A Comparison of Feature Selection and Classification Algorithms in Identifying Baseball Pitches

Adam Attarian, George Danis, Jessica Gronsbell, Gerard Iervolino, and Hien Tran *Member, SIAM*

***Abstract*—Data classification has been widely studied in engineering, market and financial analysis, medicine and biology, and other areas. In particular, there exists a large body of literature concerning classification algorithms. However, many classification algorithms are data dependent. For example, while the k-nearest neighbors algorithm (k-NN) is quite versatile in its application, most parametric classifiers are limited in their use by assumptions. Using data from Sportvision's PITCHf/x, we apply several generic classification methods to the problem of pitch classification. We place a particular emphasis on improving the accuracy of Bayesian classifiers through feature selection and dimension reduction via principal component analysis (PCA) and linear discriminant analysis (LDA), respectively. The accuracy and speed of these classification algorithms are then analyzed and compared.**

***Index Terms*—feature selection, Bayesian classifiers, principal component analysis, linear discriminant analysis, Sportvision's PITCHf/x.**

I. Introduction

DURING televised baseball games, color commentators often classify pitches based on various observable features such as speed, horizontal/vertical movement, rotation, etc. Yet even for the experienced commentator, this process is extremely subjective due to pitch idiosyncrasies; for example, the difference between a curveball and a slider may be more apparent for one pitcher than for another due to a pitcher's unique tendencies and abilities. Using machine learning techniques to consistently assign class membership can greatly reduce such human error in classification.

Data classification refers to the problem of assigning class membership to unlabeled observations. In this paper, we apply generic classification techniques to baseball pitch data from ten major league baseball (MLB) pitchers in the 2011 season. We begin with a survey of commonly used classification algorithms. In Section 2, our focus is on Bayesian classifiers, that is, those that assign class membership based on an estimated probability density function. Section 3 provides an overview of linear discriminant analysis (LDA) and principal component analysis (PCA). In Section 4, we provide a description of the PITCHf/x data set. PITCHf/x, which is created and maintained by Sportvision, is a pitch tracking system and is installed in every major league baseball (MLB) stadium since circa 2006. The system allows pitches to be analyzed and compared with unprecedented precision. In Section 5, we apply Bayesian classifiers to the problem of pitch classification using data set from PITCHf/x with a particular emphasis on improving classification accuracy with feature selection and dimension reduction via PCA and LDA, respectively. Finally, concluding remarks are provided in Section 6.

Manuscript received January 8, 2013; revised January 30, 2013. This work was carried out as part of the NC State Research Experiences for Undergraduates in Mathematics and was supported in part by the National Security Agency under Grant H9823-10-1-0252 and by the National Science Foundation under Grants DMS-1063010 and DMS-0943855.

A. Attarian was a graduate student in the Department of Mathematics, North Carolina State University, Raleigh, NC 27695 and is currently with the Advanced Concepts and Technologies, MIT Lincoln Laboratory, Lexington, MA 02420 USA e-mail: Adam.Attarian@ll.mit.edu.

G. Danis is an undergraduate student in the Department of Mathematics & Statistics, Boston University, Boston, MA 02215 USA e-mail: gjdanis@bu.edu.

J. Gronsbell is a graduate student in the Department of Biostatistics, Harvard School of Public Health, Boston, MA 02115 USA e-mail: jlg735@mail.harvard.edu.

G. Iervolina is an undergraduate student in the Department of Mathematics, Chapel Hill, NC 27599 USA e-mail: giervoli@live.unc.edu.

H. Tran is with the Department of Mathematics, North Carolina State University, Raleigh, NC 27695 e-mail: tran@math.ncsu.edu.

II. Bayesian Classifiers

A classifier is a function f that maps an unlabeled data observation $\boldsymbol{x} = [x_1, x_2, \ldots, x_l]$ consisting of l feature values to one of c classes denoted by $\omega_1, \omega_2, \ldots, \omega_c$. Prior to classification, f is learned from a training set $T = \{t_1, t_2, \ldots, t_N\}$ of N input-output pairs $(\boldsymbol{t}_i, \omega_i)$ [1].

