Outlier Detection and Predicting the Game Outcome

Aytekin YILDIZHAN
aytekinyildizhan@hacettepe.edu.tr
N18147923

Ali Taha YÜKSELDİ
alitahayukseldi@gmail.com
N18234057

CMP712 Machine Learning Final Project Report

***Abstract*- In this project, we analyze the NBA (National Basketball Association) statistics data for predicting the game outcome and detecting outliers. Firstly, some feature selection algorithms are implemented and their MSE values are calculated by ANN. NCA-C shows a better performance. Then, some classification algorithms are applied by using NCA-C. SVM shows a better performance whereas Naïve Bayes shows a worse performance. After that, we classify the data using two hidden layer (2x2 and 64x64 neurons) pattern recognition network.** `tansig` **shows a better performance whereas** `hardlim` **shows a worse performance. Secondly, when it comes to outlier detection, our aim is finding the players which break apart from ordinary or standard ones. Our outlier dataset variables are greater than one (1) so we focus on multivariate outliers. In order to find the most contributing features, we used Principle Component Analysis (PCA). It helps us extracting decreased dimensional set of features from the features approximating between 17 to 21. Afterward we practices PyOD toolkit which is specifically designed for outlier detection. We use 5 algorithms, Isolation Forest, Feature Bagging Detector, k-NN, Cluster Based Local Outlier and Average k-NN. More or less results of first 4 of them are similar but Average k-NN performs worse than others.**

I. Introduction

There are 12 datasets in NBA statistics data [1] and it is used 'team.season.txt' for predicting the outcome and 'player_playoff_career.csv' is used for outlier detection. At 'team_season.txt' each data point is the performance of each team in a season. There are 36 features and 684 information of teams. These data include from 1946 to 2004 seasons. It is used the data beginning from the 1979 season since there are losses and inconsistencies before 1979 seasons. This part is implemented with the MATLAB R2017b. On the other hand, at 'player_playoff_career.csv' there are 19 features 2055 information is available. Outstanding players are detected based on different features such as points, assists, rebounds etc. Python PyOD library, which provides access to different algorithms for detecting outliers, is made use of during the second part of the work.

Various approaches for feature selections (SFS, Relieff, and NCA) would be applied along with ANN for team win ratio of per year prediction. For outlier detection, since Principle Component Analysis (PCA) is fast and flexible unsupervised method, it is used to reduce dimensionality in dataset.

Our report is constructed as follows: (2) report on related work that has been done in the past, (3) proposed methodologies for feature selection, classification and outlier detection model and results from our experiments with those methodologies and (4) conclusion.

II. Related Work

There are many work dealing with the topic of predicting the results of sports and detecting the outlier. Bunker and Thabtah [2] proposed a novel SRP-CRISP-DM framework for classification problem. They focused on predicting team sports results. In this work, it is used ANN and match-related features. Miljkovic et al. [3] presented a system for predicting the results of NBA games. It is used Naïve Bayes and multivariate linear regression methods. It is achieved the accuracy was of 67% (a total of 778 games in season 2009/2010. Wang et al. [4] improved the existing method which is a spanning tree-inspired clustering based outlier detection technique. In this paper, they found a new global outlier factor and by using this, they proposed a new minimum spanning tree (MST)-inspired k-NN-based outlier detection method. It is used in both synthetic and real high dimensional data. Their experiment showed that their work was successfully implemented.

III. Proposed Methods and Experimental Results

A. Data Preparation

First of all, we begin by analyzing the data which is 'team.season.txt'. The last two columns of our data are number of won and lost. To determine the classes, it is calculated the winning probability by the following formula (1):

$$win\_ratio = \frac{number\ of\ won}{number\ of\ won + number\ of\ lost} x100 \qquad (1)$$

The classification of teams is shown by Table 1.

