香港中文大學
The Chinese University of Hong Kong

STAT4011 - Project - Final Report(Group 13)

Predicting NBA Games and MVP using Machine Learning Techniques

Project Members

1155159866 NG Sze Leong

1155176919 FU Tsz Ho

1155156341 ZHANG Xinyi

1155177307 CHEN Man Hin

1155147304 JUNOES Hans Nathanael

1.Introduction

In the context of sports betting, the ability to predict game outcomes and identify standout players are two sources which have become increasingly profitable in the National Basketball Association (NBA), the largest basketball league in the world. In 2023 alone, the NBA has garnered a revenue of over $11 billion USD. As such, this project will be divided into two parts: prediction of NBA games and also the MVP of the season, for each of introduction, methodology, and analysis.

A. Predicting NBA Games *(Ng Sze Leong)*

This project focuses on predicting the results of NBA games through a feature difference model, employing various machine learning algorithms including Logistic Regression, Random Forest, and K-Nearest Neighbors (KNN). By leveraging historical game data and player statistics, we aim to uncover patterns and relationships that influence game outcomes.

Initially, we utilize a feature difference model to capture the disparities between competing teams. This approach enables us to quantify the impact of individual player performances and team dynamics on game results. Subsequently, we implement several classification algorithms to evaluate their effectiveness in predicting outcomes based on our feature set.

To enhance our model's performance and reduce overfitting, we apply Principal Component Analysis (PCA) to eliminate redundant features, streamlining our dataset while retaining essential information. This step is crucial in ensuring that our predictive models operate efficiently and accurately.

B. Predicting Regular Season MVPs *(Hans Junoes)*

In addition to predicting game outcomes, we extend our analysis to identify potential candidates for the NBA Most Valuable Player (MVP) award. By employing different machine learning techniques, we aim to highlight the attributes that contribute to a player’s MVP candidacy, thereby providing insights not only into game results but also into individual player performances throughout the season.

The data is collected from multiple sources from the same website, such as player analytics, advanced metrics, team statistics, and mvp rankings of a long period of seasons, then inner and/or outer joined into a single table. After engineering some new features to improve scores, we finally use decision trees, ridge regression, and gradient descent algorithms to find our predictions. Since this is a unique ranking problem, new metrics will also be introduced that are appropriate.

Ultimately, this report aims to contribute to the growing field of sports analytics and offer a deeper understanding of the factors that influence success in the NBA.

2.Methodology

A. Predicting NBA Games

2.1.1 Data Collection *(ZHANG Xinyi)*

To predict the result of the home team winning, we used web scraping to extract the NBA information from the website ESPN(https://www.espn.com/)[1], which is the official website of a leading sports company. The dataset includes team standings, game results, player statistics of each team, and player shooting statistics of each team from 2015 through 2024. We organized and transformed the information into structured data frames, named as Home_stat, Away_stat, Player_stat, and Shooting_stat. Then, 4 data frames were combined and named Team_stat for further analysis. Team_stat includes variables such as the average of home team win/loss score, the median and SD of away team win/loss score, the average of points for each player, the average rebounds of players, and the average of three points percentage. The data frame Team_stat incorporates 21 variables in total.

2.1.2 Data Preprocessing *(ZHANG Xinyi)*

Data preprocessing steps are crucial. It was implemented before model learning, including the handling of missing values, data type conversion, etc. A binary variable of Home_Team_Winning was created based on the differences between the home team score and the away team score in a game, indicating whether the home team won or lost the game. Besides, we normalized the inconsistent team names extracted from the website by using the normalized_team_name function created and dictionaries named Club_dict. This facilitates proper data merging and analysis.

2.1.3 Feature Difference *(ZHANG Xinyi)*

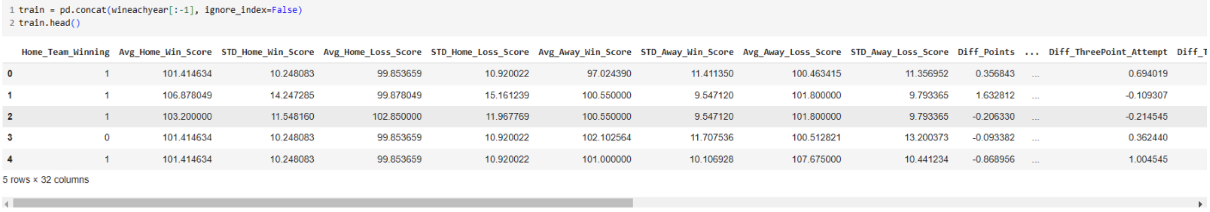
Feature difference was adopted to improve model interpretability, capture the relative strengths and weaknesses, and reduce the feature dimension by combining two teams into a single comparison. As the goal is to predict which team will win by comparing their performance, feature differences directly reflect our goal of prediction. It was implemented by the create_record function. The function first retrieved statistical data for home team (HT) and away team (AT) from the Team_stat data frame. Then, the difference of selected relevant features was calculated. The feature differences act as input variables for the predictive models, allowing models to learn relationships between team-level statistics, player-level statistics, and game outcomes.

