Analysis of NBA players data

Table of Contents

[Dataset description and data cleaning 4](#_Toc121293620)

[Correlation matrix 6](#_Toc121293621)

[Searching for outliers 6](#_Toc121293622)

[Regression models 8](#_Toc121293623)

[Univariate linear regression 8](#_Toc121293624)

[Univariate linear regression with splitted dataset 9](#_Toc121293625)

[OLS linear regression 9](#_Toc121293626)

[Multivariate linear regression 10](#_Toc121293627)

[Univariate polynomial regression 10](#_Toc121293628)

[Bivariate polynomial regression 15](#_Toc121293629)

[Ridge (alpha = 0.05) 17](#_Toc121293630)

[Lasso (alpha = 0.05) 17](#_Toc121293631)

[Elasticnet (alpha = 1.0, l1\_ratio=0.5) 17](#_Toc121293632)

[SVR Kernel test 17](#_Toc121293633)

[SVR epsilon test 17](#_Toc121293634)

[SVR C test 18](#_Toc121293635)

[Decision tree regressor (*criterion='squared\_error'*) 18](#_Toc121293636)

[Random forest regressor 18](#_Toc121293637)

[Finding best parameters for random forest regressor 19](#_Toc121293638)

[Classification models 19](#_Toc121293639)

[Logistic regression 19](#_Toc121293640)

[kNN – 3 components 20](#_Toc121293641)

[SVC Linear kernel 20](#_Toc121293642)

[SVC Polynomial kernel 3rd grade 21](#_Toc121293643)

[SVC Polynomial kernel 4th grade 21](#_Toc121293644)

[SVC Polynomial kernel 5th grade 22](#_Toc121293645)

[SVC rbf kernel 23](#_Toc121293646)

[SVC Sigmoid kernel 24](#_Toc121293647)

[Decision tree classifier with gini criterion 24](#_Toc121293648)

[Decision tree classifier with entropy criterion 25](#_Toc121293649)

[Random forest classifier default params 26](#_Toc121293650)

[Random forest classifier modified params 27](#_Toc121293651)

[PCA and TSNE with two componenets 27](#_Toc121293652)

[tSNE with X unscaled vs tSNE with X scaled (2 components) 28](#_Toc121293653)

[tSNE with X unscaled vs tSNE with X scaled (3 components) 29](#_Toc121293654)

[Clustering 29](#_Toc121293655)

[Elbow method 29](#_Toc121293656)

[kMeans with 2 clusters 30](#_Toc121293657)

[kMeans with 3 clusters 30](#_Toc121293658)

[Agglomerative clustering with 2 clusters vs 3 clusters 31](#_Toc121293659)

[Conclusion 32](#_Toc121293660)

[For regression problem 32](#_Toc121293661)

[For classification problem 32](#_Toc121293662)

[For clustering 34](#_Toc121293663)

[Bibliography 35](#_Toc121293664)

# Dataset description and data cleaning

This dataset contains individual player data from all games from start of 1996-97 season of NBA

Dataset link: https://www.kaggle.com/datasets/kenhuang41/nba-basic-game-data-by-player

**Initial dataset shape: (743423, 27)**

Dataset info

A picture containing graphical user interface

Description automatically generated

**The dataset has no missing data**

**We will use label encoding for the following columns: “RESULT”, “STARTER”**

**The following columns were discarded: “GAME\_ID”, “TEAM”, “OPPT”, “RESULT”, “PLAYER”, “STARTER”, and all columns data types will be converted to int**

**Dataset info now**

A picture containing graphical user interface

Description automatically generated

**Dataset shape: (743423, 23)**

# Correlation matrix

Chart, treemap chart

Description automatically generated

# Chart, box and whisker chart Description automatically generatedChart, box and whisker chart Description automatically generatedSearching for outliers

Chart, box and whisker chart

Description automatically generated

For PTS and FGA we have outliers.

