ISyE 6740 – Summer 2020
Final Report

Team ID: 090
Team Member Names: Qixiu Liu(903460924), Donghee Lee(903554426)

Project Title: Predict the next NBA MVP & positions of players

Partition of roles:

- Qixiu Liu: EDA on the player performance statistics data; develop unsupervised learning model on the performance data to group players in clusters; develop a predictive model for the positions of players given the performance statistics and body profile information.
- Donghee Lee: research and implementation of machine learning models to predict the next NBA MVP given players' statistics and the teams' success (NBA final championship).

Problem Statement

'The last dance' chronicled NBA stars such as Michael Jordan, Scottie Pippen, and Dennis Rodman of the legend NBA team, Chicago Bulls in the 1990s. The series was successful especially when the 2020 NBA season had to be paused due to Covid19. As a fan of NBA and Michael Jordan, we would like to predict the MVP awards. Common senses of winning the trophy say that it must come from the following three criteria: players' stats, team success, and media narratives.

For instance, Michael Jordan won the award 5 times while Chicago Bulls won the championship 6 times in his career. The only missing season's MVP was Karl Malone even though his team (Utah Jazz) did not win the championship. Below table shows the convincing statistics of both players in that season.

	Player	FG%	PPG	APG	RPG	BLKPG	3P%	2P%
3912	Karl Malone	0.550	27.4	4.5	9.9	0.6	0.000	0.555
3882	Michael Jordan	0.486	29.6	4.3	5.9	0.5	0.374	0.507

[1]

We are going to see how the NBA's MVP is decided using players' statistics and team success data. Considering multicollinearity of players' measures, only a few significant factors were picked, and details will be followed in the data source section. For team success, championship data was introduced.

Additionally, we were also interested in the followings:

[1] 1997 NBA season, comparison of two players' (Karl Malone and Michael Jordan) statistics

1. Use EDA to explore the performance statistics to see 1) if players from different positions would have different performance in some aspects; 2) if the performance statistics for players at each position and the whole NBA league vary over years.

2. Run some unsupervised and dimension reduction models on the performance statistics to see if they can group the players based on the performance statistics effectively.

3. Try to build a classifier to predict a player's position given the performance statistics information. Assuming different positions will relate to significant difference in some technical statistics and the model success will provide evidence and support to this assumption.

Data Source

To predict the NBA MVP, three datasets are combined. Here are the sources of the data.

- NBA Players stats since 1950 (Seasons_Stats.csv, individual statistics for 67 NBA seasons), *https://www.kaggle.com/drgilermo/nba-players-stats*
- NBA MVP & ABA Most Valuable Player Award Winners, *https://www.basketball-reference.com/awards/mvp.html*
- NBA Finals and MVP (nba_champion column only), *https://data.world/datatouille/nba-finals-and-mvps*

Please note the following data preprocessing:

1. Prediction of the next MVP

    - Data with missing values are omitted because players' stats vary each season. Imputing variables could highly distort our analysis. Thus, only seasons from 1987 are considered in the prediction for the MVP as players' statistics are not available before then.

    - Multicollinearity: many variables of players' statistics are correlated in seasons data. As some of the variables are a linear combination of the others, we computed the variance inflation factor (VIF). The VIF measures the proportional increase in the variance of $\hat{\beta}$, compared to what it would have been if the predicting variables had been completely uncorrelated. Thus, it will show the correlation of a variable with a group of other variables. It[2] turns out that most of the variables in players' statistics are highly correlated and among them PTS(points), 3P(3-Point field goals), FGA(Field goal attempts), ORB(offensive rebounds), FT(free throws), 2PA(2-Point Field Goal Attempts), 2P(2-Point field goals), FG(field goals), 3PA(3-point field goal attempts), TRB(total rebounds), and DRB(defensive rebounds) are especially more correlated as most of them are calculated based on other variables.

[2] Table1. VIF table – see the appendix page

Variables	VIF[3]
3P% (3-Point Field Goal Percentage)	5.378904
VORP (value over replacement)	18.133070
BLK% (block %)	19.946959
STL% (steal %)	24.215588
FTr (free throw rate)	26.002280

- Out of 45 measures, 11 primary variables are selected for predicting the MVP. The selected variables are scaled (if needed) by the number of games in each season for comparison.
- Championship data ('WinTeam') is combined with seasons data supplying a categorical variable of champion team (1=champion, 0=otherwise).
- Dataset is split into training (70%) and test (30%) dataset.

2. Prediction of positions of players

- Some statistics are missing for some player in some year and some statistics were introduced since a later year. For example, 3PA - 3-Point Field Goal Attempts is available since the 1979-80 season in the NBA; Usg% - Usage Percentage is available since the 1977-78 season in the NBA. We exclude all missing value out of the training and testing data and remain those entries with no missing value in any of the fields.

- There are 7 single positions in the raw data and some players took multiple positions in a single season. As checked, the single position entries accounted for over 96% in the dataset. So, we exclude all multi-position entries and only consider the single position cases.

Position[4]	
C	Centre
PF	Power Forward
SF	Small Forward
PG	Point Guard
SG	Shooting Guard
F	Forward
G	Guard

- datasets_1358_30676_player_data.csv is not very useful and it is lack of key to link this dataset to datasets_1358_30676_Players.csv. So, we don't use this dataset in our following work.
- Combine datasets_1358_30676_Players.csv with Seasons_Stats.csv to add player's heights and weights data onto the performance statistics dataset.