2.1.4 Define Training / Testing Dataset

Initiation: The rollover sets up a few objects:
years: An array from 2015 to 2024 which are the target years of extraction
wineachyear: An empty list for storing the data of competitions from 2015 to 2024

Rollover Loop:
The rollover loop runs up to the times equal to the length of years. In each trial, ‘i’ is substituted into each element in the object ‘years’, for example, in the first trial, i=2015, in the final trial, i=2024. During each loop: It gets the NBA standings in the year ‘i’ from the ESPN website by extracting the table in the website.
It also gets the information of each team, like their names and stats, in the year ‘i’.
Afterward, a similar method for the feature difference is executed, the data frame ‘win’ is obtained.
It puts the data frame ‘win’ into the list ‘wineachyear’.

After loop:
When the loop finishes, 10 data frames, which are the processed data from 2015 to 2024, are saved into ‘wineachyear’.
The data in the first nine years are extracted and treated as the training set, which are the statistics of the NBA competitions from 2015 to 2023.

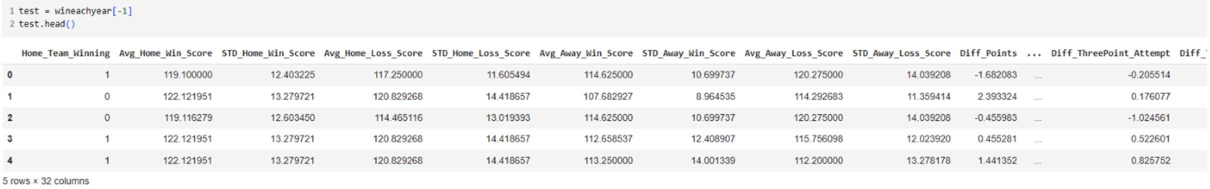


```
train = pd.concat(wineachyear[:-1], ignore_index=False)
train.head()
```

	Home_Team_Winning	Avg_Home_Win_Score	STD_Home_Win_Score	Avg_Home_Loss_Score	STD_Home_Loss_Score	Avg_Away_Win_Score	STD_Away_Win_Score	Avg_Away_Loss_Score	STD_Away_Loss_Score	Diff_Points	...	Diff_ThreePoint_Attempt	Diff_T
0	1	101.414634	10.248083	99.853659	10.920022	97.024390	11.411350	100.463415	11.356952	0.356843	...	0.694019	
1	1	106.878049	14.247285	99.878049	15.161239	100.550000	9.547120	101.800000	9.793365	1.632812	...	-0.109307	
2	1	103.200000	11.548160	102.850000	11.967769	100.550000	9.547120	101.800000	9.793365	-0.206330	...	-0.214545	
3	0	101.414634	10.248083	99.853659	10.920022	102.102564	11.707536	100.512821	13.200373	-0.093382	...	0.362440	
4	1	101.414634	10.248083	99.853659	10.920022	101.000000	10.106928	107.675000	10.441234	-0.868956	...	1.004545	

5 rows × 32 columns

The data in the last year serves as the testing set, which are the statistics of the NBA competitions in 2024.



```
test = wineachyear[-1]
test.head()
```

	Home_Team_Winning	Avg_Home_Win_Score	STD_Home_Win_Score	Avg_Home_Loss_Score	STD_Home_Loss_Score	Avg_Away_Win_Score	STD_Away_Win_Score	Avg_Away_Loss_Score	STD_Away_Loss_Score	Diff_Points	...	Diff_ThreePoint_Attempt	Diff_
0	1	119.100000	12.403225	117.250000	11.605494	114.625000	10.699737	120.275000	14.039208	-1.682083	...	-0.205514	
1	0	122.121951	13.279721	120.829268	14.418657	107.682927	8.964535	114.292683	11.359414	2.393324	...	0.176077	
2	0	119.116279	12.603450	114.465116	13.019393	114.625000	10.699737	120.275000	14.039208	-0.455983	...	-1.024561	
3	1	122.121951	13.279721	120.829268	14.418657	112.658537	12.408907	115.756098	12.023920	0.455281	...	0.522601	
4	1	122.121951	13.279721	120.829268	14.418657	113.250000	14.001339	112.200000	13.278178	1.441352	...	0.825752	

5 rows × 32 columns

2.1.5 Implementation/ Experimentation *(Ng Sze Leong)*

1. Logistic Regression (LR)*(Ng Sze Leong)*

Logistic regression is a widely used statistical method for predicting binary outcomes, making it a fundamental tool in various fields, including medicine, finance, and sports analytics. Unlike linear regression, which predicts continuous values, logistic regression estimates the probability that a given input belongs to a particular category. This is achieved through the logistic function, which maps any real-valued number into a range between 0 and 1. In this project, our objective is to predict whether the home team will win (getting 1 in logistic regression) or not (getting 0 in logistic regression), the logistic regression will help us to find the result.

$$[P(Y = 1 \mid X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_n X_n)}}]$$

