily selected, like-handed pitchers. For left-handed pitchers, *n*- was consistently accurate. For right-handed pitchers, *n*- was ± 1 pitch. Figures like the following were generated for all Cy Young winners in the sample (see APPENDIX IV). These figures indicate the computed number of **pitch_type**s within the samples as well as the number of **pitch_type**s labeled by MLB-AM.



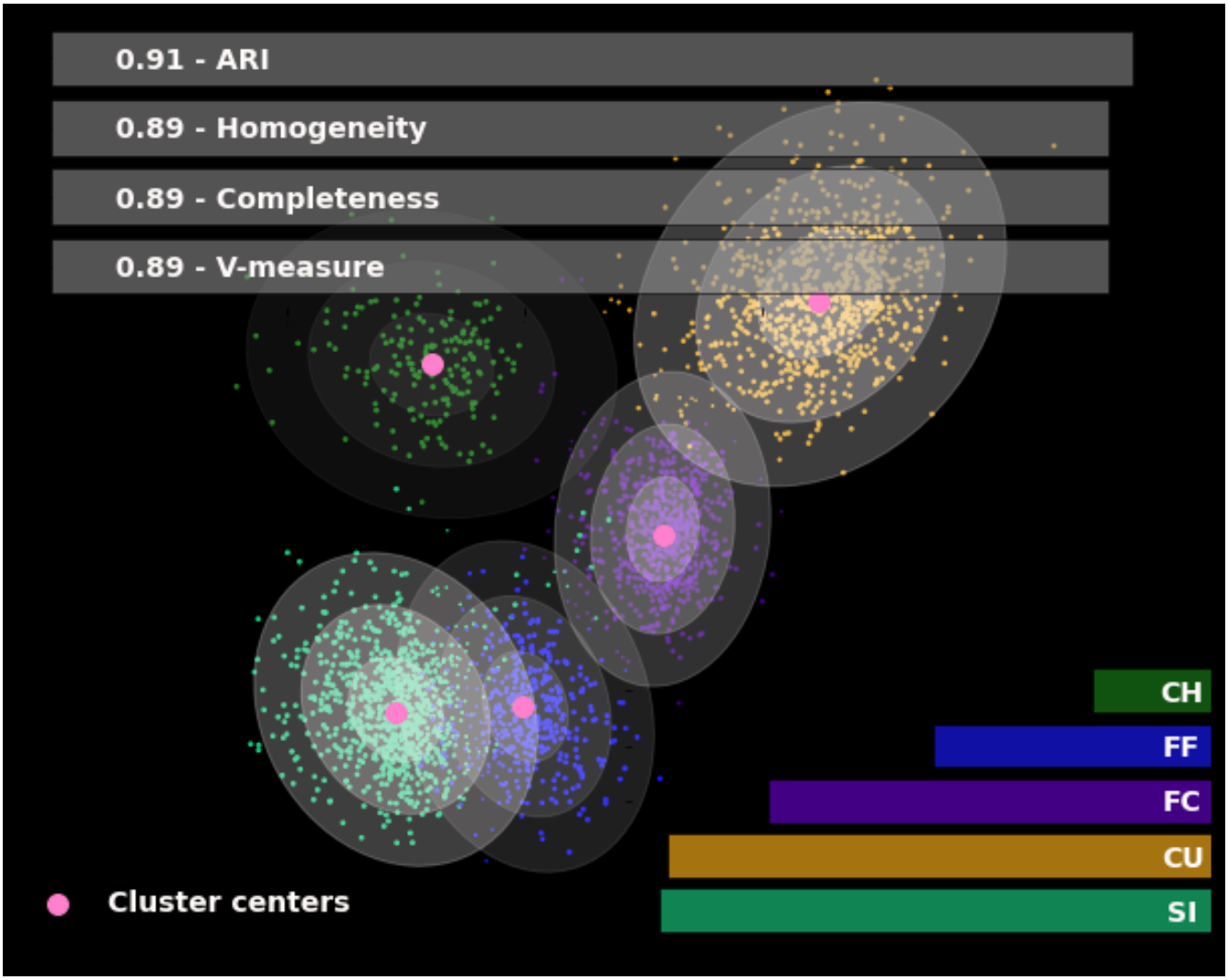
PLOTTING CLUSTERS USING GAUSSIAN MIXTURE MODEL

After estimating the number of different **pitch_type**s within the sample, Krebs" movement/velocity plot was revised to identify and indicate cluster centers. Combined with robust scaling (**RobustScaler**), this resulted in increased scores on clustering metrics and distance between centers.

The resulting 2D-plots utilized GMM to identify cluster centers. Concentric rings surrounding the centers indicated approximated decision boundaries and covariance. Classification probability was reflected in the size of individual pitch markers. Generally, the farther from the cluster center, the smaller the dot was rendered.

The color-coded plot seemed to indicate clear delineation between CH, CU, and SL for all pitchers. There appeared to be significant overlap between the 4 different types of Fastball. Attempts to induce sparsity

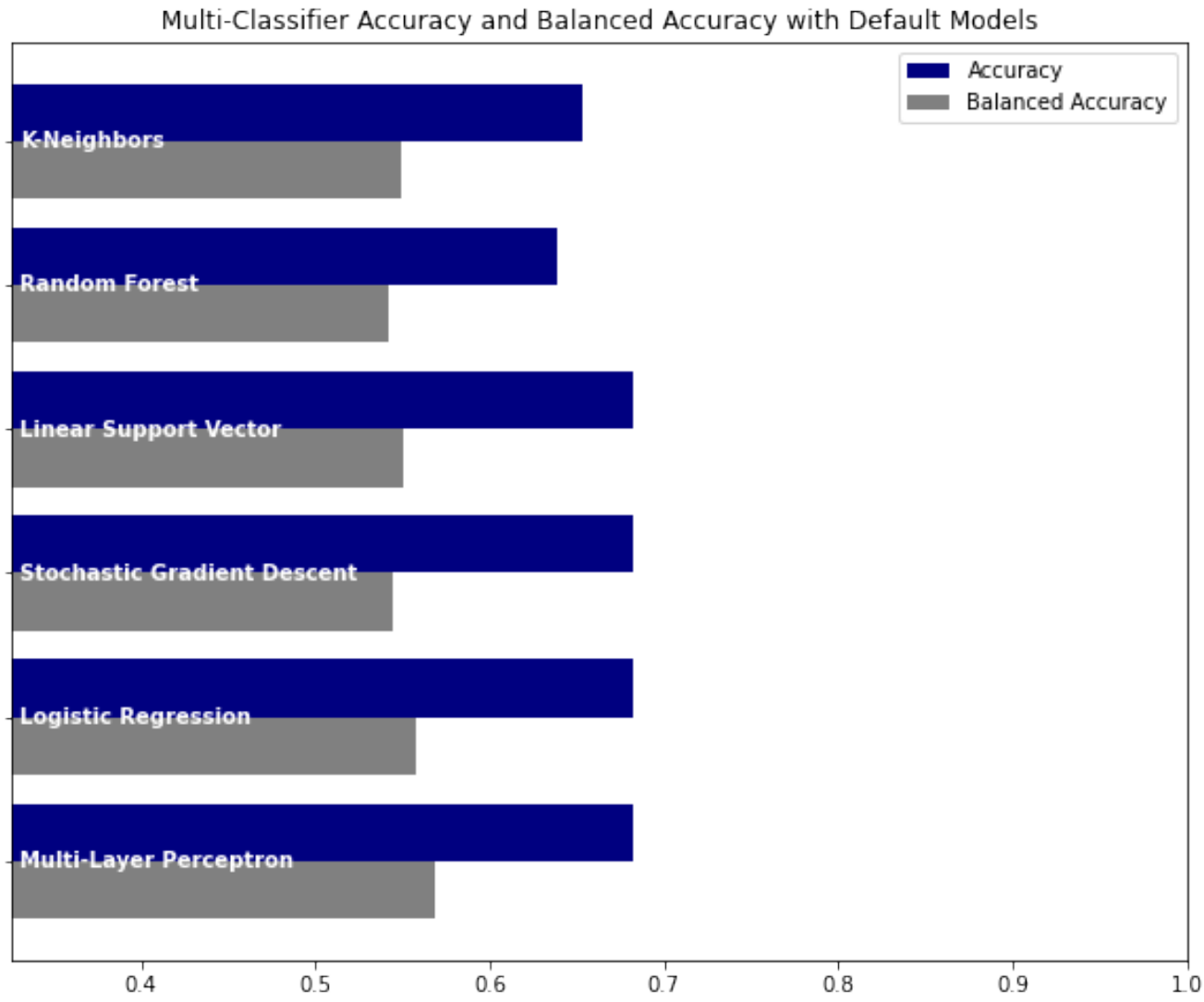
for Fastballs using various transformers were unsuccessful. For samples of mixed pitchers, areas of high density overlapping multiple classes were visible. This issue was not as pronounced in plots of data for individual pitchers.



PREDICTIVE METHODS

A number of predictive models were explored for use in classifying the data set. Classifier implementations with built-in support for **One-Versus-Rest** (**One-Versus-All**) multi-classification were examined. These included: **K-Neighbors**, the ensemble method Random Forest Classification (**RandomForestClassifier**), Linear Support Vector Classification (**LinearSVC**), Linear classifiers with Stochastic Gradient Descent training (**SGDClassifier**), and **LogisticRegression**. In addition, Multi-Layer Perceptron Classifier (**MLPClassifier**) was examined because of its similarity to MLB-AM's own neural network.

Preprocessing used with the descriptive methods was retained for the predictive methods. This included the use of **RobustScaler**. The categorical feature **p_throws** was added as well as the use of **RBFSampler** to reduce the model learning time. Because of the class imbalance within the data set, both balanced accuracy and accuracy were used for comparison metrics.



Based on these results, **MLPClassifier**, **SGDClassifier** (using **LogisticRegression** for loss), and **LinearSVC** were selected for further study.

FEATURE SELECTION

As previously stated regarding data preparation for Unsupervised Learning, initial feature selection involved the removal of categorical features which were used as reference within the database. These included: **pitcher_id**, **ab_id**, **type_confidence**, & **inning**. The removal of other features was explored in an effort to improve classifier results.

An examination of feature variance indicated a wide range of variances within the data set. Principal Component Analysis was explored, but eventually abandoned. The variance of **spin_rate** was such that it dwarfed the results for other features. Resultant feature selection included the following filtering methods (Sharma 2018):

- Low Variance filtering
- High Correlation filtering

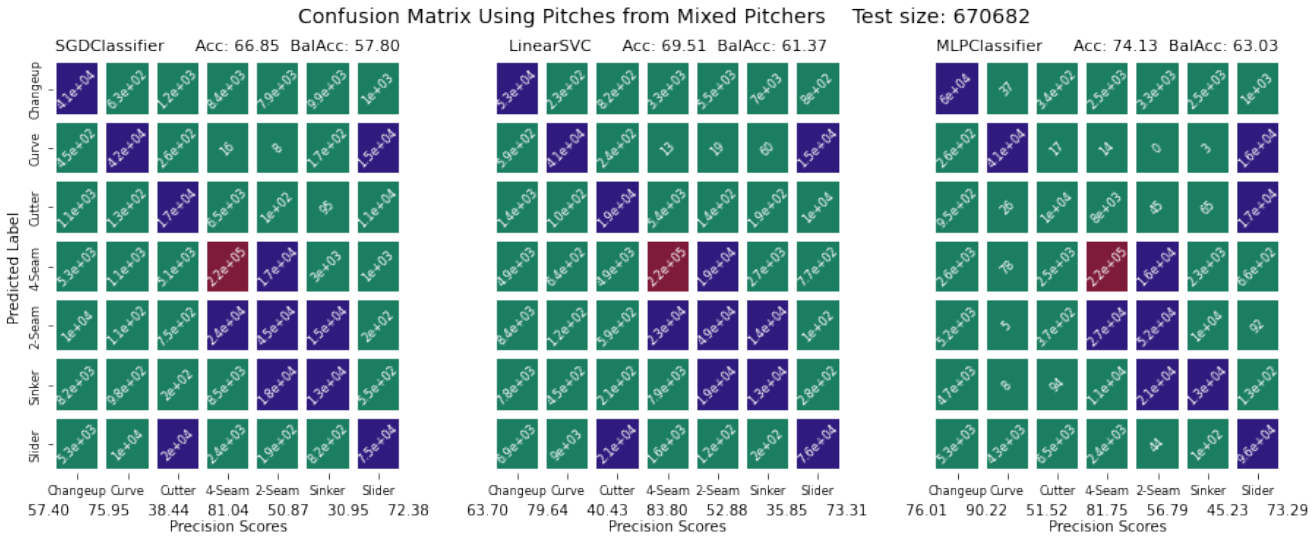
Low Variance filtering indicated 6 candidates for removal **x0**, **y0**, **z0**, **break_y**, **px**, & **pz** (see APPENDIX V). The correlation matrix (see APPENDIX V) used in High Correlation filtering indicated 3 features exhibiting positive correlation with **pitch_type**: **vz0**, **px**, & **break_y**. Negative correlation was notably exhibited by **spin_rate**, **spin_dir**, **x0**, **az**, **break_length**, **p_throws**, & **pfx_z**.

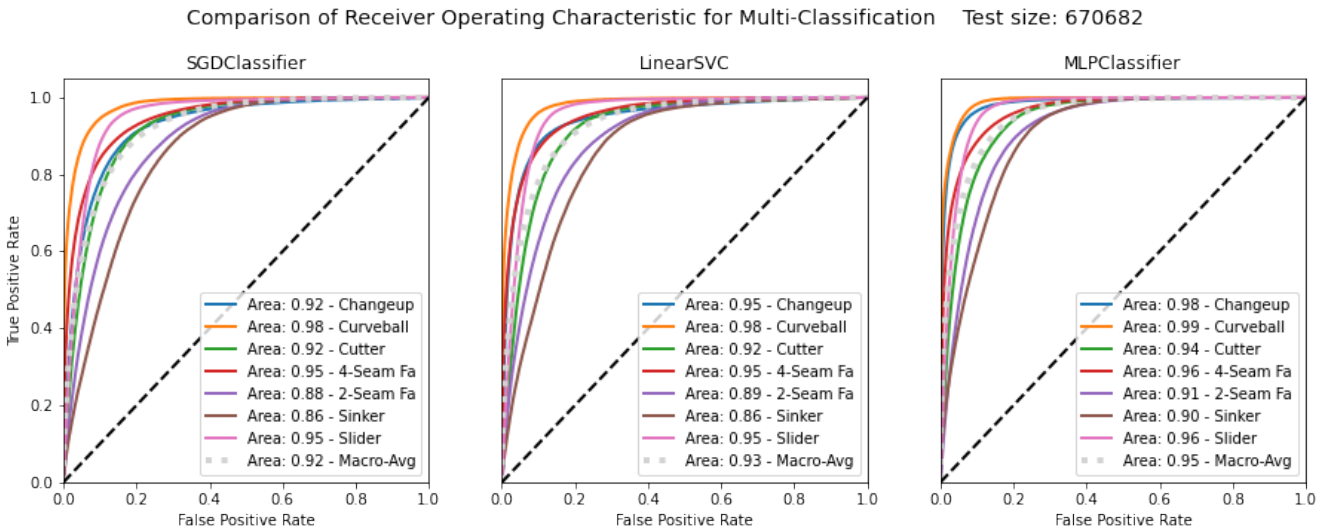
After combining the results, the remaining features used in the predictive model were: **p_throws**, **az**, **vz0**, **break_angle**, **spin_dir**, **spin_rate**, & **pfx_z**.