[3] A part of the VIF table from seasons data
[4] Position abbreviations and positions

- In the following analysis, we further simplify the positions into 3 types: "C", "F" and "G". "F" includes "PF" and "SF", and "G" includes "PG" and "SG".

Methodology

1. Prediction of the next MVP

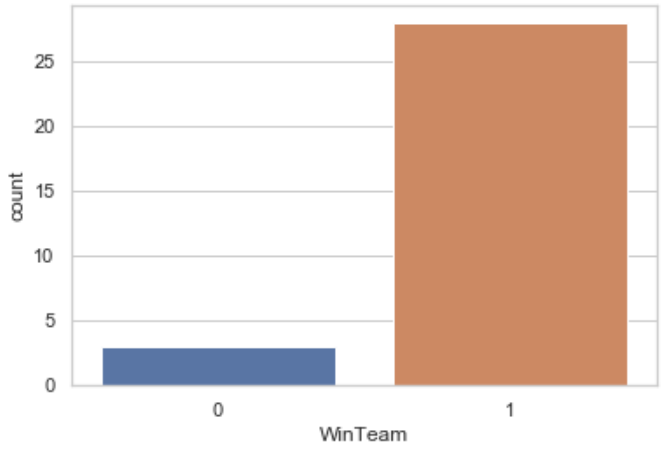
   - Lasso & Logistic regression: we modeled the probability of winning the MVP given the following predictors: ['PPG', 'APG', 'RPG', 'PER', 'WinTeam'] selected with L1 penalty out of 12 total features. First 4 predicting variables are players' stats and the last variable, 'WinTeam' is a categorical variable showing if the team won the championship. The response variable is binary.

```
Optimization terminated successfully.
         Current function value: 0.063589
         Iterations 10
                         Results: Logit
================================================================
Model:              Logit            Pseudo R-squared: -2.699
Dependent Variable: MVP              AIC:              1202.5406
Date:               2020-07-27 00:48 BIC:              1238.2707
No. Observations:   9377             Log-Likelihood:   -596.27
Df Model:           4                LL-Null:          -161.21
Df Residuals:       9372             LLR p-value:      1.0000
Converged:          1.0000           Scale:            1.0000
No. Iterations:     10.0000
----------------------------------------------------------------
             Coef.   Std.Err.     z      P>|z|    [0.025   0.975]
----------------------------------------------------------------
PPG           0.2743    0.0434    6.3130  0.0000   0.1891   0.3594
APG          -0.6352    0.1267   -5.0128  0.0000  -0.8836  -0.3868
RPG          -1.5832    0.1303  -12.1538  0.0000  -1.8385  -1.3279
PER          -0.1974    0.0172  -11.4822  0.0000  -0.2311  -0.1637
WinTeam       1.7553    0.2991    5.8685  0.0000   1.1690   2.3415
================================================================
```

[5]

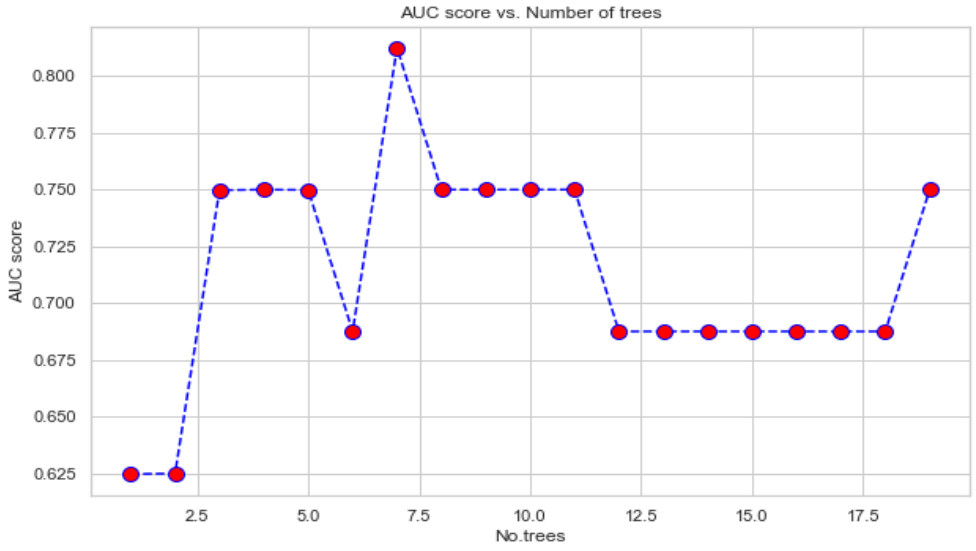
Above logistic regression results show coefficients and p-value of the selected measures in the model. As expected, all variables are statistically significant at the significance level of 0.01. For instance, the best estimate for 'WinTeam', is 1.76. That is, when the team wins the championship the odds of winning the MVP trophy is exp (1.76) = 5.81 times higher than otherwise. The interpretation makes sense since the trophy is often awarded to one of the players in the championship.