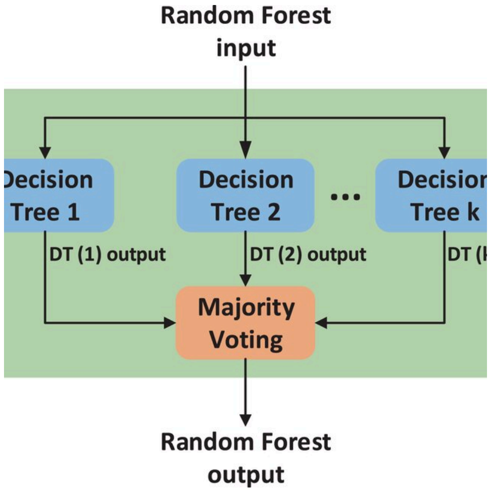
The strength of logistic regression lies in its ability to model complex relationships between independent variables and the log-odds of the dependent variable. By analyzing factors such as team performance metrics, player statistics, and historical data, logistic regression provides insights into the likelihood of specific outcomes, such as winning or losing a game. Its interpretability and efficiency make it an invaluable tool for us to use it to get the prediction of the games' results.

Code Snippet

```
X_train = train.drop('Home_Team_Winning',axis=1).values
y_train = train['Home_Team_Winning'].values
X_test = test.drop('Home_Team_Winning',axis=1).values
y_test = test['Home_Team_Winning'].values
lr = LogisticRegression(random_state=0,
solver='saga',multi_class='ovr').fit(X_train, y_train)
pX_test = lr.predict_proba(X_test)
pclass_test = lr.predict(X_test)
```

2. Random Forest (RF) *(Ng Sze Leong)*

Random Forest is a powerful ensemble learning method used for both classification and regression tasks, particularly effective in handling complex datasets. This helps us to classify whether the home team is the winning team or not. By constructing a multitude of decision trees during training and outputting the mode of their predictions (for classification), this is the reason why it is called a 'forest' and enhances predictive accuracy and robustness.



(ResearchGate, n.d., Figure 5, [2])

One of its key strengths is its ability to manage large datasets with high dimensionality and to handle missing values effectively as we have a lot of game data. The method also reduces the risk of overfitting, which is a common issue in single decision tree models. Additionally, Random Forest provides insights into feature importance. This may help us to know how the features impact the results, giving us insights if we want further investigation.

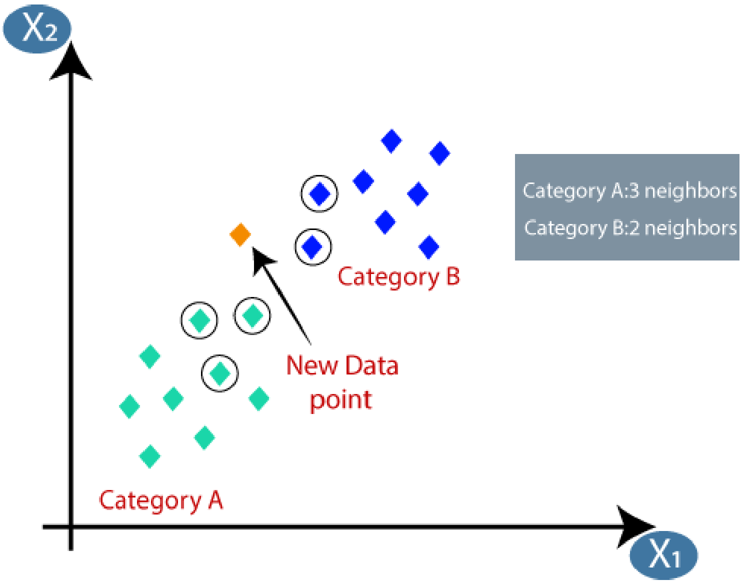
In the context of predicting NBA game results, Random Forest is particularly valuable due to its capacity to capture complex interactions between various performance metrics, such as player statistics and team dynamics. By using this method, we can get a more precise result even if we have a huge number of features and data like this project.

Code Snippet

```
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)
y_pred_rf = rf_model.predict(X_test)
```

3. K-nearest Neighbors Algorithm (KNN) *(Ng Sze Leong)*

The K-Nearest Neighbor (KNN) algorithm is a widely-used classification algorithm that is known for its simplicity. It is a non-parametric and lazy learning algorithm where the function is only approximated locally and all computation is deferred until function evaluation. It uses a database containing data points that are separated into several classes to predict the classification of a new sample point. In KNN, an object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its K nearest neighbors (K is a positive integer, typically small).
If K = 1, then the object is simply assigned to the class of that single nearest neighbor.



Javatpoint (n.d.) explains the k-nearest neighbor algorithm. [3]

Code Snippet

```
knn_model = KNeighborsClassifier(n_neighbors=3)
knn_model.fit(X_train, y_train)
y_pred_knn = knn_model.predict(X_test)
```

2.1.6 Reinforcements - Principal Component Analysis(PCA) *(Ng Sze Leong)*

Principal Component Analysis (PCA) is a statistical technique widely used for dimensionality reduction, particularly effective in managing datasets with a large number of features. So we expect this would be helpful for improving our model's performance. By transforming the original correlated variables into a new set of uncorrelated variables called principal components, PCA captures the maximum variance in the data with fewer dimensions.