FINAL MODEL SELECTION

To select a final model, the three remaining classifiers were instantiated with default hyperparameters and trained. A confusion matrix and ROC AUC plot was then produced for each model using tuning data. Of the remaining candidates, **MLPClassifier** exhibited the most consistent and accurate performance on a sample composed of mixed pitchers.

The confusion matrices indicate that the number of false positives generated by **MLPClassifier** was fewer than for the other two. The associated accuracy and balanced accuracy scores were measurably higher for **MLPClassifier**. Lastly, the respective precision scores for all but one class within the data set was higher than that found using the other classifiers.

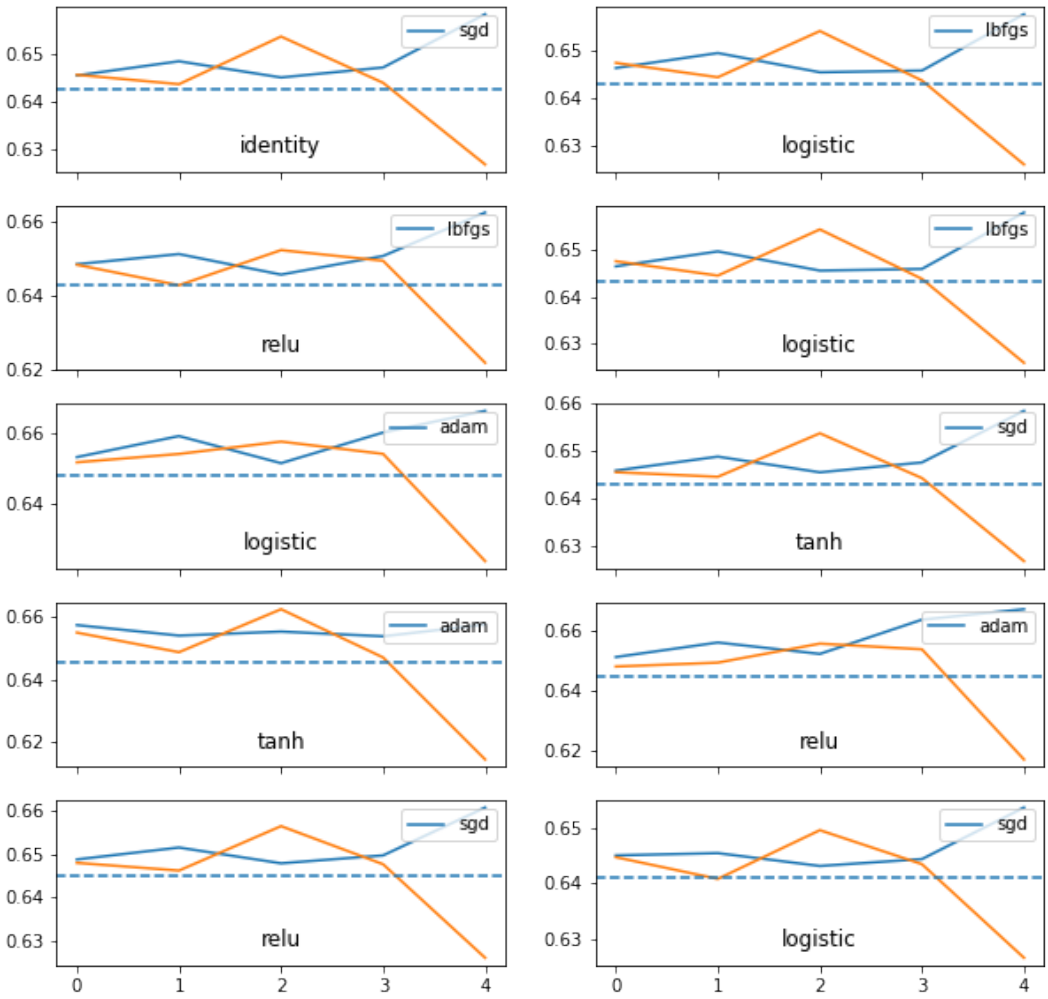




The composite ROC AUC plots indicated a higher true positive rate for every class when using **MLPClassifier**. Also of note, the coverage for all 3 classifiers was very similar. This would imply that, with tuning, the other 2 classifiers would probably be capable of performing just as well.

MODEL TUNING

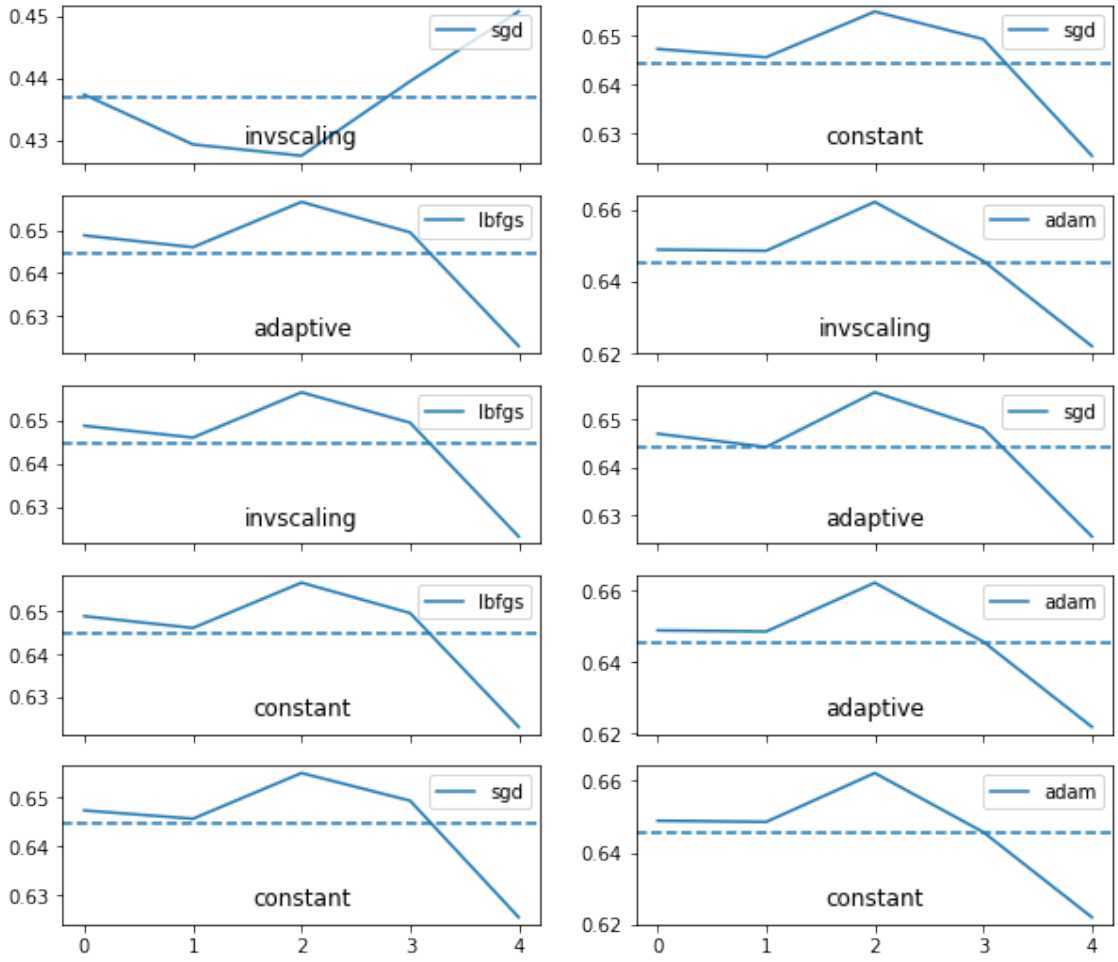
Model tuning began with randomized 5-fold cross-validation. The initial hyperparameters under test were **solver** (*sgd*, *adam*, & *lbfgs*), **activation** (*identity*, *logistic*, *tanh*, & *relu*), and **learning_rate** (*constant*, *invscaling*, & *adaptive*). The following plot displays the train & test scores for all 5 folds as well as the mean test score. The plots were ordered by mean_fit_time.



The results were inconclusive. No combination of the 3 hyperparameters appeared more effective within the comparison. The next step was to utilize grid search.

GRID SEARCH CROSS-VALIDATION TUNING

As before, 5-fold cross-validation was used. The hyperparameters under test were reduced to **solver** and **learning_rate,** utilizing the same options. This test was run twice with identical results both times. The most obvious takeaway was that the *sgd* / *invscaling* combination, while the fastest, was approximately 20% less accurate than any of the other options in test. The remaining results were all markedly similar. As such, *lbfgs* was selected due to its shorter convergence time, indicated by its earlier placement in the plot. This could be problematic in the future. *lbfgs* is optimized for smaller data sets. As the project grows, *sgd* should be reconsidered.



The same approach of incrementally tuning pairs of hyperparameters was continued. Final selected hyperparameter values were:

- **activation**: *identity*
- **learning_rate:** *invscaling*
- **hidden_layer_sizes:** *50*
- **alpha:** *1e-4* (default value)
- **tol:** *1e-4* (default value)

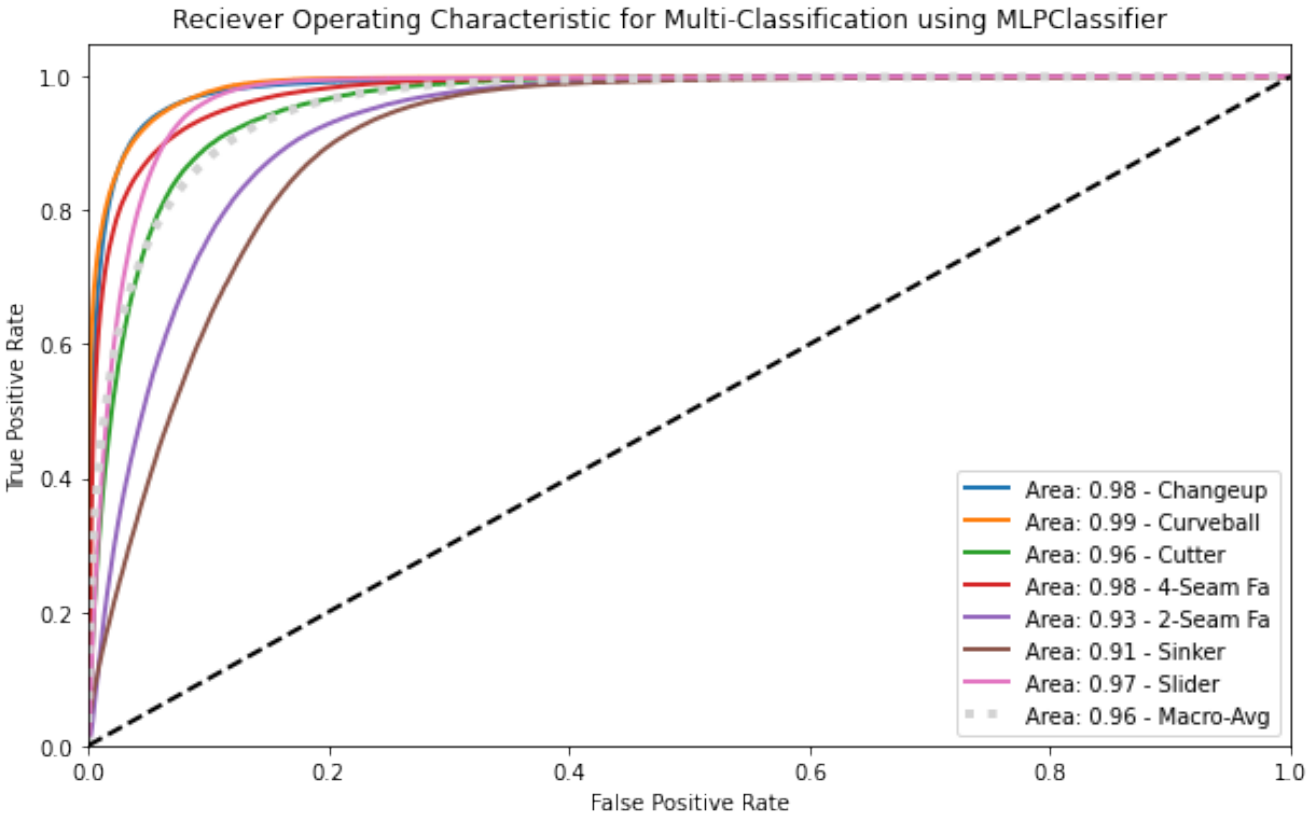
MODEL VERIFICATION

A model using the previously defined hyperparameters was instantiated and trained using the combined training and tuning data. Predictions on the holdout data were made using this model. Those predictions were used to generate a classification report and ROC AUC plot.

```
pitch_type_final initialized to default state
              precision    recall  f1-score   support

           0       0.77      0.87      0.82     73064
           1       0.85      0.78      0.81     58606
           2       0.68      0.33      0.45     38649
           3       0.87      0.93      0.90    247115
           4       0.55      0.62      0.58     80653
           5       0.51      0.22      0.31     56669
           6       0.76      0.89      0.82    119320

    accuracy                           0.77    674076
   macro avg       0.71      0.66      0.67    674076
weighted avg       0.76      0.77      0.75    674076
```



GETTING STARTED

REQUIREMENTS

The User Dashboard requires the following packages. Versions indicated are minimum:

- Python 3.6
- Jupyter Notebook 6.0.3
- Numpy 1.18.2
- Pandas 1.0.3
- SciKit-Learn 3.2.1
- Seaborn 0.10.0

INSTALLATION

To install Python, download the appropriate version for your operating system HERE. Follow the installation instructions. When complete, open Terminal and verify the install by typing:

```
python3 --version
```

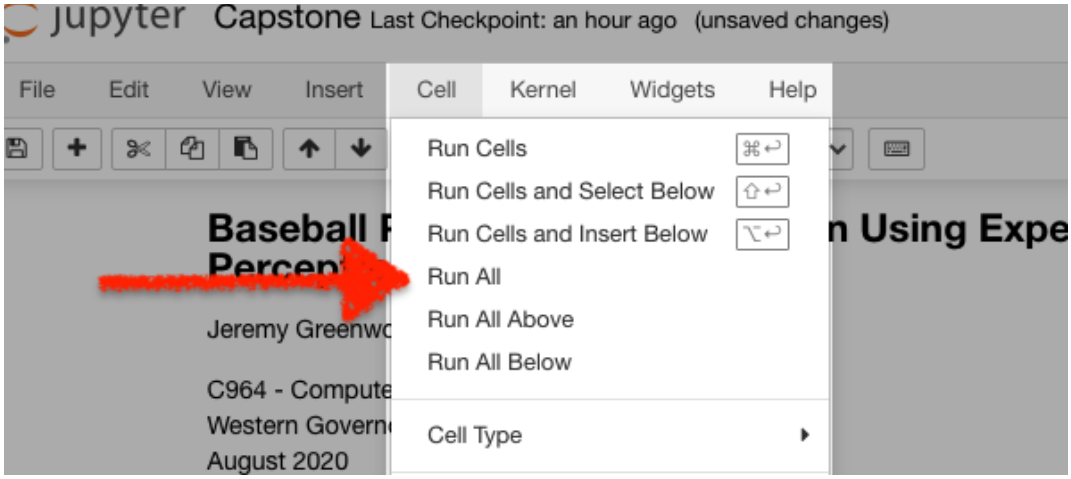
To install the necessary support packages, enter the following:

```
pip install numpy
pip install pandas
pip install jupyter
pip install matplotlib
pip install seaborn
pip install scikit-learn
```

Next, download the Capstone_Submission.zip file from HERE. When complete, move the .zip file to your desired directory and extract the contents. Return to Terminal and enter the following:

```
cd <directory where you extracted .zip file>
cd Capstone_Submission
jupyter notebook Capstone.ipynb
```

A new window will open in your default browser. Once loaded, open the Cell tab in the menu bar and select Run All.