TABLE 1. CLASSIFICATION OF TEAMS

Class	Win Ratio	Number of Teams
Class 1	0%-35%	135
Class 2	35%-50%	190
Class 3	50%-65%	227
Class 4	65%-100%	132

It is applied several feature selection methods. We have 36 features. We eliminate the 'team', 'leag', 'won' and 'lost' features since 'team' and 'leag' features are nominal (include words) and 'won' and 'lost' features tell about the class information so 32 features are left. We have to reduce the

dimensionality of data to help to lighten the cost in term of time and space for algorithms to analyze a big scale of data.

B. Feature Selection Algorithms

First, feature selection models are applied. Using 75% of the dataset as the training data and the remaining 25% as test data. The data was split into training and test randomly using the sampling method without replacement:

Sequential Feature Selection: There are backward and forward sequential selection and forward selection is applied. When forward selection is applied, one of the filter methods, QAD (Quadratic Discriminant Analysis) is used for increasing the success of selection. Then, we got 7 features being selected. These features (columns) are 12 31 26 20 3 16 11. ('o_to', 'd_pts', 'd_stl', 'd_fta', 'o_fga', 'o_pts', 'o_stl').

ReliefF Algorithm: ReliefF is an iterative, randomized, and supervised approach that estimates the quality of the features according to how well their values differentiate data samples that are near to each other; it does not discriminate among redundant features, and performance decreases with few data [5]. In this experiment, we use 10 nearest neighbors and the features are listed according to their ranking:

28 9 24 16 31 7 22 23 17 2 4 5 26 25 13 19 20 8 12 10 11 14 21 6 32 15 18 27 1 3 29 **30**

Therefore, 28th feature is the most important and 30th is the least important. We use the first ten important features, 28 9 24 16 31 7 22 23 17 2. ('d_blk', 'o_asts', 'd_asts', 'o_pts', 'd_pts', 'o_dreb', 'd_dreb', 'd_reb', 'd_fgm', 'o_fgm').

NCA (Neighborhood Component Analysis) for Classification and Regression Algorithm: NCA is a learning algorithm to measure the Mahalanobis distance used in the k-NN classification algorithm [6]. The weights of the irrelevant features should be close to zero. NCA for classification (NCA-C) result is shown by Figure 1.

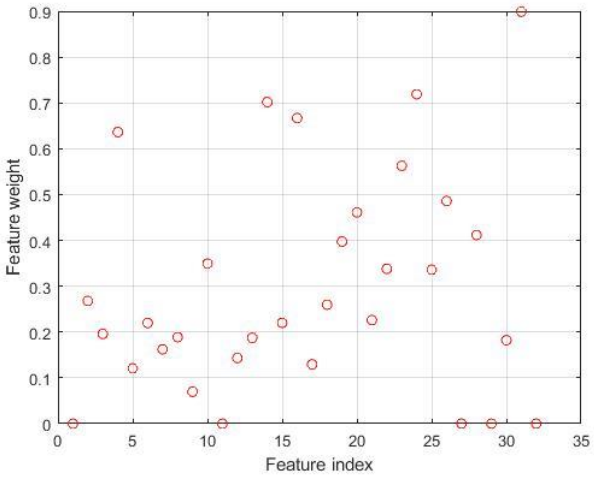

Fig. 1. NCA for Classification without error fitting

To improve the performance, we apply error fitting for classification and it is obtained Figure 2.

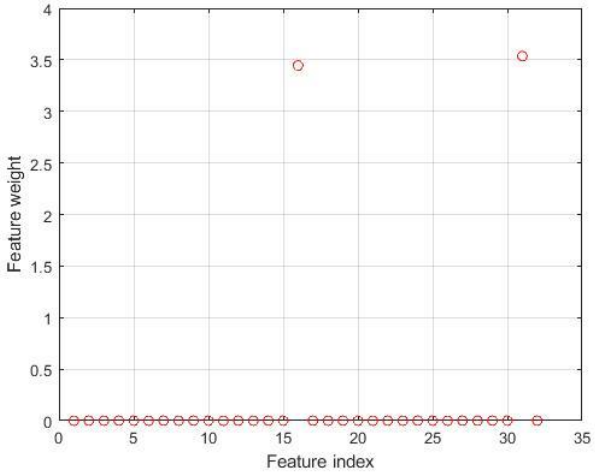

Fig. 2. NCA for Classification with error fitting

As you can see from Fig.2, 16th and 31st features are selected in this method ('o_pts' and 'd_pts').