One of the key advantages of using PCA is its ability to simplify complex datasets while preserving essential information. This is particularly beneficial in our project, predicting NBA games results, where numerous performance metrics can lead to challenges such as multicollinearity and overfitting. By reducing the dimensionality of the dataset, PCA enhances model interpretability and performance.

Additionally, PCA helps in visualizing high-dimensional data, allowing us to identify patterns and relationships that may not be easily discernible. Its application in sports analytics enables a more focused analysis of the most impactful features, ultimately leading to improved predictive accuracy in game outcomes.

By integrating PCA into our project, we tried to exclude the features with lower impact to the prediction and successfully get better prediction results. To perform PCA, first, we first standardized the features we use for the predictions. Second, we plotted out the explained variance of the components. Setting a threshold of 0.01, we got rid of the features with low variance, which meant less impact on the predictions. After excluding the features with low variance and fitting the data into the models. We found slight improvement on the three models.

	Logistic Regression Before PCA	Logistic Regression After PCA	Change in %
Accuracy	64.2%	65.4%	+1.2%
Precision	61.9%	62.8%	+0.9%
Recall	66.7%	69.2%	+2.5%
F1-Score	64.2%	66.5%	+2.3%

	Random Forest Before PCA	Random Forest After PCA	Change in %
Accuracy	54.3%	59.3%	+5%
Precision	52.1%	56.8%	+4.7%
Recall	64.1%	64.1%	Unchanged
F1-Score	57.5%	60.2%	+2.8%

	KNN Before PCA	KNN After PCA	Change in %
Accuracy	56.8%	51.8%	-5%
Precision	54.2%	50%	-4.2%
Recall	66.7%	64.1%	-2.6%
F1-Score	59.8%	56.2%	-3.6%

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \quad \text{Precision} = \frac{TP}{TP+FP}$$
$$\text{Recall} = \frac{TP}{TP+TN} \quad \text{F1-score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

From the table above, we can see that LR achieves the highest accuracy among the three models which is around 10% higher than the other two models with or without performing PCA. For random forest and KNN, they both performed at 54.3% and 56.8% respectively. The higher precision and recall rates indicates logistic regression's ability to accurately predict home team winning or losing compared to the other two models. So we will use the logistic regression to perform the simulation in the later parts.

For logistic regression, the reason why it outperforms other models is estimated to be due to the relatively few features compared to the games we used for the predictions. Logistic regression performs better in situations like this, avoiding the risk of overfitting. While for more complex models like Random Forest, there might not have enough data points to support the complexity. Also, as the dataset shows a clear separation between classes, logistic regression may perform better compared to the other two models.

After performing PCA, both logistic regression and random forest have improvement on almost all metrics, while KNN's performance dropped.

For logistic regression, the improvement might be due to the reduction of the multicollinearity. This helps logistic regression to be more stable and reliable.F
For random forest, as it is an ensemble method. Reducing the irrelevant features helps the model to generalize better to unseen data, which helps to improve the model's performance.

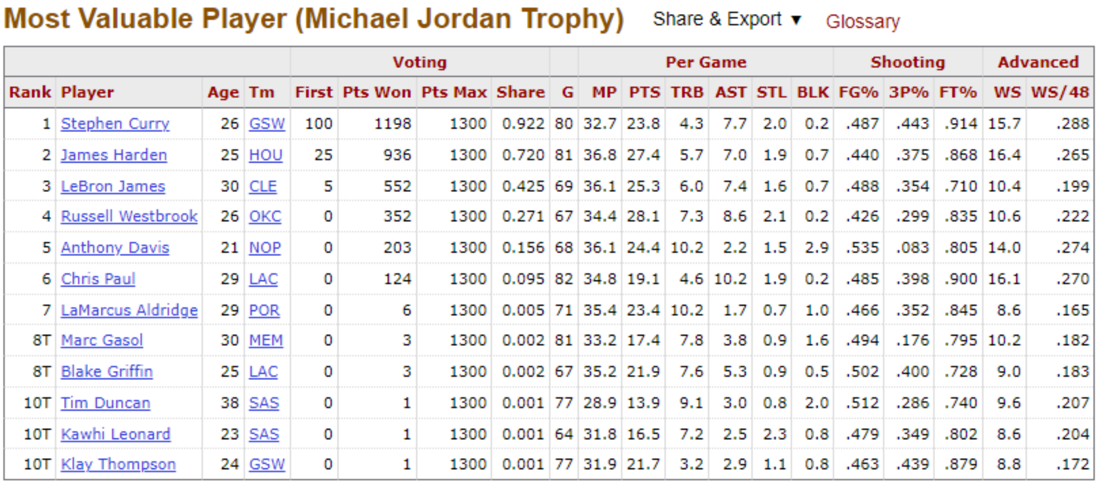
For KNN, as PCA reduces dimensions, if the resulting dimensions still do not represent the data structure well, KNN may suffer from the curse of dimensionality which makes the distance between points become less meaningful, causing the performance to be worse.