CONTROL PANE

The Control Pane is located on the left side of the Dashboard screen. This is where the user can select what pitcher's data to explore.

Step 1: *Select Player*:

The available pitchers were all Cy Young winners during the available seasons. The option to select "All" pitchers from the league is available, as well as a randomly selected pitcher from the data set.

Step 1a: *Outer number*:

This option is only available when *Select Player* is set to "All". Its use is explained under *Pitch Counts*.

Step 2: *Season*:

Seasons may be viewed individually or in toto. The option to pair different seasons together is not available. Selectable seasons will change to reflect those during which the selected pitcher participated.

RESULTS PANE

The Results Pane is the largest segment of the Dashboard, found on the right portion of the screen. This is where the pitchers' data will be displayed. At the top of the Results Pane is where the user will find 4 tabs for selecting what data to display.

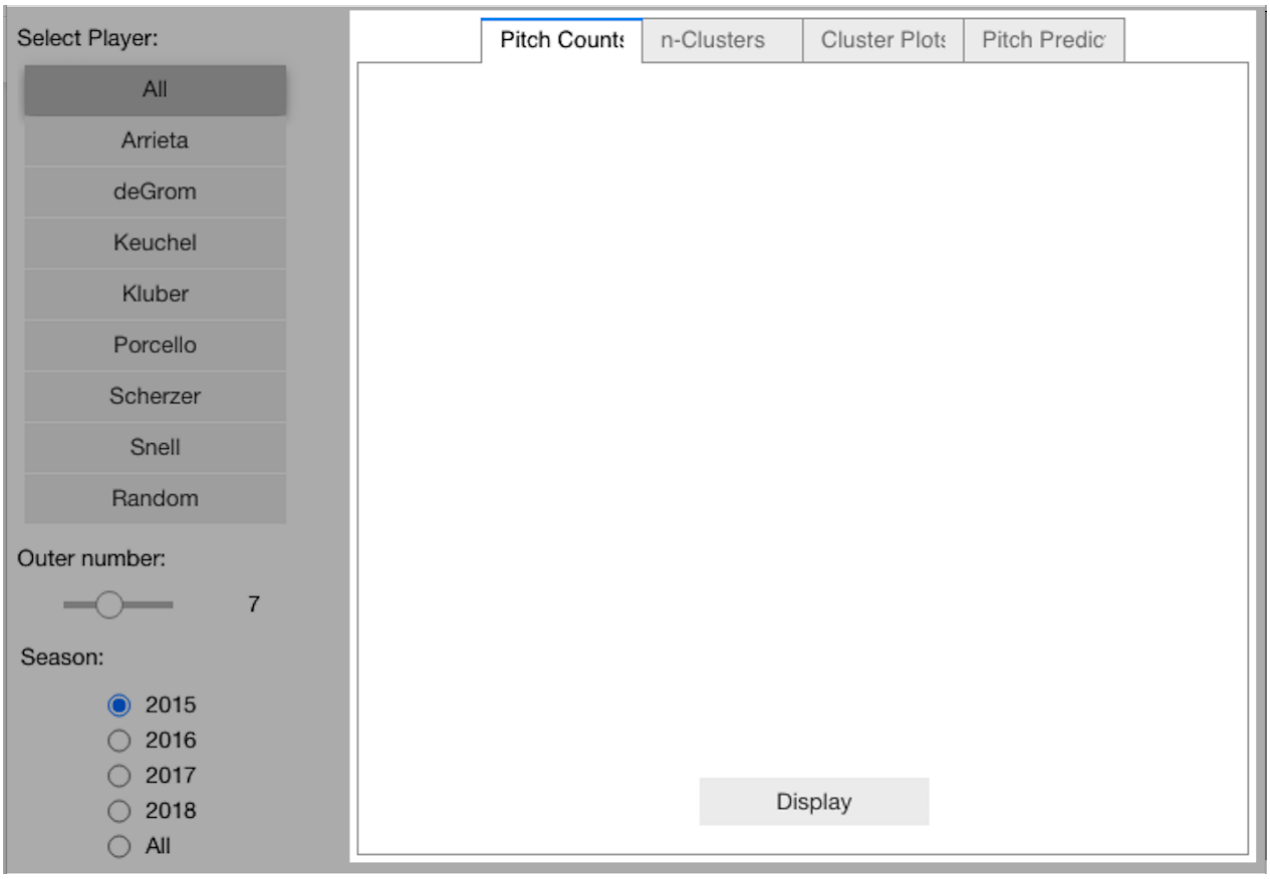
Step 3: *Pitch Counts* / *n-Clusters* / *Cluster Plots* / *Pitch Predict*

Pitch Counts : This tab displays a pie chart of the number of pitches thrown by the selected pitcher during the selected season. The chart is color-coded to enable the user to more quickly compare pitch percentages between different pitchers and different seasons. If "All" players are selected, the *Outer number* option allows the user to change the number of pitch types displayed in the outer ring of the pie chart. This allows the user to adjust spacing to enhance number readability.

n-Clusters : This tab displays the probable number of different pitch types associated with the pitchers' data as well as the number initially determined by MLB-AM. The *n-Clusters* display is unavailable for "All" pitchers.

Cluster Plots : This tab displays a cluster plot of the selected pitchers' pitch data. Cluster centers are indicated within the plot. A color-coded bar graph representing the pitch counts (lower right) is also displayed to allow the user to more easily intuit the level of density within a cluster. Lastly, a bar graph of clustering metrics is overlaid to provide a quantifiable measure of the cluster density and cluster overlap within the plot. The *Cluster Plots* display is unavailable for "All" pitchers.

Pitch Predict : This tab allows the user to enter pitch data and receive a pitch type classification determined by a machine learning model. This classification is generalized across the entire data set. Future versions will utilize models trained for individual pitchers and allow the user to compare that submitted pitch to the selected pitchers' pitch history. This tab is the only element not controlled by or associated with the Control Pane.

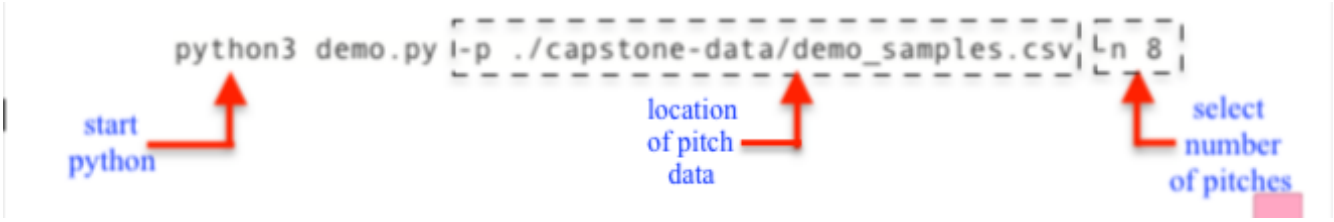


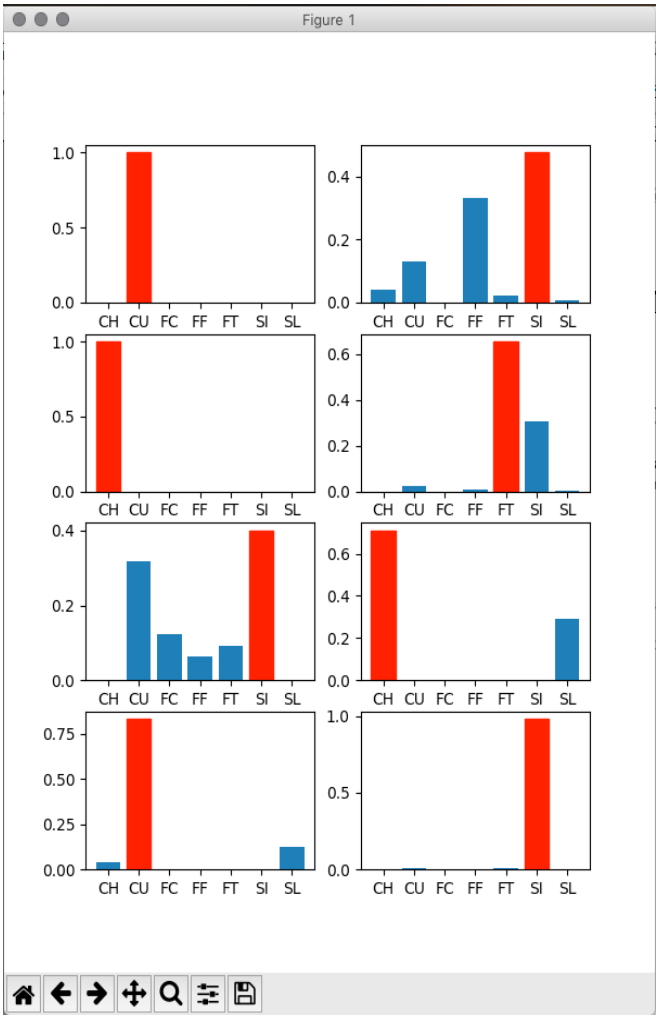
BATCH PREDICTIONS

In addition to entering pitches individually from the Dashboard, TLaP will allow batch processing and analysis of pitches. Given appropriately formatted pitch data in a .csv file, the program will render predictive analysis results.

OPERATION

Open the Terminal and change to the Capstone_Submission directory. At the command prompt, enter:





RESULTS

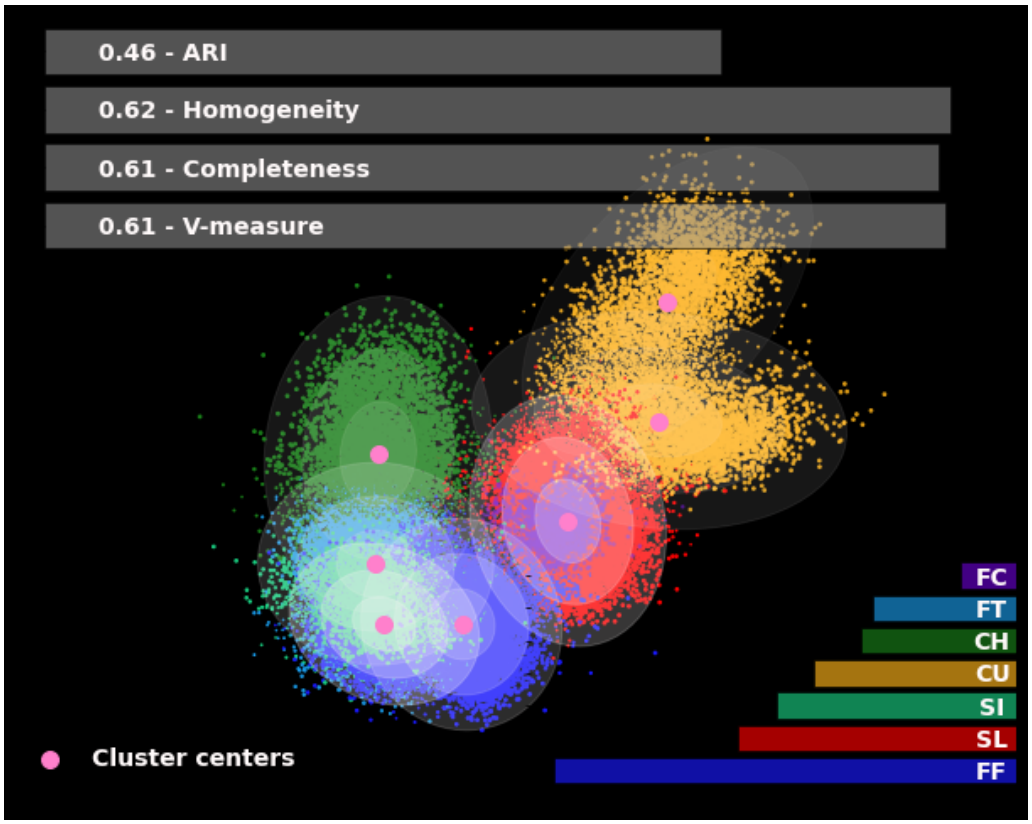
The pitch prediction results will be displayed in a new popup window. The results display the relative likelihood for each pitch with each of the pitch classes.

In this demo version, the pitches are randomly selected from the supplied pitch data. The number of pitch results displayed is limited to integers between 4 and 10.

FINAL THOUGHTS

PRODUCT ACCURACY

There exists a confounding reality in pitch classification: two pitchers can throw an identical pitch and call it by two different names. Pitches exist on a continuum (Kim 2020). If one were to examine the cluster plots found in Appendix IV for all right-handed pitchers overlaid on top of each other, one would notice that all of the pitch clusters form a J-shaped line. The same is true of left-handed pitchers (the J is inverted about the y-axis).