Bayesian classifiers assign class membership to an unlabeled observation through maximization of a posterior probability density function. These density functions are calculated via the Bayes Rule:

$$P(\Omega|\boldsymbol{x}) = \frac{P(\boldsymbol{x}|\Omega)P(\Omega)}{P(\boldsymbol{x})}, \tag{1}$$

where Ω denotes the set of classes, $P(\boldsymbol{x}|\Omega)$ is the likelihood, $P(\Omega)$ is the prior, and $P(\boldsymbol{x})$ is a normalizing factor [2]. We describe two supervised Bayesian Classifiers, the k-Nearest Neighbors algorithm (k-NN) and the Naive Bayes classifier.

A. k-Nearest Neighbors (k-NN)

The k-nearest neighbors algorithm is a simple and intuitive classification method yet often outperforms more sophisticated learners. The method can be summarized as follows ([3]):

k-NN Algorithm:

- User chooses a value for k
- Calculate the distance between $\boldsymbol{x}$ and all training points $\boldsymbol{t}_i$ for $i = 1, 2, \ldots, N$ (using user defined metric)
- Find the k nearest points to $\boldsymbol{x}$
- Classify $\boldsymbol{x}$ based on the class membership of the training points by majority vote

ISBN: 978-988-19251-8-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

The output of this algorithm is clearly dependent on both the user defined metric and the value of k. Although a common heuristic is to set k equal to $\sqrt{N}$ [3], metric choice is data dependent. While squared Euclidean distance

$$\sum_{i=1}^{l}(x_i - t_i)^2 \tag{2}$$

is a standard choice, statistical regularities may often be better estimated by other metrics such as Mahalanobis distance, which accounts for nonzero covariances between features, and Manhattan distance, a generalization of the Minkowski function [3]. These metrics are defined as follows:

Mahalanobis distance:

$$D^i_{Mahal}(\boldsymbol{x}) = \sqrt{(\boldsymbol{x} - \mu_i)^T \Sigma_i^{-1} (\boldsymbol{x} - \mu_i)}.$$

where μ_i and Σ_i, $i = 1, 2, \ldots, c$ are the mean and covariance of the ith class.

Manhattan distance:

$$D_{Man}(\boldsymbol{x}, \boldsymbol{t}) = \sum_{i=1}^{l} |x_i - t_i|.$$

It is important to note that while k-NN does not explicitly involve the computation of pdf's, it is often grouped with Bayesian classifiers because of its ties to Bayes Decision Theory [4].

B. Naive Bayes

The Naive Bayes classifier computes the posterior probability that a feature vector $\boldsymbol{x}$ is in class ω_i for $i = 1, 2, \ldots, c$ given its particular set of features via Bayes rule. The main assumption of the Naive Bayes classifier is that the presence (or absence) of a particular feature of a class is unrelated to the presence (or absence) of any other feature, given the class variable; that is, the features are conditionally independent given the class label. Therefore, equation (1) becomes

$$P(\Omega | x_1, x_2, \ldots, x_l) = \frac{\prod_{i=1}^{l} P(x_i|\Omega) P(\Omega)}{\prod_{i=1}^{l} P(x_i)}. \tag{3}$$

As the denominator in equation (3) is simply a normalizing factor, the class membership of $\boldsymbol{x}$ may be calculated according to the following rule

$$\arg\max_{\Omega} \prod_{i=1}^{l} P(x_i|\Omega) P(\Omega).$$

Although the independence assumption of the Naive Bayes classifier is most often false since the features are usually dependent, the resulting model is easy to fit and works surprisingly well in practice [2].