[5] Logistic regression result



[6]

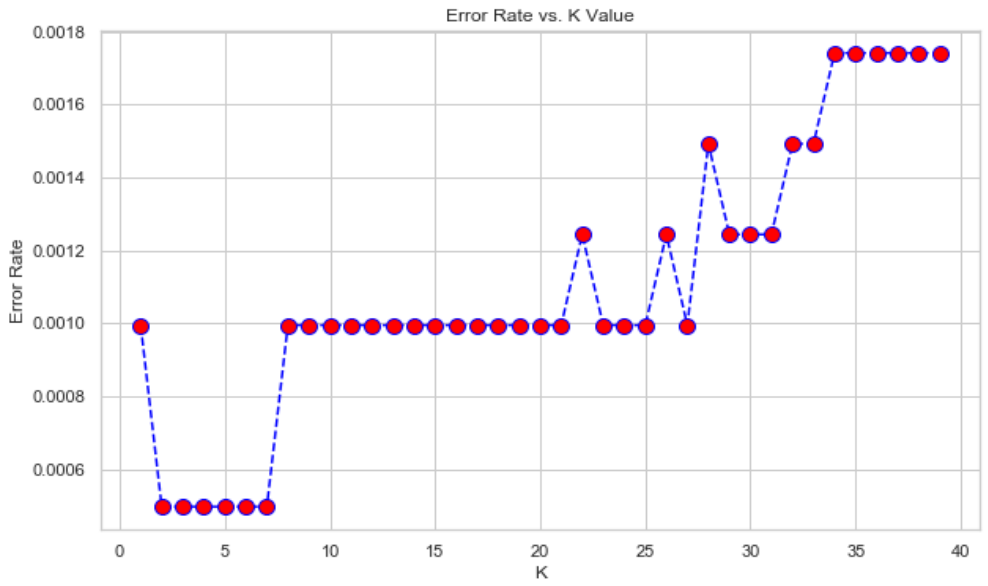
- Random Forest: previously selected 11 variables plus championship variable are predictors. 20 different number of decision tree classifiers were tried ranged from 1 to 20. The number of trees of 6 was chosen which results the best accuracy score. One of the good things of random forest model is that it handles a mixed type of data.



[7]

- KNN: it is a non-parametric algorithm which means it does not make any underlying distribution assumption. By using averages of nearest neighbors, the model is used to classify MVPs vs otherwise. The number of neighbors is chosen based on the lowest validation error. We can see that the validation error rate reached minimum at the k-value from 1 to 7. As the model is sensitive to the scale of the data, variables are standardized using sklearn.StandardScaler.

[6] Categorical count plot showing how many MVPs are from a championship team vs non-championship team
[7] AUC score vs number of trees from random forest model



[8]

- SVM: with support vector machine model, we tried two different approaches. One is with linear hyperplane and the other is kernelized SVM using GridSearch. The linear SVM results accuracy level of 56% as the data cannot be simply divided linearly. The decision boundary did not perform well as in the case of logistic regression (accuracy level of 62.5%). Yet, logistic regression performed slightly better than linear SVM as SVM typically works for 'clear' separable classes. On the other hand, logistic regression approaches classification with probability which gives a smooth objective. For better fitting the data, kernelized SVM was also performed. It tries to find a mapping function which maps the non-linearly separable dataset into a higher dimension. As a result, accuracy rate increased to 87.5%.

2. Prediction of players' positions

2.1. EDA

- To explore the time series of each performance statistics over years. Differentiate the 3 position types in each statistics scenario.
- The performance statistics we check in its time series are: 'TS%','3PAr', 'FTr', 'ORB%', 'DRB%', 'TRB%', 'AST%', 'STL%', 'BLK%', 'TOV%','USG%', 'WS/48',
- '3P%', '2P%', 'eFG%', 'FT%', 'height', 'weight'.
- We calculate the mean for each position for each year.
- All the 18 plots are attached in the appendix.
- Assuming there exists distinction in statistics trend over years for each position type.

[8] Prediction error vs number of neighbors

2.2. Unsupervised learning

- The first model is K-mean clustering. Considering all the performance statistics are numerical, it is good to have a go on trying Euclidean distance to describe the similarity between players' performance.
- The performance statistics used for clustering are 'TS%', '3PAr', 'FTr', 'ORB%', 'DRB%', 'TRB%', 'AST%', 'STL%', 'BLK%', 'TOV%', 'USG%', 'OWS', 'DWS', 'WS', 'WS/48', 'OBPM', 'DBPM', 'BPM', 'VORP', 'FG', 'FGA', 'FG%', '3P', '3PA', '3P%', '2P', '2PA', '2P%', 'eFG%', 'FT', 'FTA', 'FT%', 'ORB', 'DRB', 'TRB', 'AST', 'STL', 'BLK', 'TOV', 'PF', 'PTS'.
- In the raw dataset, a field called "PER" (i.e. The Player Efficiency Rating) is a per-minute rating developed by ESPN.com columnist John Hollinger. "The PER sums up all a player's positive accomplishments, subtracts the negative accomplishments, and returns a per-minute rating of a player's performance." So, it could be treated as an overall evaluation of a player's performance and it could be somewhat useful for predicting MVP players. We will use this rating scores to check the clustering result and what we expect is the clustering of players should match the rating score intervals and the players from the same cluster should have fallen into the same score interval.
- While "PER" is a per-minute rating score, but many of the performance statistics used for clustering are accumulative, so we rescale the score by multiplying it with "MP" (i.e. minutes played). In other words, the final score we used to check the clustering result is "PER" * "MP".
- The players are grouped into 5 clusters.
- The second unsupervised learning output is T-distributed Stochastic Neighbor Embedding (t-SNE). It used the same dataset with K-mean clustering. The performance score used in the visualization is also "PER" * "MP".
- Use this dimensionality reduction technology to convert the data cloud into a 2-D dataset and bin the performance scores into 4 intervals and use these bins to check how well t-sne capture the variance in the data cloud and project the majority variance into a 2-D plane.