There is no clear decision boundary between clusters for mixed pitchers. Adjacent clusters overlap – some greatly so! Given this, the accuracy results found in Model Verification are rather exceptional, considering the generalized nature of the final model.

ACCEPTANCE

Unequivocally, the project goal was successfully met. The machine-learning model developed is able to predict pitch classification with accuracy well above that of chance. The model is able to effectively operate on the majority of pitches used in modern professional baseball.

APPENDIX I:

ENTITY RELATIONSHIP DIAGRAM

APPENDIX II: DATA DEFINITIONS

Attribute explanations for *pitches.csv* are summarized from those located within the Kaggle.com data definitions, and as explained by Fast, M. (2007), Slowinski, S. (2012), and Krebs, C. (2013). Attributes are grouped into: game state, derived properties, & telemetry readings.

	Identifier	Type	Unit	Description
Measured Attributes	**x0**	float	feet	horizontal distance from center of the pitch at release
	y0	float	feet	distance from home plate at release
	z0	float	feet	vertical distance from ground at release
	vx0	float	ft/sec	velocity at release
	vy0	float	ft/sec	" " "
	vz0	float	ft/sec	" " "
	ax	float	ft/sec^2	acceleration at release
	ay	float	ft/sec^2	" " "
	az	float	ft/sec^2	" " "
	px	float	feet	horizontal distance from the center of strike zone as it crosses the plate
	pz	float	feet	vertical distance from the ground as the pitch crosses home plate
	start_speed	float	MPH	speed of the pitch at 50' mark (PITCHf/x) or at release (TrackMan)
	end_speed	float	MPH	speed of the pitch as it crosses the plate
	spin_rate	float	RPM	pitch's rate of rotation
	spin_dir	float	degrees	angle at which pitch is spinning
	break_angle	float	degrees	angle of deflection from the release point to the front edge of home plate
	break_length	float	inches	measurement of the max deviation between the trajectory of the pitch at any point and the front of home plate
	break_y	float	feet	distance from home plate where pitch achieved its greatest deviation from a straight-line path
	sz_bot	float	feet	distance from the ground to the bottom of the current batter's rulebook strike zone
	sz_top	float	feet	distance from the ground to the top of the current batter's rulebook strike zone
	x	float	?	the horizontal position of the pitch as it crosses home plate, using old Gameday system
	y	float	?	the vertical position of the pitch as it crosses home plate, using old Gameday system

APPENDIX II: DATA DEFINITIONS

	Identifier	Type	Unit	Description
Derived Properties	**pfx_x**	float	inches	horizontal movement of the pitch between the release point and home plate
	pfx_z	float	inches	vertical movement of the pitch between the release point and home plate
	pitch_type	char(3)	-	most probable pitch type according to MLB-AM's neural net classification algorithm from Ross Paul. See below
	type_confidence	float	-	value of the weight of classification algorithm output; multiplied by 1.5 if known to be a part of the pitcher's repertoire
	nasty	float	?	uncertain: *appears to indicate pitch hitting difficulty; pitches located in the corners of the strike zone rate higher*
	zone	float	?	uncertain: *appears to indicate which sector of the strike zone the pitch is in as it crosses the plate*
	Identifier	**Type**	**Unit**	**Description**
Game State	**ab_id**	integer	-	at-bat ID; unique identifier for each pitcher/batter matchup
	pitch_num	integer	-	pitch count of current at-bat
	event_num	integer	-	event number of at-bat; useful for tracking ejections
	s_count	integer	-	strikes in the current at-bat
	b_count	integer	-	balls in the current at-bat
	outs	integer	-	number of outs before pitch is thrown
	on_1b	Boolean	-	if runner on first
	on_2b	Boolean	-	if runner on second
	on_3b	Boolean	-	if runner on third
	b_score	integer	-	current score for the batter's team
	code	char(3)	-	recorded result of the pitch. See below
	type	char(3)	-	simplified code: (S)trike) (B)all or (X) in play

APPENDIX II: DATA DEFINITIONS

Valid values for ***code*** and ***pitch_type*** in *pitches.csv*:

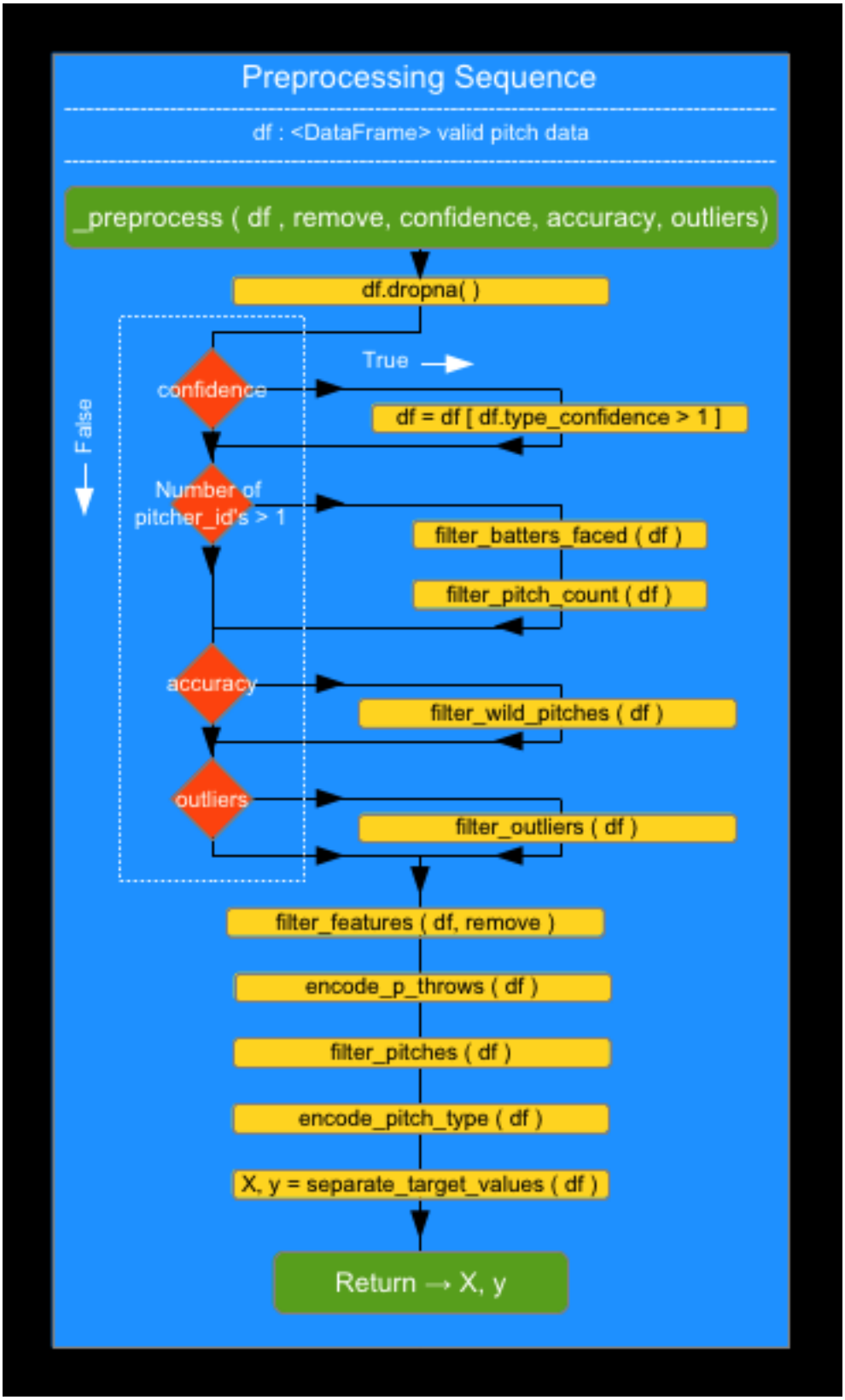
Code	Value	Code	Value
P	Pitchout	**T**	Foul Tip
Q	Swinging pitchout	**L**	Foul Bunt
R	Foul pitchout	**W**	Swinging Strike (Blocked)
I	Intentional Ball	**M**	Missed Bunt
B	Ball	**X**	In play, out(s) •
***B**	Ball in dirt	**D**	In play, no out •
S	Swinging Strike	**E**	In play, runs •
C	Called Strike	**H**	Hit by pitch •
F	Foul		

Values marked • only occur on last pitch of at-bat

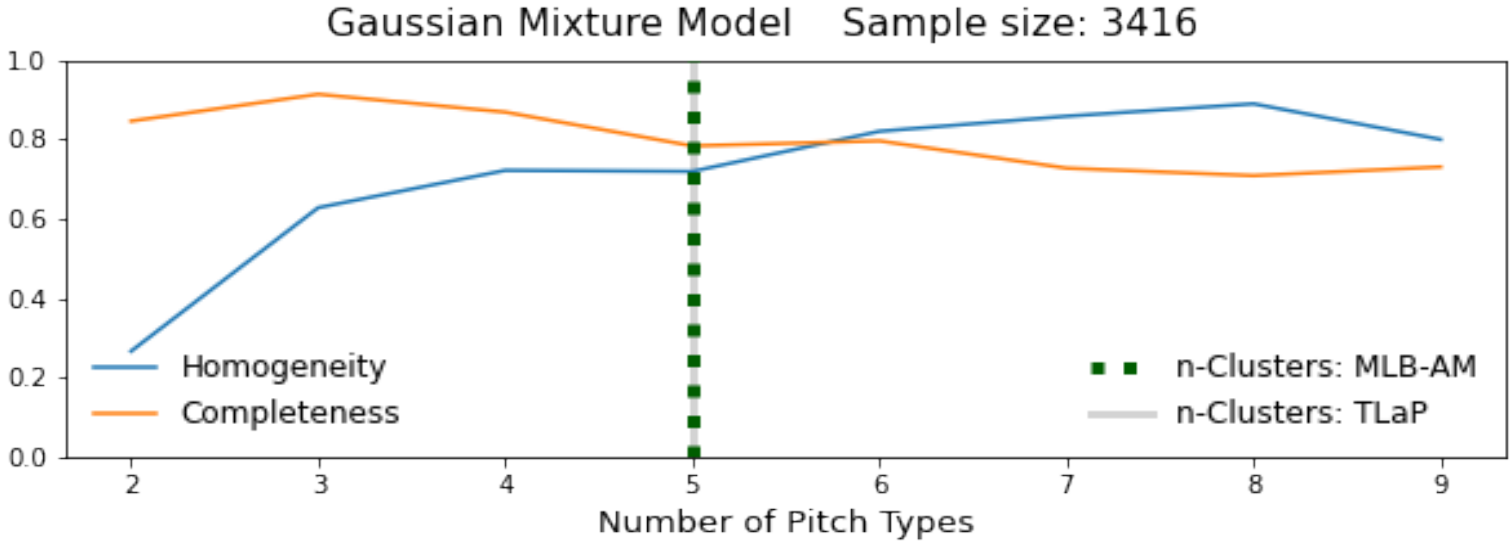
Pitch_Type	Abbr	Pitch_Type	Abbr
Changeup	CH	Knuckle-curve	KC
Curveball	CU	Knuckleball	KN
Eephus	EP ••	Intentional Ball	IN
Cutter	FC	Pitchout	PO
4-seam Fastball	FF	Screwball	SC ••
Forkball	FO ••	Sinker	SI
Splitter	FS	Slider	SL
2-seam Fastball	FT		

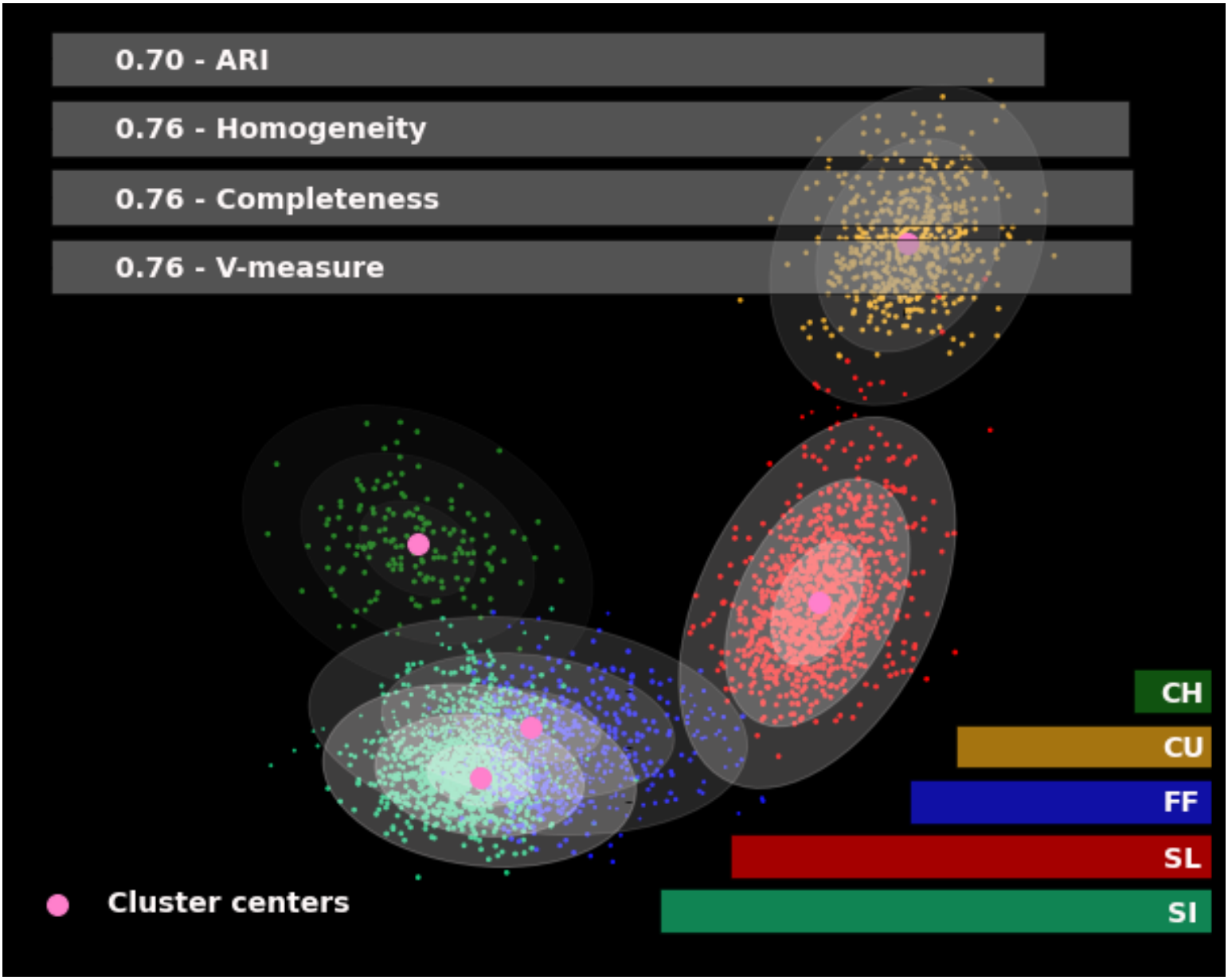
Pitches marked •• rarely occur in actual gameplay. MLB.com
**Due to a 2017 rule change* (Bieler 2017) IN *ends in 2016.*