After outliers handling

Chart, box and whisker chart

Description automatically generated

Chart, box and whisker chart

Description automatically generated

Chart, box and whisker chart

Description automatically generated

Removing remaining PTS outliers

Shape after handling outliers: **(728112, 23)**

Number of outliers: 15311

# Regression models

For the regression problems, we want to predict field goals attempted by using minutes played as feature. For multi-valued regressions, we add “total points” as a feature

For the first regression models, we have R^2 and execution times for the full dataset and for the trimmed one

## Univariate linear regression

Background pattern

Description automatically generated

Univariate linear regression R^2: 0.7298672387333168 0.7168365599165089

Lin reg exec time: 0.1692979335784912 0.01358485221862793

## Univariate linear regression with splitted dataset

Background pattern

Description automatically generated

Univariate linear regression splited dataset R^2 : 0.7292715673468013 0.7166562492841991

Univariate linear regression splitted dataset exec time: 0.09502601623535156 0.0597085952758789

## OLS linear regression

Graphical user interface, text

Description automatically generated

OLS exec time: 0.2756638526916504 0.06481599807739258, and 0.881 R^2

We can observe that OLS has a higher R^2 value than the models presented before, and the execution time is slightly increased

## Multivariate linear regression

Chart

Description automatically generated

R^2 Multivariate Linear Regression: 0.8553742663896492 0.8519306996384368

Multivariate linear regression exec time: 0.16484951972961426 0.03317880630493164

## Univariate polynomial regression

Background pattern

Description automatically generated with medium confidence

Univariate polynomial regression 2nd degree R^2: 0.7334636726804015

Univariate polynomial regression 2nd degree exec time: 0.27399277687072754

Background pattern

Description automatically generated with medium confidence

Univariate polynomial regression 3rd degree R^2: 0.7340628893494789

Univariate polynomial regression 3rd degree exec time: 0.20400309562683105

Background pattern

Description automatically generated

Univariate polynomial regression 4th degree R^2: 0.7347345911009611

Univariate polynomial regression 4th degree exec time: 0.2499988079071045

Background pattern

Description automatically generated

Univariate polynomial regression 5th degree R^2: 0.7347418725011717

Univariate polynomial regression 5th exec time: 0.29399824142456055

Background pattern

Description automatically generated

Univariate polynomial regression 6th degree R^2: 0.7347525237974357

Univariate polynomial regression 6th degree exec time: 0.31601786613464355

Background pattern

Description automatically generated

Univariate polynomial regression 7th degree R^2: 0.734769215945446

Univariate polynomial regression 7th degree exec time: 0.4379899501800537

Background pattern

Description automatically generated

Univariate polynomial regression 8th degree R^2: 0.7347773990784456

Univariate polynomial regression 8th degree exec time: 0.5250029563903809

Background pattern

Description automatically generated

Univariate polynomial regression 9th degree R^2: 0.7347781786249864

Univariate polynomial regression 9th degree exec time: 0.5899999141693115

Chart

Description automatically generated

Univariate polynomial regression 10th degree R^2: 0.733992977810007

Univariate polynomial regression 10th degree exec time: 0.5657339096069336

## Bivariate polynomial regression

Chart

Description automatically generated

Bivariate polynomial regression 9th degree R^2: 0.8564746161282775

Bivariate polynomial regression 9th degree exec time: 4.408908843994141

For the trimmed dataset

Chart

Description automatically generated

Chart

Description automatically generated

POLY Multivariate R^2: 0.8533056675075275

Multivariate poly reg exec time: 0.6795823574066162

## Ridge (alpha = 0.05)

Full dataset Trimmed dataset

Ridge R^2: 0.7292715673473285 Ridge R^2: 0.716656249264487

Ridge exec time: 0.10999369621276855 Ridge exec time: 0.0314335823059082

## Lasso (alpha = 0.05)

Lasso R^2: 0.7292718759888268 Lasso R^2: 0.7166520799191645

Lasso exec time: 0.12700510025024414 Lasso exec time: 0.030445337295532227

## Elasticnet (alpha = 1.0, l1\_ratio=0.5)

ElasticNet R^2: 0.7292000675845649 ElasticNet R^2: 0.7165063049970547

ElasticNet exec time: 0.13499855995178223 ElasticNet exec time: 0.02519059181213379

For the following models, the dataset was trimmed to 100000 entries

## SVR Kernel test

Line chart

Description automatically generated with low confidenceRegression with kernel linear and score: 0.7127001601799997

Time of execution: 354.6275107860565 seconds

Regression with kernel poly and score: 0.5955668381718753

Time of execution: 1648.4259049892426 seconds

Regression with kernel sigmoid and score:

-137017.92136434338

Time of execution: 360.06326627731323 seconds

Regression with kernel rbf and score: 0.7212857513608553

Time of execution: 241.22344136238098 seconds

## SVR epsilon test

Regression with epsilon 0 and score: 0.7213511622260207

Time of execution: 277.22791147232056 seconds

Regression with epsilon 0.01 and score: 0.7212995552063458

Chart, line chart

Description automatically generatedTime of execution: 670.025139093399 seconds

Regression with epsilon 0.1 and score: 0.7212857513608553

Time of execution: 239.8785696029663 seconds

Regression with epsilon 0.5 and score: 0.7208447403891828

Time of execution: 205.81772303581238 seconds

Regression with epsilon 1 and score: 0.7158019104929585

Time of execution: 412.7857210636139 seconds

Regression with epsilon 2 and score: 0.6980899118539192

Time of execution: 362.8528501987457 seconds

Regression with epsilon 4 and score: 0.6143328160421295

Time of execution: 544.4271202087402 seconds

## Chart, line chart Description automatically generatedSVR C test

Regression with C 10 and score: 0.7213637678508715

Regression with C 20 and score: 0.7213646336850346

Regression with C 30 and score: 0.7213627730263783

Regression with C 40 and score: 0.7213635052645782

Regression with C 50 and score: 0.721366621241895

Regression with C 60 and score: 0.721366141041464

Regression with C 70 and score: 0.7213629276337369

Regression with C 80 and score: 0.7213623255351538

Regression with C 90 and score: 0.7213611404644906

## Decision tree regressor (*criterion='squared\_error'*)

Decision tree score: 0.721770120426478

Execution time: 0.03259921073913574

## Random forest regressor

Random forest regressor score: 0.6853272465870539

Exec time: 0.723644495010376

## Finding best parameters for random forest regressor

Random forest regressor with best params score: 0.7234004290077604

Best params: {'max\_depth': 300, 'n\_estimators': 50}

Exec time: 147.81057024002075

# Classification models

For the classification problem, we try to classify the game as won or lost based off the goal attempts and the minutes played of a player

## Logistic regression

Chart, treemap chart

Description automatically generated

Calendar

Description automatically generated

Logistic score: 0.5060333333333333

Time of execution: 0.14763188362121582 seconds

## Chart, treemap chart Description automatically generatedkNN – 3 components

kNN score without cross-validation: 0.499

Time of execution: 14.469719886779785 seconds

Cross-validation scores: [0.49875 0.505875 0.503625 0.504625 0.510125 0.496875 0.511875 0.502625

0.495375 0.505125]

Average cross-validation score: 0.5034875

Exec time 4.478180170059204

## Chart, treemap chart Description automatically generatedSVC Linear kernel

KERNEL: linear

Score 0.5042

Training score: 0.502

[[ 4398 15631]

[ 4201 15770]]

Time of execution: 214.3196897506714 seconds

A picture containing text, music

Description automatically generated

Text, calendar

Description automatically generated with medium confidence

## Chart, treemap chart Description automatically generatedSVC Polynomial kernel 3rd grade

Calendar

Description automatically generated

Bar chart

Description automatically generated with low confidence

## Chart, treemap chart Description automatically generatedCalendar Description automatically generatedSVC Polynomial kernel 4th grade

Bar chart

Description automatically generated with low confidence

## SVC Polynomial kernel 5th gradeChart, treemap chart Description automatically generated

Calendar

Description automatically generated

A picture containing text, music

Description automatically generated

## Chart, treemap chart Description automatically generatedSVC rbf kernel

A picture containing graphical user interface

Description automatically generatedCalendar

Description automatically generated

## Chart, treemap chart Description automatically generatedCalendar Description automatically generatedSVC Sigmoid kernel