2.3. Predictive modelling

- To set up the target variable as position with 3 main types: "C", "F" and "G". We exclude the entries with multiple positions for one player like 'PG-SG'. We also convert 'SG' and 'PG' into 'G', 'PF' and 'SF' into 'F'. The model is developed to use performance statistics and body profile information to predict a player's position.
- The statistics 'ORB', 'DRB', 'TRB', 'AST', 'STL', 'BLK', 'TOV', 'PF', 'PTS', 'FG', '3P', '2P' and 'FT' are accumulative score. They are time dependent. To remove the time effect, we divide them with "MP" to create a new set of features 'ORBmp', 'DRBmp', 'TRBmp', 'ASTmp', 'STLmp', 'BLKmp', 'TOVmp', 'PFmp', 'PTSmp', 'FGmp', '3Pmp', '2Pmp', 'FTmp'.
- 3 body profile fields are involved: 'age', 'height' and 'weight'.
- Some fields are not useful such as 'G' (Games), 'GS' (Game Started), 'Year', 'PER'.
- The candidate models are: Stochastic gradient descent classifier, Stochastic gradient descent classifier with polynomial features (degree = 2), Naïve Bayes,

KNN, Random forest, Gradient boosting, XGBClassifier, SVM and shallow neural network.

- Use 5-fold cross validation to search the hyperparameter spaces for each candidate model, tune the hyperparameter into an optimal model, reach a tradeoff of bias and variance.
- Set up stacking model to see if taking the input from those candidate model, a higher layer of learner can reach a better performance.
- Some feature engineering efforts were introduced to see if it helps with the performance improvement. These efforts include 1) create a binary classifier respectively for predicting "SF" and "PF" position. Take the output of the binary classifier as a new feature into the 3-class position predictive model; 2) try some polynomial features on some candidate models to see if there is any improvement;
- Use accuracy as the evaluation metric to assess the candidate models and select one to be the final model.
- For stacking model, try different way of voting and different higher layer of learner to see if any difference.
- Use 75% data as the training data and 25% as testing.

- Candidate models:

    - Stochastic gradient descent classifier: The regularization constant alpha, the loss function and the penalty regularizer matter. The maximum number of iterations affects the convergences and hence the accuracy of the model.
    - Naïve Bayes: The entries are not independent for a player can have multiple entries in the data for each season he was involved. So, this model's assumption was not satisfied, and this would affect the performance of this model.
    - KNN: it is simple and, in this case, as all features are numerical, this model can efficiently reach a good accuracy with lower computational cost.
    - Random forest and Gradient boosting are both ensemble algorithms which are supposed to be robust. The number of estimators (weak learners) matters. XGBoosting is an enhanced algorithm on gradient boosting. This normally outperformed others.
    - Regularization parameter can significantly affect SVM's performance. "rbf" kernel was used.
    - Shallow neural network: the structure of hidden layers largely determines the performance of the neural network performance.
    - For stacking model, the set of candidate model affects the stacking performance. Some candidate model will reduce the accuracy and should be removed from the stacking layer to reach a better outcome. In a voting mechanism, we can try assigning weights for different candidate models to differentiate the importance and reliability of each model.

Evaluation and Final Results

1. Prediction of the next MVP

- Lasso & Logistic regression

N=4020	Predicted: No	Predicted: Yes
Actual: No	4012	0
Actual: Yes	6	2

AUC score: 0.625

- Random Forest

N=4020	Predicted: No	Predicted: Yes
Actual: No	4011	1
Actual: Yes	3	5

AUC score: 0.81

- KNN

N=4020	Predicted: No	Predicted: Yes
Actual: No	4012	0
Actual: Yes	2	6

AUC score: 0.875

- Linear SVM

N=4020	Predicted: No	Predicted: Yes
Actual: No	4012	0
Actual: Yes	7	1

AUC score: 0.563

- Kernelized SVM

N=4020	Predicted: No	Predicted: Yes
Actual: No	4012	0
Actual: Yes	2	6

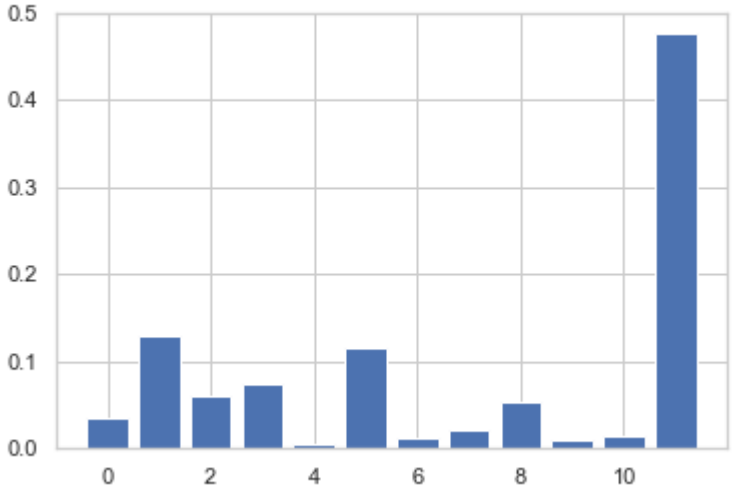
AUC score: 0.875

As shown above, accuracy level increased from Logistic regression to random forest, KNN, and kernelized SMV model. In general, non-MVPs are predicted very well due to the large number of samples for non-MVPs. The best models overall for predicting the MVP were kernelized SVM and KNN models.