APPENDIX III: PREPROCESSING SEQUENCE

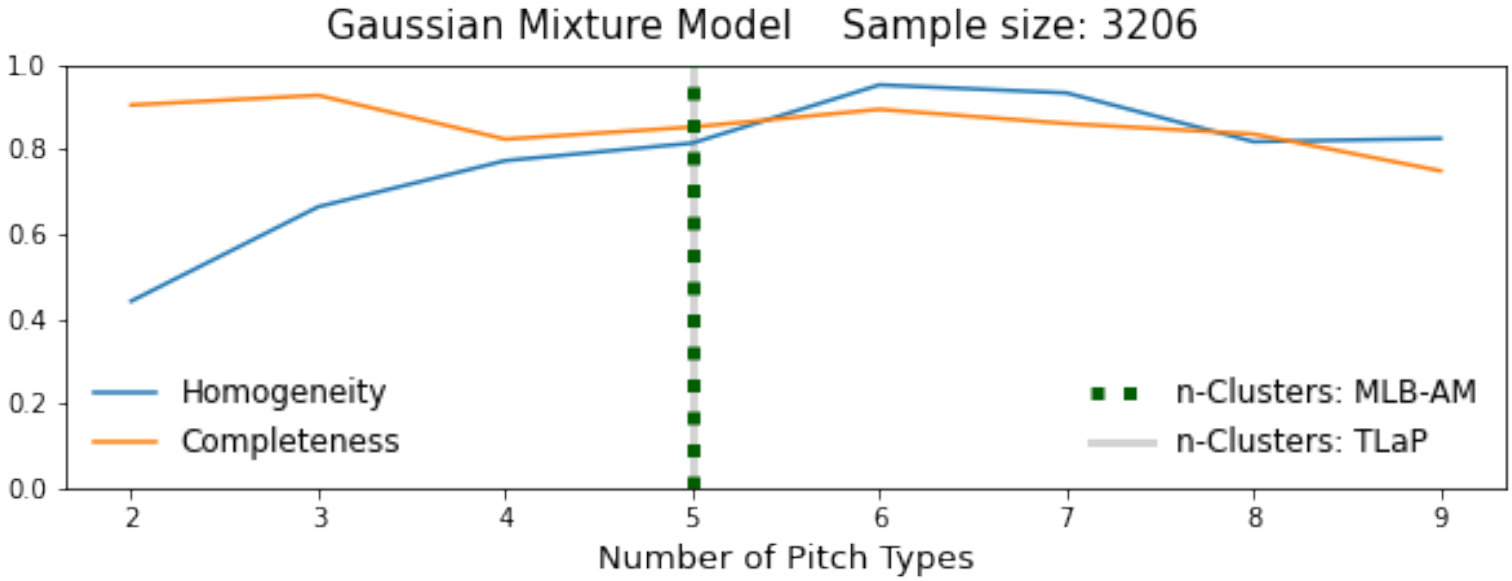


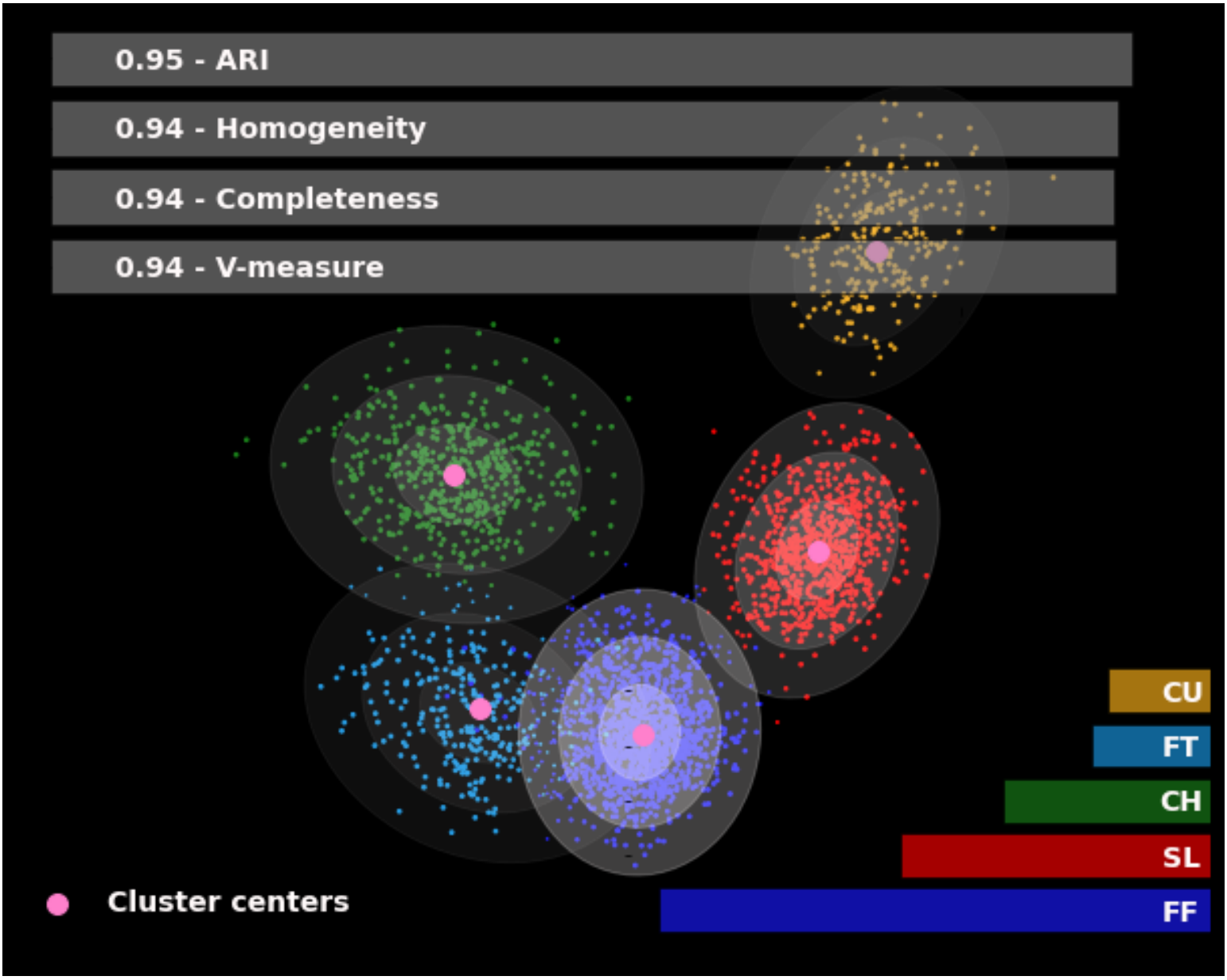
- ARRIETA, JAKE #453562 (2015) -



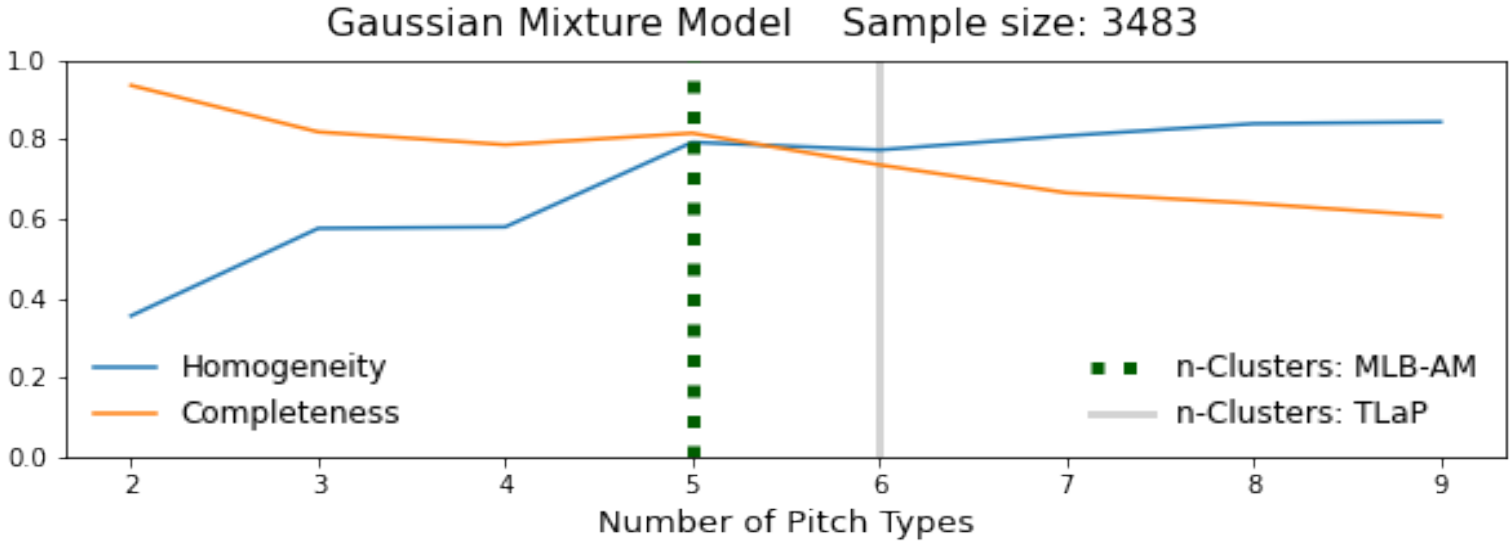


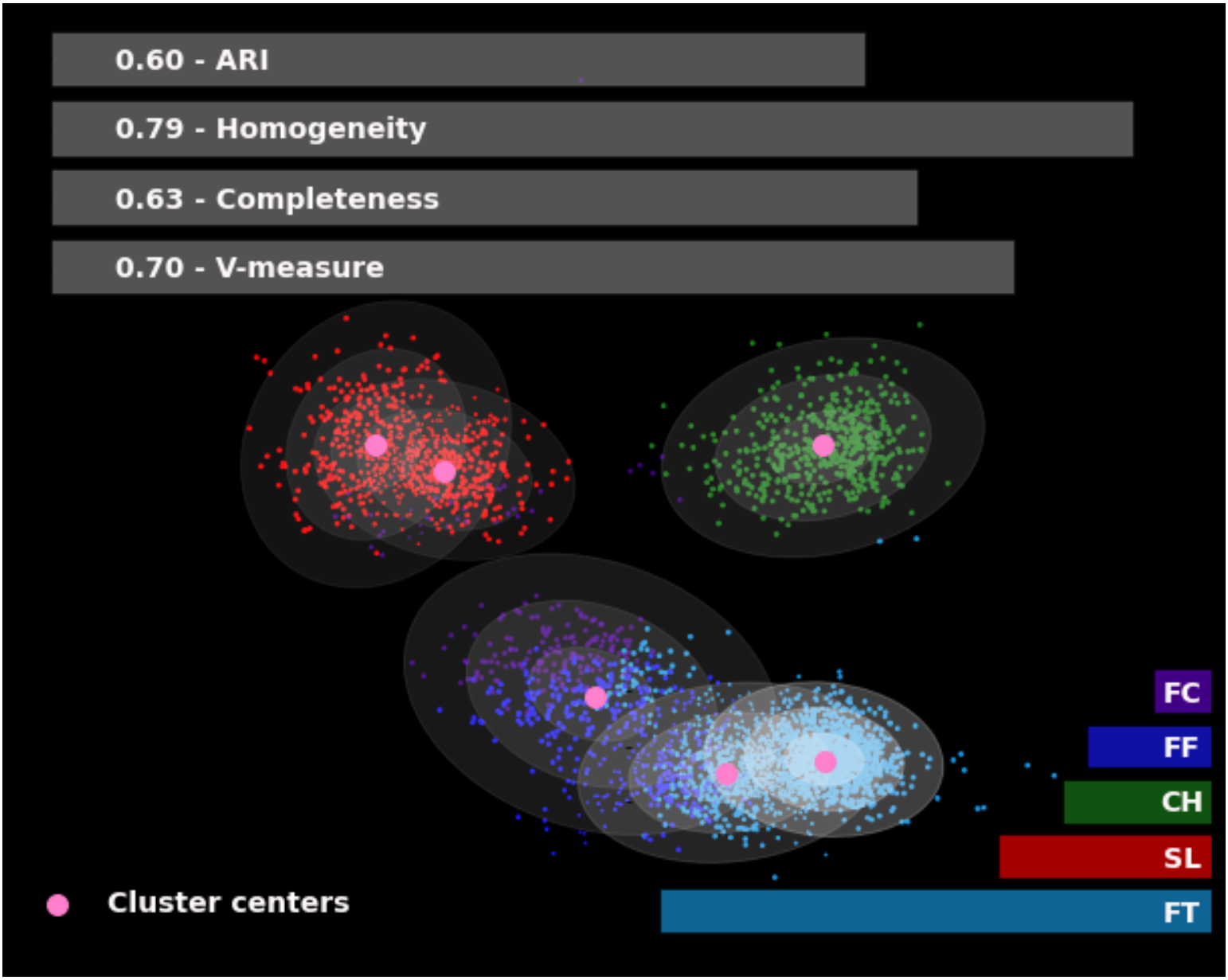
- deGROM, JACOB #594798 (2018) -



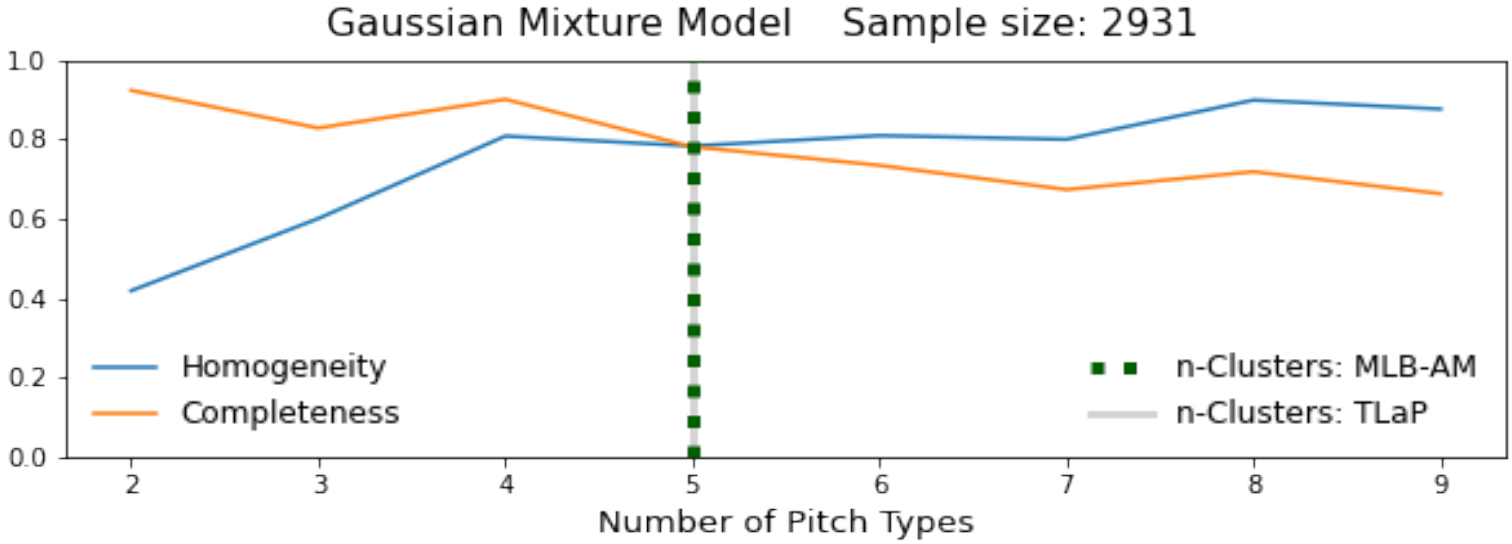


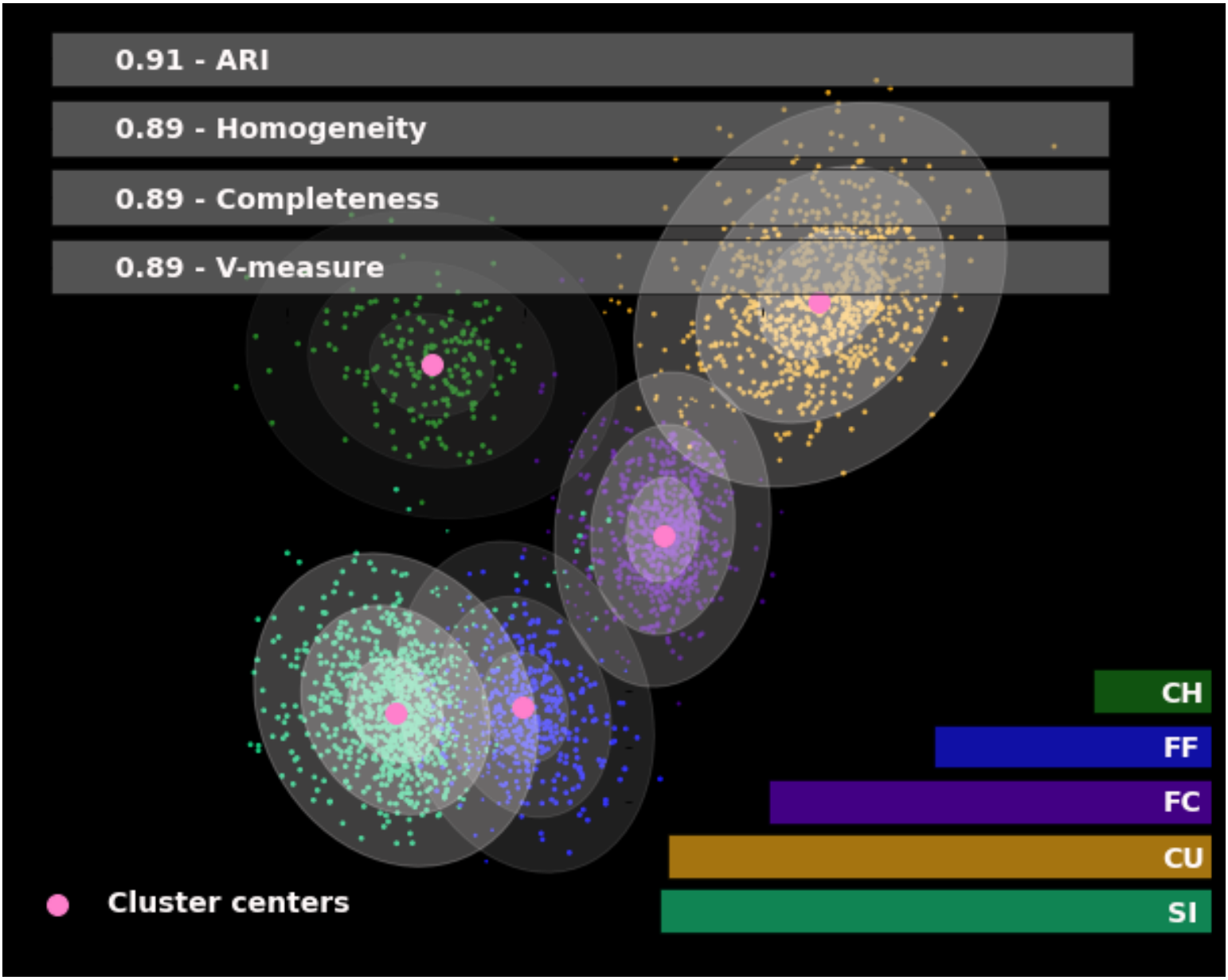
- KEUCHEL, DALLAS #572971 (2015) -



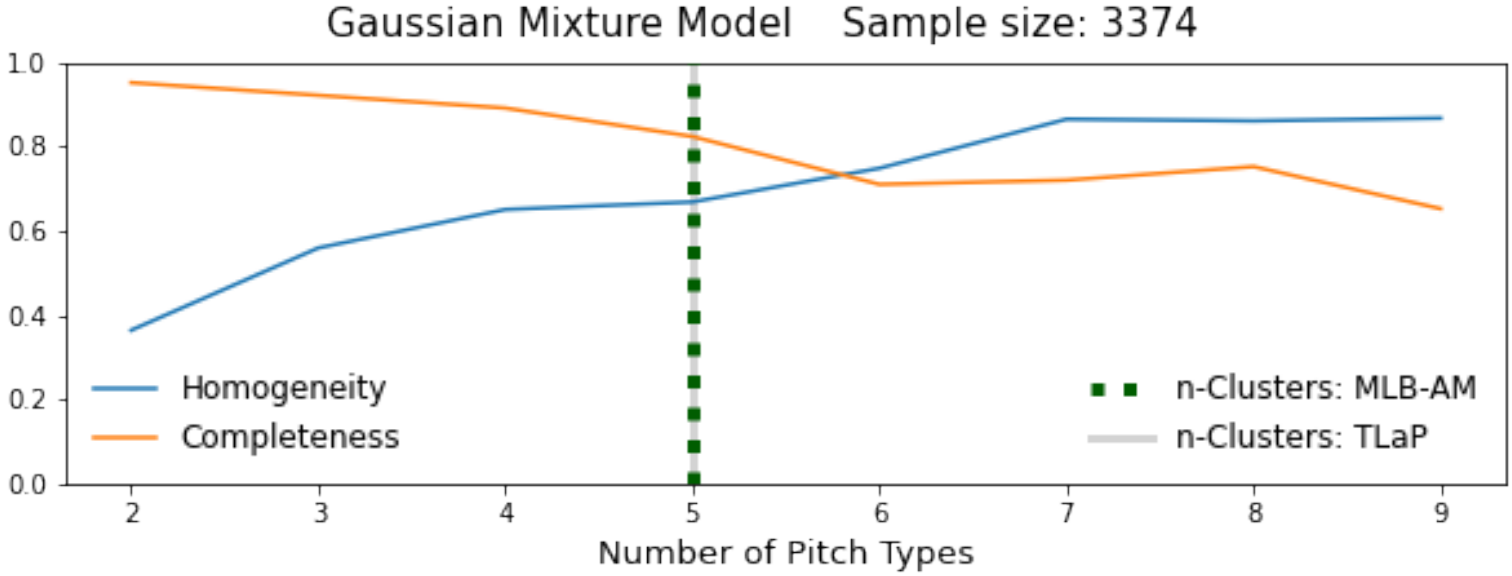


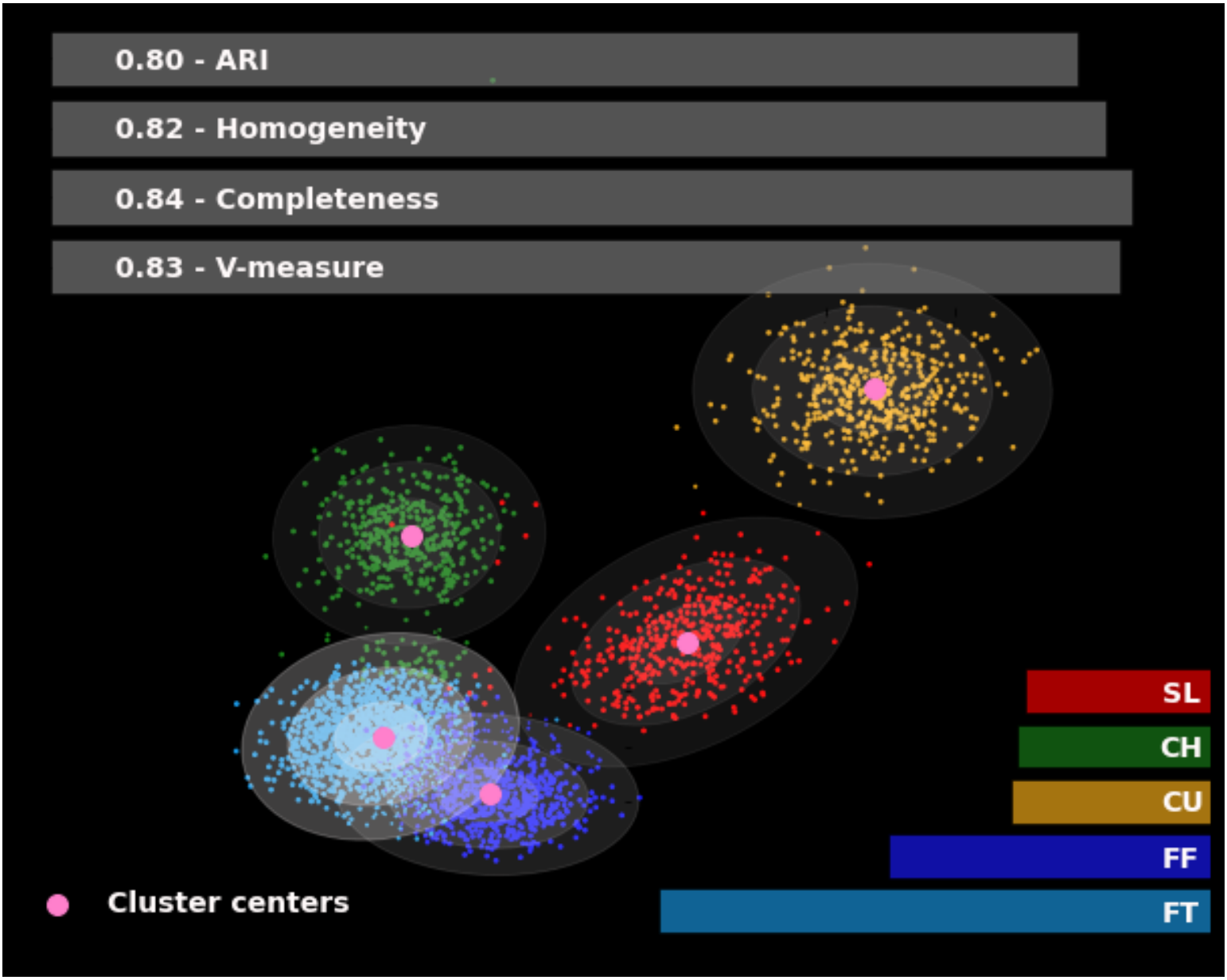
- KLUBER, COREY #446372 (2017) -



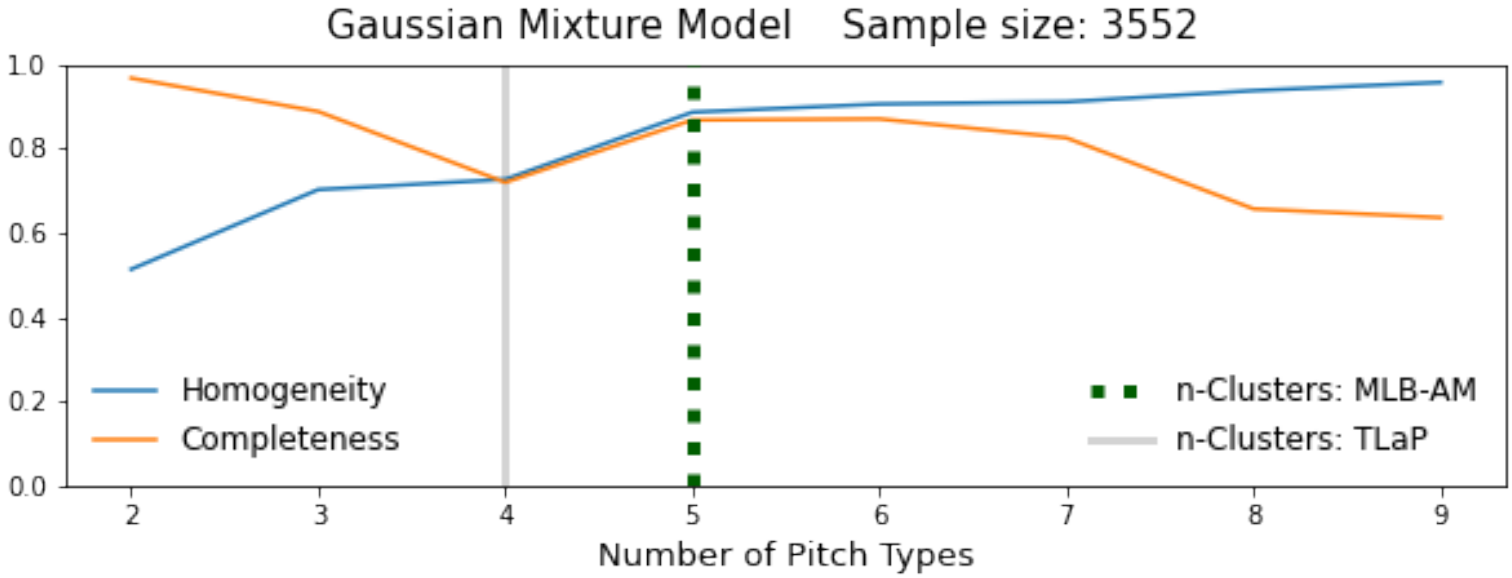


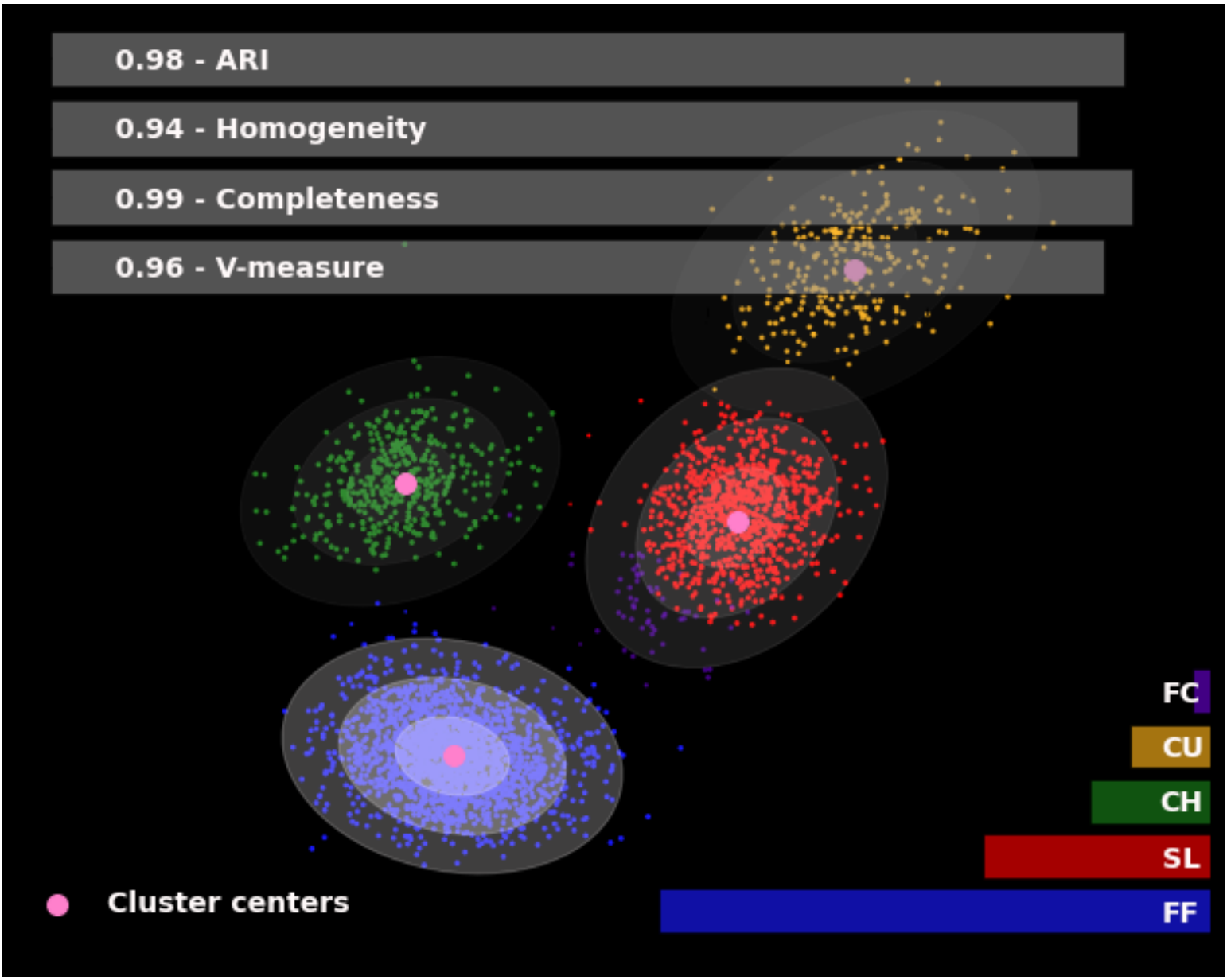
- PORCELLO, RICK #519144 (2016) -



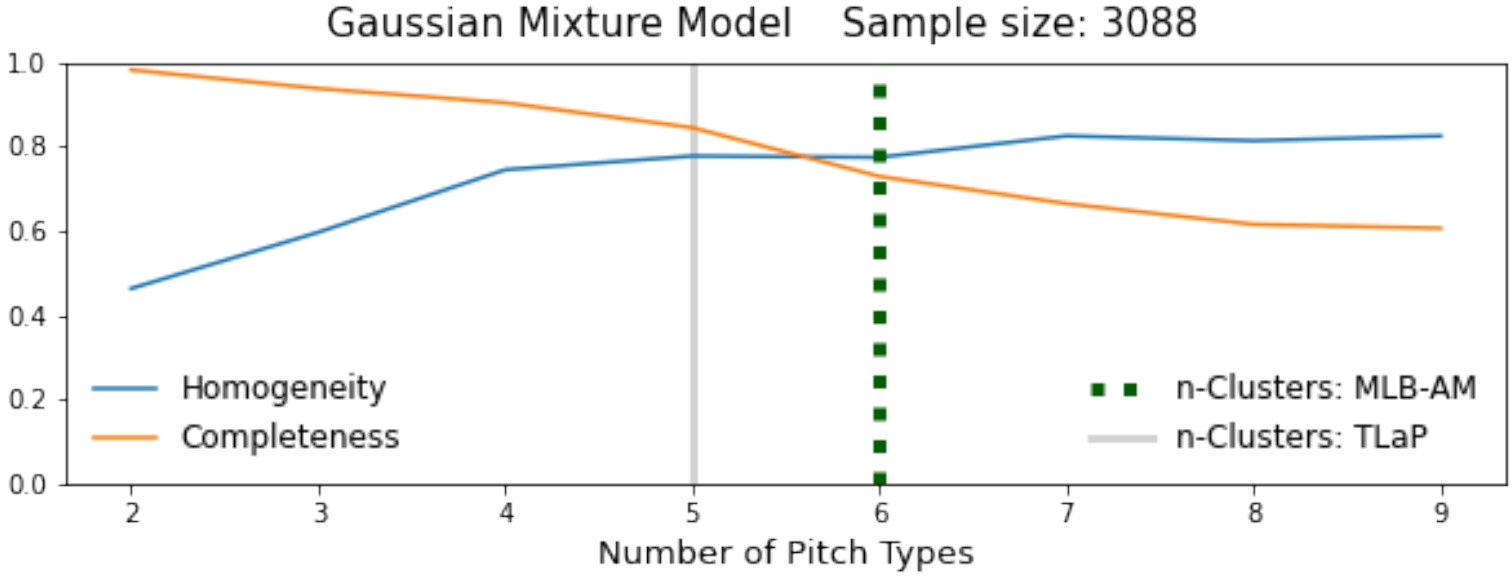


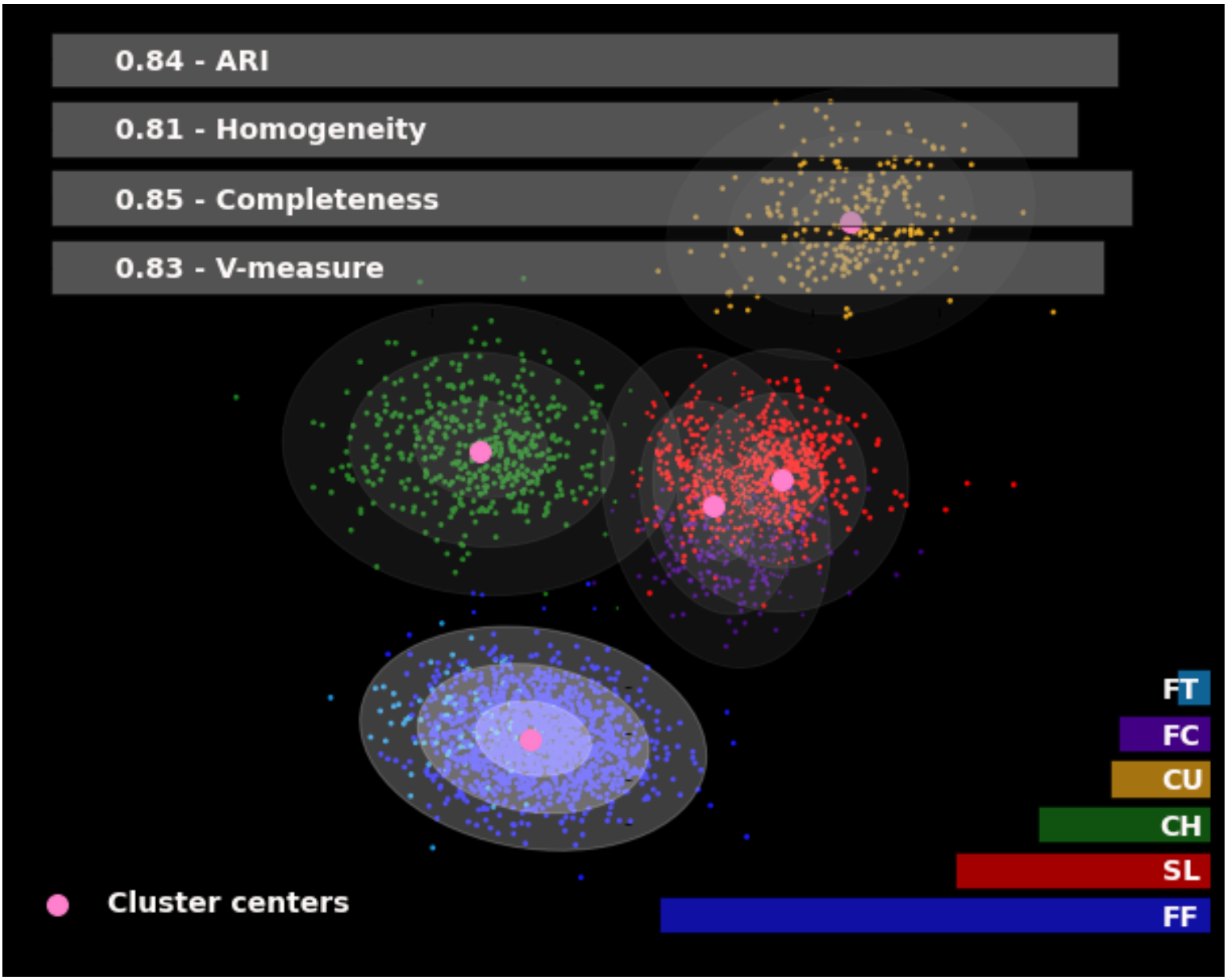
- SCHERZER, MAX #453286 (2016) -



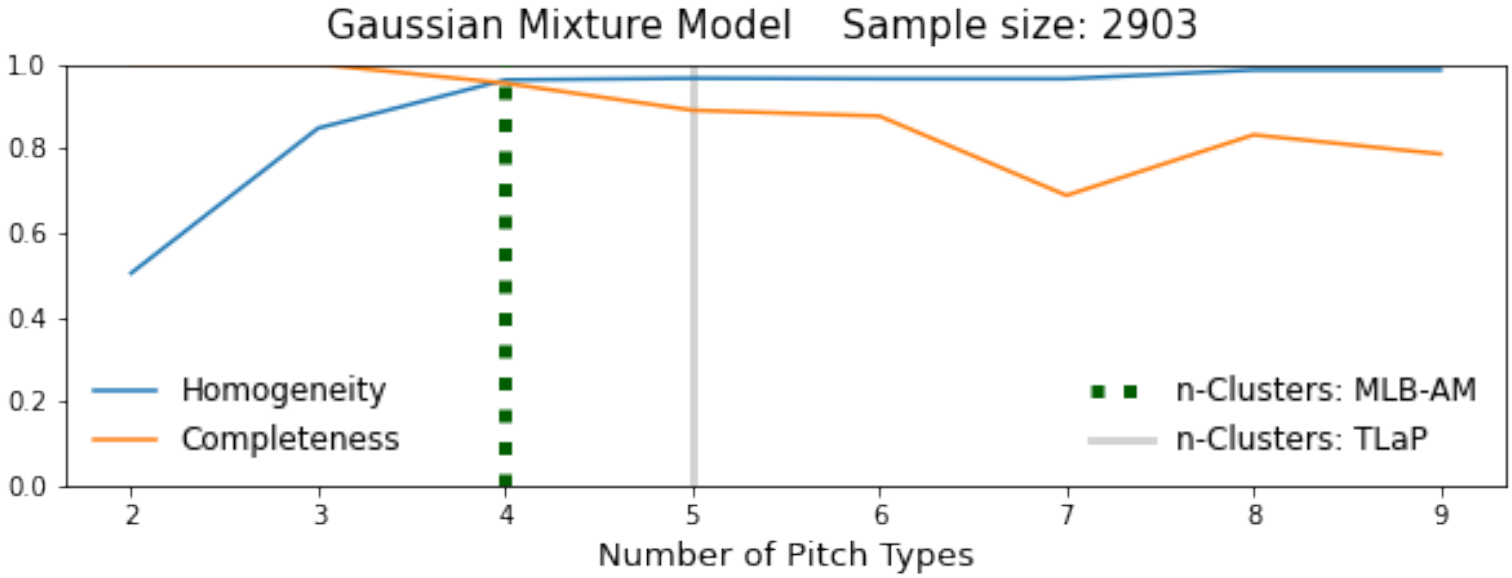


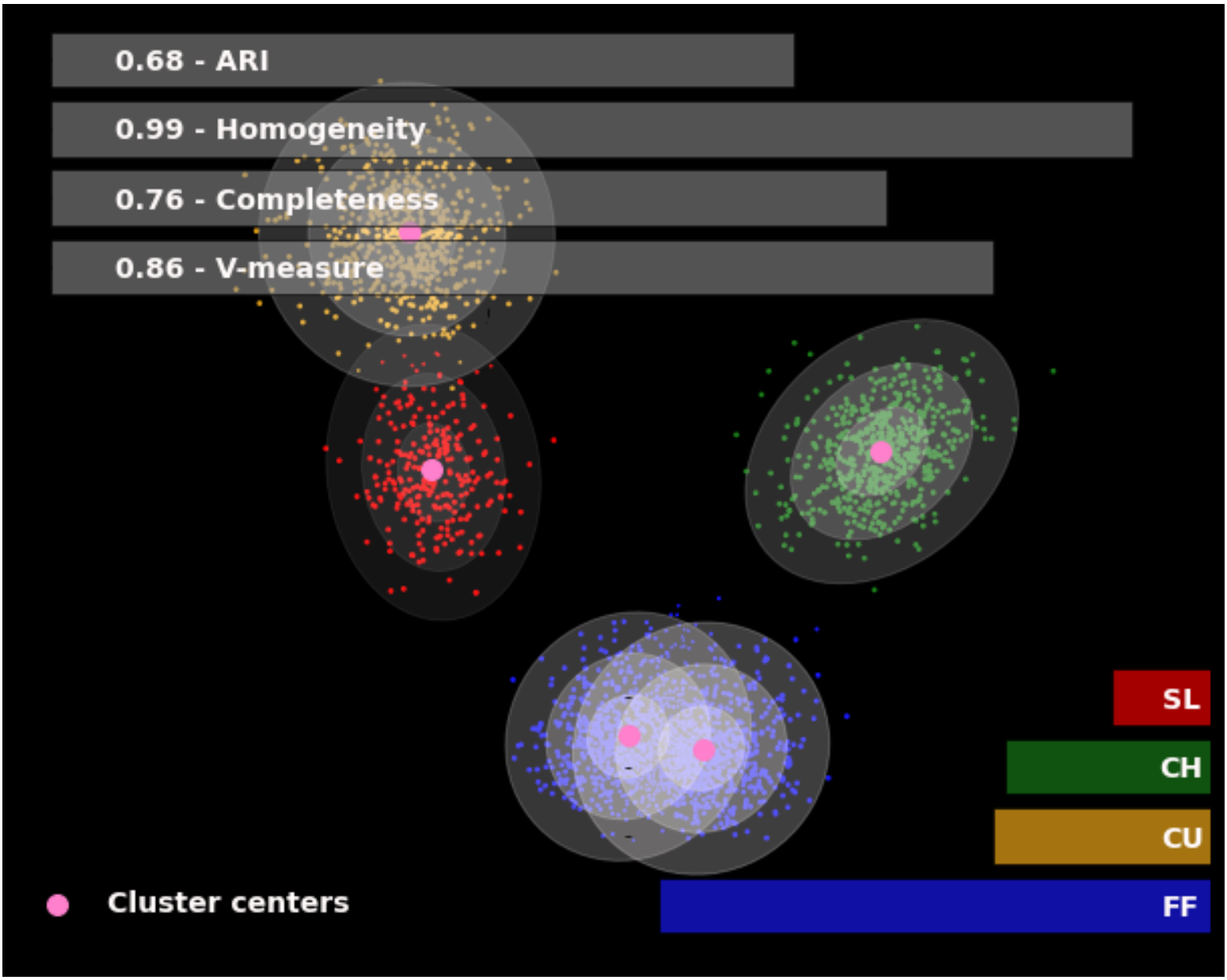
- SCHERZER, MAX #453286 (2017) -

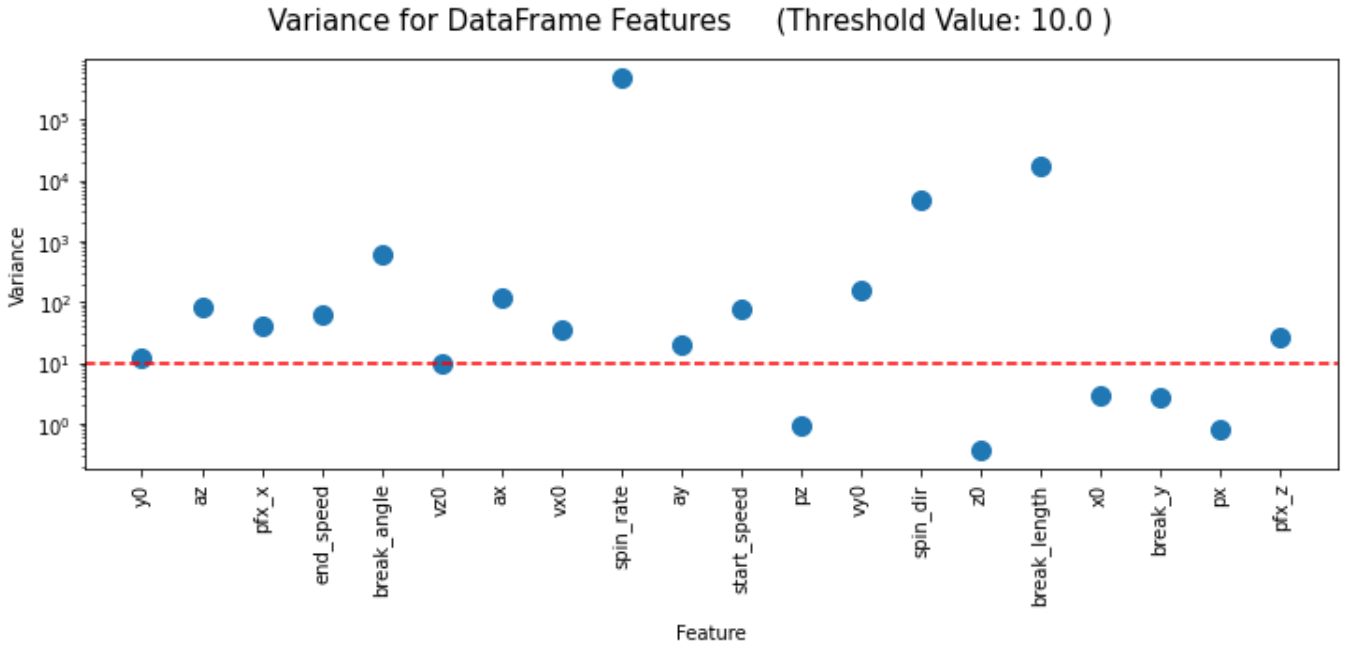


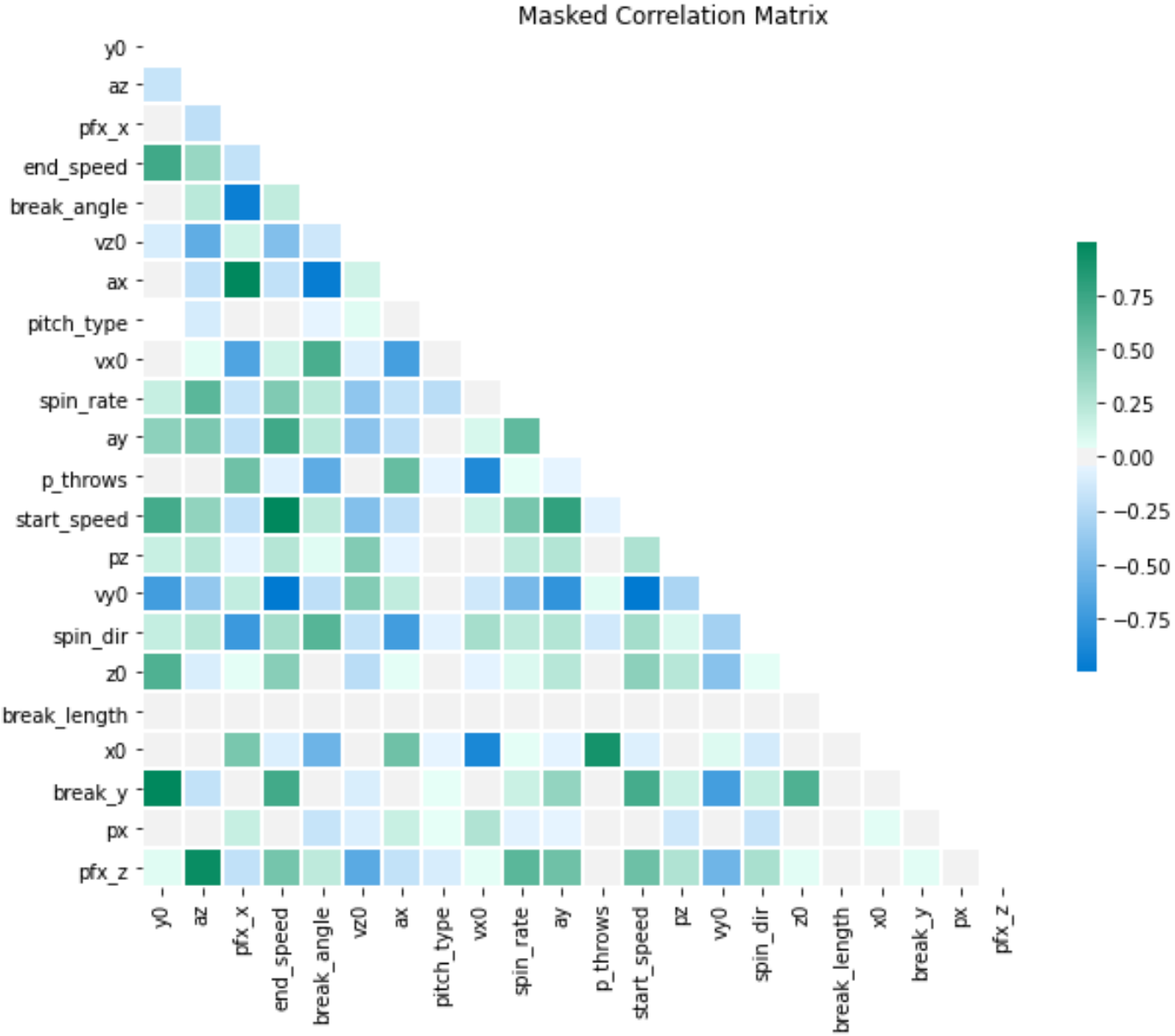


- SNELL, BLAKE #605483 (2018) -






Variance for DataFrame Features (Threshold Value: 10.0)
Variance
Feature
y0
az
pfx_x
end_speed
break_angle
vz0
ax
vx0
spin_rate
ay
start_speed
pz
vy0
spin_dir
z0
break_length
x0
break_y
px
pfx_z


Masked Correlation Matrix
y0
az
pfx_x
end_speed
break_angle
vz0
ax
pitch_type
vx0
spin_rate
ay
p_throws
start_speed
pz
vy0
spin_dir
z0
break_length
x0
break_y
px
pfx_z
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75

BIBLIOGRAPHY