III. Dimension Reduction and Feature Selection

A. Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) can be used to reduce data dimensionality while preserving as much class discriminatory information as possible. Intuitively, for the two-class problem, this amounts to finding the optimal linear projection of the data $y = \boldsymbol{w}^T \boldsymbol{x}$ that maximizes class separability. While it would seem natural to choose the distance between the projected means as a measure of separation, this distance does not account for within-class variance [2]. Fisher's separation criteria provides a much better indicator of separation and is given as follows.

For each class, define the *scatter* as

$$S_i = \sum_{\boldsymbol{x} \in \omega_i} (\boldsymbol{x} - \mu_i) (\boldsymbol{x} - \mu_i)^T,$$

where μ_i is the mean of the ith class. Also define the quantity $(S_1 + S_2)$ as the within-class scatter matrix S_w of the projected examples, the quantity $\boldsymbol{w} (\mu_1 - \mu_2) (\mu_1 - \mu_2)^T$ as the between-class scatter matrix S_b and the criterion function $J(\boldsymbol{w})$ as

$$J(\boldsymbol{w}) = \frac{\boldsymbol{w}^T S_b \boldsymbol{w}}{\boldsymbol{w}^T S_w \boldsymbol{w}}. \tag{4}$$

Maximizing equation (4) effectively yields an optimal projection where data points from the same class are projected close to each other while the projected means remain as far apart as possible. A solution to $\arg\max [J(\boldsymbol{w})]$ is then given by solving

$$S_w^{-1} S_b \boldsymbol{w} = \lambda \boldsymbol{w},$$

where $\lambda = J(\boldsymbol{w})$ is a generalized eigenvalue problem.

LDA can be readily extended to the c-class problem and used to reduce data dimension to $c - 1$ [2]. For a multi-class classification problem, the within-class and between-class scatter are defined as follows:

$$S_w = \sum_{i=1}^{c} S_i,$$

$$S_b = \sum_{i=1}^{c} N_i (\mu_i - \mu) (\mu_i - \mu)^T,$$

where $S_i = \sum_{\boldsymbol{x} \in \omega_i} (\boldsymbol{x} - \mu_i) (\boldsymbol{x} - \mu_i)^T$, μ_i and N_i are the mean and sample size of the ith class, and μ is the global mean. As the projection matrix $\boldsymbol{w}$ we are looking for is no longer a scalar (rather, it has dimension $c - 1$), the ratio in equation (4) is instead maximized by taking the determinant of both the numerator and the denominator. Similar to the two-class case, the solution to this expression may be solved by the generalized eigenvalue problem $(S_b - \lambda_i S_w) \boldsymbol{w}_i = 0$.

B. Principal Component Analysis

Principal component analysis (PCA) is a widely used dimension reduction technique that performs a linear transformation on a set of data using the so called *principal components* that best depict the variance within the data set. If we let X be our data (consisting of n observations and l features), P represent an orthonormal matrix where each row corresponds to a principal component and Y be the matrix of transformed data, PCA can be represented as:

$$PX = Y. \tag{5}$$

More specifically, our goal in using PCA is to compute an orthonormal basis that will (i) reduce the dimension of our data and (ii) capture the variance of our data. This is accomplished by transforming our data so that the covariances between the distinct features are zero. Mathematically, our problem becomes solving for an orthonormal matrix P such that $\mathrm{Cov}(Y)$ is diagonal.

ISBN: 978-988-19251-8-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

By definition,

$$\text{Cov}(Y) = \frac{1}{n-1} YY^T. \quad (6)$$

Substituting in equation (5), equation (6) becomes

$$\begin{aligned}\text{Cov}(Y) &= \frac{1}{n-1} PX(PX)^T \\ &= \frac{1}{n-1} PGP^T,\end{aligned}$$

where $G = XX^T$. Since G is symmetric,

$$G = EDE^T,$$

where D is diagonal and the columns of E contain the eigenvectors of G. Letting $P = E^T$, ie. letting the rows of P be the eigenvectors of $\text{Cov}(X)$, we get the desired result

$$\begin{aligned}\text{Cov}(Y) &= \frac{1}{n-1} PP^T DPP^T \\ &= \frac{1}{n-1} D\end{aligned}$$

since P is an orthogonal matrix and therefore $P^T = P^{-1}$.