Logistic regression model after applying L1 penalty was easy to interpret thanks to the logit function. However, this classification problem turns out to be non-linear and thus, we could not lower the prediction error further even after adjusting collinearity effect. With random forest, we could improve the prediction accuracy when compared with logistic regression. As random forest works well with a mixture of numerical and categorical features, it suits our data better than the logistic regression. KNN performed even better than random forest for both classes. Also, it is intuitive to interpret especially when compared with kernelized SVM due to its hyperparameters.

As shown in the below table from random forest model and Figure.4 from logistic regression model, we can see that the variable 'WinTeam' is the most significant factor for deciding the MVP. That is, individual player's statistics are important, but team's success plays a bigger role to win the trophy.

	Feature importance
FG%	0.035192
PPG	0.128672
APG	0.059278
RPG	0.074025
BLKPG	0.004090
PER	0.114992
TS%	0.010671
3PAr	0.019973
FTr	0.052881
3P%	0.010071
2P%	0.013054
WinTeam	0.477101



[9]

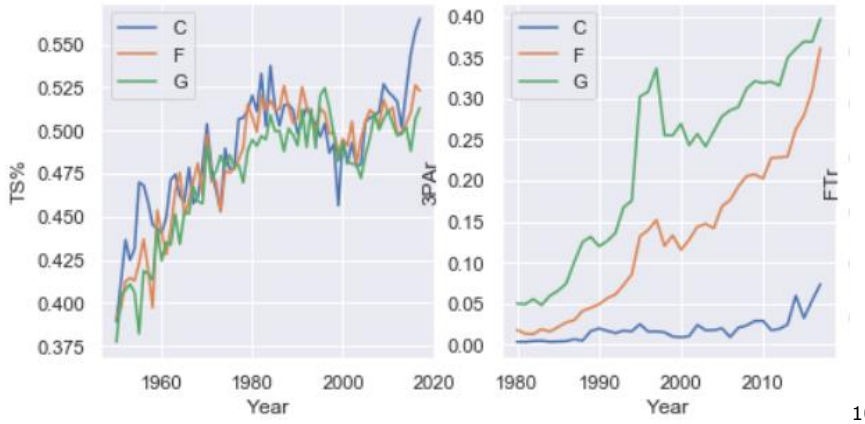
2. Prediction of players' positions

2.1. EDA

- As checked in the time series curves, we can confirm the assumption for difference between position and time is correct.
- For example, for TS%, i.e. the True Shooting Percentage, we can see the average rate increases over time for all types of position but little difference among position types until recent years that the center players seem to have a higher average true shooting percentage than other positions. For 3PAr, i.e. 3-Point Field Goal Attempts rate, we can see the average performance among different positions differs very much to each other. This convinced us that the data can somehow help build a predictive model to distinguish the position the players took, given the performance statistics. In other words, the position groups in the raw data were separated well to some extent. Also, the guard and forward players have a higher and higher 3PAr

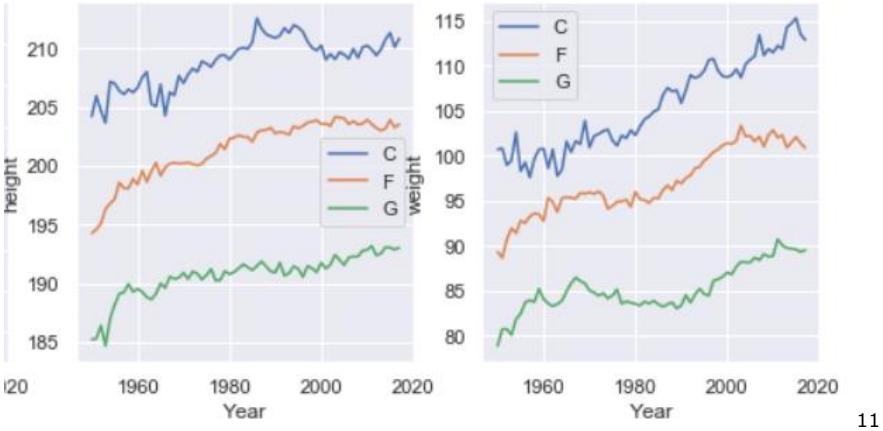
[9] Feature importance from random forest model

rate as time pasts while the center players stay at a much lower rate with minor change until recent years that they start to demonstrate an increase trend on that rate. This indicates the change of the way the players play the games.



[10]

- For players' height and weight, it is interesting that there are two trends here: 1) the average height and weight of players in NBA league go upwards over time; 2) There are significant difference in average height and weight among position types: normally center players are the tallest and heaviest while the forward players are in the middle and the guard players have lowest average height and weight.