Anderson, R.J. (2017, June 6). How Statcast has changed MLB and why not everybody seems all that happy about it. Retrieved on April 17, 2020, from CBSsports.com

Arthur, Rob. (2017, April 28). Baseball's New Pitch-Tracking System Is Just A Bit Outside. Retrieved on April 17, 2020, from FiveThirtyEight.com

Bernier, Doug. (n.d.). *Types of Pitches*. [Web log post]. Retrieved April 16, 2020, from probaseballinsider.com.

Bieler, Des. (2017, February 21). MLB is eliminating the four-pitch intentional walk, and not everyone is thrilled. Retrieved on April 15, 2020, from WashingtonPost.com

Boswell, Thomas. (1979, March 21). Koufax: Hall of Famer Back in Baseball After Years of 'Wandering'. *The Washington Post* Retrieved on April 15, 2020, from https://www.washingtonpost.com

Casella, Paul. (2015, April 24). Statcast primer: Baseball will never be the same. Retrieved on April 15, 2020, from MLB.com

Diemert, Joshua. (2017, April 6). Did any pitchers actually throw harder on Opening Day? Retrieved on April 15, 2020, from pinstripealley.com

Dimeo, Nate. (2007, August 15). Baseball's Particle Accelerator: The new technology that will change statistical analysis forever. Retrieved on April 15, 2020, from Slate.com

Dhakar, Lokesh. (2017, September 19). *Baseball pitches illustrated*. Retrieved on Jun 27, 2020 from lokeshdhakar.com

Fast, Mike. (2007, August 2). Glossary of the Gameday pitch fields. *Fast Balls*. Retrieved on April 16, 2020, from FastBalls Blog

Kim, DT. (2020, August 4). Pitches of Baseball – Part 1: Types. Retrieved on August 27, 2020, from Medium.com