NCA for regression (NCA-R) method is also applied and it is shown by Figure 3.

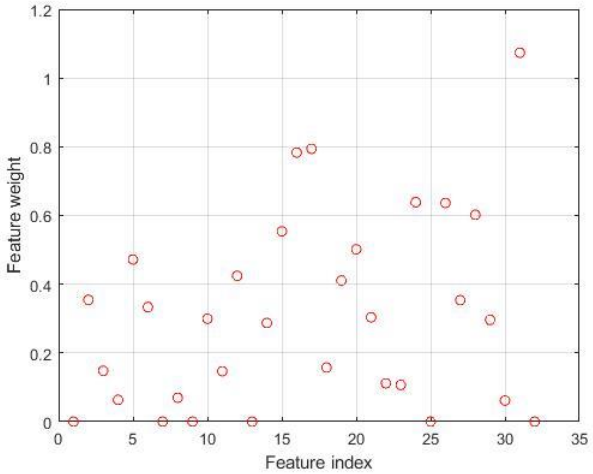

Fig. 3. NCA for Regression without error fitting

To improve the performance, we apply error fitting for regression and it is obtained Figure 4.

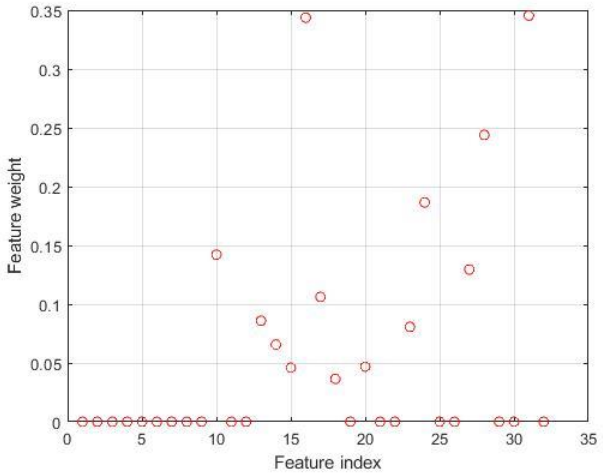

Fig. 4. NCA for Regression with error fitting

As you can see from Fig.4, 13 features are selected. These features are 10, 13, 14, 15, 16, 17, 18, 20, 23, 24, 27, 28 and 31 ('o_pf', 'o_blk', 'o_3pm', 'o_3pa', 'o_pts', 'd_fgm', 'd_fga', 'd_fta', 'd_reb', 'd_asts', 'd_to', 'd_blk', 'd_pts').

C. Team Performance Prediction

In this part, feature selection algorithms that mention about section B, have been compared to each other by using ANN.

During the ANN performance, we choose the number of neurons (R) of the single hidden layer. We consider the cases with R = 2, 64 and 128 neurons. The default values are initialized by MATLAB code `'feedforwardnet'`. Target value is winning ratio which is calculated by Eqn (1). For these ANN training, the first 60% of data is used to training, whereas the next 20% is used for the validation step and the remaining 20% for the testing step. Activation function is hyperbolic tangent sigmoid transfer function `(tansig)` and training function is Levenberg-Marquadt Back-Propagation `(trainlm)` algorithm. The errors are measured by Mean Squared Error (MSE). Every individual training trial is terminated after the completion of 5000 epochs, the epoch error increases for 10 consecutive epochs, MSE value is equal to zero or minimum gradient value is reached. Finally, the results of ANN are shown by Table 2-4.