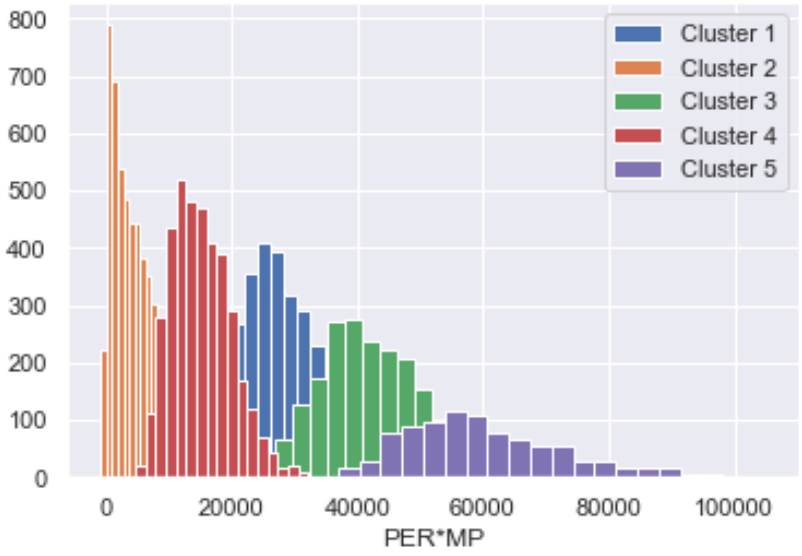


[11]

2.2. Unsupervised learning

- K-means clustering

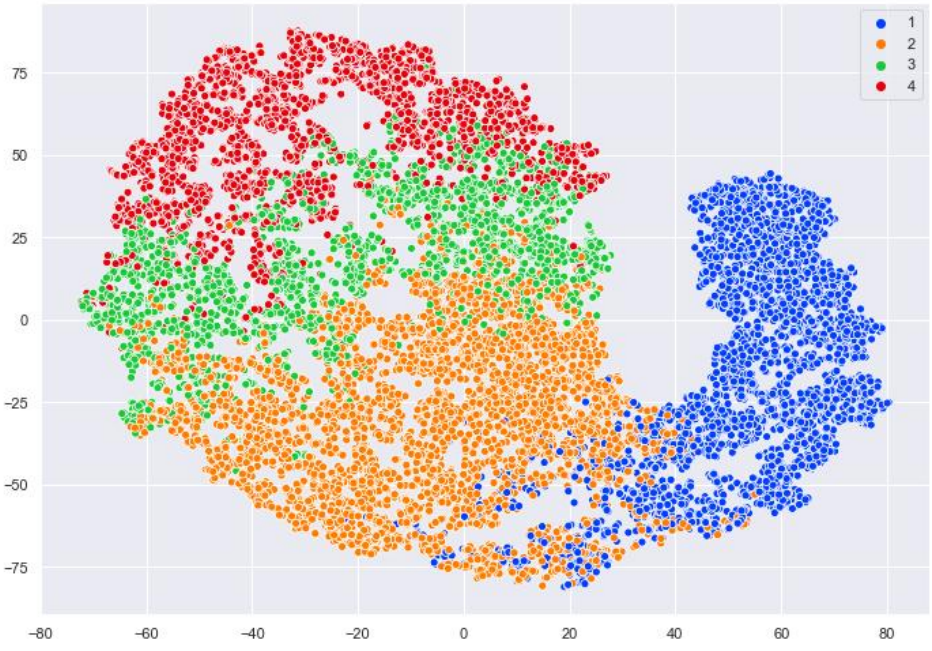
[10] Change in TS% and 3PAr over years
[11] Change in height and weight over years



[12]

- The plot shows the performance score ("PER" * "MP") distribution for each cluster generated by a K-mean clustering algorithm. There are totally 5 clusters.
- We can see each distribution significantly differentiate from each other despite that they have some minor overlapping among clusters. This means the clustering algorithm produce a useful feature which can be used to predict player's MVP status or salary etc. that directly relates to the player's performance.

- T-SNE



[13]

[12] K-means clustering result
[13] T-SNE result

- We can see this nonlinear dimensionality reduction technique compress the data well, remaining the similarity between data points in the high dimensions.
- The players in the same score interval are physically close to each in the t-sne plot and we can still find some overlapping parts in the boundary between two score intervals. The labels derived by t-sne can be a feature to provide some information about the ranking of the play performance and can be used for some prediction tasks.

2.3. Predictive modelling

- The performance of each candidate model tuned by a 5-fold CV process is listed below:

Classifier	Hyperparameters	Accuracy
SGDClassifier	penalty="l1", loss='hinge', max_iter=1000, alpha=0.001	0.7836
GaussianNB		0.77
KNeighborsClassifier	n_neighbors=8, weights='distance'	0.8638
RandomForestClassifier	max_depth=12, random_state=0, n_estimators=500	0.8613
GradientBoostingClassifier	learning_rate=0.15, n_estimators=500, subsample=0.9	0.8635
XGBClassifier	n_estimators=200	0.8807
SVC	gamma='scale', C=0.9	0.8508
MLPClassifier	alpha=0.005, learning_rate='invscaling', max_iter=2000, hidden_layer_sizes=(15, 6, 3),early_stopping=True, learning_rate_init=0.001	0.833
SGDClassifier, polynomial n=2	penalty="l2", loss='hinge', max_iter=1000, alpha=0.0001	0.7895
stacking model by voting	XGBClassifier + KNeighborsClassifier + GradientBoostingClassifier	0.8777
stacking model by XGBClassifier	all candidate model are included	0.8807