In conclusion, the logistic regression model is the most effective model for predicting NBA games results.

B. Predicting Regular Season MVPs

2.2.1. Data Collection *(Hans Junoes)*

Data was collected from the *Basketball Reference* website which records various parameters on matches from 1991 to 2024 (Basketball-Reference, 2024). Four webpages in particular were read to obtain the MVP rank with voting shares, average player statistics per game, advanced player statistics, and team division standings throughout the years which will be joined in the future into a single table. The reason behind the use of advanced player statistics is to improve the model's performance. There are many advanced statistics used for decades in the NBA, such as PER, VORP, BPM, etc. each with their own calculations derived from basketball analysts. These advanced metrics have shown historically to be a good indicator for a player's ability, and the calculations for each can be accessed in the glossary[2]. Below is one such table for the 2014-15 season MVPs obtained from the website.



Most Valuable Player (Michael Jordan Trophy)

				Voting					Per Game						Shooting			Advanced	
Rank	Player	Age	Tm	First	Pts Won	Pts Max	Share	G	MP	PTS	TRB	AST	STL	BLK	FG%	3P%	FT%	WS	WS/48
1	Stephen Curry	26	GSW	100	1198	1300	0.922	80	32.7	23.8	4.3	7.7	2.0	0.2	.487	.443	.914	15.7	.288
2	James Harden	25	HOU	25	936	1300	0.720	81	36.8	27.4	5.7	7.0	1.9	0.7	.440	.375	.868	16.4	.265
3	LeBron James	30	CLE	5	552	1300	0.425	69	36.1	25.3	6.0	7.4	1.6	0.7	.488	.354	.710	10.4	.199
4	Russell Westbrook	26	OKC	0	352	1300	0.271	67	34.4	28.1	7.3	8.6	2.1	0.2	.426	.299	.835	10.6	.222
5	Anthony Davis	21	NOP	0	203	1300	0.156	68	36.1	24.4	10.2	2.2	1.5	2.9	.535	.083	.805	14.0	.274
6	Chris Paul	29	LAC	0	124	1300	0.095	82	34.8	19.1	4.6	10.2	1.9	0.2	.485	.398	.900	16.1	.270
7	LaMarcus Aldridge	29	POR	0	6	1300	0.005	71	35.4	23.4	10.2	1.7	0.7	1.0	.466	.352	.845	8.6	.165
8T	Marc Gasol	30	MEM	0	3	1300	0.002	81	33.2	17.4	7.8	3.8	0.9	1.6	.494	.176	.795	10.2	.182
8T	Blake Griffin	25	LAC	0	3	1300	0.002	67	35.2	21.9	7.6	5.3	0.9	0.5	.502	.400	.728	9.0	.183
10T	Tim Duncan	38	SAS	0	1	1300	0.001	77	28.9	13.9	9.1	3.0	0.8	2.0	.512	.286	.740	9.6	.207
10T	Kawhi Leonard	23	SAS	0	1	1300	0.001	64	31.8	16.5	7.2	2.5	2.3	0.8	.479	.349	.802	8.6	.204
10T	Klay Thompson	24	GSW	0	1	1300	0.001	77	31.9	21.7	3.2	2.9	1.1	0.8	.463	.439	.879	8.8	.172

During the web scraping process with a Selenium chromedriver, a sleep timer needs to be set within the for loops so that not more than 20 pages are accessed per minute according to regulations from the Basketball-Reference website. If more than 20 pages are accessed per minute, the IP address will be banned for an hour. Furthermore, some rows were found to be rendered by javascript after loading the page in the website. After testing, in order to ensure the completeness of scraped data, javascript code to scroll the page from to bottom was utilized to load rows that would otherwise be hidden. Additionally, this also fixes the problem of correctly typing player names with special characters, such as the 'ć' in Nikola Jokić.

2.2.2 Data Preprocessing *(Hans Junoes)*

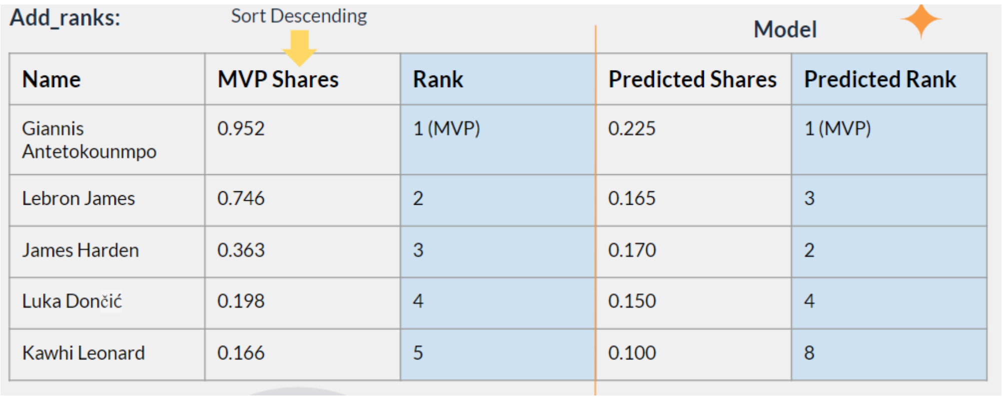
Data preprocessing starts with a different notebook, 'predictors.ipynb'. The data was first parsed via html code to remove overheaders and to obtain the correct tables if there are multiple in a single webpage. Afterwards, it was cleaned to get rid of redundant columns, null values, and maintain consistency in player and team names. Finally, using groupby() functions and join() statements, all four tables are combined into a singular, massive dataset consisting of both normal and advanced metrics of each player, their mvp vote shares, along with their team statistics.