TABLE 2. THE MSE RESULTS OF R = 2 NEURONS

	Without F.S.	**Forward F.S.**	**ReliefF**	**NCA-C**	**NCA-R**
Train Error	0.0132	0.0013	0.0013	0.0014	0.0086
Test Error	0.1310	0.0013	0.0010	9.2378e-04	0.0355
Validation Error	0.0266	0.0158	0.0026	0.0022	0.0178

TABLE 3. THE MSE RESULTS OF R = 64 NEURONS

	Without F.S.	**Forward F.S.**	**ReliefF**	**NCA-C**	**NCA-R**
Train Error	0.0085	5.3322e-04	0.0032	0.0014	6.5047e-04
Test Error	0.3481	0.0161	0.0097	0.1368	0.2352
Validation Error	0.0252	0.0209	0.0031	0.1368	0.0840

TABLE 4. THE MSE RESULTS OF R = 128 NEURONS

	Without F.S.	**Forward F.S.**	**ReliefF**	**NCA-C**	**NCA-R**
Train Error	4.4431e-06	0.0011	0.0010	0.0012	8.8884e-04
Test Error	0.2441	0.0032	0.0032	0.0141	0.2205
Validation Error	0.0639	0.0108	0.0807	0.0141	0.0684

As can be seen from Table 2-4, when R=2, NCA-C has the least error. When R=64, ReliefF has the least error. When R=128, ReliefF and forward feature selection has the least error. When all data are examined, the least error is NCA-C with 2 neurons. As a result, among for the feature selection algorithm, NCA-C has been used for team performance classification experiment.

D. Team Performance Classification

In this part, the project has been done to predict team performance by classification methods. Table 1 is used for class. SVM (Support Vector Machine), Naïve Bayes, Binary Decision Tree and k-NN (k-nearest neighbor) methods are used for classification methods. For k-NN methods, the distance function is Euclidean function and neighbor number is k=3. Using 75% of the dataset as the training data and the remaining 25% as test data. The data was split into training and test randomly using the sampling method without replacement. The accuracy of our classification methods are shown by Table 5.

TABLE 5. ACCURACY OF DIFFERENT CLASSIFIERS MODELS

	Without F.S.	**NCA-C**
SVM	77.28%	80.99%
Naïve Bayes	44.88%	35.38%
Binary Decision Tree	44.01%	69.74%
k-NN	53.07%	74.56%

As can be seen from Table 5, SVM performs better than the other methods. NCA-C increases the success of classification except Naïve Bayes.

E. Team Performance Classification with ANN

In the last part of classification, we use multi-layer pattern recognition ANN. This part compares four activation functions in multi-layer pattern recognition network.

In this experiment, it consists of two hidden layers and the activation functions in hidden layers are the same. During this

experiment, we choose the number of neurons (R) of the two hidden layers. We consider the cases with R = 2x2 and 64x64 neurons. The default values are initialized by MATLAB code `'feedforwardnet'`. Classes are determined as in Table 1. For these ANN training, the first 60% of data is used to training, whereas the next 20% is used for the validation step and the remaining 20% for the testing step. These functions are hyperbolic tangent sigmoid transfer function `(tansig)`, log-sigmoid transfer function `(logsig)`, linear transfer function `(purelin)`, and hard-limit transfer function `(hardlim)`. Output neuron is a soft max transfer function `(softmax)`.Training function is Levenberg-Marquadt Back-Propagation `(trainlm)` algorithm. The errors are measured by MSE. Activation function with the least MSE value will be determine. Every individual training trial is terminated after the completion of 5000 epochs, the epoch error increases for 10 consecutive epochs, MSE value is equal to zero or minimum gradient value is reached. Finally, the results of MSE are shown by Table 5 and 6.