Intuitively, we are calculating the first principal component to account for the most variance within our data, the second component to capture the second most variance and also be orthonormal to the first principal component and so on. This process is repeated until we obtain l principal components. Our hope is to solve for $r < l$ principal components that capture most of the variance so that we can reduce the dimensionality of our data set from l to r. This is achieved by multiplying X by an $r \times l$ matrix comprised of the eigenvectors of $\text{Cov}(X)$ associated with the r largest eigenvalues, i.e. the r most dominant principal components [5].

While the above implementation provides us with a transformed set of data, it is often not useful in classification tasks. Unlike LDA, PCA is an unsupervised dimension reduction technique and thus class separation is often lost in the transformation [6]. It is therefore more useful for us to use PCA for feature selection. The most dominant features can be identified via a QR decomposition of $\text{Cov}(X)$,

$$\text{Cov}(X)E = QR,$$

where Q is unitary, R is upper triangular, and E is a permutation matrix such that the column permutation of E is chosen so that the values along the diagonal of R are decreasing and thus indicate the most dominant features of our data set.

IV. Pitch Data

The datasets used in this paper are from Sportsvision's PITCHf/x tool, which tracks and records pitch data from Major League Baseball (MLB) games. Each MLB stadium is equipped with three tracking cameras. The cameras use sensors to follow the baseball from when it leaves the hand of the pitcher until it crosses home plate, recording and registering characteristics in the system [7]. Overall, there are 41 qualitative and quantitative features that the PITCHf/x tool records [8] (see Appendix A for the complete list). Among them are 16 quantitative features including horizontal movement, vertical movement, spin rate, spin direction, break angle, and start speed, etc.

We randomly selected ten starting pitchers to examine from the 2011 MLB regular season. We then divided their total number of pitches into two halves; the first half of the pitches were used for the training set and the second half of pitches were used as the test set. Table I describes the size of the training and test sets, as well as the number of pitch types (fastball, cut fastball, change up, slider, curve, split-finger, etc.).

TABLE I
Data for each Pitcher

Pitcher	Training Size	Test Size	Pitch Types
Dickey	1473	1472	2
Hamels	1565	1564	4
Kershaw	1689	1688	4
Lester	1593	1593	5
Duke	607	606	4
Price	1728	1727	4
Sabathia	1745	1744	5
Verlander	1905	1905	4
Weaver	1878	1877	5
Buehrle	1571	1570	4

V. Methods

Six subsets of data from the ten pitchers were analyzed using four supervised learning algorithms: the k-NN classifier with squared Euclidean distance, the k-NN classifier with Mahalanobis distance, the k-NN classifier with Manhattan distance, and the Naive Bayes classifier. The first three subsets of data investigated used two quantitative features from the PITCHf/x website: horizontal and vertical movement, horizontal movement and start speed, and vertical movement and start speed. PCA was then used to identify the most dominant features across all ten pitchers and classification was performed using the four and two most dominant features from this analysis. LDA was also used as a dimension reduction technique to obtain a $c-1$ dimensional transformed data set. These six subsets, as depicted in Table II, are compared to the data set comprised of all 16 quantitative valued features.