[14]

- The difference of accuracy between classifiers are obvious. XGBClassifier outperformed.
- To check the detail of the classification errors, we look at the confusion matrix for each candidate model. In each confusion matrix, the row names are true positions and the column names are predicted positions. For most candidate models, they are relatively weak on identifying the center players or distinguish between center and forward players. Guard players seem to be easier to classifier than the other two positions.
- The best model goes to XGBClassifier which reaches an accuracy score of 0.8807. Even in this winning classifier, the position "C" accuracy is still less than 80% and almost erroneous cases in position "C" are misclassified into "F".
- The reason behind the difficulty on dealing with some "C" and "F" cases might be some forward players or center players take multiple positions in his career. So even he is a center player, he can play like a forward. likewise, some forward player can have a set of performance statistics quite like a center player. Some players took turns on these two positions in his team as requested.
- The body profile features contribute to the performance improvement.

[14] Performance comparison by classifier

Stacking with voting

	C	F	G
C	0.776596	0.223404	0.000000
F	0.055223	0.851630	0.093147
G	0.000000	0.069240	0.930760

Stacking with XGBClassifier

	C	F	G
C	0.785106	0.212766	0.002128
F	0.052562	0.857618	0.089820
G	0.000000	0.070466	0.929534

SGDClassifier

	C	F	G
C	0.568085	0.427660	0.004255
F	0.097139	0.753826	0.149035
G	0.001838	0.122549	0.875613

SGDClassifier +Polynomial n=2

	C	F	G
C	0.317021	0.678723	0.004255
F	0.028609	0.813706	0.157685
G	0.000000	0.096814	0.903186

GaussianNB

	C	F	G
C	0.851064	0.146809	0.002128
F	0.234198	0.586161	0.179641
G	0.000000	0.083946	0.916054

KNeighborsClassifier

	C	F	G
C	0.719149	0.280851	0.000000
F	0.055223	0.853626	0.091151
G	0.000000	0.085172	0.914828

RandomForestClassifier

	C	F	G
C	0.685106	0.312766	0.002128
F	0.045243	0.854291	0.100466
G	0.000000	0.081495	0.918505

GradientBoostingClassifier

	C	F	G
C	0.757447	0.240426	0.002128
F	0.069860	0.829674	0.100466
G	0.000000	0.074755	0.925245

XGBClassifier

	C	F	G
C	0.785106	0.212766	0.002128
F	0.052562	0.857618	0.089820
G	0.000000	0.070466	0.929534

SVC

	C	F	G
C	0.670213	0.327660	0.002128
F	0.041916	0.841650	0.116434
G	0.000000	0.088848	0.911152

MLPClassifier

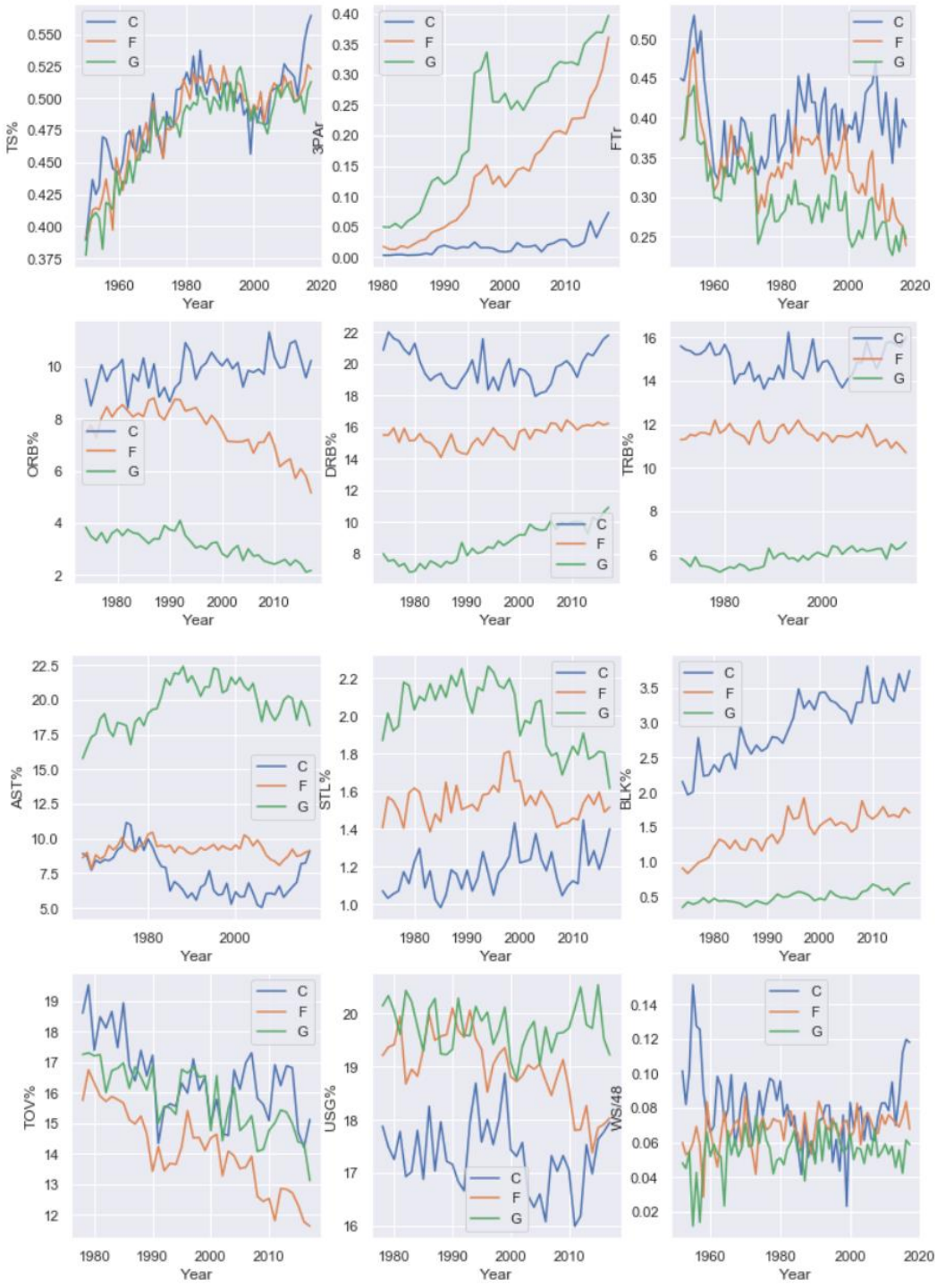
	C	F	G
C	0.631915	0.368085	0.000000
F	0.055888	0.820359	0.123752
G	0.000000	0.097426	0.902574