2.2.3. Data Engineering *(Hans Junoes)*

In the third and last notebook, ‘machine_learning.ipynb’, some features are redefined in the form of new variables to aid the training process. For example, ratios per year was created as a normalized value for the total points of a player compared to the average player. This value would replace the total points per player for multiple features across the years. Categorical encoding was given to different teams so that it can enter the model as numeric data.

The feature ‘Shares’ was determined through the ratio of total points gained over the maximum points available in a given season. With this information, add_ranks() adds a new feature to every player according to their MVP shares where the first rank was set as the MVP. This rank would be used in the model to determine the predicted MVP through the sortation of the predicted shares.

Data training and testing were done through backtesting or rollover, similar to part A. Cross-validation was not applied as it does not respect the temporal nature of the data which must be trained and tested in year batches, as rankings are only relevant when inside a given year (for example, rank = 1 in 2015 is not relative to rank = 2 in 2020). This makes the usage of *GridsearchCV* and *RandomsearchCV* difficult or impossible to be applied.



Add_ranks:

Sort Descending

Model

Name	MVP Shares	Rank	Predicted Shares	Predicted Rank
Giannis Antetokounmpo	0.952	1 (MVP)	0.225	1 (MVP)
Lebron James	0.746	2	0.165	3
James Harden	0.363	3	0.170	2
Luka Dončić	0.198	4	0.150	4
Kawhi Leonard	0.166	5	0.100	8

2.2.4. Evaluation Metrics *(Hans Junoes)*

Evaluation metrics include the i) mean average precision (MAP), ii) ‘mean MVP rank’, and the iii) ‘MVP accuracy’. Below is the formula for average precision (AP):

$$AP@K = \frac{1}{N}\sum_{k=1}^{K} Precision\,(k) \times rel\,(k)$$

Where N is the number of relevant items, Precision(k) is the precision calculated at each position, and rel(k) equals 1 if the item at position k is relevant and 0 otherwise (Evidently AI Team, 2024). In our case, we have chosen the cutoff value k = 5. The i) mean average precision (MAP) is the arithmetic mean of all average precision values across the 10 years between 2015 and 2024.

A perfectly precise model will give precise matches for all k=5 relevant items, resulting in a value of 100%.

The ii) ‘mean MVP rank’ is the average predicted rank of the real MVP in the model, wherein a perfect model will always predict the MVP’s rank as 1.

The iii) ‘MVP accuracy’ is the number of times the model correctly predicted the MVP, wherein a perfect model will score 10/10 correct guesses for 2015-2024.

3.Analysis

Implementation/ Experimentation (Chen Man Hin)

Simulations (Betting on Home team wins):

Two main tasks:

-Evaluating the best fit model for betting (1.)

-Simulate a betting scenario based on that model's predictions. (2.)

3.1.1 Model Comparison

We used 3 methods to obtain the results: Logistic Regression ,Random Forest Metrics and KNN Metrics. We determine the higher the accuracy, precision and F-1 Score are, the more likely we would choose the model since we are trying the place bet based on the model predictions.

Results:The Logistics Regression have the highest accuracy, precision and F-1 score. So we choose it for the betting model and start simulation.

3.1.2 Betting Simulation

Parameters: The simulation sets up several parameters:

num_simulations: Maximum **1000** of betting simulations to run.

initial_bet: The amount placed on each bet is **$50**.

accuracy_threshold and precision_threshold: Minimum required of accuracy and precision for placing a bet is **0.6**.

initial_balance: Starting balance for the betting is **1000**.

Odds for Home Team: The odds for betting on the home team are defined by collecting the most recent 200 games home team wins odds for websit 'odds portal'(odds references)[1], and calculate the average odds. The result is **2.538341014**. The appendix would include the 200 games data file 'odds'.

Simulation Loop:

The loop runs up to num_simulations times. In each trials:

It checks if the logistic regression model's accuracy and precision exceed the thresholds(0.5).

If they do, it simulates a bet:

A win or loss is determined based on a comparison of two columns in the pX_test array ,it represents predicted probabilities of home team wins or away team wins. If the first columns is higher than the second columns, which means home team wins. And we only place bet for home team wins whenever accuracy and precision exceed the thresholds.