TABLE II
Data Subset Legend

Data Subset	Description
HV	Using horizontal movement and vertical movement
VS	Using vertical movement and start speed
HS	Using horizontal movement and start speed
PCA2	Two most important features determined by PCA
PCA4	Four most important features determined by PCA
LDA	Data reduced by LDA (to $c-1$)

VI. Results

To get a baseline for accuracy and speed improvement, we first ran all four classification algorithms on the data consisting of the 16 quantitative features from the PITCHf/x data set (see appendix for a complete list of all features). With this data, k-NN with Manhattan distance performed best (on average of all ten pitchers, 93.63% of pitches classified correctly by the k-NN with Manhattan distance classifier), more than 4% better accuracy as compared to the Naive Bayes classifier (see Table III). However, the Naive Bayes

ISBN: 978-988-19251-8-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

TABLE III
ACCURACY COMPARISON USING ALL 16 QUANTITATIVE FEATURES. SYMBOLS: k-NN WITH EUCLIDEAN DISTANCE (k-NN), k-NN WITH MANHATTAN DISTANCE (k-NN MAN), k-NN WITH MAHALANOBIS DISTANCE (k-NN MAHAL),AND NAIVE BAYES (NB)

Classifier	Dickey	Hamels	Kershaw	Lester	Duke	Price	Sabathia	Verlander	Weaver	Buehrle	Average
k-NN	99.3886	96.2916	97.9858	82.5487	91.2541	89.3457	95.2982	89.7638	87.7997	89.9363	91.9613
k-NN Man	98.9810	94.4373	96.2678	81.9209	87.4587	85.2924	94.2661	89.0814	87.2136	88.4713	93.6295
k-NN Mahal	99.4564	97.7621	98.519	85.2725	93.0693	93.6303	96.4450	90.2887	89.8775	91.9745	90.3391
NB	99.7962	93.4142	98.3412	88.8890	92.0792	64.9102	92.8326	88.8714	85.8817	88.8714	89.3887

TABLE IV
CPU TIMES (SECONDS)

CPU Time	Dickey	Hamels	Kershaw	Lester	Duke	Price	Sabathia	Verlander	Weaver	Buehrle
k-NN	1.29	1.87	1.61	2.04	0.57	1.68	2.36	2.06	2.24	2.28
k-NN Man	0.28	1.38	1.54	1.5	0.66	1.57	1.65	1.8	1.81	1.36
k-NN Mahal	0.52	0.63	0.67	0.71	0.13	0.73	0.79	0.89	0.92	0.66
NB	0.02	0.09	0	0.01	0.01	0.02	0.03	0.03	0.04	0.04

classifier was the most computationally efficient algorithm as depicted in Table IV.

Table V depicts the average accuracies among all ten pitchers as compared across various combinations of features and data reduction. Results using two features, namely, horizontal movement and start speed, produce the best accuracies with all four classification algorithms. It is noted that applying LDA to reduce data dimension (to size $c-1$) show considerable decrease in accuracy (average classification accuracy with LDA - 72.18%; and without - 86.36%). This is due in most part to low accuracy readings from four pitchers (Duke, Price, Verlander and Weaver) whose classification accuracy decreased considerably. For this reason, it is interesting to remove these pitchers from all calculations of accuracy, both pitcher specific accuracies and algorithm specific accuracies. Doing so shows that LDA can be applied to greatly improve classification performance; that is, results for pitchers Dickey, Hamels, Kershaw, Lester, Sabathia, and Weaver show individual improvement with LDA (see Table VI).

TABLE V
TABLE OF ACCURACIES: FEATURE SELECTION AND DATA REDUCTION

Features	k-NN	k-NN Mahal	k-NN Man	NB
HS	94.2207	93.9591	94.1055	89.9257
HV	85.2484	85.0569	85.1084	78.978
VS	82.0438	81.1194	81.8329	79.4
LDA	72.2211	71.5969	72.1757	72.7302
PCA2	85.3128	83.5532	83.5827	77.6029
PCA4	92.894	92.0231	93.0323	88.2861

TABLE VI
PITCHER ACCURACY IMPROVEMENT (AVERAGED OVER THE 4 BAYESIAN CLASSIFIER ALGORITHMS) WITH LDA

Pitcher	Accuracy Improvement
Dickey	2.27098571
Hamels	0.52068571
Kershaw	1.15972857
Lester	1.76307143
Sabathia	0.9993
Weaver	3.88151429
Average	1.76588095