Shape, arrow

Description automatically generated with medium confidence

## Decision tree classifier with gini criterion

A picture containing table

Description automatically generated

Timeline

Description automatically generated with medium confidence

## Decision tree classifier with entropy criterion

Calendar

Description automatically generated with low confidence

Timeline

Description automatically generated

## Random forest classifier default params

Calendar

Description automatically generated

## Random forest classifier modified params

Calendar

Description automatically generated

# PCA and TSNE with two componenets

Chart, scatter chart

Description automatically generated

A picture containing text, vector graphics

Description automatically generated

# tSNE with X unscaled vs tSNE with X scaled (2 components)

A picture containing text

Description automatically generated

# tSNE with X unscaled vs tSNE with X scaled (3 components)

Background pattern

Description automatically generated

# Clustering

## Elbow method

Shape

Description automatically generated

# kMeans with 2 clusters

Chart, scatter chart

Description automatically generated

DB coefficient: 0.6100047050362262

# kMeans with 3 clusters

Chart, scatter chart

Description automatically generated

DB coefficient: 0.6800155201989435

# Agglomerative clustering with 2 clusters vs 3 clusters

Background pattern

Description automatically generatedBackground pattern

Description automatically generated

DB coefficients: 0.6872315810783166 0.726256757000454

# Conclusion

## For regression problem

|  |  |  |
| --- | --- | --- |
| Regression model | Score | Execution time |
| OLS | 0.882 | 0.275663853 |
| Bivariate polynomial regression 9th degree R^2 | 0.856474616 | 4.408908844 |
| R^2 Multivariate Linear Regression | 0.855374266 | 0.16484952 |
| Univariate polynomial regression 9th degree R^2 | 0.734778179 | 0.589999914 |
| Univariate polynomial regression 8th degree R^2 | 0.734777399 | 0.525002956 |
| Univariate polynomial regression 7th degree R^2 | 0.734769216 | 0.43798995 |
| Univariate polynomial regression 6th degree R^2 | 0.734752524 | 0.316017866 |
| Univariate polynomial regression 5th degree R^2 | 0.734741873 | 0.293998241 |
| Univariate polynomial regression 4th degree R^2 | 0.734734591 | 0.249998808 |
| Univariate polynomial regression 3rd degree R^2 | 0.734062889 | 0.204003096 |
| Univariate polynomial regression 10th degree R^2 | 0.733992978 | 0.56573391 |
| Univariate polynomial regression 2nd degree R^2 | 0.733463673 | 0.273992777 |
| Univariate linear regression R^2 | 0.729867239 | 0.169297934 |
| Lasso R^2 | 0.729271876 | 0.1270051 |
| Ridge R^2 | 0.729271567 | 0.109993696 |
| Univariate linear regression splited dataset R^2 | 0.729271567 | 0.095026016 |
| ElasticNet R^2 | 0.729200068 | 0.13499856 |
| Random forest regressor with tunned params | 0.723400429 | 147.8105702 |
| Decision tree regressor | 0.72177012 | 0.032599211 |
| SVR rbf kernel, epsilon 0, C 50 | 0.721366621 |  |
| Random forest regressor | 0.685327247 | 0.723644495 |

The regression model with the highest accuracy score is “OLS”, and the one that has the lowest score is random forest regressor. In terms of execution time, the lowest execution time is for the “Decision tree regressor”, and the one that has the highest execution time is “Random forest regressor with tunned params”.

For the polynomial regression model, the degree that achieved the highest score is 9th degree, and the one with the lowest score was the 2nd degree.

## For classification problem

|  |  |  |
| --- | --- | --- |
| Classificaiton model | Score | Execution time |
| SVC rbf | 0.513575 | 279.8457 |
| Decision tree gini | 0.51335 | 0.175774 |
| Decision tree entropy | 0.51335 | 0.151148 |
| Random forest default | 0.5109 | 4.354203 |
| Random forest modified params | 0.51055 | 8.033483 |
| SVC sigmoid | 0.509425 | 190.6903 |
| Logistic | 0.506033 | 0.147632 |
| SVC linear | 0.5042 | 214.3197 |
| kNN - 3 components CV | 0.503488 | 4.47818 |
| SVC polynomial 3rd grade | 0.5016 | 179.6800 |
| SVC polynomial 5th grade | 0.5007 | 204.5473 |
| SVC polynomial 4th grade | 0.5004 | 198.2244 |
| kNN - 3 components | 0.499 | 14.46972 |

The model with the highest accuracy is “SVC with rbf kernel”, and the one with the lowest accuracy score is “kNN with 3 components”. Also, SVC with rbf has the highest execution time, and the “Logistic” model has the lowest one.

Chart, bar chart

Description automatically generated

The model with the most true positive values is SVC polynomial 5th degree, which also has the most false positive values.

The model with the most true negatives values is SVC linear, which also has the most false negative values.

## For clustering

The elbow method tells us that the optimal number of clusters is 2 or 3.

For kMeans with 2 clusters, the DB coefficient is 0.6100047050362262, which is better than kMeans with 3 clusters with DB coefficient of 0.6800155201989435.

For agglomerative clustering we trimmed the data to 30000 entries,

For two clusters we got the DB coefficient 0.6872315810783166, which is better then three clusters with 0.726256757000454

# Bibliography

<https://www.kaggle.com/code/brixkt/polynomial-and-multiple-regression-in-3d-plot/notebook>

<https://pandas.pydata.org/docs/reference/frame.html>

<https://numpy.org/doc/stable/index.html>

<https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html>

<https://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html>

<https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html>

<https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html>

<https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html>

<https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html>