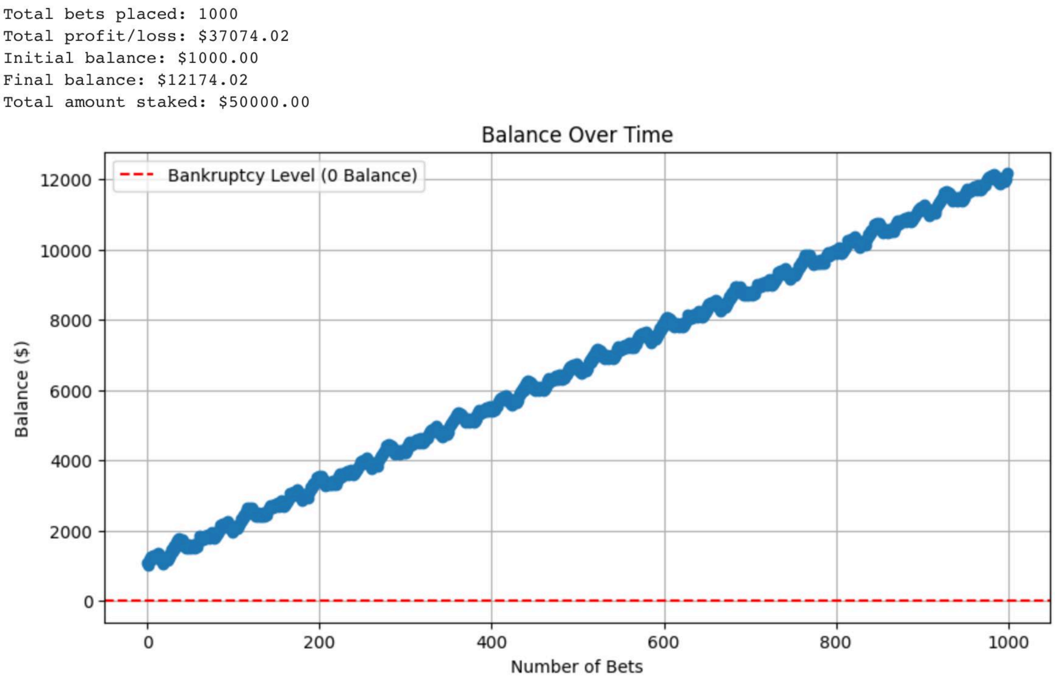
-If the bet **wins**, the balance is updated with the winnings minus the initial bet.

-If the bet **loses**, the balance is decreased by the initial bet.

The current balance is recorded after each bet.

The loop stops if the balance reaches zero or below, indicating bankruptcy.

Balance Plot:



```
Total bets placed: 1000
Total profit/loss: $37074.02
Initial balance: $1000.00
Final balance: $12174.02
Total amount staked: $50000.00
```

The graph illustrates that the balance is increasing over time as we place bets. Which is a really good outcome. However, this betting strategy is based on the logistics model with f-1 score 0.6420 ,usually it should higher than 0.7 in order to simulate the most realistic scenario. In reality, there were still many factor may cause the games to be not as predicted, such as:

- **Injuries**: Key injuries to players can alter a team's chances and overall performance.

- **Home Court Advantage**: Teams often perform better at home due to familiar surroundings and fan support.

- **Matchups**: Specific player matchups can be advantageous or disadvantageous, affecting how teams perform against each other.

- **Coaching Decisions**: Strategies, substitutions, and time management by coaches can impact the game's outcome.

B. Predicting Regular Season MVPs

3.2.1. Results *(Hans Junoes)*

Four methods were used to obtain the results: random forest, ridge regression, extra trees, and gradient boosting. For the random forest method, the number of decision trees N was set to 500 and 3 minimum samples were taken. The same parameters were also set for the extra trees method. For ridge regression, an alpha of 0.1

The obtained results are given in the following table:

	Mean Average Precision	MVP Accuracy	Mean MVP Rank
Random Forest	**90.32%**	8/10	**1.2**
Ridge Regression	89.54%	5/10	1.5
Extra Trees	89.85%	8/10	1.5
Gradient Boosting	89.38%	**9/10**	**1.2**

The mean average precision ranges between 89.38% to 90.32% across all methods with the random forest method achieving the highest precision. The MVP accuracy obtained was 9 correct guesses out of 10 for the gradient boosting method, 8/ 10 for the random forest method and extra trees method, and 5/10 for the ridge regression method. The mean MVP rank ranges from 1.2 to 1.5 with a lower value being better. The random forest method and gradient boosting method obtained the mean rank of 1.2 whereas the rest obtained a mean rank of 1.5.

3.2.2. Interpretation *(Hans Junoes)*

Year	Actual MVP	Predicted Rank			
		Random Forest	Ridge Regression	Extra Trees	Gradient Boosting
2015	Stephen Curry	1	2	1	1
2016	Stephen Curry	1	1	1	1
2017	Russell Westbrook	2	1	5	1
2018	James Harden	1	2	1	1
2019	Giannis Antentokounmpo	1	2	1	1
2020	Giannis Antentokounmpo	1	2	1	1
2021	Nikola Jokić	1	1	1	1
2022	Nikola Jokić	1	1	1	1
2023	Joel Embiid	2	2	2	3
2024	Nikola Jokić	1	1	1	1

The table above represents the predicted rank of each MVP from 2015 to 2024 from each method. Inaccuracies can be found on the predictions, particularly in 2023 where all methods fail to predict the MVP correctly. We can further interpret this result by looking at the events that happened during that year.