TABLE 6. THE MSE RESULTS OF R = 2X2 NEURONS

	tansig(x2)	logsig(x2)	purelin(x2)	hardlim(x2)
Without F.S.	0.2624	0.2710	0.2335	0.3256
NCA-C	0.2351	0.2357	0.2678	0.3246

TABLE 7. THE MSE RESULTS OF R = 64X64 NEURONS

	tansig(x2)	logsig(x2)	purelin(x2)	hardlim(x2)
Without F.S.	0.2468	0.2437	0.2837	0.3314
NCA-C	0.1871	0.2370	0.1871	0.2899

In Table 6, MSE values with NCA-C are less than MSE values with not using feature selection except `purelin`. `tansig` is the least MSE value whereas `hardlim` is the highest MSE value.

In Table 7, MSE values with NCA-C are less than MSE values with not using feature selection. `tansig` and `purelin` is the least MSE value whereas `hardlim` is the highest MSE value.

To sum up, NCA-C reduces the MSE values. For 128x128 hidden neurons in multi-layer pattern recognition networks, it lasted more than sixteen hours so it is not practical. Finally, it is stated that NCA-C increases the success of classification.

F. Outlier Detection and Outstanding Player Detection

The dataset is formed from 5 different parts. These are (I) Statistics of players in regular season (19113 records), (II) Statistics in playoff season (7544 records), (III) Statistics in all-star season (1463 records), (IV) Statistics in playoff career (2055 records) and (V) Statistic in regular season career (3760 records).

In data exploration, there are 19 features in each dataset and 3 of them are nominal. These features are player's id, name and last name so these nominal features are not taken into consideration throughout analysis. The remaining features are scaled into 0 to 1 in order to standardize independent variables.

Furthermore in data preparation stage, other features in the dataset was comprised of sums of all scores so in order to avoid most played player to be a natural outlier, we divide all the scores of players into match played.

per_match_score =

$$\frac{total\ number\ of\ distinct\ score\ \ for\ each\ player}{number\ of\ matches\ played\ for\ each\ player} \quad (2)$$

In the case of dimensionality reduction on player_playoff_career data, PCA (Principal Component Analysis) is applied. Because PCA takes lower dimensional set of features from a higher dimensional dataset while capturing as much information as possible. For PCA results of Playoff career dataset is shown below.

TABLE 8. PCA FOR PLAYER_PLAYOFF_CAREER DATASET

PCA'S	Variance	Cumulative Variance
PCA1	0.502874	50.29
PCA2	0.192634	69.55
PCA3	0.126686	82.22
PCA4	0.0461349	86.83
PCA5	0.0338315	90.21
PCA6	0.0235394	92.56
PCA7	0.0223254	94.79
PCA8	0.0169766	96.49
PCA9	0.0149214	97.98
PCA10	0.00972426	98.95

Table 8 shows that first 7 principle components captures nearly 95% of variance in dataset.

For the dataset playoff career, the dataset stretches from 1949 to 2004 and old data older than 1980's lacks values on some features. So we decided to eliminate these data and take only the ones after 1980. PCA results of playoff season dataset is shown below (Table 9).

TABLE 9. PCA FOR ALLSTAR DATASET

PCA'S	Variance	Cumulative Variance
PCA1	0.850226	85.02
PCA2	0.0693685	91.96
PCA3	0.0246625	94.43
PCA4	0.0197133	96.4
PCA5	0.0136819	97.77
PCA6	0.00633051	98.4
PCA7	0.00611065	99.01
PCA8	0.00350275	99.36
PCA9	0.00230317	99.59
PCA10	0.0017241	99.76

For the dataset all-star players, we did not per match calculation since dataset contains only 1 match statistics. PCA results of playoff season dataset is shown below (Table 10).

TABLE 10. PCA FOR PLAYOFF_SEASON DATASET

PCA'S	Variance	Cumulative Variance
PCA1	0.566743	56.67
PCA2	0.166589	73.33
PCA3	0.0718594	80.52
PCA4	0.0485056	85.37
PCA5	0.032174	88.59
PCA6	0.0298594	91.58
PCA7	0.0245657	94.04
PCA8	0.022483	96.29
PCA9	0.0154817	97.84
PCA10	0.0145132	99.29

For outlier detection, Python PyOD library is used which is a scalable library for detecting outliers in multivariate data. Different outlier detection algorithms are used, these are (I) Feature Bagging Outlier Detection (ABOD), (II) Isolation Forest, (III) k-Nearest Neighbors Detector, (IV) Cluster-based Local Outlier Detection and (V) Average K-NN.