Krebs, Caitlin. (2013). *Analysis of Softball Pitch Trajectories by the Cluster Method*. Retrieved on May 6, 2020, from The Physics of Baseball

Marchi, Max. (2011, February 25). Fine tuning PITCHf/x location data. *The Hardball Times*. Retrieved on April 18, 2020, from FanGraphs.com

Mills, Brian. (2015, January 11). Pitch Classification with Mclust. *Exploring Baseball Data with R*. Retrieved May 8, 2020, from BaseballWithR.com

Nathan, Alan M. (2007, December 21). *Effect of the Magnus Force in the PITCHf/x Tracking System.* Retrieved on August 18, 2020, from The Physics of Baseball

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O. and ... Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, 2825-30.

Schifman, Gerald. (2018, March 27). The Lurking Error in Statcast Pitch Data. *The Hardball Times*. Retrieved on April 17, 2020, from FanGraphs.com

Sharma, Pulkit. (2018, August 27). The Ultimate Guide to 12 Dimensionality Reduction Techniques (with Python codes). *Analytics Vidhya*. Retrieved on July 23, 2020, from AnalysticsVidhya.com.

Slowinski, Steve. (2012, April 4). Pitch Type Abbreviations & Classifications. Retrieved on April 16, 2020, from FanGraphs.com

Staff. (2015, November 18). Cubs' Jake Arrieta wins 2015 NL Cy Young. *BBWAA*. Retrieved May 8, 2020, from Baseball Writers Assoc. of America

Staff. (2015, November 18). Dallas Keuchel becomes 1st Astros' pitcher to win AL Cy Young. *BBWAA*. Retrieved May 8, 2020, from Baseball Writers Assoc. of America