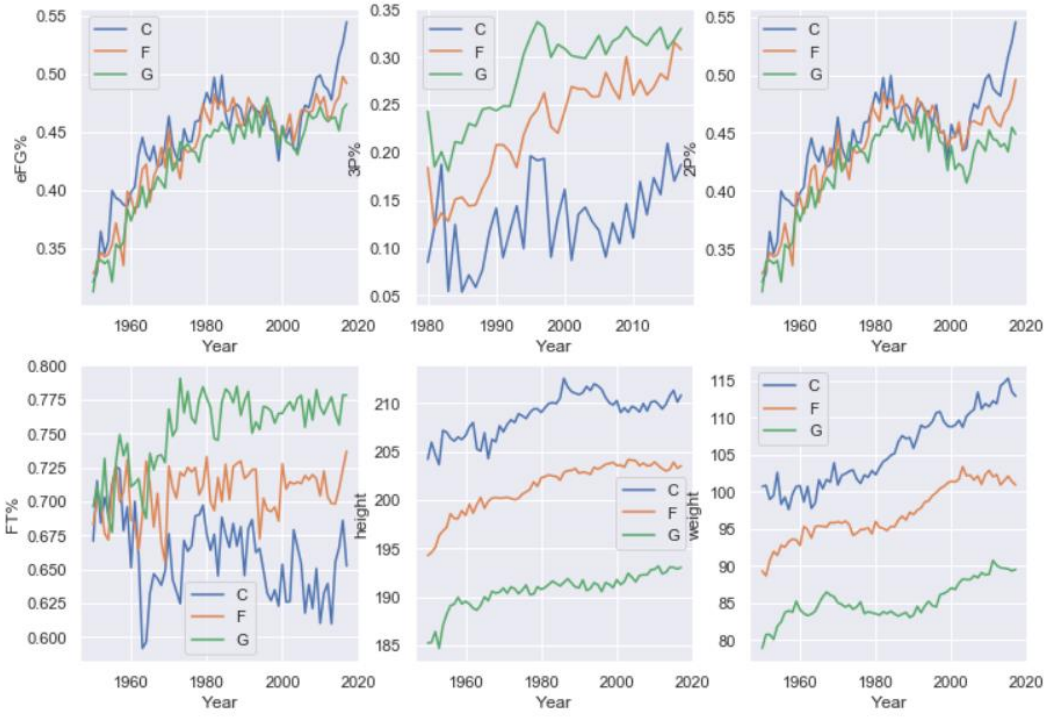
[15]

- The feature engineering effort we tried didn't bring too much difference in the model performance. i.e. 1) the output from the binary classifiers for predicting "SF" and "PF" position respectively cannot solve the problem of distinguish some "C" cases from "F" cases; 2) The polynomial features did not show significant contribution to the candidate model outcome but meanwhile it cause a rapid increase in computation. The overlapping part in feature distribution between "C" groups and "F" groups is a bottleneck in our modelling for gaining higher accuracy.
- Stacking models do not outperform all single candidate model and reach an optimal accuracy score.
- Next step: 1) based on the current dataset, try more feature engineering methods to add in new features to solve the bottleneck, including adding the unsupervised learning outcome into the supervised learning tasks; 2) introduce new data like salary, MVP or other performance statistics to lift the accuracy outcome; 3) try deep learning architecture to see if it helps for boosting the accuracy;

Appendix

[15] Model performance per classifier


TS%
3PAr
FTr
ORB%
DRB%
TRB%
AST%
STL%
BLK%
TOV%
USG%
WS/48
Year
C
F
G


eFG%
3P%
2P%
FT%
height
weight
Year
C
F
G