Compared to LDA's improvement, the added improvement of classifying pitches based on the four most dominant features—spin rate, spin direction, break angle, and start speed—was rather minute and in some cases diminished classification accuracy. For example, for k-NN with Mahalanobis distance PCA4 improves classification accuracy by 1.68% (when compared with the average accuracy over all ten pitchers as shown in the last column of Table III). However, the Naive Bayes classifier decreases the accuracy from 89.39% (with all 16 quantitative features) to 88.29% (with 4 most dominant features). It also noted that there was no accuracy improvement when taking the two most dominant features determined by PCA; compared to classification algorithms applied to data using all 16 quantitative features, PCA2 yielded a loss of 8.82% classification accuracy on average.

VII. CONCLUSION

While we obtained a relatively low misclassification rate overall with our generic classification methods, we had significantly less accurate results with four of the ten pitchers using LDA. LDA is a good starting point for dimension reduction but is subject to data specifics. Namely, LDA performs best when the discriminatory information is contained within the mean rather than the variance and can therefore fail on data sets whose classes are separated by variance rather than mean. Because of this, overlapping and tightly clustered classes can greatly reduce the efficacy of LDA. As discussed below, the PITCHf/x classification scheme is subject to this issue and therefore LDA was not the best choice for dimension reduction on our particular data set. Accessing in-between and within-class scatter matrices before and after performing LDA would address this issue.

Being a supervised learning task, we used the classification scheme provided by PITCHf/x. This scheme, however, has significant overlap between different classes due to the subjectivity involved in pitch classification. It would be beneficial to use an unsupervised technique to come up with a classification scheme that provides better separation between classes and perhaps more uniform guidelines for pitch classification. The k-means algorithm, for example, could be used to find new class labels. The Gaussian Mixture Model would also be a good choice for this task, that is, fitting multiple GMMs by varying the number of components using different subsets of features to find what best encapsulates our data.

Creating multi-dimensional classification algorithms would further improve robustness. Multi-layer algorithms could be created that access the separation of classes with

ISBN: 978-988-19251-8-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

different combinations of features and perform classification using the features that provide the most discriminatory information at each layer.

Lastly, while pitch classification is useful for statistical records, pitch prediction is more useful in practice. Although classification algorithms cannot provide us with the ability to predict the next pitch, we can alter our generic classifiers to see how accuracies change across different game settings, i.e. ball park locations, score differentials, pitch count. For example, we can add additional features when performing linear classification to see if a specific game setting has bearing on the accuracy of classification. We could access whether a 3-0 pitch count affects our classification rate by creating a new binary variable that takes on the value of 1 when we are at a 3-0 count and a zero otherwise. Thus, the pitch count affect is only taken into account when it is 3-0 and can be compared to a linear classifier that excludes this binary variable.

APPENDIX A
RECORDED FEATURES BY SPORTSVISION'S PITCHF/X

The following 41 features are recorded by Sportsvision's Pitchf/x tool [9]:

1) Pitcher: name of pitcher
2) Pitcher Handedness: pitching hand of pitcher
3) gid: unique identification number per pitch within a specific game
4) Stadium: stadium at which game is being played
5) Umpire: home-plate umpire calling balls and strikes
6) Pitching Team: team of the pitcher
7) Batting Team: team of the batter
8) Pitch Count: number of pitches thrown at particular point in a game
9) Pitch Results: outcome of pitch, i.e. ball, strike, in play, etc
10) Result Type: abbreviated pitch result, i.e. B=Ball, S=Strike
11) Batters Faced: number of batters faced in a specific game
12) Atbat_pitch_num: number of pitches recorded against a specific batter
13) Pitch_type: classification of pitch type, i.e. FF=Fourseam Fastball, SL= Slider, etc
14) Type Confidence: the value of the weight at the classification algorithm's output node corresponding to the most probable pitch type
15) Atbat Result: outcome of at-bat, i.e. pop out, strikeout, single, double, etc
16) Batted Ball Type: abbreviated at-bat Result for balls put in play
17) Batter: name of batter
18) Batter Handedness: batting hand of batter
19) Balls: number of balls in the count
20) Strikes: number of strikes in the count
21) Outs: number of outs during an at-bat
22) On_first: binary column; display 1 if runner on first, 0 otherwise
23) On_second: binary column; display 1 if runner on second, 0 otherwise
24) On_third: binary column; display 1 if runner on third, 0 otherwise
25) Inning: inning during a particular pitch
26) Start_speed: pitch speed, in miles per hours, measured from the initial position, y0.
27) End_speed: pitch speed when the ball crosses home plate.
28) sz_top: the distance in feet from the ground to the top of the current batter's rulebook strike zone
29) sz_bot: the distance in feet from the ground to the bottom of the current batter's rulebook strike zone
30) px: the left/right distance, in feet, of the pitch from the middle of the plate as it crosses home plate
31) pz: the height of the pitch in feet as it crosses the front of home plate
32) x0: the left/right distance, in feet, of the pitch, measured at the initial point
33) z0: the height, in feet, of the pitch, measured at the initial point
34) pfx_x: the horizontal movement of the pitch, in inches, from release point to home plate
35) pfx_z: the vertical movement of the pitch, in inches, from release point to home plate
36) Break_y: the distance in feet from home plate to the point in the pitch trajectory where the pitch achieved its greatest deviation from the straight line path between the release point and the front of home plate
37) Break_angle: the angle, in degrees, from vertical to the straight line path from the release point to where the pitch crossed the front of home plate, as seen from the catcher's/umpire's perspective
38) Break_length: the measurement of the greatest distance, in inches, between the trajectory of the pitch at any point between the release point and the front of home plate, and the straight line path from the release point and the front of home plate
39) Spin_dir: direction of spin of the baseball measured in degrees
40) Spin_rate: measured as the number of rotations the ball makes per minute
41) sv_id: a date/time stamp of when the PITCHf/x tracking system first detected the pitch in the air; it is in the format YYMMDD_hhmmss

ACKNOWLEDGMENT

The authors would like to thank Drs. Lori Layne and David Padgett of MIT Lincoln Laboratory for their scientific guidance and expertise.

REFERENCES

[1] R. Duda, P. Hart, and D. Stork, *Pattern classification*, ser. Pattern Classification and Scene Analysis: Pattern Classification. Wiley, 2001. [Online]. Available: http://books.google.com/books?id=YoxQAAAAMAAJ

[2] S. Theodoridis and K. Koutroumbas, *Pattern Recognition, Third Edition*. Orlando, FL, USA: Academic Press, Inc., 2006.

[3] R. Gutierrez-Osuna, "The k nearest neighbor rule (k-nnr)," 2002, k-NN Lecture Notes.

[4] S. Theodoridis, A. Pikrakis, K. Koutroumbas, and D. Cavouras, *Introduction to Pattern Recognition: A Matlab Approach*. Academic Press, 2010.

[5] J. Shlens, "A tutorial on principal component analysis," March 2003, derivation, Discussion and Singular Value Decomposition.

[6] A. M. Martínez and A. C. Kak, "Pca versus lda," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 2, pp. 228–233, Feb. 2001. [Online]. Available: http://dx.doi.org/10.1109/34.908974

ISBN: 978-988-19251-8-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

[7] (2012) Pitchf/x technology. [Online]. Available: http://www.sportvision.com/base-pitchfx.html
[8] (2012) Brooks baseball. [Online]. Available: http://www.brooksbaseball.net/
[9] M. Fast. (2007, August) Glossary of the gameday pitch fields. [Online]. Available: http://fastballs.wordpress.com/2007/08/02/glossary-of-the-gameday-pitch-fields/

ISBN: 978-988-19251-8-3
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)