Feature Bagging Detector: It fits a number of base detectors. It uses base estimator as Local Outlier Factor as default but other estimators such as k-NN or Angle Based Outlier Detector could be user.

In playoff career dataset, there are 20 outliers and 2036 Inliers. Threshold value is -0.7118109714421073.

In playoff season dataset, there are 33 outliers and 4430 inliers. Threshold value is -6231948.694033984.

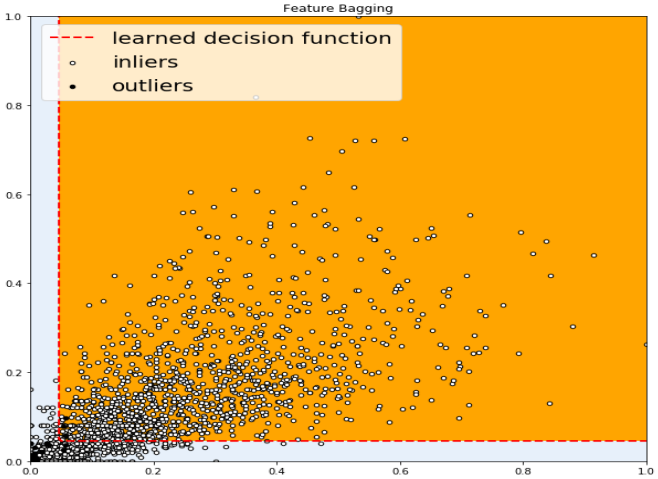

Fig. 5. Distribution of Points with Feature Bagging Detector

Isolation Forest: This method makes data partitioning by using a set of trees. It calculates an anomaly score examining how much isolated the point in the structure. The anomaly score is then used to identify outliers from normal observations.

In playoff career dataset there are 21 outliers and 2034 inliers. Threshold value is -0.14277197028200625.

In playoff season dataset there are 45 outliers and 4418 inliers. Threshold value is - - -0.1391610505888.

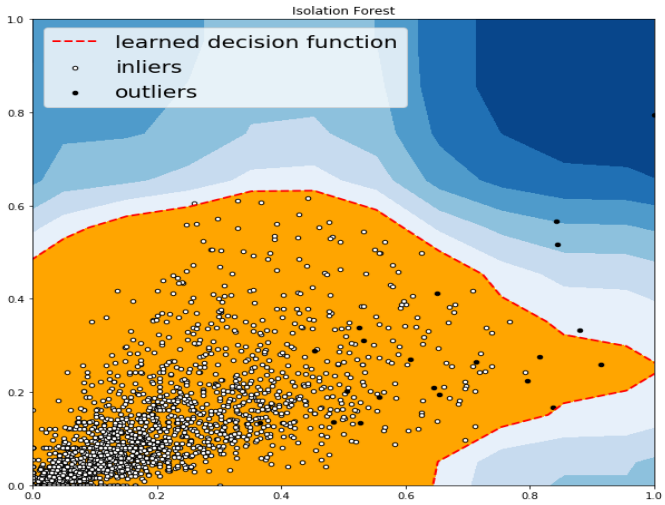

Fig. 6. Distribution of Points with Isolation Forest Detector

K-Nearest Neighbors: The distance to its kth nearest neighbor could be viewed as the outlying score.

In playoff career dataset, there are 16 outliers and 2039 inliers. Threshold value is -0.07808484312452174.

In playoff season dataset there are 34 outliers and 4429 inliers. Threshold value is -0.05731006134932328.