According to our model and the general public in 2023, Nikola Jokić was objectively the best player in 2023. However, Joel Embiid was chosen as the MVP instead due to critics and analysts agreeing that Nikola Jokić has gotten the previous two MVPs which resulted in a bias against selecting Nikola Jokić for the third time in a row, dubbed 'voting fatigue' (Beame, 2023).

Conversely, 2017 was a year where 4 candidates each had solid reasons to win an MVP as they individually had a great season. The 4 players were Russell Westbrook, James Harden, Kawhi Leonard, and Lebron James. Strictly looking at pure statistics and numbers, it will be difficult to pinpoint the MVP. However, there are many things that can not be represented in the data, as one article has suggested. Harden was coming off a terrible previous season, James already had a wealthy resume under his belt and is by far the most popular player in the NBA, hence suffering from voter fatigue; Leonard had always lacked charisma from the start of his career, ultimately leading to Westbrook taking home the trophy (Hughes, 2017). Another source was bent on Harden winning MVP for the nearly the entire season, until Westbrook started hammering memorable game-winners and triple-doubles toward the end of the season, creating recency bias when it was time to vote (Yahoo Sports Staff, 2020).

4.Conclusion

A. Predicting NBA games results *(Ng Sze Leong)*

With basketball being a more and more popular sport around the globe, sportsbetting is also getting a lot more popular. Using machine learning to perform sports analytics has become a hot topic for gamblers. As basketball enthusiasts and students in Statistics, we tried to combine what we like and we have learnt to perform the prediction on NBA games results, not only to learn money, but also to see whether our favourite team can win in the future and maybe earn a little using machine learning techniques. Although the performance of the models is not that good and may not be feasible for real-world uses, we understand that it is very difficult to use past years data to predict the current season. This is because there are a lot of features to be measured if we truly want to get the results very accurately. For example, there are roster changes, rules, coaches, rookies, the tactics, trade of players…etc. So it is expected that the way we are using to predict the games may not be accurate enough. As there are already thousands of features to be used in NBA.com, to get a more accurate result, we will need to include more and more features. Yet, this may also cause a lot more time and money.

B. Predicting Regular Season MVPs *(Hans Junoes)*

The objective of predicting regular season MVPs was to gain a deeper understanding towards the factors that best indicate an MVP caliber player, and to possibly bet on the sport to gain money. The two best models in predicting the MVP for the last 10 years were Random Forest and Gradient Boosting Regressor, with 8/10 and 9/10 accuracy in 2015-2024 respectively. Both models present a reasonable justification for application in real-world betting scenarios.

As has been gleaned in the previous analysis, there are factors and storylines for players in the NBA that are extremely difficult if not impossible to be represented in data alone, which can and have flirted with our cognitive bias as humans with emotions watching the game. Further improvements regarding notable historical seasons, recency bias, and voting fatigue can be explored in the future.

References

1. ESPN. (n.d.). *ESPN: The worldwide leader in sports*. Retrieved December 12, 2024, from https://www.espn.com/
2. ResearchGate. (n.d.). *Example of random forest classifier [Figure].* Retrieved December 12,2024,from

https://www.researchgate.net/figure/Example-of-random-forest-classifier_fig5_368878301

3. Javatpoint. (n.d.). *K-nearest neighbor algorithm for machine learning.* Retrieved December12,2024,from https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning
4. [Main webpage of the website]. (n.d.). Basketball Reference. https://www.basketball-reference.com/
5. Evidently AI Team. (2024). *Mean average precision (MAP) in ranking and recommendations*. Evidently AI. Retrieved November 22, 2024, from https://www.evidentlyai.com/ranking-metrics/mean-average-precision-map
6. Beame, A. (2023). *Why Nikola Jokic winning a third NBA MVP in a row would be bad for basketball. SBNation*. Retrieved November 30, 2024, from https://www.sbnation.com/nba/2023/3/22/23651077/nikola-jokic-nba-mvp-denver-nuggets-vote-please-no
7. Hughes, G. (2017). The case against every 2017 NBA MVP candidate. Bleacher Report. Retrieved December 11, 2024, from https://bleacherreport.com/articles/2701679-the-case-against-every-2017-nba-mvp-candidate
8. Yahoo Sports Staff. (2020). *Woulda, coulda, shoulda: Was Russell Westbrook a deserving MVP in 2016-2017*? Yahoo Sports. Retrieved December 11, 2024, from https://sports.yahoo.com/woulda-coulda-shoulda-was-russell-westbrook-a-deserving-mvp-in-201617-200035672.html
9. OddsPortal. (n.d.). *NBA results*. Retrieved December 14, 2024, from https://www.oddsportal.com/basketball/usa/nba/results/