BIBLIOGRAPHY

Staff. (2016, November 16). Max Scherzer adds NL Cy Young after winning in AL. *BBWAA*. Retrieved May 8, 2020, from Baseball Writers Assoc. of America

Staff. (2016, November 16). Rick Porcello narrowly wins AL Cy Young. *BBWAA*. Retrieved May 8, 2020, from Baseball Writers Assoc. of America

Staff. (2017, November 15). Back-to-back: Nationals' Max Scherzer wins 2nd consecutive Cy Young, 3rd overall. *BBWAA*. Retrieved May 8, 2020, from Baseball Writers Assoc. of America

Staff. (2017, November 15). Indians' Corey Kluber wins 2nd Cy Young in 4 seasons. *BBWAA*. Retrieved May 8, 2020, from Baseball Writers Assoc. of America

Staff. (2018, November 14). Mets' ace Jacob deGrom rolls to landslide victory for Cy Young Award on the strength of 1.70 ERA. *BBWAA*. Retrieved May 8, 2020, from Baseball Writers Assoc. of America

Staff. (2018, November 14). Rays' Blake Snell wins tight race for AL Cy Young. *BBWAA*. Retrieved May 8, 2020, from Baseball Writers Assoc. of America

Tangotiger Blog. (2017, April 5). Pitch velocity: new measurement process, new data points. Retrieved on April 17, 2020, from Tangotiger.com

Wikipedia contributors. (2020, April 30). Doug Bernier. In Wikipedia, The Free Encyclopedia. Retrieved 00:34, May 7, 2020, from Wikipedia.org

Wikipedia contributors. (2019, December 26). Pitch (baseball). In Wikipedia, The Free Encyclopedia. Retrieved 16:36, April 16, 2020, from Wikipedia.org