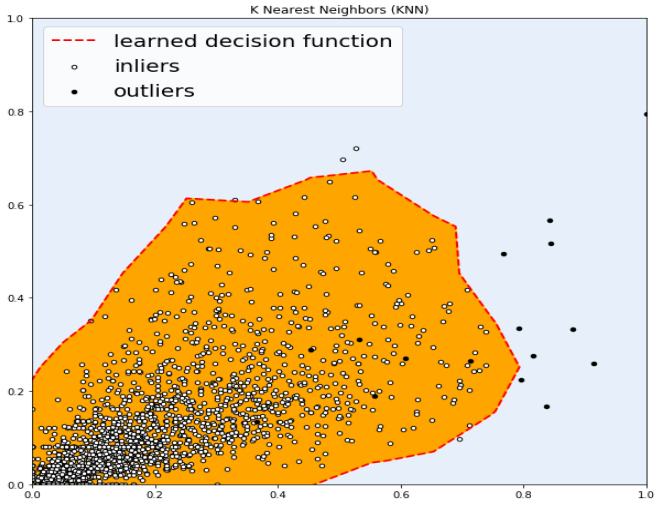

Fig. 7. Distribution of Points with k-NN Detector

***Clustering Based Local Outlier*:** It classifies the data into small clusters and large clusters. The anomaly score is then calculated based on the size of the cluster the point belongs to, as well as the distance to the nearest large cluster.

In playoff career dataset, there are 21 outliers and 2034 inliers. Threshold value is -0.6599293211770738.

In playoff season dataset, there are 45 outliers and 4418 inliers. Threshold value is -0.6953386755417442.

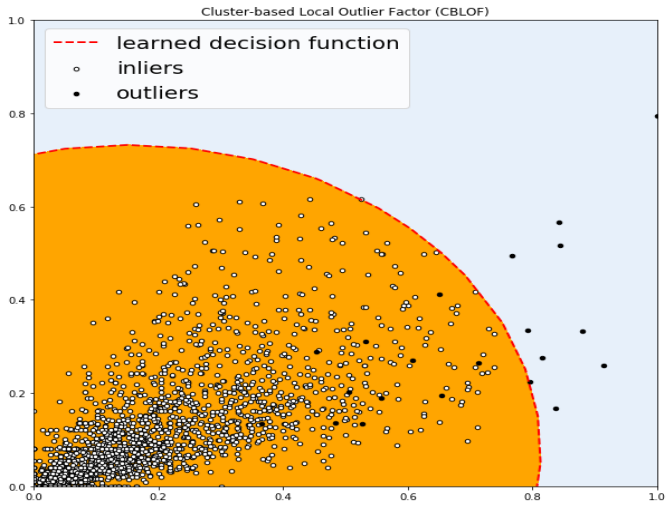

Fig. 8. Distribution of Points with Cluster-Based Local Outlier Detector

***Average k-NN*:** It uses the average of all k neighbors as the outlier score.

In playoff career dataset, there are 10 outliers and 2045 inliers. Threshold value is -0.04585606196467415.

In playoff season dataset, there are 7 outliers and 4456 inliers. Threshold value is -0.04706805441399679.

We considered a comparison should be made with NBA official website data against our results and methods. In unsupervised learning there is no ground truth for the comparison of results. In order to tackle the issue, we decided to get Most Valued Players of year 2004 from NBA official website. These players are detected with the help of votes from fans, critical assessment of players' performance on important matches and other factor like fair player and discipline on matches.

Here are the list of 2004 MVP players in Table 11.

TABLE 11. NBA MVP OF YEAR 2004

Player	Pts Won	Share of Vote
Steve Nash	1066.0	0.839
Shaquille O'Neal	1032.0	0.813
Dirk Nowitzki	349.0	0.275
Tim Duncan	328.0	0.258
Allen Iverson	240.0	0.189
LeBron James	93.0	0.073
Tracy McGrady	44.0	0.035
Dwyane Wade	43.0	0.034
Amar'e Stoudemire	41.0	0.032
Ray Allen	41.0	0.032
Kevin Garnett	15.0	0.012
Gilbert Arenas	4.0	0.003
Vince Carter	3.0	0.002
P.J. Brown	1.0	0.001
Marcus Camby	1.0	0.001
Shawn Marion	1.0	0.001

We compared our results with the official NBA results in Table 12.

TABLE 12. COMPARISON OF 2004 NBA MVP AND OUR RESULTS

Algorithm	Detected Outlier	Comparison Percentage
Feature Bagging Detector	12	0.75
Isolation Forest	12	0.75
k-NN	14	0.875
Clustering Based Local Outlier	13	0.8125
Average k-NN	8	0.50

IV. Conclusion

This project presents a prediction the outcome of NBA games by predicting team performance per season and detection outstanding players per season in NBA.

In first experiment for predicting team performance, we define the classes, consider four different feature selection algorithms and having 2, 64 and 128 hidden neurons with `tansig` activation function and `trainlm` training function for a single layer feedforward network. NCA-C displays a better performance (least MSE value) among four feature selection algorithms. Then by using NCA-C, four classification algorithm implemented. SVM with NCA-C displays a better performance (80.99%) among four classification algorithm.

In second experiment for predicting team performance, we consider four different neuronal activation functions, `trainlm` training function, NCA-C feature selection algorithm and two hidden layers with 2x2 and 64x64 neurons for a multi-layer pattern recognition network. `tansig` displays a better performance whereas `hardlim` displays a worse performance among four neural activation functions.

While the number of neurons or hidden layers increases, the elapsed time and MSE values also increases. This approach may not be practical.

For detecting outliers in our datasets, we have make use of Python Outlier Detection which is an open-source Python toolbox for performing scalable outlier detection on multivariate data[7].

In feature selection process, we have used PCA and turned its disadvantage of generating principle component without looking at target into advantage on our unsupervised learning problem. It helped us on reducing dimensionality of data from 19 to 7.

The algorithms we have used calculated a threshold value for separating normal ones with the anomalies or outperformer. We use an outlier fraction parameter during our research since our aim is finding observations that are not similar to rest of data. First we take it 0.05 and we have come up with nearly 60 outliers so after some iterations we hold the value on 0.01.

k-NN shows greater performance on our dataset and in average its success rate is 87%, so it makes him superior in comparison with other algorithms. But it is important to keep in mind that, in unsupervised learning researches it is difficult to detect basic truth. Even though our algorithms shows 87% success, voting preferences of fans and emotional factors may change MVP voting results.

Future works may include different dataset, may increase the number of activation functions, the number of different training functions, hidden layers or neurons. Also different classification models, network configurations like Bayesian Belief Network, Radial basis neural network etc. and outlier detection models may be applied and compared to each other.

REFERENCES

[1] NBA Statistics, http://www.cs.cmu.edu/~awm/10701/project/databasebasketball2.0.zip.

[2] Bunker, R.P. and Thabtah, F. : A machine learning framework for sport result prediction. *Applied Computing and Informatics*, vol. 15, pp. 27-33, 2019.

[3] Miljkovic, D., Gajic, L., Kovacevic, A. and Konjovic, Z. : The Use of Data Mining for Basketball Matches Outcomes Prediction. *IEEE* 8th International Symposium on Intelligent Systems and Informatics, pp 10-11, 2010.

[4] Wang, X., Wang, X. L., Ma, Y. and Wilkes, D. M.: A fast MST-inspired k-NN-based outlier detection method. *Information Systems*, vol. 48, pp. 89-112, 2015

[5] Liu, H., Motoda, H.: Computational Methods of Feature Selection. Chapman & Hall (2008)

[6] Malan, N.S. and Sharma, S.: Feature selection using regularized neighbourhood component analysis to enhance the classification performance of motor imagery signals. *Computers in Biology and Medicine*, vol. 107, pp. 118-126, 2019.

[7] Zhao, Yue & Nasrullah, Zain & Li, Zheng. (2019). PyOD: A Python Toolbox for Scalable Outlier Detection.